Universität Bremen
Fachbereich 3, Mathematik/Informatik
Studiengang Informatik

Diploma Thesis

Spatio-Temporal Real-Time Analysis of Dynamic Scenes in the RoboCup 3D Soccer Simulation League

Tobias Warden
Mail: warden@projectiwear.org, Matrikelnr.: 1546401

December 12, 2007

1. Gutachter: Dr. habil. Ubbo Visser
2. Gutachter: Prof. Dr. Kerstin Schill

Official Statement

I hereby confirm that I have written this masters thesis (Diplom) at hand without interference by third parties. I have not used resources and means other than those explicitly declared throughout the thesis. All text passages which are either literally or analogously gathered from scientific publications are indicated adequately.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Diplomarbeit selbständig ohne die Hilfe Dritter verfasst habe. Ich habe keine anderen als die explizit angegebenen Quellen und Hilfsmittel verwendet. Wörtliche oder sinngemäße Zitate sind in der Arbeit stets als solche gekennzeichnet.

Bremen, December 12, 2007

Tobias Warden

Contents

1 Introduction

1.1 Motivation

The plan for this masters thesis (Diploma) was conceived as a result of the author's continuous involvement in Virtual Werder 3D, the RoboCup soccer simulation league team of the University of Bremen since its foundation back in March 2004. Due to this the motivation section begins from the point of view of a long-time developer of the existing multi-agent framework which includes a short digest of the work covered so far in our working group[1]. Starting from this viewing angle the research problem is identified and the scientific motivation for this thesis is developed.

In 2005 Virtual Werder 3D participated without noteworthy success in the RoboCup world championships in Osaka, Japan. Back then the team used the second version of the Virtual Werder multi-agent code base[2] [KLP+05] which was not yet competitive on a world-class level neither against other German simulation league teams[3] nor against strong international teams[4].

In the wake of the aforementioned RoboCup event an in-depth analysis of the existing framework infrastructure was conducted at the end of which the team felt that without a radical new multi-agent architecture implemented from scratch, further progress would be very hard to achieve.

This is why in 2006 Virtual Werder 3D participated with a new code base at the RoboCup world championships in Bremen, Germany [LRS+06, LPR+06]. Our team was able to reach the quarter finals, ranking at place eight from 32 participating teams at the end of the competition. The result proved that due to continued reviews and optimizations the code base had matured enough to be competitive under tournament conditions. It had also reached a level where it was feasible to consider using it to foster scientific research dedicated to a deeper treatment of some major challenges in the development of a team of simulated soccer agents which were still untouched by the team so far.

One of those challenges motivated the research done in this masters thesis. The chal-

1 A more elaborate treatment of the history of Virtual Werder is presented in section 2.2.
2 The first embodiment used in 2004 was build on top of the agenttest code base provided together with the soccer server. In 2005 Virtual Werder 3D used a self-developed code base for the first time.
3 such as Brainstormers 3D (Osnabrück), AT Humboldt 3D (Berlin) or RoboLog 3D (Koblenz)
4 such as Aria 3D (Iran), Caspian (Iran) or ZJUBase (China)

lenge has been outlined in a clear and concise manner by Stanton & Williams in [SW04, p.758]:

"*One of our motivations [...]is to build a robot soccer multi-agent system in which each robot knows it is playing soccer and understands all the important elements of the game such as it is a ball game with a set of rules that govern how it is played.*"

The challenge thus is to enable simulated soccer agents to understand the dynamic nature of a game of (simulated) soccer. At the time when the proposal for this diploma thesis was compiled the agent's knowledge representation built upon an assumption which involved a coarse simplification of the understanding of *dynamic scenes*[5] which may be summarized as follows: In order to understand the situation on the field it is satisfactory to assemble and maintain sufficiently complete knowledge about the geometrical layout of the dynamic scene on the soccer field where geometrical layout refers to absolute or relative positions of game-relevant objects and the spatial relations between those objects.

An appropriate metaphor for the perception of an agent at that time is that of a camera snapshot taken from the agent's point of view at a particular moment during the game.

In server versions prior to 0.5 an omni-directional camera with a complete 360 degrees field of vision was used. At that time the observation of the agents was made up of a single snapshot. Subsequent perceptions completely replaced the agent's observation. In order to extend the agent's knowledge about the situation in the temporal dimension, a sequence of observations was maintained, some of them recent observations describing the immediate past, some of them artificially assembled observations hypothesizing the immediate future.

In early 2006 the 0.5 series of the soccer server was introduced, which replaced the omni-cam with a camera featuring a 180 degrees field of vision. Thus, the relation of perceptions and observations was revised. Since each single perception was guaranteed to be incomplete, each new perception was now integrated into the existing observation in order to calculate an approximation of the current observation. The acquisition of complete snapshots got more complicated yet the system of knowledge representation built upon the snapshot sequence remained the same.

The critical question that remained was: Given the properties of the task environment (playing a game of simulated soccer in a 3D world with the goal of winning that game) is a snapshot sequence an adequate representation of the dynamic scenes under inspection?

A classification of the task environment according to the categorization introduced by Russel & Norving in [RN95, chapter 2.3] produces the following result: In comparison to other task domains simulated soccer is particularly hard and complex. First of course it is only *partially observable*. Due to the complexity of the environment and the role of random in the generation of noise for the agent's sensations the task environment certainly appears to be *stochastic*. It is *non-episodic* and thus sequential. It is a *dynamic environment* by nature with *discrete time* but several *continuous aspects in the agent sensors*. Finally, it is a *multi-agent environment* with *cooperative* (team play) as well as *competitive aspects* (due to the opponent team).

In the light of this classification at least two problems arise with regard to the adequacy of a knowledge representation based on the snapshot sequence metaphor that immediately catch the reader's eye.

The first problem arises due to the sequential nature of the task environment. The actions which are performed by soccer agents at a single point in time have a natural temporal dilation spanning multiple observations. When only the snapshot sequence is used information concerning the unfolding of agent actions is stored implicitly within

5 According to Miene in [Mie04a] a *dynamic scene* "*consists of a set of objects that change their position over time*" (translation by the author).

the geometric flow[6]. Thus there's little or no chance to actively exploit this kind of information.

The second problem which can be identified is due to the fact that actions carried out by the remaining 21 agents on the soccer field cannot simply be attributed to the environment itself as the philosophy behind the snapshot metaphor implies. On the contrary: The classification of the task environment dictates that there are 22 intentional entities involved in the simulation. Intention in this context means that the soccer agents interact with their environment in order to achieve certain goals. Whether these goals are explicitly formulated or implicitly weaved into the agent code is a subordinated fact. The important issue is that the simulated soccer players should be treated as *rational agents*[7] whose actions are performed with purpose.

The explicit representation of those intentional actions lies beyond the capabilities of the snapshot-based knowledge representation due to the fact that it works in an unintrusive manner with respect to the state of mind of the other actors in the dynamic scene. Only those pieces of information can be represented which can be derived by unbiased visual observation of the geometric layout/flow in the dynamic scene.

In order to understand what is happening in the soccer environment inhabited by 22 intentional agents the dynamic scenes need to be interpreted in light of soccer expert knowledge. Since it is safe to attribute intentionality to the scene actors their actions should be recognized by exploiting not only the fundamental information provided by the geometrical flow in the dynamic scenes but also expert knowledge about the temporal composition of intentional actions. The latter allows the retrieval of actions within the snapshot sequence. To bring about this higher level of understanding which implies a qualitative aspect in knowledge representation a variation of spatio-temporal analysis of dynamic scenes as introduced by Miene [Mie04a] will be applied.

1.2 Scope and Intention

As already mentioned in the previous section the goal of this thesis is the design and implementation of a system for spatio-temporal analysis of dynamic scenes in the RoboCup *3D Soccer Simulation League* [KO04].

The design of a system for general-purpose spatio-temporal analysis in arbitrary dynamics scenes originating from a broad range of application domains marks a long-term goal in our research group [LPR+06]. Noteworthy fundamental work in this research field with immediate relevance for this thesis has been acquired both by Miene [MV02, MVH04, Mie04a] and Gehrke [Geh05] (cf. section 4.1.1) in chapter 'Spatio-Temporal Analysis of Dynamic Scenes - An Overview', p.29). Due to the constraints imposed by scope and duration of a masters thesis the work presented here is restricted to the application domain of simulated soccer.

It is worthwhile to identify the purpose of the analysis to be performed in the context of further developments of the Virtual Werder team. First and foremost the approach presented in subsequent chapters is meant to improve the grounding situation of the soccer agents playing autonomously on the simulated soccer field. The thesis seeks to make a contribution on our way towards an intelligent soccer agent that actually *understands* the complex situation in its immediate environment.

6 Geometrical flow in this context should be understood as the change of the geometrical layout of a dynamic scene over the course of time

7 as introduced by Russel & Norvig in [RN95, section 2.2]

1.2.1 Enabling a Rich Qualitative Representation

This thesis proposes the use of a rich qualitative representation within the agent's world model provided by means of a spatio-temporal analysis of dynamic scenes as a step in this direction. The qualitative representation complements rather than replaces the accordant quantitative representation. The soccer agents thus gain a hybrid knowledge base.

While all relevant information about the course of the game is already encoded in the quantitative representation, by providing a qualitative layer in the agent knowledge base, knowledge about concepts with a certain temporal dilation, describing dynamic aspects of a soccer game, are encoded in an explicit and concise manner.

The qualitative representation should be rich in that it entails both knowledge about actions or action sequences as well as knowledge suitable to describe the situation context in a game of simulated soccer. An example for a relevant situation context is the ball control state which among other things describes whether the ball is controlled by single team, hard-fought for or free. Such pieces of qualitative information have a lower complexity in comparison to single- or multi-player actions. Yet they may not be neglected when a multi-layered and thus versatile qualitative representation is formulated as a goal which, while per se not leading to an immediate improvement in the agents' game performance, constitutes a mandatory prerequisite for subsequent research efforts in neighboring research fields which bear the promise of an increased competitiveness of the soccer agents. Examples for such research fields comprise amongst others plan recognition, action/intention prediction [Elf07] and the learning of opponent models.

1.2.2 Indirect Benefit for Overall Agent Performance

Plan recognition approaches typically seek to identify the current strategy of agent groups which characterize the typical style of play of an agent team such as a *quick change of wings and subsequent attack over the left wing concluded by a high cross right into the penalty area taking the adversary's defense off-guard by placing the ball in the defender's back*. Plans bear noteworthy importance in the domain of simulated soccer due to the fact that, even though to a lesser extent than American football, soccer is a tactic-oriented team sport. If the plans of an adversary team can be recognized in due time effective counter measures are likely to be more effective as compared to pure reactive behavior.

A proper qualitative representation of the dynamic scene comes in handy due to the fact that the strategic moves which are considered can be thought of as compositions of simpler actions performed by the agents that are involved in the plan execution. Thus the basic qualitative building blocks are provided to fuel plan recognition approaches.

A noteworthy multi-stage research project which couples spatio-temporal analysis and plan recognition has been demonstrated by the *VITRA* project [Her95a] (cf. chapter 4 on page 29) where the *SOCCER* system [HG89] provided qualitative input data such as recognized actions for the *REPLAI* system [RS88, Ret91] dedicated to high-level plan and intention recognition.

The ability to recognize plans leads to the compilation of sophisticated opponent models which in turn allows the development of opponent-sensitive auto-adaptive behaviors for the own team. It is immediatly apparent that such behaviors bear the potential to outperform static behaviors which disregard the specific properties of the respective adversary team and cannot possibly exploit potential weaknesses in the opponent strategy.

The analysis approach developed in this thesis should be considered as a first step on towards an intelligent use of the rich amount of information provided by the environment through sensor data.

1.2.3 Immediate Benefit for Online Statistics

It should also be noted that while not constituting the driving purpose of the assembly and maintainance of a qualitative representation of the world the availability thereof provides immediate benefits such as the ability to maintain online game statistics based on the situation context. Such statistics can provide clues whether or not certain strategies are successful against the respective adversary team. The term *strategies* in this context relates to several levels of granularity such as team-wide game strategies, the system of play (e.g. *4-4-2*, *Catenaccio*[8]) or global style of play (e.g. *kick'n'rush*, *short-pass-oriented*) as well as to preferences with respect to the character of action execution (e.g. low vs. high passes). The exploitation of online game statistics has already been proposed for integration into the Virtual Werder team [LRS+06, p.16].

1.2.4 Immediate Benefit for the Development Process

The preceding paragraphs render obvious that the main focus of a spatio temporal analysis of soccer scenes in this thesis lies upon the improvement of the grounding situation of the soccer agents. Nevertheless the qualitative description of soccer scenes also provides a surplus value for the agent developers in more than one respect. First the online creation of comprehensive, i.e. intuitive and human-understandable, game progression reports is rendered possible since adequate raw data is provided. An investigation of several approaches reviewed for the state-of-the-art survey in chapter 4 shows that a basis for the development of automated moderation/commentator systems is made available. While for the implementation of this kind of systems additional research in the area of artificial discourse generation would be imperative, comparatively simple written log reports lie within immediate reach. Such reports, available right after the final whistle, are valuable in that they improve the ability of multi-agent developers to understand the decision processes which drove the choice of actions of the soccer agents during the course of the game. This is particularly true since due to the qualitative character of the representation the situation is described in terms which are familiar and intuitive for a human soccer expert. The agent relates to actions and the situation context on the field in a way which is appropriate for the domain of discourse through use of suitable vocabulary.

Long-term experience in the development of the Virtual Werder multi-agent system also suggests that the operational availability of a suitable qualitative representation can facilitate the software development process of additional agent skills and behaviors considerably due to the fact that high-level information is made immediately accessible in the knowledge base. "*Provided the [qualitative] concepts correspond to those in (human) soccer theory, it [...]eases the task of agent specification for the designer.*" [SFL06, p.1]. The level of abstraction which is used throughout the code is raised rendering the whole code-base easier to understand and maintain both for adept developers and novices. The simplification can be achieved mainly through an avoidance of low-level coding fruequently found when working directly on quantitative data such as Cartesian or polar coordinates.

1.3 Thesis Outline

The purpose of this section is to give the reader an idea about the structure of this diploma thesis now that the scope and intentions have been outlined in the last section.

Chapter 2, 'RoboCup as Domain of Discourse', will begin with a survey of RoboCup including a historical digest of the RoboCup challenge and the soccer simulation league in particular. The Virtual Werder team is also introduced.

8 tactical system in soccer with strong emphasis on defense and tactic fouls (cf. http://en.wikipedia.org/wiki/Catenaccio (visited:03/11/2006)

Chapter 3, 'Scenarios and Demands', gives a comprehensive overview of the concepts – facts, events, actions and action sequences – the analysis proposed in this thesis is supposed to recognize in dynamic soccer scenes. The demands on the analysis are summarized precisely in the same chapter.

Chapter 4, 'Spatio-Temporal Analysis of Dynamic Scenes - An Overview' provides a survey of related scientific approaches and, based on the compiled set of demands from the previous chapter, identifies relevant aspects of the respective work which should be considered for the development of an own approach.

Chapter 5, 'A Conceptual Approach for 3D Soccer Simulation' will describe in detail the approach for an incremental real-time spatio-temporal analysis of dynamic scenes which is proposed in this thesis. This chapter constitutes the core contribution of this thesis. It also comprises a presentation of relevant techniques for the qualitative description of time, space and action.

Chapter 6, 'The Realized Analysis Framework', describes the implementation of the incremental recognition approach introduced in chapter 5 within the Virtual Werder multi-agent system.

Chapter 7, 'Evaluation', shows the results of that implementation in a series of tests tries to measure the success and the quality of the approach proposed in chapter 5.

Finally chapter 8, 'Conclusion', summarizes the results of the thesis as a whole. The primary question of interest here is to which degree the developed methodology/system prototype is able to fulfill the demands established in chapter 3. Furthermore, possible paths for future work are highlighted.

2

RoboCup as Domain of Discourse

The second chapter of this thesis is devoted to a survey of the *Robot World Cup Initiative*. The purpose of this survey is to present a comprehensive research context for the work found in subsequent chapters.

In section 2.1 RoboCup is introduced as a grand challenge for artificial intelligence and robotics research. The long-term vision of the RoboCup initiative is outlined as well as the role of the simulation- and hardware leagues in the realization of this vision. A review of the milestones in the development of RoboCup beginning with the first official championships Nagoya (Japan) in 1997 to the most recent championship in Bremen (Germany) in 2006 is presented. The review puts special emphasis on the development of both the *3D Soccer Simulation League* and the *Humanoid League* which complement each other in many respects. It is shown that both leagues seem to have reached a state where indicators for big leaps forward towards *human robot soccer players* exist. These indicators are examined and the relation to the contribution proposed in this thesis is established. The review ends with a description of the currently planned joined road-map for both simulation and humanoid league which reflects the joined progress towards the final goal of RoboCup .

After the global view upon RoboCup as domain of discourse, section 2.2 introduces Virtual Werder, the simulation league team of the University of Bremen in more detail. The purpose of this section is to outline the progress of the team since the introduction of the 3D simulation league in 2004. This is done due to the fact that the work presented in this thesis deeply relies upon the ground work done by the Virtual Werder developer team so far.

2.1 The RoboCup Challenge

The *Robot World Cup Initiative* was proposed in the early and mid 1990's in a series of papers [KAK+95, KTS+97] by a group of Japanese computer scientists, amongst others Hiroaki Kitano and Minoru Asada[1]. These researchers felt that the time had come

1 At the time of writing Minoru Asada is the acting president of the international RoboCup Federation.

to propose a new grand challenge[2] in their research community that would be widely approved as a standard benchmark for novel state-of-the-art scientific approaches to research problems in both *robotics* and *multi-agent systems*[3] and that would retain relevance for several decades to come. RoboCup was proposed at a time where the grand challenge of 'classical' AI, to win in a chess match against the reigning world champion was about to be solved[4]. Even though the research community had learned to handle problems with simple task environments as found in board games[5] it had become clear that once those problems had been cracked by brute force as in the case of chess the applied solutions became "*irrelevant for handling more complex aspects of the real world in which none of the classical assumptions [...] hold*" [Roj06]. Thus, robot soccer was proposed as the new grand challenge due to its complex task environment (cf. section 1.1) which rendered it directly relevant for real world applications while at the same time providing a setting where research was still feasible in terms of technical and financial prerequisites. Soccer was also chosen due to its immense popularity worldwide.

In order to provide the research community with a long-term perspective in [KA00] Kitano stated the vision of the RoboCup challenge as follows: "*Our goal is to develop a team of fully autonomous robots that can compete against the World Soccer Champion team by the year 2050.*"

Since the first RoboCup tournament 1997 in Nagoya [NSH+98], a large number of researchers from all over the world has joined the initiative turning RoboCup into one of the biggest competitions in AI which draws attention not only from professionals in computer science and engineering disciplines but also from an interested public. The latter can be attributed to the innovative research presented in attractive competition in the various leagues but also to the choice of the right game.

When RoboCup was introduced in the mid-1990's it started with both a hardware (*Small-Size-* and *Middle-Size* Robot League) and a software (*2D Soccer Simulation League*) track of competitions. In order to work towards the vision of RoboCup as stated above the global task to construct an artificial, humanoid soccer player capable of competing with the reigning human soccer world champion were decomposed in two sub-tasks that provided the research focus for both the hardware and software track. The first sub-task which was associated with the hardware leagues comprises the engineering task to develop a suitable humanoid body with a proper set of sensors and actuators and the lower cognitive functions of the robot's 'brain' such as reflexes, basic locomotion and cognition. The second sub-task which was associated with software leagues comprises the development of the higher cognitive functions of the robot such as knowledge representation, deliberation, communication and learning. The subsequent paragraphs will investigate the improvements that could be attained since the introduction of RoboCup in the accomplishment of these tasks.

2 [HM05] presents a characterization for the concept of a grand challenge as follows: "*A grand challenge for research in science and engineering pursues a goal that is recognized [...]decades in advance; its achievement is a major milestone in the advance of knowledge and technology celebrated not only by the researchers themselves but by a wider scientific community and the general public. It has no guarantee of success.*" To develop a program which can beat the reigning world champion in chess is an example for a grand challenge that was conducted and completed successfully.

3 Concrete problems in robotics included acting in a real-world environment, handling of incomplete and uncertain knowledge, sensor systems, (real-time) image processing and actuator design while in the multi-agent systems domain cooperation and communication strategies, planing and knowledge representation were in the focus of attention

4 In February 10, 1996 the computer system called *Deep Blue* was the first machine to win a single match against the reigning world champion Garry Kasparov under regular time controls. However Kasparov was able to win this first contest comprised of a total of six games. In 1997 an improved version of *Deep Blue* manages to win the rematch, thereby solving the grand challenge. [Hsu02, New02]

5 which were also referred to as 'toy-problems'

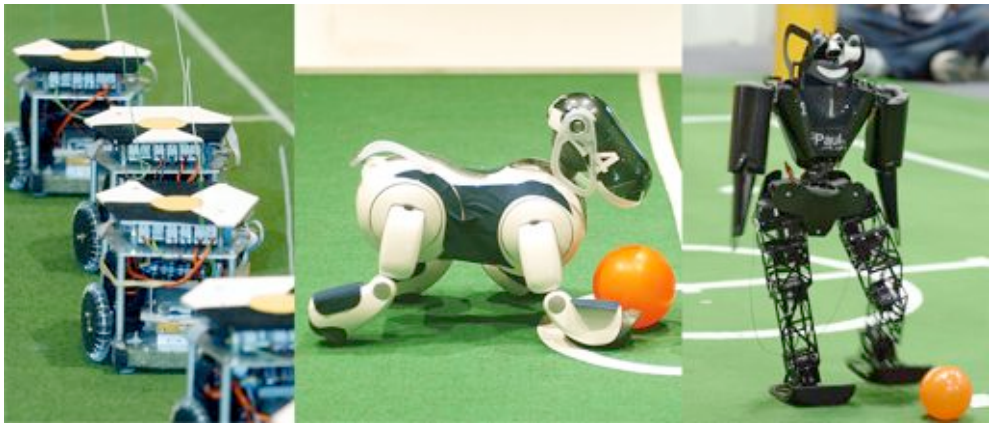


Figure 2.1: The evolutionary progress of the RoboCup Soccer Hardware Leagues (left: *Small-size Robot League*; center: *Four-legged League*; right: *Humanoid League*), official RoboCup image material

2.1.1 Development in the Hardware Leagues

When the first official RoboCup championship took place 1997 in Nagoya (Japan) [NSH+98], the event started with two hardware leagues which differed primarily in the size and level of autonomy of the soccer robots while the setup in both leagues featured a simple wheel-based design and unarticulated bodies. In the *Small-size Robot League* five robots per team each of them no bigger than 18 cm are controlled remotely by a central dedicated server which is connected to a ceiling-mounted camera system that provides bird-view images of the situation on the soccer field. In the *Middle-size Robot League* each of the four robots in a team is fully autonomous during soccer matches. The robots are no bigger than 50cm, carry their own processing unit and use an omni-directional camera system to perceive its immediate environment. Both small-size and middle-size players belong to the first generation of RoboCup robot systems.

In 1999 the *Four-legged Robot League* was first introduced at the RoboCup in Stockholm (Sweden). *AIBO* robot dogs, which were commercially produced by Sony[6], were proposed as a standardized, autonomous, four-legged platform for 4-4 soccer games. Since the robot platform is a given the *Four-legged Robot League* is a competition for the best programming making the best of the available resources. The new league marked a milestone in RoboCup soccer since a decisive first step away from the wheel-based systems (as found in the original *Small-size* and *Middle-size* robot leagues) towards legged locomotion was taken. The *AIBO*s constitute the second generation of RoboCup robot systems.

The RoboCup 2002 in Fukuoka (Japan) marked the next important milestone with the introduction of the *Humanoid League*. In contrast to the *Four-legged Robot League* the *Humanoid League* is concerned with biped locomotion of autonomous humanoid robots. The league started with previews[7] and technical challenges. First regular 2-2 games were conducted at the RoboCup 2005 in Osaka (Japan). In 2006 at the RoboCup in Bremen (Germany) the *Humanoid League* was divided in two sub-leagues, *kids-size* and *teen-size* which differ primarily in the size of the robots. As this league matures so does the design of the robots. The typical design currently used in competition is based on servo motors used as actuators and a simple but robust control method. The main focus of research in

6 Information about the AIBO platform is available on a dedicated web site: http://support.sony-europe.com/aibo/index.asp (visited:06/08/2006)

7 such as one-on-one or penalty-kick scenarios

the *Humanoid League* lies upon dynamic walking and acceptable stability/robustness of the hardware.

These humanoid robots which feature the coarse physical layout of a human soccer player belong to the third generation of RoboCup robot systems. They demonstrate the progress of the hardware leagues towards a suitable artificial human shell and motion controllers (lower cognitive functions of the human brain) to take on the human soccer world champion in 2050 but they also have a significant drawback: Even though those robots' principal layout already resembles the human archetype due to the servos the motion apparatus is way too bold and heavy in order to allow for human-like elegance in treatment of the ball as well as in dynamic walking/running.

The RoboCup in Bremen offered a glimpse at what might become the next generation of humanoid robots by means of the introduction of the new teen-size robot model *Lara*[8]. *Lara* is a prototype designed and built by the University of Darmstadt featuring a revolutionary architecture compared to her fellows which is based on a compact endo-skeleton and artificial muscles.

It can be argued that *Lara* marks a shift in the hardware research focus that is of extraordinary importance with respect to the global vision of RoboCup . For the first time researchers within the RoboCup community are not satisfied to build a robot system that mimics the human archetype in terms of body layout and motion patterns using traditional technical means borrowed from other fields of activity. They are also interested to learn from the advanced design principles provided by nature in order to develop new technological means to build for example a motion apparatus adhering to the same principles and thus inheriting abilities and advantages of the natural equivalent.

2.1.2 Development in the Software Leagues

While the hardware track in RoboCup *Soccer* saw a steady progress towards advanced humanoid robots and a continued adaption and diversification of the respective leagues since Nagoya 1997 [NSH+98], the development of the soccer simulation can be described simply in terms of two eras: the classical *2D Soccer Simulation* (since 1997) and the *3D Soccer Simulation* (since 2004).

During the first seven years of RoboCup tournaments the *2D Soccer Simulation League* was unique in that it was both the only software league in RoboCup *Soccer* and the only league that featured a full set of eleven players per team right from the beginning. The simulation of the simplified soccer environment relied upon the *Soccer Server* originally developed by Itsuki Noda et al. [Nod95, NMHF97]. The *2D Soccer Simulation League* was introduced as a refuge for those developers which, especially in the first years of RoboCup , would not want to wait for the hardware leagues to mature enough to host soccer matches where tactical considerations rather than plain ball control were crucial factors for a team's success. This is why a simulation with a high level of abstraction and some major simplifications such as a missing space dimension could be tolerated especially since a lot of progress could be made by improvements in the multi-agent system (MAS) development of the teams while the changes to the server were only mild and - compared to hardware leagues - conservative. Due to the long development time the Soccer Server became quite robust and with additions such as a coach in 2001 strengthened its dominance as basis for research in multi-agent-systems even beyond the introduction of its successor, the 3D Soccer Server. Most of the work groups whose approaches to spatio-temporal analysis of dynamic scenes are reviewed in chapter 4 are situated in the *2D Soccer Simulation League*.

Due to the simplified simulation model of the *2D Soccer Simulation League*, a three-

8 Information about Lara is available on a dedicated web site:
http://robocup.informatik.tu-darmstadt.de/humanoid/lara/index.html (visited:06/08/2006)

Figure 2.2: The evolutionary progress of the RoboCup Soccer Simulation Leagues, Rendering by Heni Ben Amor

dimensional physical simulation was created with a new 3D soccer simulator [KO04, Kög03] designed to be capable of addressing additional classes of problems beyond those already coped with in 2D. First according to [MBdSG+06, p.3] "*Articulated agents create the problem of coordinating several actions of the same agent among each other, as well as with the global team behavior*". Second, "*Decision making procedures have to deal with a much higher complexity of the decision space, compared to 2D Soccer Simulation League*".

The *3D Soccer Simulation League* was introduced at the 2004 RoboCup in Lisbon (Portugal). At that time the *3D Soccer Server* featured a three-dimensional environment with a soccer pitch compliant with FIFA soccer rules. It also shipped with a largely improved physics model in terms of realism[9] compared to the 2D equivalent. However the initial agent design used in 2004 was simplistic even from the point of view of the 2D simulation. The soccer agents were represented as solid spheres 44cm in diameter with homogeneous omni-drive and omni-kick abilities. No possibility was implemented to control the ball other than to kick it along the vector spanned by the agent's and ball's 2D coordinates. The agents used an omni-camera which allowed the perception of all objects on the soccer pitch with only slight simulated noise in the vision perceptor. Communication was not implemented. At this point 3D simulation was still in its initial maturation process where a stabilization of the 3D soccer server and the development of low level controllers were in main focus while the development of strategic play, communication strategies, or planning was deferred and typically being investigated further in the well-established *2D Soccer Simulation League*. Regardless of the simplicity of the agent design the introduction of the *3D Soccer Simulation League* nevertheless marked a milestone in soccer simulation since it initiated a steady transition from the undoubtedly restricted scenario in terms of the task environment classification that was 2D simulation towards a realistic simulation of the soccer game mentioned in the vision formulated for 2050. As such this transitions bear some resemblance to the transition from the grand challenge 'chess' to 'soccer'.

While the 2005 RoboCup in Osaka (Japan) saw improvements in the low level controllers in the *3D Soccer Simulation League* which resulted in the possibility for fast dribblings with the ball, improved pass play and better communication-less coordination within the teams noticeable development of the league itself was delayed. In 2006 the *3D Soccer*

9 Application and treatment of forces for acceleration and deceleration, details such as air resistance, proper collision detection.

Simulation League managed to adopt a list of features first introduced in the *2D Soccer Simulation League* such as a restricted field of vision, a camera which was mounted on a movable neck which allowed pan-tilt actions, restricted communication of information reminiscent of shouting commands in a real soccer match and a first step towards player differentiation with the introduction of a goal keeper capable of catching a ball within a certain radius. The changes were used in the 2006 RoboCup in Bremen (Germany) and let to technically attractive games over the course of the competition. The focus of research interest had changed towards strategy due to the broad availability of controllers with acceptable or even excellent quality.

However the *3D Soccer Simulation League* still needed to officially reach the next milestone in its development through the adoption of articulated bodies thus addressing the second class of additional problems identified in [MBdSG+06]. Since 2005 the server development team worked on the integration of a biped agent design in the *3D Soccer Server*. A first technical preview of the new design was released in 2005 with only rudimentary functionality. In order to foster the development of controllers for biped walking it was possible to submit a walking controller as qualification material for RoboCup 2006. In 2007 finally, the forceful change was brought about with the abandonment of the spheres in favor of an all-new server based on the *Spark* framework [OR05] and a complete[10] humanoid agent model. As a consequence, the center of attention was shifted from high-level tactical aspects to enabling sufficient locomotion and kick controllers. Successful attempts of writing such controllers and transferring knowledge from the *Humanoid League* were presented at the RoboCup 2007 in Atlanta (USA). As of writing these lines there are further ambitious efforts underway towards far-reaching cooperation and a joint road map for the *Humanoid League* and the biped flavor of the *3D Soccer Simulation League* [MBdSG+06] initiating a league convergence towards 2050.

While the developments towards humanoid simulation illustrate the server-side changes in soccer simulation until late 2006 significant progress could also be made client-side by the RoboCup community towards more sophisticated high-level cognitive functions. Among the trends which can be identified and which bear relevance for the work presented in this thesis is a.) the ongoing transition from purely reactive agent behaviors towards (online) deliberation and planning and b.) the transition from purely quantitative knowledge representations towards hybrid representations which take advantage of both quantitative and qualitative aspects. Active areas of research in the latter direction comprise amongst others the development of appropriate spatial representations [WH05], formalizations of concepts from soccer theory [DLM+05], real-time constrained reasoning upon qualitative scene descriptions [DFL03] and recognition of motion situations, agent actions and tactical moves in soccer matches [MV02, Mie04a]. This research is an indicator for a development analogous to that initiated by *Lara* in the humanoid league. Research seeks to understand the structure of human reasoning and to build software agents whose own reasoning processes are designed to feature increased cognitive adequacy with respect to the human archetype.

2.2 RoboCup Soccer Simulation in Bremen

Virtual Werder 3D was initiated by Ubbo Visser in March 2004 as a successor to Virtual Werder [DHS+01], a self-organized student project at the Intelligent Systems Department of the Center for Computing Technologies of the University of Bremen in March 2004. The goals that were formulated initially comprised the establishment of a competitive soccer team in the recently introduced RoboCup *3D Soccer Simulation League* and the effective transfer of knowledge and experience obtained during the four years of active involvement of Virtual Werder in the *2D Soccer Simulation League*.

10 complete in terms of a body layout with upper and lower extremities

Figure 2.3: Scenes from a match in the RoboCup 3D Soccer Simulation League (Virtual Werder 3D takes on SEU 3D in a friendly match during preparation for RoboCup 2006 in Bremen)

In June 2004 Virtual Werder 3D which at that time comprised five students assembled a first rough framework based on the sample agent that shipped with the 3D Soccer Server and participated in the first real 3D competitions at the RoboCup in Lisbon (Portugal).

In October 2004 Virtual Werder 3D became part of the RoboCup *Transfer* [11] project. RoboCup *Transfer* is dedicated to the transfer of scientific findings and technical expertise into everyday-life and the regional economy. The project is supported by the Bremen Senator for Education and Science. Of particular interest is the integration of results from the accompanying German Research Council (DFG) focus project 1125 titled '*Cooperating Teams of Mobile Robots in dynamic Environments*' [12] which concentrated on the anticipation of intentions of agents located in the near range environment and own actions derivable based on the aforementioned information. Due to the association with RoboCup *Transfer* Virtual Werder acquired additional support and expertise from PHD-student Andreas Lattner.

In 2005 Virtual Werder participated at the RoboCup in Osaka (Japan) with a new code base that had been designed and written completely from scratch. This second embodiment of the Virtual Werder 3D framework, even though it could not yet satisfy the high expectations in direct comparison with the international competitors, helped to generate valuable insights in the fields of MAS-architectures, knowledge representation and controller design. It opened up new vistas and initiated the successful second complete reimplementation of our framework that was due for the following year.

In 2006 Virtual Werder played its home game participating at the RoboCup 2006 in Bremen (Germany). As mentioned in the motivation for this thesis the team's performance was quite noteworthy. This is not only reflected in the end result which was rank eight out of 32 participants qualified for the 3D soccer simulation competitions. Virtual Werder 3D managed to complete the first three round-robin qualification rounds (19 games in total) without a single goal against, winning five and reaching a draw in 14 games. While the team was thus competitive as a whole exceptional performance could be attributed to the defense[13]. The results at the RoboCup in Bremen showed that Virtual Werder 3D had run through a maturation process which had let to a situation where the team's

11 Information about this project is available on a dedicated web site: http://www.tzi.de/index.php?id=150&L=1&tx_projectdisplay_pi1[showUid]=118&cHash=674cf3ea90 (visited 18/08/2006)

12 Information about this project is available on a dedicated web site: http://www.ais.fraunhofer.de/dfg-robocup/ (visited 18/08/2006)

13 In fact besides Virtual Werder 3D only two other teams managed not to receive a goal in the round-robin phase: FC Portugal (rank 1) and ZJU Base (rank 2).

code base was competitive on the national[14] as well as on the international level thereby achieving the initial goal formulated back in 2004.

Also in the wake of the championships in Bremen 2006 the team members began to adopt a new view point upon Virtual Werder 3D and its code base. While until Osaka, the main focus lay upon competitiveness in tournaments, the research aspect was shifted into focus to a larger extent. At the time of writing, three additional diploma theses besides the one at hand are in the making. A PhD thesis in the field of temporal pattern mining in dynamic environments [Lat07] was also to a large extent motivated by the development of the Virtual Werder team.

14 Virtual Werder 3D was the only German as well as one of the two European teams to reach the RoboCup quarter finals.

3

Scenarios and Demands

The third chapter of this thesis is devoted to a specification of the demands on the concrete approach for a spatio-temporal analysis of dynamic scenes which is used to obtain the desired qualitative representation of a soccer game called for in the introduction.

First, section 3.1 introduces an exemplary set of possible scenarios which are likely to be encountered frequently in similar fashion in regular games of the RoboCup *3D Soccer Simulation League*. These examples are provided in order to give the reader a coarse idea of both actions including parameter values and game situation aspects which bear relevance to the soccer game and shall therefore be recognized adequately by the proposed analysis approach. Following the scenario presentation section 3.2 a comprehensive list of general demands is compiled which constitutes mandatory constraints for the concrete implementation of the desired analysis approach such as upper bounds for the acceptable runtime duration for the processing required to advance the analysis a single analysis step. In section 3.3 the prerequisites which need to be fulfilled in order to apply a spatio-temporal analysis are highlighted. The desired concrete scope of the analysis with respected to recognizable concepts is finally described in section 3.4.

3.1 Game Scenarios

In this section three short scenarios from the RoboCup *3D Soccer Simulation League* are presented and aspects of the respective dynamic scene which should be recognized by the proposed approach to spatio-temporal analysis are *highlighted*.

3.1.1 Attempted High Cross Pass, Leading to Fight for Ball

At the starting point of this scenario outlined in figure 3.1 the game is in *play mode* 'playon' which corresponds to 'normal course of the game' as opposed to a set situation such as a free kick. The ball is located on the left-wing side (from team A's point of view) and close to the half-way line. While *rolling slowly* into team B's half of the soccer field the ball is currently *free*. A player from team A *is approaching* the ball. This player is *closer to the ball than the players of team B*. Eventually *the player and the ball meet* and the agent *receives* the ball thus *gaining exclusive ball control*. The player circles around

Figure 3.1: Schematic Overview for the cross pass scenario.

the ball in search for a suitable pass target while *remaining in ball control*. Since the left wing is covered dense by the players of team B while the right wing is only sparsely occupied the player decides to initiate a change of wings. In order to perform a *cross pass* to a fellow player waiting beyond the center circle on the right wing he *initiates a ball transfer* by *kicking the ball high along the half-way line with considerable force*. The ball *is leaving ground* and over-flies two players from team B that are located not far from the center spot. Eventually the flight trajectory nears an end, the ball *touches the ground again* while still rolling with noteworthy speed. The fellow player from team A *approaches the ball*. He is not alone since an adversary player is also already *very close to the ball* and *approaching it*. Both agents manage to *meet* the ball at the same time *immediately engaging in a fight for exclusive ball control*. The *pass attempt has thus been baffled* by the quick reaction of team B. While before the pass team A was clearly *in ball control* the situation know first has to be resolved either by a *retreat from the ball* from either of the combatants engaged in the tackle or by another *kick of the ball*. So long the *tackle* continues.

3.1.2 Tackle Resolution followed by Successful Dribbling

The scenario described above continues as outlined in figure 3.2. The *tackle* goes on for a while before finally the agent from team B can *initiate a kick which rebounds from the other combatant* yet *rolls on ground level in a direction advantageous for the player from team B*. While the player from team A is confused, the kick initiator *approaches the ball* which *has been kicked with only little power* in order to resolve the *tackle*. The player *meets the ball* and performs a *ball reception* thus *gaining exclusive ball control*. From a global point of view *the ball control thus is conquered by team B*. Since the agent is an attacker and the foremost among his fellow players there's no suitable pass partner and the agent *initiates a self assist* where *the ball is played steep with a lob*. The agent immediately follows the ball *receiving the ball* again. During the whole self assist the ball *remains close to the initializing player*. The player repeats the self assist action. The

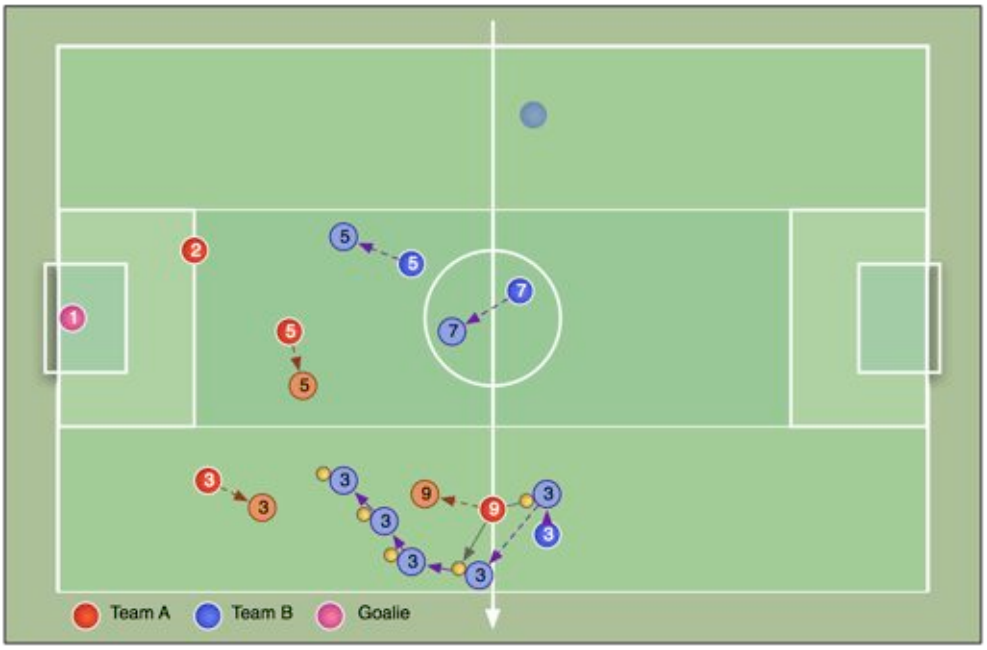


Figure 3.2: Schematic Overview for the tackle resolution and dribble scenario.

sequences of those self assist *constitutes a dribbling* towards the adversary goal by this player which is not interrupted by adversary players from team A.

3.1.3 Scoring Attempt Baffled by Adversary Goal Keeper

The previous scenario described a dribbling of a forward player from team towards the adversary goal over the left wing. While still performing the dribbling action the player eventually reaches a position from which a score attempt is feasible (cf. figure 3.3). Since defenders from the enemy are *rapidly closing in on the player* he decided to *initiate a score attempt* while this is still possible due to the fact that he *exerts exclusive ball control*. Since the player is still relatively away from the goal for a score attempt he *kicks the ball with full power* and *medium height* in order to avoid a collision of the ball with the forthcoming adversary defenders. The ball flies over the adversary defense towards the goal. The adversary goal keeper reacts and performs a reposition action. When the ball is *very close* the keeper successfully catches the ball. Thus *the score attempt is baffled by the adversary goal keeper*.

3.1.4 Discussion

The scenarios above have been presented intentionally in the style of a live radio broadcast as the latter is a good example for a qualitative real-world simultaneous description of the dynamic scene unfolding during a soccer match which actually works quite well for the most part of the listening audience. Due to the highlighting of those pieces of information which should be recognized by the analysis for whom the demands to be fulfilled are unfolded throughout the remainder of this chapter it becomes apparent that an important subset of a complete scene description is compiled. At the same time the scenario descriptions also give hints as to which concepts are subject of future work on the way towards a complete scene description.

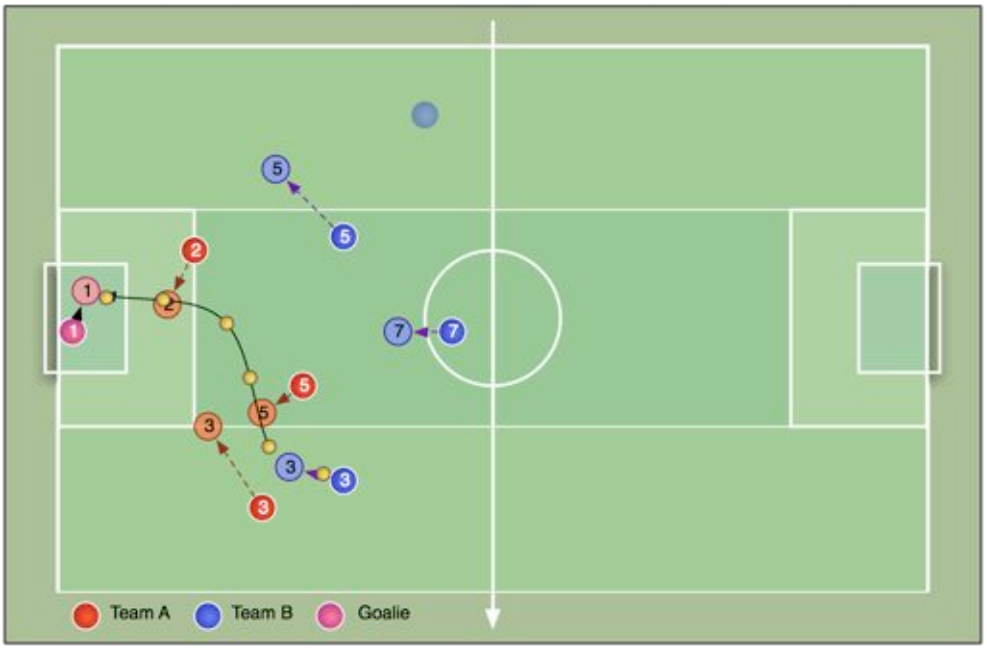


Figure 3.3: Schematic Overview for the score attempt scenario.

3.2 Demands on the Spatio-Temporal Analysis

An important goal with mandatory character for the analysis approach developed in this thesis is unconfined applicability either by the Virtual Werder 3D soccer agents or by the Virtual Werder 3D coach [BKN+04, KLP+05, LPR+06] during the course of regular friendly matches against befriended soccer teams provided by other universities.

Several important issues arise from the desire to perform an online analysis which constitute important demands outlined below.

3.2.1 Adequate Performance for Online Application

To start with, the analysis needs to operate as intended while the duration of a single analysis step is bounded by well-defined temporal constraints that arise from the mode of operation of the RoboCup 3D Soccer Server.

Under normal circumstances[1] soccer agents in the RoboCup *3D Soccer Simulation League* are provided with sensation messages which contain an egocentric vision as well as setup/game state information every 20 simulation steps where each simulation step accounts for 10 milliseconds of simulated time. In an *average simulation environment* as it can be found both in RoboCup competitions and in our team's laboratory the processing of a server message received by the agent[2] can be accomplished in about two simulation steps. This assertion applies to the standard Virtual Werder 3D soccer agent as of November 2006 which is essentially an accelerated version of the agent used during the RoboCup 2006 championships and still relies on a simple, classic knowledge base without an explicit qualitative layer [LRS+06, pp.13].

1 standard configuration of the server

2 the processing task entails the parsing of the server message string, the integration of the resulting worldstate perception and the last worldstate observation into a new current observation and the precalculation of the progression of the simulation in the immediate future (three simulation cycles where each cycle has a duration of 20 simulation steps)

When the spatio-temporal analysis of the soccer scene is performed in order to update the qualitative representation following the completion of the standard knowledge base update additional simulation steps will be required.

The theoretical effective upper bound for the analysis computation time is specified by the duration of the server simulation cycle and the fraction thereof already consumed before analysis start. However in order ensure the competitiveness of the agent the analysis actually needs to be performed in only a fraction of the remaining maximum number of simulation steps thereby making sure enough time is left to perform the regular reasoning and acting duties. Typically, the above-mentioned Virtual Werder 3D agent can perform a complete reason-act operation within a single simulation step.

Due to the time/communication model of the 3D Soccer Server it is possible for each agent to reason and act more than once within a single simulation cycle if there is enough time left in the cycle to do so. This modus operandi is reasonable as it allows the agent to obtain additional timing information from the server between consecutive worldstate messages and thus to schedule and execute actions within a simulation cycle with high accuracy. Especially in situations where an agent handles the ball, it can be advantageous first to start acting as soon as possible after a complete knowledge base update and to act multiple times in succession in order to achieve the desired overall effect for the actions performed within the current simulation cycle. If a bigger fraction of the simulation cycle remains for reason-act this means the agent can either reason more deeply or more often.

Due to this fact the complete update of the agent knowledge base including the qualitative layer should be completed with at least half of the simulation cycle left for normal operation within the soccer game. Thus the fact that an analysis of the dynamic scene is applied by the agent is likely not to interfere with the character and quality of the agent's performance due to its time complexity. It also leaves a certain temporal buffer in order to handle situations where either the current state of the game demands more intensive computation as compared to the average processing or the soccer simulation is performed on less efficient simulation environments.

It is a major goal of this work to show that spatio-temporal analysis of dynamic scenes is feasible under these circumstances.

While the evaluation of the time complexity of the spatio-temporal analysis of soccer scenes in the context of the RoboCup 3D Soccer Simulation can be performed based on the simulation time that is consumed in each step due to the fixed simulation environment as specified for the thesis evaluation in section 7.1 on page 123, with regard to a desired future transfer of the developed analysis into further real-world application domains, it is also compulsory to obtain additional data about the time complexity measured in real system time (milliseconds) per analysis pass.

3.2.2 Feasibility in Realistic Game Situations

The feasibility of the approach to spatio-temporal analysis of soccer scenes developed in this thesis should be proved in realistic game situations like those encountered during regular RoboCup championships rather than in well-prepared lab scenarios. While especially crafted, simplified test scenarios are imperative in order to evaluate parts of the analysis system during the development stages the final test scenario should feature two complete teams each consisting of 11 players. The matches are to proceed using the normal set of soccer rules and server parameterization which comes as a default with the 0.5.x series of the RoboCup 3D Soccer Server. The rationale for this decision is twofold. Proof can be acquired that the analysis may be applied by a whole team of eleven soccer agents while playing a normal game of simulated soccer without significant performance disadvantages. Thus once the thesis is completed the results thereof are appropriate for a technology transfer into the main development line of the Virtual Werder 3D multi-agent-system

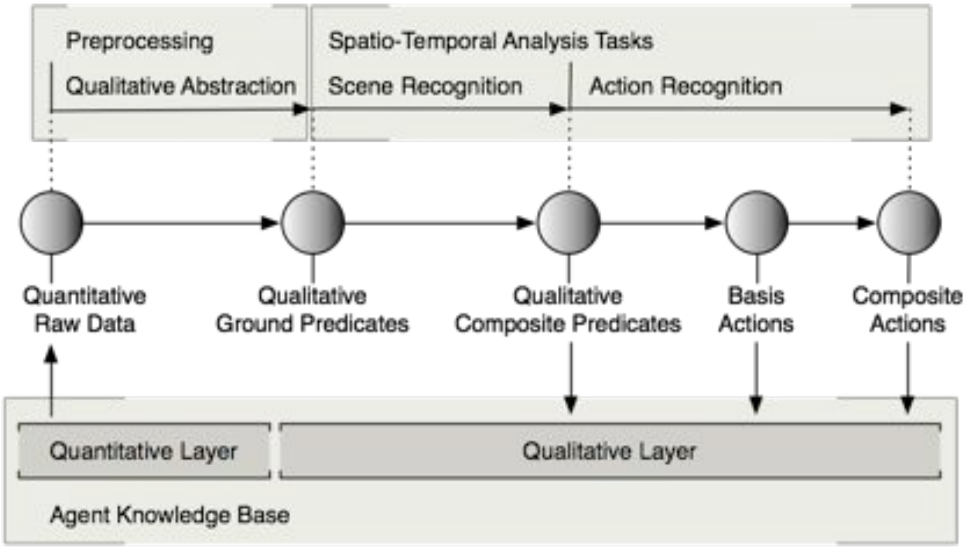


Figure 3.4: Schematic overview of the distinct task areas touched by spatio-temporal analysis and reference to generated data.

(MAS) as described in [LRS+06].

3.2.3 Handling of Incomplete and Noisy Sensor Input

In the introduction to this thesis (cf. section 1.1 on page 1) it was already mentioned that in the 0.5.x series of the RoboCup 3D Soccer Server the soccer agents are equipped with a restricted field of vision and sensor readings are normally distributed to a certain amount around their true value by the server thus simulating a certain inaccuracy in the positional sensor values obtained with the server messages. Both measures have been introduced to the server in order to obtain a simulated environment which features increased realism by confronting the agents with problems which are drawn from real-life counterparts.

In the desire to supply the Virtual Werder 3D agents with an acceptable hybrid knowledge base the existence of both incompleteness and noisiness of sensor data needs to be acknowledged and explicitly addressed in the development of the proposed approach. The spatio-temporal analysis of soccer scenes used to maintain the qualitative knowledge base part needs to work in an acceptable fashion both in a simplified scenario where vision is complete and unaltered and a real-life scenario where the standard conditions to be encountered during RoboCup championships. Thus the analysis needs to be robust and provide stable results. The assumption which can hopefully be proven in the thesis evaluation is that the decrease of analysis quality without perfect perception is mild where *quality* needs to be evaluated along at least two dimensions: overall recognition of concepts and suitable temporal classification of those recognized concepts.

3.3 Prerequisites for the Analysis Application

This section discusses the prerequisites which need to be fulfilled in order to apply the concrete spatio-temporal analysis. The prerequisites refer to both the required set of raw input data (cf. section 3.3.1) and the preprocessing which is to be performed in order to obtain a qualitative input suitable for further use (cf. section 3.4, upper-left part). The decision for an early transformation from quantitative raw data to qualitative

ground predicates is motivated in the light of the aforementioned general demands (cf. section 3.3.2).

3.3.1 Availability of Suitable Input Data

General Information: Simulation and Game State

The following set of information needs to be provided with regard to the course of the simulation and the respective game of soccer simulated therein:

1. The discrete *simulation time* which is measured in the number of simulation steps since simulation start
2. The *game time* measured in seconds which indicates the simulated time already played in the respective soccer match
3. The *play mode* information which indicates the current state of the soccer game[3]

Information about Objects on the Soccer Field

For the spatio-temporal analysis information about the movable objects acting in the simulation is required while information about stationary objects such as flags or goal posts are neglected for the time being.

It is expected that for each movable object the following pieces of information can be provided:

1. the *object type* which can be either 'Ball' or 'Player'
2. the *allocentric, absolute position* of the respective object on the field which is represented in three-dimensional Cartesian coordinates. The position is measured in meters.
3. the *velocity* of the respective object which is represented in three-dimensional Cartesian coordinates where the length of this vector signifies the object's position delta since the last simulation cycle. The velocity is thus measured in meters/sim-cycle.

For players the information given above is not sufficient and thus additional data needs to be provided:

1. Information about the agent's *team membership*. This is provided both by a play-half indication which states whether the player's team is playing on the left or right half of the soccer pitch[4] and the name of the team.
2. The *tricot number* which unambiguously identifies the player within his own team. The tricot number bears additional information as in the current setup of the RoboCup 3D Soccer Server, the player with the tricot number one is automatically the respective team's goal keeper[5]

3 At this point it should be mentioned that the fact that play mode information can be directly accessed is convenient from the analysis point of view. If the play mode was not accessible directly due to its significance for the analysis itself an inference of the play mode based on the location of the ball, last ball contact etc. would have been necessary [Mie04a, pp.120].

4 In the RoboCup *3D Soccer Simulation League the teams do not switch sides for the second half of the game.*

5 The reason for this fact is that a decent differentiation of player types within a team where each player type is equipped with a certain set of capabilities such as the ability to catch the ball within a certain area (for the goal keeper) is subject to future development of the RoboCup 3D Soccer Server.

In section 3.2.3, it is specified as a demand that the spatio-temporal analysis is applicable both in a test scenario with complete and accurate sensor data and with a realistic scenario where the sensor data is incomplete due to restricted agent vision and where noise is inserted by the soccer server in order to decrease the sensor quality.

In order to compare the results of the analysis approach in both scenarios it should be possible to have access to two sources of data that both describe the same soccer game. The data which is provided from the first source should be complete which means that accurate information about all movable objects on the soccer pitch is available at all times. In order to obtain data of such quality an additional coach agent needs to be introduced into the simulation which is equipped with perfect vision sensors. The data which is provided from the second source may be incomplete which means that not all movable objects are necessarily visible all the time and less accurate sensor values are provided. However for an object which is actually within vision range, the set of information should be complete if not necessarily accurate. This kind of data can be obtained directly from the Virtual Werder 3D agents which actively participate in simulated soccer games.

3.3.2 Qualitative Abstraction of Raw Input Data

Qualitative Ground Predicates

During the first step of the spatio-temporal analysis a suitable set of qualitative ground predicates should be generated from the quantitative raw data.

The concrete set of required predicates is given below. While the first five predicate types describe the motion situation of single movable objects on the soccer field, the following predicate types describe aspects of the motion situation of two movable objects respectively which are considered in relation to each other.

- the *velocity* of a movable object
- the *acceleration behavior* of a movable object
- the *motion direction* of a movable object
- for the ball the *vertical_position* which indicates whether the ball is airborne or located on the ground
- for the ball the *vertical_trend* which indicate whether the ball is rising in the air (due to a kick or bounce effect), is stable with respect to its height above ground or is falling towards the ground.
- the *distance* between two movable objects
- the *distance trend* for two movable objects

Rationale for Early Qualitative Abstraction

The utilization of qualitative ground predicates in general as starting point for further analysis steps as opposed to a direct use of the original quantitative raw data can be motivated in different ways.

A sequence of quantitative raw observations which contains for the most part position and velocity information for each of the 23 movable objects normally found on the soccer pitch in the RoboCup *3D Soccer Simulation League* constitutes a massive collection of data which is not feasible to process under real-time constraints as formulated in section 3.2.1 in a spatio-temporal analysis approach. This statement can not only be justified

with a reference to the total amount of raw data for each new observation but also with the fact that even though a comprehensive set of information may be contained within the raw data the information is for the most part stored in an implicit way. Thus the need for a complex data retrieval contributes to prohibitive computation costs.

Therefore the transition to a qualitative representation should be performed as early as possible in a preprocessing step before the actual analysis. In doing so a significant decrease of data which needs to be processed further can be achieved while the representation of the dynamic scene based on qualitative predicates still bears a suitable expressiveness for the analysis system.

The concise character of the representation is to some extent due to the capability of qualitative predicates to represent dynamic concepts such as aspects of the motion situation of single or groups of movable objects in a scene that possess a natural dilation in time. For example it is easy to express the fact that an adversary player is drawing closer to the agent of the own team currently in ball control for 5 simulation cycles so far.

Due to its qualitative character the representation introduces a suitable symbolic abstraction from concrete quantitative values. An appropriate example is the symbolic representation of distances for two movable objects in the domain of simulated soccer. While in a quantitative representation the numerical value for the distance usually changes continuously the qualitative representation can deliver the information that the objects are close and approaching each other eventually leading to a situation where both objects meet. While the relevant information is preserved the symbolic values demonstrate stability with respect to their temporal validity.

Given a suitable, robust method for the generation of qualitative ground predicates the symbolic values are also less susceptible to noise. This is an noteworthy argument especially with respect to the demand specified in section 3.2.2 to develop an analysis approach which is feasible in real-life scenarios where noisification of positions and the dependent velocities pose serious problems which need to be addressed. If this problem is addressed as early as during the generation of qualitative ground predicates the actual spatio-temporal analysis of dynamic scenes which is applied afterwards can safely ignore the presence of noise contained in the quantitative raw data.

Implementation of a Focus Strategy

When a human is actively engaged in a game of real world soccer it can be observed that he/she is not interested in all regions on the soccer field for each moment of the game with a uniformly distributed intensity. During the whole course of the game he concentrates especially on certain regions on the field such as the dynamic region around the ball or his immediate neighborhood on the soccer pitch. He may also decide to keep an eye on those players which for some reason bear a special interest. Such a special interest may be given as a consequence of the observing player's position and his role on the field (consider 1on1-covering as one such example). It may also arise due to outstanding capabilities of certain adversary players for whom it would be fatal to leave them unobserved even for short amounts of time. To conclude humans apply focus strategies in order to concentrate on important aspects of the soccer game with respect to their own performance. The analysis system proposed in this thesis should adopt this behavior in order to reduce the computation load for each new analysis step. Qualitative basis information should be generated in a preprocessing step to act as immediate analysis input data only for those objects which are currently focused and thus considered relevant.

3.4 Scope of the Spatio-Temporal Analysis

This section outlines the desired scope of the spatio-temporal analysis. First the set of *composite qualitative predicates* is introduced. Afterwards the remainder of the section continues with a presentation of both the *basis actions* and *composite actions* to be recognized on the basis of the composite qualitative predicates (cf. figure 3.4, upper-right part).

3.4.1 Recognition of Composite Qualitative Predicates

Based on the qualitative ground predicates which have been illustrated in section 3.3.2, p. 22 a rich qualitative description of the situation context within a running soccer game should be provided by means of *composite qualitative predicates*. In order to obtain predicates of this kind the low-level description of the respective dynamic soccer scene at hand needs to be exploited and conclusions need to be drawn based on the temporal relation of sets of *qualitative ground predicates* where the character of the temporal relation is not constrained per se. Examples for possible temporal relations are amongst other simply an isochronic appearance of certain *qualitative ground predicates* or, generalized, the appearance thereof in a certain chronology which can entail concurrency, sequential occurrence and interlacing.

The *composite qualitative predicates* which should be subject of the analysis encompass situation descriptions which reflect durative aspects of the game context such as the state of the game with respect to ball control. They also encompass game-relevant events such as the ball reception of a certain player or a retreat from the ball which can also imply a retreat from a tackle situation. The latter two groups of composite qualitative predicates are within primary focus and are thus subsequently discussed in more detail.

Events

As events which can be identified by the analysis approach possess a dual functionality first to provide relevant information in their own right and second to enable the recognition of actions which is described in the following section 3.4.2 the concrete set of relevant events which actually need to be recognized is geared to the requirements for the latter mentioned further analysis.

Ball Reception by a single player or a group of players (cf. section 3.4.2, p.25)

Kick of the Ball again by a single player or a group of players (cf. section 3.4.2, p.25)

Tackle Engagement where a single player starts a struggle for ball control in engaging a player that previously controlled the ball exclusively or engages in an ongoing struggle for ball control which already involves multiple players

Ball Retreat where an agent gives up its ability to control the ball, thus either retreating voluntarily from a struggle for ball control (without the ball being kicked) or leaving the ball alone entirely, potentially in order to transfer the responsibility to build up the game to a teammate

3.4.2 Recognition of Actions

Recognition of Basis Actions

With regard to the scope of the action recognition *ball-centered basis actions* constitute the primary focus for development and implementation in the context of this thesis. Common to all *ball-centered basis actions* is that they are initialized by either a single or a group of players on the soccer field by means of a ball kick event. Furthermore these actions are characterized and referred to as *basis actions* due to the fact that for each respective action there is only a single entity which plays the role of the active action initiator. *Basis actions* cannot be decomposed into simpler sub-actions which is the case for the *action sequences* described in section 3.4.2. The incremental recognition process for a concrete *ball-centered basis action* should eventually (in the context of this thesis towards the end of the action) lead to the one of the following conclusions

Pass The ball has been passed successfully or has been lost due to a failed pass.

Clear The ball has been cleared either into a free area on the soccer field or outside the soccer field where the resulting play mode change allows to determine the exact characteristic of the respective clearing effort such as clearing to the side (adversary kick-in) or to the back (adversary corner kick).

Score The ball has been shot in order to score a goal. The score attempt has been successful or not where in case of failure an appropriate differentiation should clarify if the shot missed the goal, the shot has been caught by the adversary goal keeper or the shot has been baffled by an adversary player.

Self Assist A player has successfully performed a self assist.

FightDissolve The ball action has brought an end to a situation where players from both teams were engaged in a fight for the ball. The fight could either be dissolved successfully since the ball has been received by only one team or cleared to a standard situation. Another possibility is that the fight for the ball continues immediately at ball reception with a new set of engaged combatants.

Handling of Ambiguities

In the RoboCup *3D Soccer Simulation League* situations frequently occur where a group of players, either due to a fight for the ball or disaccord with respect to the responsibility for the ball, commonly exerts control over the ball. The analysis approach should handle consistently situations where a ball-centered action is obviously initialized originating from such an ambiguous situation where it is impossible to determine with certainty which amongst the possible players initiated the ball transfer. The following scenarios are possible:

1. The ball has been kicked by a particular player in such a way that it immediatly bounces off the body of a second player before leaving the sphere of influence of the player group.
2. The ball has been kicked by more than a single player. All players have kicked virtually at the same time.
3. The ball has been kicked by only a single player amongst those that actually had the opportunity to initiate a kick. The other player were not involved in the particular action.

The problem which has been discussed above for kick initiation applies in a similar fashion also to the reception of the ball. A consistent approach to the handling of such ambiguous ball situations should avoid a solution which on first thought seems to suggest itself and is pursued in other approaches to action recognition such as [Mül02] or [Mie04a] where based on the conclusion that more than a single player might have initiated an action it is assumed by default that all players have actually deployed their opportunity to do so. Such an approach leads to an attempt to recognize several *ball-centered basic actions* where only a single action has actually been initialized. The solution which is favored instead for the analysis proposed in this thesis is to allow actions to be initialized by single agents as well as groups of agents. Those groups may be both heterogeneous or homogeneous.

Recognition of Action Characteristics

As already mentioned in section 1.2 on page 3, an important demand is that the recognition process for a concrete *basis action* that has just been triggered already yields a useful, yet due to the early stage of action execution possibly incomplete characterization. The part of the characterization which should be available right from the start of the recognition process for the respective action entails a differentiation in which way the ball has been kicked.

A suitable description of the individual kick style that should be provided by the analysis comprises the following aspects:

1. The *direction* of the kick. In a soccer game it makes sense to differentiate between *backward kicks*, *cross kicks*, *diagonal kicks* and *steep kicks*.
2. The kick style with respect to the *flight trajectory*. A ball may be kicked *on ground level*, *medium* or *steep*.
3. The kick may be performed with *low*, *medium* or *full strength*.

With a combination of the above mentioned characteristics it is possible to describe special kicks such as lobs[6]. Eventually a pass can be characterized as for example a *long lateral pass* which is conducted on ground level.

Recognition of Action Sequences

The analysis should be able to recognize not only *basis actions* but also *composite actions* which can be thought of as *action sequences*. The recognition of those sequences should be based on the recognition of the *basis actions* described above. A mandatory demand in this context is that while the concepts to be recognized become more complex the actual recognition thereof remains tractable by exploiting what has already been recognized so far. This means that the recognition of an *action sequence* directly relies upon the recognition of the constituent *basis actions*.

Concrete *action sequences* to be treated comprise:

Extended Dribbling of the ball by a certain player. The complete dribbling consists of single atomic dribblings performed in sequence by that player.

OneTwo Pass also called *give'n'go pass* as an example for the handling of a so-called dyad or multi-player combination where more than one entity[7] actively kicks the ball. A

6 in German: 'Heber' or 'Lupfer'
7 in this special case a single player

onetwo pass can be performed in straight-forward where the second pass by the receiver of the first back to the initiator of the first pass is performed without the second player performing any further basis action. A onetwo pass may also feature a short dribbling by the first receiver before the ball is passed back.

Extendibility towards Play without Ball

Non ball-centered basic actions bear a noteworthy relevance in the RoboCup *3D Soccer Simulation League*. This class of basic actions which entails for example man-for-man marking or participation in the setup of an off-side-trap have not been considered in section 3.4.2. However the analysis should be capable in principle to support this class of actions even though they are not in the primary focus. An introduction of *non ball-centered basic actions* should not require principal changes to the way the analysis works.

4

Spatio-Temporal Analysis of Dynamic Scenes

The fourth chapter of this thesis is devoted to a survey of related research in spatio-temporal analysis of dynamic scenes and qualitative knowledge representation. The survey concentrates on approaches which have been applied in sport application domains which are characterized by swift, dynamic changes to the respective situations and a large number of relevant scene actors. The domain of *robotic soccer* is given special treatment with a survey of research recently compiled by the RoboCup research community. Also, several research efforts located in the area of sports analysis are reviewed covering both soccer and American Football as featured team sports.

The chapter is concluded with a discussion of the approaches outlined in the survey with regard to the demands which have been defined previously in chapter 3. This is done in order to identify both ideas and means which can be either directly applied or used as a starting point for the development of the approach proposed in this thesis.

4.1 Review of Related Research

4.1.1 Spatio-Temporal Analysis of Dynamic Scenes

At the Center for Computing Technologies (TZI) at the University of Bremen, Andrea Miene developed a comprehensive approach to *Spatio-Temporal Analysis of Dynamic Scenes* in the context of her doctoral thesis[1] [Mie04a, Mie04b] which was also picked up as a central issue in several associated papers such as [MVH04, MV02]. In these publications Miene presents a general analysis solution which is bound neither by applicability in only a limited set of problem domains nor by certain scenarios for the utilization of analysis results. The analysis rather seeks to provide a rich multi-tier qualitative representation of dynamic scenes and explores ways in which such a representation can be constructed starting from low-level sensor values. For the concrete theoretical development and the prototypical implementation of her analysis Miene chose the RoboCup *2D Soccer Simulation League* as demanding realistic application domain, stating however that domain alternatives such as cell tracking or driver assistance systems are also feasible [Mie04a, p.2]. In order to

1 Author's Note: The doctoral thesis was written in German. The title thereof has been translated by the author for the sake of better understanding. The original title is: *Räumlich-zeitliche Analyse von dynamischen Szenen* (cf. [Mie04a])

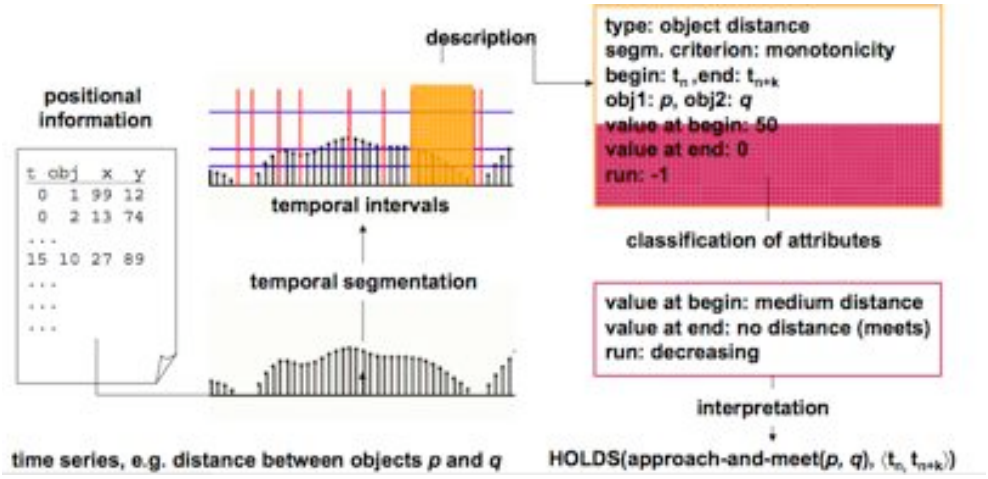


Figure 4.1: Schematic overview of Miene's processing pipeline from quantitative raw data until qualitative ground predicates. [MLVH04, p.3]

demonstrate the functionality of the implementation, simple human-readable log reports were generated where the recognized aspects of the dynamic scenes were used to maintain a simple qualitative description of the course of simulated soccer games. It should be noted however that the logs are primarily intended as a human-readable feedback from the analysis system rather than as fully-fledged automated commentaries in the tradition of ROCCO/SOCCER (cf. section 4.1.3 on page 37) or MIKE[2] [TNF+98].

An important contribution of Miene's approach is the emphasis upon the investigation and interpretation of both the spatio-temporal movement situation for single movable objects in a scene and, even more important, the movement situation which results from the spatio-temporal relations between arbitrary groups of movable objects.

Miene uses monitor log-files from the RoboCup 2D Soccer Server [NMHF97] as raw input for her analysis. With regard to a rating of the character of the input data the analysis is prepared to accept, it is important to understand that these monitor log-files provide the exact Cartesian position of all movable objects within the simulation environment for each discrete moment in a soccer game [CNO+02]. Thus, the analysis developed by Miene is designed to build upon complete information which is of high quality with regard to accuracy of the position values within the bounds of the simulation environment. An adoption of the analysis to incomplete input data remains subject to future work.

The first step in the analysis is a qualitative abstraction of the raw input into a set of qualitative ground predicates describing continuities which occur either in the motion of single objects, such as velocity and heading, or in the spatial relation between pairs of objects, such as distance and heading. Due to their respective temporal dilation these predicates feature a validity interval and they are referred to by Miene as *Object Motion Intervals* (OMI) and *Spatial Relation Intervals* (SRI) [MV02, p.3]. Both OMIs and SRIs are obtained incrementally in a multi-tier process, as raw data for successive moments in time is ingested and handled by the analysis algorithm (cf. figure 4.1).

In a first step the raw data which is interpreted as a set of *time series* each representing the quantitative value run of a certain movement aspect over (discrete) time is pre-processed, where these movement aspects comprise amongst others object velocities,

2 Abbrev.: Multi-agent Interactions Knowledgeably Explained

velocity bearings, inter-object distances. The *time series* are subjected to a series of transformations such as *gradient smoothing* and removal of single runaway values in order to compensate errors in the sensor readings. Using these means the time series are put in a suitable shape such that they can be used in the next processing step where both a monotonicity and a threshold-based segmentation of the time series data into intervals is performed.

Miene motivates the use of both segmentation methods with the notion that a monotonicity based segmentation is well suited in order to capture value trends such as object acceleration/deceleration or approach/retreat, while a threshold-based segmentation captures value states such as the distance between objects. The segmentation process is executed automatically, driven by a numerical parameterization which is tuned to the characteristics of each respective time series. Up to this point in the processing pipeline no domain-specific semantics is attributed to the generated intervals. In applying this course of action Miene establishes a clear distinction between a.) the segmentation- and b.) the interval classification phase.

The classification is the last step which needs to be executed in order to obtain OMIs and SRIs. The classification allows for an interpretation of the numeric interval descriptors such as the average value of an threshold-based segmented interval as member of a symbolic and thus qualitative class. The domain-specific specification of the respective class layouts is performed manually based on established formalisms for qualitative descriptions for spatial relations [CH01], distances [HCDF95] and directions [Her94].

Starting from the set of qualitative ground predicates, Miene's analysis approach strives to derive more complex correlations which require the examination of multiple constituent predicates. In her analysis approach Miene explicitly addresses the recognition of composite situation properties, events and actions that are performed by the actors in a scene, following the conceptual distinctions introduced by Allen in [All84].

The basic concept of Miene's approach to recognize complex entities relies upon two assumptions. First, complex concepts can be described in terms of simpler concepts with a certain temporal entanglement thus constituting a characteristic temporal pattern (cf. pattern 4.1 and pattern 4.2 for example patterns). The pattern is a generalized description of the respective concept which corresponds to the required common denominator of actual instances of the concept occurring in dynamic scenes. It can be compiled manually by a domain expert due to the fact that the standardized, pattern-based description of concepts such as domain-specific actions is not artificially crafted for the analysis' sake but reflects the way an expert thinks about actions. The second assumption is that the declarative formalism which describes the concept patterns is flexible enough to subsume the by far largest part of concrete concept instances that occur in typical dynamic scenes. Thus, once a pattern such as *pass* has been assembled with necessary care it should be suitable for all concrete *pass flavors* regardless of parameters such as the respective situation, the acting team or the mode of play. So while the first assumption states that complex concepts can – in principle – be described in terms of their internal composition, the second assumption states that concrete concept blueprints can indeed be constructed, that actually apply without restrictions in actual dynamic scenes.

Miene does not restrict the type of constituents which are used in the specification of a composite concept pattern. For example, in pattern 4.2 both actions ($OCCURRING(\ldots)$) and situation properties ($HOLDS(\ldots)$) are used to specify the one-two pass pattern. This fact bears special importance since it allows a hierarchical organization of concept patterns as opposed to a flat organization. For the description of higher-order concept patterns other concept patterns of lower order can be immediately reused. Thus, while the complexity of the entities described by the concept patterns grows, the complexity of the pattern description is kept low, as, in a natural way, higher-order concepts are described on a suitably high level of abstraction. This can be exemplified with Miene's definition of

Motion Pattern 4.1 Concept Pattern for a successful *pass* from player p to q. [Mie04a, p.103], translated from German

$$
\begin{aligned}
&OCCURRING(Pass(p,q),i) \Leftrightarrow \exists h,j: \\
&\quad OCCUR(Kick(p),h) \\
&\wedge \quad HOLDS(BallControl(p),i) \\
&\wedge \quad OCCUR(Reception(q),j) \\
&\wedge \quad InOrEquals(h,i) \wedge FinishedByOrEquals(i,j) \\
&\wedge \quad Team(p) = Team(q)
\end{aligned}
$$

Motion Pattern 4.2 Concept pattern for a *one-two pass* between players p_1 and p_2. [Mie04a, p.104], translated from German

$$
\begin{aligned}
&OCCURRING(OneTwo(p_1,p_2),i) \Leftrightarrow \exists j,k,l: \\
&\quad OCCURRING(Pass(p_1,p_2),j) \\
&\wedge \quad HOLDS(BallControl(p_2),k) \\
&\wedge \quad OCCURRING(Pass(p_2,p_1),l) \\
&\wedge \quad Meets(j,k) \wedge FinishedOrEquals(l,k) \wedge Starts(j,k) \wedge Finishes(l,i)
\end{aligned}
$$

the one-two pass pattern (pattern 4.2) which is constructed on the basis of two separate simple pass actions.

The temporal entanglement of constituent concepts within a concept pattern is expressed in terms of the possible interval relations which have been proposed by Allen [All84, All83, AF94] and Freksa [Fre92] in their respective research in the area of qualitative temporal reasoning, allowing for pattern descriptions which encode some flexibility as to the concrete temporal predicate configurations. This abstract representation constitutes a key-factor in order to obtain the required generality of concept patterns for a sufficient coverage of concrete concept instances.

Miene demonstrates the scalability of her analysis approach with respect to the recognition of complex situations where multiple actors are involved with a treatment of off-side situations in simulated soccer [MVH04].

Due to the character of the recognition concept outlined above the analysis is retrospective in character. This means that the occurrence of concept instances is stated a posteriori whenever a complete concept instance that occurred in the currently considered time frame is matched successfully against a certain concept pattern. Thus, an early recognition of partially completed action instances is laborious as the concept pattern for the respective complete action would need to be broken down into sub-patterns which then could be recognized upon completion. Thus an incremental recognition of action which are just about to happen is not supported immediately by the analysis.

Miene shows that her analysis approach can handle both domain-specific and domain-independent concepts. The latter allude to general movement patterns such as movement configurations (parallel movement, in-line movement, chase). Miene claims that, in supporting domain-independence, her analysis can be easily adapted to new domains since even in the absence of domain-specific semantics the general concepts still prevail.

The analysis approach proposed by Miene is described as being suited for simultaneous application due to the fact that the integration of raw input data associated with a new set of sensor values provided by attached sensors is intertwined with the processing of the recognition. Real-time capability is discussed briefly as theoretically possible. However this claim is not substantiated further as concrete deployments of the analysis implementation were performed offline as could be verified in a direct correspondence with Miene. While the quality of the analysis was the primary subject of evaluation no notion was given as to the runtime performance.

The basic approach proposed by Miene for the generation of qualitative ground predicates from quantitative raw sensor data and the subsequent bottom-up recognition of complex (domain-dependent) situation properties, events and (multi-agent) actions based on a matching process between concept patterns and the qualitative description of the dynamic scene at hand has been applied in the context of the ASKOF project[3] by Gehrke et al. for the qualitative representation and description of traffic scenes [MLVH04, GLH04, Geh05] and tested for feasibility under real-time constraints in synthetic traffic scenarios with a varying number of traffic participants. These tests showed that for a mapping interval of 100-150 milliseconds up to seven such actors could be handled without efficiency problems. Beyond that number the scene analysis started to limp behind significantly with the effect increasing with time of a simulation run which was attributed to the costs of interactions with the knowledge base rather than the cost for qualitative abstraction. Due to the fact that the number of traffic participants to be considered in the example scenarios was relatively small in most cases the general approach was proven to be suitable for real-time applications within certain bounds.

4.1.2 Behavior Recognition in the AT Humboldt

Jan Wendler considered the issue of behavior recognition in the context of his doctoral thesis [Wen03] in which he proposes a novel method for the automatic modeling of agent behaviors in complex multi-agent environments[4]. The approach comprises three constituent aspects, namely the above-mentioned *behavior recognition*, the *construction of behavior models* and the application of the assembled behavior models for the *prediction of agent behaviors*.

For the concrete development and implementation of his *behavior modeling* approach Wendler chose the RoboCup *2D Soccer Simulation League* which he – in accordance with Miene – characterizes as demanding, dynamic multi-agent environment. In the context of this application domain Wendler's behavior modelling approach suits an *opponent modelling* purpose. Similar to other team sports the global performance of a (simulated) soccer team is determined to a large extent by an adequate choice of team strategy. However adequacy cannot be measured detached from a concrete game situation but only in dependence of the respective counter strategy. The construction of an opponent model is a means to enable agents to understand the adversary and automatically adopt the team's own strategy in order to increase competitiveness. Wendler suggests alternative application domains such as robot rescue where behavior modelling is performed in order to optimize the cooperation efficiency in agent groups.

Wendler distinguishes two phases which are relevant for an effective behavior recognition. The first recognition phase is taking place offline and comprises the manual specification

3 ASKOF, 'Architektur und Schnittstellen für Kognitive Funktionen in Fahrzeugen', was a preliminary project at the Center for Computing Technologies (TZI) supported by the German Research foundation, a dedicated web site is available at: http://www.informatik.uni-bremen.de/agki/www/grp/agki/projects/index.html?project=askof&site=short (visited:16/12/2006)

4 Author's Note: The doctoral thesis is written in German. The original title is: *Automatisches Modellieren von Agenten-Verhalten; Erkennen, Verstehen und Vorhersagen von Verhalten in komplexen Multi-Agenten-Systemen*

Motion Pattern 4.3 Behavior Pattern for a *pass* from player p_1 to p_2. [Wen03, p.51]

$$pass(p_1, p_2, t_0, t_n, ballSpeed, playerMovement) : -$$

$$XBallControl_2(p_1, t_0, t_1)\ ,\ ballFree(t_2, t_3)\ ,$$
$$XBallControl_1(p_2, t_n, t_n)\ ,\ follow(t_1, t_2)\ ,\ follow(t_3, t_n)\ ,$$
$$ballFastDeparting(p_1, t_2, ballSpeed)\ ,\ teammateInKickRegion(p_1, t_2)\ ,$$
$$movement(p_2, t_2, t_n, playerMovement)\ ,\ sameTeam(p_1, p_2)\ ,$$
$$notSame(p_1, p_2)\ ,\ VirtuallyPlayOn(t_0, t_n)$$

of relevant behaviors by a domain expert. In this context the term 'behavior' relates both to simple actions actively performed by a single player such as a `pass` (pattern 4.3) or `dribbling` as well as to complex group actions such as the implementation of off-side traps.

In order to make effective use of behavior recognition results in the construction of behavior models Wendler claims that the common scheme to recognize only a sparse set of behavior attributes such as the initiator/receiver, the basic behavior type and – optionally – the success of execution for concrete behavior instances does not provide sufficient expressiveness. Due to this fact Wendler proposes the recognition of additional behavior-specific attributes that allow for a more precise characterization of behavior instances. An interpretation of carefully chosen additional attributes allows for the classification of a behavior instance as a distinct specialization of the basic behavior type. A suitable example for this course of action is Wendler's specification of the pass pattern that features two additional descriptive attributes: the speed of the ball directly after the kick by the pass initiator and the accumulated motion vector of the pass receiver during the whole pass action (cf. pattern 4.3 → `ballSpeed,playerMovement`). Based on these attributes Wendler demonstrates several possible pass distinctions which are not exclusive but rather meant to complement each other. Thus, a pass can be played direct or indirect, as a cross pass, backward pass, etc.

Wendler's approach to behavior recognition relies upon the assumption that behavior patterns can be described exclusively in terms of a gapless temporal sequence[5] of constituent qualitative ground predicates. An arbitrary overlapping of predicates cannot be modeled directly. Wendler uniformly refers to the modeled predicates as events. It should be noted that, contrary to the common notion of the term, Wendler's events feature a temporal dilation that can comprise either a single moment in the discrete time model of the underlying simulation environment (*instantaneous events*) or a time period (*durative events*). The denotation of the ground predicates as events can be misleading as for the most part Wendler's events describe situation properties such as exclusive ball control rather than conventional events such as a kick of the ball. A description of complex behaviors such as dyads as a sequence of both simple behaviors and events is not intended by Wendler. Due to this constraint the specification of complex behaviors does not scale well as the basic vocabulary for the specification remains bound to ground predicates only. This fact is exemplified by Wendler's specification of *pass* (pattern 4.3) and *one-two pass* (pattern 4.4). The latter pattern is bloated considerable, especially when compared to the equivalent pattern by Miene (pattern 4.2, p.32). In Wendler's approach behaviors which have been recognized already have no value for the further recognition process.

For the second recognition phase which is executed automatically based on the specified

5 In the pattern declarations this requirement if enforced by the explicit `meets(...)` relations [All84] called `follows(...)` by Wendler (cf. pattern 4.3)

Motion Pattern 4.4 Behavior Pattern for a *one-two* pass between players p_1 and p_2. [Wen03, p.79]

$$pass(p_1, p_2, t_0, t_n, bS1, pM1, positioningMovement, bS2, pM2) : -$$

$$XBallControl_2(p_1, t_0, t_1)\ ,\ ballFree(t_2, t_3)\ ,$$
$$XBallControl_3(p_2, t_4, t_5)\ ,\ ballFree(t_6, t_7)\ ,$$
$$XBallControl_1(p_1, t_n, t_n)\ ,$$
$$follow(t_1, t_2)\ ,\ follow(t_3, t_4)\ ,$$
$$follow(t_5, t_6)\ ,\ follow(t_7, t_n)\ ,$$
$$ballFastDeparting(p_1, t_2, bS1)\ ,\ teammateInKickRegion(p_1, t_2)\ ,$$
$$movement(p_2, t_2, t_4, pM1)\ ,\ positioning(p_1, t_2, t_5, positioningMovement)\ ,$$
$$ballFastDeparting(p_2, t_6, bS2)\ ,\ teammateInKickRegion(p_2, t_6)\ ,$$
$$movement(p_1, t_6, t_n, pM2)\ ,\ sameTeam(p_1, p_2)\ ,$$
$$notSame(p_1, p_2)\ ,\ virtuallyPlayOn(t_0, t_n)$$

behavior patterns Wendler proposes an object-oriented incremental recognition algorithm for behaviors which can be applied in a real-time scenario by a dedicated observer agent which possesses a complete and noise-free perception of the observed agents acting in the simulation environment. In the concrete implementation, the behavior recognition is performed online by the dedicated coach agent, observing the behavior of a single team of soccer agents during regular matches in the RoboCup *2D Soccer Simulation* environment.

In the object-oriented implementation each behavior pattern is represented by a single *recognition instance* which is due to detect occurrences of the specified behavior over the course of the game. Each *recognition instance* carries an internal state which indicates the respective current state of the recognition process. Initially a *recognition instance* is waiting for an initial match of the first element in the *event sequence* within the respective world state. While no such match can be found the associated behavior is not happening. When an initial match is found the recognition process is started as the match might be a first hint that the behavior has been triggered. Thus the possible initialization of a new *behavior instance* in the simulation is investigated. The *recognition instance* is now pending. In order to fully trigger the concrete recognition of a new *behavior instance* in subsequent processing steps further successful matches of world state and active event in the *event sequence* need to occur until finally the initial event sub-sequence which represents the complete *behavior trigger* has been processed. However if the matching cannot be continued until the passing of the trigger threshold the *recognition instance* is reset as the hypothesis with regard to the start of a new *behavior instance* is discarded.

After the recognition of the *behavior trigger* the remainder of the *event sequence* that constitutes the *behavior pattern* is matched step by step until either the sequence could be matched completely which corresponds to the recognition of a successfully executed *behavior instance* or no further match can be found. In the latter case the recognition algorithm tries to identify the reason for the mismatch. Upon success a new *behavior instance* has been recognized which could not be executed successfully and led to a certain undesired outcome. Otherwise the *recognition instance* is reset.

During the course of the recognition procedure the attributes which characterize the respective behavior instance are allocated with values by and by during the whole recognition of the behavior instance whenever sufficient data has been acquired. Thus early in the recognition the characterization is fragmentary and grows more and more complete

towards the completion of the recognition cycle.

The recognition algorithm developed by Wendler is designed to recognize only a single instance of a certain behavior pattern at a time which is sufficient for the treatment of ball-oriented behaviors in the soccer domain. By allowing only a single active behavior instance at a time Wendler also implicitly handles the problem of multiple ambiguous interpretations of the same behavior which could arise when the source of a kick initiating a new behavior cannot be determined with certainty as more than a single agent might have had the opportunity to act. The single-instance bears the drawback – convenient as it may be for the considered application scenarios – that it cannot be immediately applied for behaviors such as *cover* which may be executed by a significant part of a whole team while in a defensive situation. For a closely related approach which implements a multiple-recognition capability Wendler refers to the work by Müller [Mül02].

Wendler claims that the data associated with each moment in time is processed only once during the whole recognition process in a sequential order. Also, the assignment of the behavior parameterization occurs as time progresses. Thus both with respect to the utilization of its input data and the recognition process as such Wendler's approach has an incremental character.

Due to the further application of his approach for the modelling of behaviors, Wendler sets aside the compilation of a rich qualitative representation which comprises situation properties, events and actions. Thus, while events and situation properties appear in the description of behavior patterns, they are solely used as building stones for those patterns. In the concrete implementation they are pulled immediatly[6] from the quantitative knowledge representation only when this is required by the *world state – event* matching process outlined above.

Wendler evaluated the concrete implementation of the behavior recognition for seven distinct behavior types[7] with a series of test data-sets obtained from games played at the RoboCup *German Open 2001* both with respect to required computation time for each processing cycle and recognition quality. The coach program was executed on an average PC (400 MHz AMD CPU) and managed to complete the computation of a single recognition cycle in at most 1 millisecond. Wendler claims that the required computation time is linear to the number of applied behavior patterns and hence concludes that the developed recognition approach is suitable for unconfined real-time use. In regard to recognition quality the first result with respect to coverage is that on average the recognition provides results for roughly 98% of the overall game time. Wendler also tested for correctness and completeness of the recognized behaviors based on a comparison between the results provided by the implemented algorithm and a ground truth provided by the result of an objective human examination of the applied test games. The baseline of these tests is that, based on complete, noise-free input data, the recognition performance is comparable to the human equivalent.

With respect to a possible application of his recognition approach for agents actively participating in the observed soccer matches, Wendler concludes that, due to the restricted and imprecise vision of players in comparison to the coach, a direct transfer of the proposed approach is not feasible. He assumes that once behavior patterns are described in terms of few *compulsory events* and otherwise *optional events* an application for soccer players becomes an option. Other problems which might arise out of an reduced quality of the input data due to sensor noise are not considered.

6 i.e. obtained via query functions

7 `balltransfer, pass, dribble, score, clear,one-two pass` and `'fight for exclusive ball control'`

4.1.3 VIsual TRAnslator (VITRA)

In the research context of the CRC[8] 'Künstliche Intelligenz und wissensbasierte Systeme' funded by the *German Research Foundation*, Herzog and colleagues were concerned with the development of knowledge-based systems, capable of the interpretation of visual input (i.e. imagery from a dynamic scene) and a translation thereof in an appropriate natural language description of ongoing events. The project was called VIsual TRAnslator (VITRA) [HBG+96]. The desire to enable an automated soccer commentary system provides a motivation for an *incremental recognition strategy* in contrast to *a posteriori strategies*.

VITRA was comprised of several sub projects one of which was concerned with the automated generation of commentaries for short video broadcast sequences from regular soccer games based on a high-level interpretation of the observable occurrences in dynamic soccer scenes. The system was called SOCCER [HG89]. While the research on SOCCER was focused on the recognition of high-level concepts based solely on the observable geometrical flow in a dynamic scene an add-on project by Retz-Schmitdt, called REPLAI[9] [Ret92, Ret91], was focused on an interpretation of the recognition results provided by SOCCER, based on the assumption that soccer players are intentional agents whose actions have a purpose in the respective context of the game.

SOCCER: Incremental Spatio-Temporal Analysis of Soccer Scene for Automated Commentary

In order to provide automated commentator capabilities for SOCCER, a major step comprises the interpretation of events on the soccer field using high-level scene analysis. The desire to assemble real-time comments for player actions while the soccer game unfolds implies special demands with regard to the recognition strategy and the availability of recognition results for further processing. Since event recognition and discourse generation are executed in parallel or at least intertwined it is mandatory that the discourse component is supplied not only with information about completed actions but also about those actions about to proceed. The rationale is that both discourse planning and, ultimately, the utterances for the comments require a certain amount of time, such that if only completed actions were considered the discourse generation would start to significantly lack behind the events on the field.

The incremental event recognition proposed by the VITRA group [Her95b, AHR88, RHA87, HG89] is based upon *event models* which describe the a-priori expert knowledge about the common structure of events, The core of each *event model* is a so-called *course diagram*, a special, labeled directed graph where each edge of the graph is defined by a tuple `(source, goal, condition, type)`. An example for such a *course diagram* is presented in figure 4.2. While `source` and `goal` of an edge refer to nodes in the graph and define the traversal direction the *edge condition* is comprised of a conjunction of sub-events in a specified state. If the conjunction is true, which means that the required set of sub-events could be recognized for the respective discrete time step, the traversal condition for the edge is satisfied and the recognition proceeds. A complete successful recognition of a particular event occurrence in the soccer scene thus corresponds to an individual complete traversal of the *course diagram* from the graph's source node to the goal node.

Herzog et al. state that, for an optimal further exploit of preliminary recognition results, the common binary distinction between events that have occurred and those that have not is insufficient [Her95b, AHR88]. Hence, they introduce an extended set of activity

8 Abbrev.: Collaborative Research Center, dedicated funding instrument of the German Research Foundation

9 Abbrev.: 'REcognition of PlAns and Interactions'

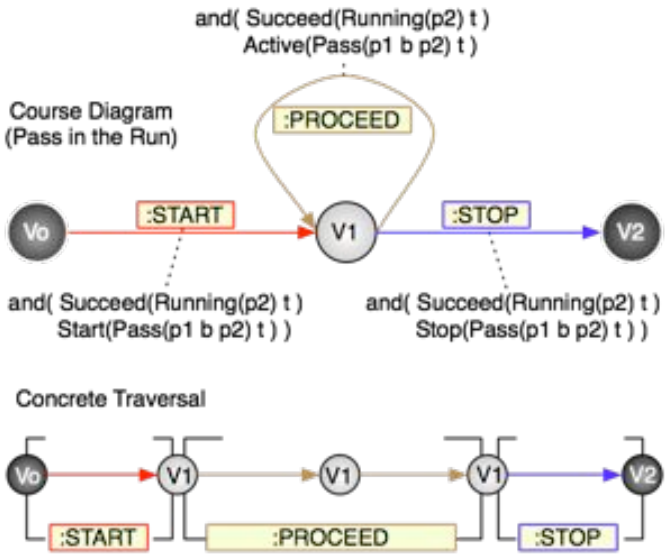


Figure 4.2: Simple course diagram for *pass in the run* action (reproduction of figure in [RHA87, p.9]) and concrete sample traversal of course diagram.

states for events, namely `:START`[10], `:PROCEED`, `:STOP` and a special state `:SUCCEED` for durative events. In order to maintain information about the activity state of an event during the recognition/traversal process the edges of the *course diagram* are typed with one of the states introduced above. The type of the edge which has been traversed in the current step thus indicates whether the respective event occurrence has just been started, is still proceeding or has just been finished. The concrete event recognition task consists of an instantiation of the predefined event models based on information about the dynamic scene obtained from the SOCCER knowledge base. "*As soon as new input data are provided [...] the recognition component continues traversing course diagrams already activated and tries to trigger new ones.*" [HG89, p.9]

It should be mentioned that course diagrams as introduced by Herzog et al. are designed for the complete recognition of successful event occurrences. The premature abortion of an event occurrence can be determined if for a certain time step the course diagram offers no out-going edge from the currently occupied graph node whose traversal condition is satisfied. This means that a course diagram disposes of only a single sink node for success whereas failure conditions are not modeled. Thus, it is not possible to assert why an event failed. Further research in that direction has been done by Cavazza and Palmer in [Cav00] which propose the use of extended course diagrams with additional *exception edges*.

Herzog et al. propose an object-oriented implementation for their incremental recognition which for each recognition step applies a bottom-up strategy by starting with the recognition of atomic events and progressing upwards in the event hierarchy which is formed pragmatically by defining the level of each event model as one above the maximum level of any sub-event used in any edge condition [RHA87].

Herzog discusses the problem that the specification of an appropriate set of course diagrams seems to be 'rather lavish' in comparison with declarative interval-based concept descriptions [Her95b, p.6] as in the approach by Miene (cf. section 4.1.1 on page 29) and identifies the following problems. First, a declarative description such as figure 4.3 is easier to comprehend for the knowledge engineer. Second, course diagrams are not particularly

10 also referred to as `:TRIGGER` in publications other than [Her95b]

$(defevent \quad PENALTY\_KICK(p * playerg * goalkeeper)$

:subconcepts

$RUN\_UP(p)[I_1],\ SHOOT(p,b)[I_2],\ MOVE(g)[I_3],\ PARRY(g)[I_4]$

:temporal relations

$[I_1] \vdash starts \dashv [PENALTY\_KICK],\ [I_1] \vdash before \vee meets \dashv [I_2],$
$[I_3] \vdash during \dashv [PENALTY\_KICK],\ [I_3] \vdash meets \dashv [I_4],$
$[I_4] \vdash finishes \dashv [PENALTY\_KICK],\ [I_2] \vdash finishes \dashv [I_3]$

Figure 4.3: Declarative Description for a *penalty kick*. [Her95b, p.7]. The intervals have temporal relationships as described by Allen [All84].

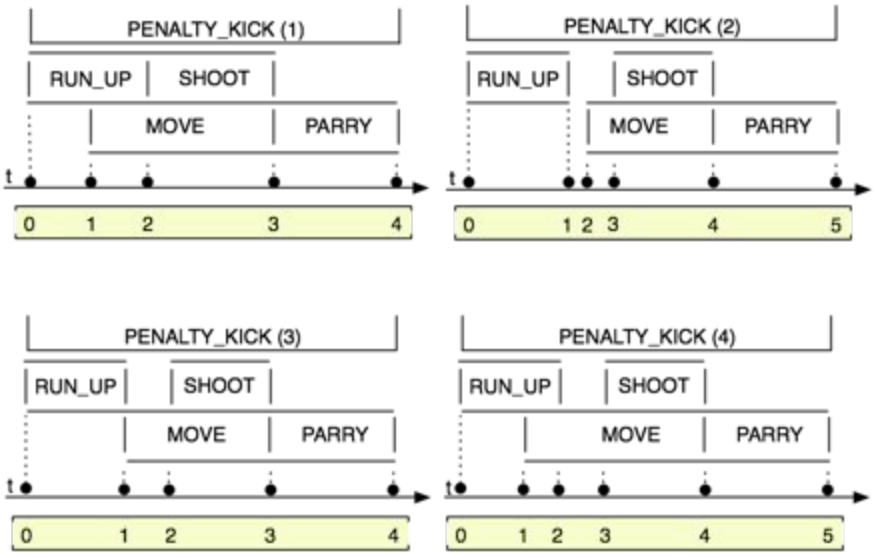


Figure 4.4: Possible configurations of the sub-intervals from figure 4.3, calculated by the temporal reasoner. Since the description contains both an explicit (RUN_UP & $SHOOT$) and an implicit (RUN_UP & $MOVE$) relation disjunction, the mapping from description to course diagram is ambiguous. [Her95b, p.8]

well suited as basis for a knowledge inference procedure where the occurrence of certain unobservable events in the domain is derived from other events which actually could be observed.

In order to overcome these restrictions Herzog proposes the deployment of a dual representation, thereby acknowledging that while course diagrams are a key concept for the realization of an incremental recognition strategy, the declarative description is desirable for further reasoning tasks based on the recognition results. He proposes to use an automated translation from declarative interval-based concept descriptions to course diagrams. The translation is understood as a temporal reasoning problem to be solved with temporal constraint propagation techniques. Given the respective temporal entanglement of the constituent predicates in a certain description[11] the temporal reasoner[12] tries to find the temporal relation between each pair of predicates. If no disjunctions[13] occur the process

11 such as `starts, before, meets, during` and finishes in figure 4.3
12 Herzog mentions the TIMELOGIC system by Koomen as reasoner of choice
13 alternative for a concrete temporal relation

corresponds to a projection of the predicate validity intervals onto the temporal axis which then allows a non-ambiguous diagram construction. However if disjunctions are found, as in the example (figure 4.3), the reasoner enumerates the alternatives where each single one corresponds to an own concrete course diagram (figure 4.4).

This statement of affairs is interesting as it highlights the fact that, generally speaking, the expressiveness of both formalisms is *equal*. However, the declarative description sometimes corresponds to a family of course diagrams due to the flexibility which is obtained by an application of the qualitative temporal relations between predicates as proposed by Allen [All84]. In the translation procedure, the flexibility of the description is represented as alternatives in the course diagram formalism. Based on this fact it can be stated, that both Herzog and colleagues and Miene (cf. section 4.1.1 on page 29) use rather expressive basic formalisms to specify their respective descriptions/patterns in comparison to Wendler (cf. section 4.1.2 on page 33) who effectively specifies behavior patterns on the complexity level of course diagrams. With respect to the proposal of a dual representation and automatic (one-way) translation, SOCCER provides a formal bridge between the contributions from Miene and Wendler.

The translation procedure outline in [Her95b] is rather expensive as the applied algorithm has exponential runtime complexity. Its application is feasible nevertheless due to the low expected number of constituent predicates ($\#pred < 10$), thanks to hierarchical ordering of events, and the fact that the translation needs to be computed only once in advance before any time-critical recognition is actually performed.

Eventually the basic research done for the SOCCER commentator system provided the basis for a more sophisticated multi-media-capable system, the RoboCup Commentator (ROCCO)[14] [VAHR99] designed to accept log data generated by the Soccer Server in regular matches of the RoboCup *2D Soccer Simulation League*. ROCCO uses the same incremental recognition strategy as its predecessor and proved its unconstrained applicability during RoboCup 1998 where it was awarded the annual RoboCup Scientific Award for an exceptional research contribution.

REPLAI: Recognition of Plans and Interactions (in Soccer Games)

In the context of the research performed for the VIsual TRAnslator (VITRA) introduced above Gudula Retz-Schmidt was concerned with the development of the REPLAI subsystem that would accept the events recognized by the SOCCER system and perform a super-ordinate *keyhole recognition* for executed plans and interactions in a dynamic scene.

In the motivation of her approach outlined in [Ret91] Retz-Schmidt claims that systems such as SOCCER are limited in that they describe only immediatly observable spatio-temporal aspects of dynamic scenes. However "*human observers do not only pay attention to the spatio-temporal aspects of motion. They interpret what they see.*" [Ret91, p.174]

Due to the considerable flow rate of raw data comprised of both events[15] and spatial relations from the SOCCER recognition system Retz-Schmidt introduces a *focus mechanism* for REPLAI in order to filter out the sub-set of data which is considered bearing relevance for further processing. Retz-Schmidt justifies this approach with a hint to human observers which are also limited significantly by a restrictive *visual focus* which allows to keep track of no more than half a dozen objects at a time. Yet obviously humans are reasonably good in attributing intentional behavior to observed entities. For each processing step REPLAI determines a sub-set of interesting players on the soccer field based on *domain-specific focus heuristics*. The system considers players carrying out interesting

14 A dedicated web site is still maintained for the ROCCO system on http://www.dfki.de/imedia/robocup/ (visited:06/08/2006)

15 actions of agents such as `(dribble player4)` and dynamic agent properties such as `(have_ball player3)`

actions (`dribble`) or having interesting properties (`have_ball`), players with an interesting spatial relation with an already focused player, or special static objects such as the goals on the soccer field. Beyond that in REPLAI there is a notion of *star players* which due to their special capabilities must be kept in focus regardless of the immediate situation as decisive actions for the course of the game must be expected being carried out by those players at all times. During normal operation the *focusing criteria* are kept invariant, thus rendering the focus deployment a ground step in the bottom-up strategy for subsequent plan and interaction recognition. However, Retz-Schmidt introduces the possibility to expand the focus temporarily replacing the standard heuristics by demand of higher-level recognition components.

In REPLAI distinct system components are responsible for the recognition of *plans* which apply for single agents and for *interactions*, composite plans which are carried out by groups of agents where each agent is contributing by trying to achieve an individual partial plan.

The plan recognition is based upon a domain-specific plan library which has been assembled manually using expert knowledge such as coaching experience and textbooks on soccer strategy. Actions and their hierarchical relation are represented in a special data-structure, an *AND/OR-Tree*. This tree structure describes both the hierarchical ordering of plans where the abstraction level of the plans grows towards the tree's root and the respective plan decomposition in a sequence of plan constituents which can be either sub-plans or elementary actions whose execution by individual soccer players can be directly observed. The decomposition specifies the temporal order of plan constituents. Successive plan steps are interconnected with temporal arcs that express immediate succession. Since besides compulsory forward-directed arcs additional reflexive and backward-directed arcs are allowed the plan description is quite expressive as plans can contain sequence, alternatives and repetitions. The nodes in the plan hierarchy are associated with a set of preconditions and intended effects. The preconditions specify constraints that have to be satisfied in order for the respective action/plan to be adequate at a certain moment during a soccer game.

The actual plan recognition in REPLAI is described as a top-down search and instantiation process. Two recognition modi can be distinguished. The first applies when no current plan exists for an agent. The system then proceeds with a top-down search of the plan hierarchy checking the plan preconditions along the way to identify those plans which are plausible given the current situation. The leaves of the selected subtrees which represent observable actions are matched against the input data observed from the SOCCER system. If a match can be found a new plan hypothesis is instantiated. At the end of the search process in general a set of possible plan hypotheses has been gathered each stating that the agent is supposedly executing a certain plan and has proceeded up to a certain execution step in the process. Now that hypotheses exist the second recognition mode applies where the further progress in the realization of presumed hypotheses is monitored. Since search and tracking are applied in a mutually exclusive fashion and hypotheses which are no more supported by the incoming data are dismissed the number of remaining active hypotheses dwindles as the game progresses eventually leading to either an empty hypothesis set (which spurs another hypothesis search) or a single hypothesis in which case the associated plan is considered as recognized. Once its last decomposition part is executed the plan is considered observed.

The recognition process provides the necessary data for the dependent recognition of agent intentions. Retz-Schmidt distinguishes *state-directed intentions* and *action-directed intentions*. The latter intentions are especially noteworthy as they indicate what the agent currently executing a plan is determined to do in the immediate future. The *action-directed intentions* are in principle comprised of all potential next plan steps for the respective plan hypothesis. If more than a single next step exists REPLAI does not evaluate a probability distribution over the incidence of possible steps. It rather finds

the next unambiguous plan which comprises all alternatives and declares this plan as the current intention thereby making a less specific but safe statement.

REPLAI is an interesting research effort due to the fact that it demonstrates a working connection of a system for the spatio-temporal analysis of dynamic scenes in a certain domain[16] and a plan recognition system. Based on the provided qualitative data and additional expert knowledge encoded explicitly in the plan library, the recognition system is able to recognize higher-level concepts. The recognition process is advanced from the recognition of 'simple' actions to strategic moves. At the same time due to the availability of both recognized actions and plans schemas it becomes feasible to make assumptions about the immediate intentions of players on the field thus going from recognition to prediction.

4.1.4 Application of Qualitative Reasoning to Robotic Soccer

Steinbauer et al. discuss the application of qualitative reasoning for robotic soccer [FSW04, SWW05]. They provide a detailed description of the development of a robust *hybrid control system* for autonomous robots in the RoboCup *Middle-size Robot League* (cf. section 2.1.1 on page 9) which is a combination of a *reactive control system* and a *deliberative control system*. Both aspects – reactivity and deliberation – are retained in the target system as they complement each other well. Due to their small control loop which maps sensor information immediatly to executable actions reactive control is suitable in situations where a swift action execution is mandatory such as in the soccer defense. On the other hand deliberation, the explicit planning of (team) actions using classical artificial intelligence techniques[17], is required in order to execute specific tactic moves and thus adhere to a super-ordinate team strategy.

The desire to implement a planning system leads to the demand for a qualitative representation. The authors argue that 'planning tasks are particularly appropriate for application of qualitative techniques' and provide a list of advantages which are gained by performing planning based on a suitable qualitative representation [FSW04, pp.3]. First the search space for the planning problem is kept small as a qualitative model equals an infinite number of numerical models. This is possible by subsuming situations which can be considered equivalent in the context of the application domain from a high-level point of view even though they bear differences with respect to their respective *geometrical layout*. Furthermore due to the qualitative abstraction *graceful degradation* is achieved meaning that the robot can keep operating in the presence of errors in its available quantitative raw data. The authors also note that an explicit treatment of uncertainty and missing knowledge is possible in a qualitative model.

For their qualitative reasoning approach Steinbauer et al. start with the definition of a domain-specific *qualitative representation language* that comprises a suitable set of first-order *qualitative ground predicates* such as `inReach(x)`. For each such predicate an associated *interpretation function* is defined which for a specific moment in a soccer game allows to evaluate whether or not the predicate holds for the current moment in time given the state of the world as perceived on the quantitative level. The *interpretation function* is the means by which the abstraction step from quantitative raw data to meaningful, human-readable, symbolic facts is performed which can be subsequently used as input for the classical AI planner in order to determine the next course of action for the robot.

In real-life scenarios as found in the RoboCup *Middle-size Robot League* the mapping from a quantitative model to symbolic predicates in a dynamic and uncertain environment leads to two major problems:
First, the truth value of predicates is calculated using thresholds, i.e. there are sharp

16 which yields a well-grounded qualitative representation
17 In their own approach Steinbauer et al. use a state space regression planner and refer to [Wel94]

boundaries for the quantitative input values which considered in the *interpretation function*. Thus, even slight changes of the environment lead to truth value changes which can lead to instability in the high-level decision making process as plans grow more and more likely to be interrupted eventually. This is counter-productive to the desire to achieve a certain commitment once a plan has been chosen for execution.
Second, Situations can occur where the truth value changes continually in subsequent computations due to a value oscillation of the quantitative input values around the sharp numerical boundaries which are used for predicate validity checks. Such an oscillation can be caused by the inherent fluctuations in the robot's sensor input (sensor noise) due to seemingly indiscernible changes in the environment (such as the lighting conditions during a match) and the constrained perception capabilities of the robots based upon sensors with limited precision which cannot be worked out by the respective localization techniques (such as particle filters).

In order to mitigate the identified problems Steinbauer et al. propose to take into account the predecessor state of a predicate for the consecutive evaluation step and to introduce a certain resistance of predicates against state changes. Borrowing from the field of electrical engineering where the technique at hand is known as *hysteresis function* a change of the predicate state can only be brought about by a sufficiently strong force that works against the predicate's persistence effort. In the context of robotic soccer 'strong force' refers to a significant change in the environment of the robot which is reflected in the value run of the quantitative data considered in the validity evaluation. By applying *predicate hysteresis* for the change behavior of predicate truth values it is possible to trade off necessary reactivity of predicates against the desire to suppress validity oscillation and obtain increased stability. *Predicate hysteresis* is modeled by a temporary extension of the numerical value boundaries which are used in evaluations of the predicate truth value [SWW05]. The extent of the *predicate hysteresis* needs to be tuned for each specific predicate and given application scenario by a modulation of the *hysteresis size* (increase of the bounded numerical value range).

Steinbauer et al. performed a thorough evaluation of predicate generation with applied *hysteresis* compared to their basic setup and were able to confirm the assumption that significant improvements in predicate stability can be obtained with small *hysteresis sizes* while retaining enough reactivity for an adequate adoption to situation changes [SWW05].

4.1.5 Learning the Sequential Coordinated Behavior of Teams from Observations

Kaminka, Veloso et al. focus on the task to apply autonomous unsupervised learning for the recognition of sequential behavior of both single agents and agent teams based on the observation of their behavior traces [KFCV03]. The RoboCup *2D Soccer Simulation league* is used as test-bed for the development of their two-tier approach.

An observing agent first uses a set of simple basic behavior recognizers in order to parse the stream of incoming raw observations which comprises a multi-variate time series of sensor readings and compile it into a stream of recognized behaviors which are referred to as events such as `pass(player1,player2)` or `dribble(player3)`. As can be seen in the examples the atomic events are annotated with their respective participating players. It should be noted that to the best knowledge of the author, no further annotations are maintained such as action execution time intervals or additional attributes that would allow for the assessment of action execution characteristics. In the construction of the recognizers which is performed manually by a designer well acquainted with the application domain (such as simulated soccer) their function is restricted to the recognition of atomic events.

The recognizers remain ignorant of ways how these simple atomic events may be com-

bined in order to form higher level events and operate independently from each other. Thus while the multi-variate input stream is fed into the system, the recognizers are applied in parallel, each computing only the occurrences of its associated atomic event over the course of a game. In order to detect those occurrences each recognizer is built as a logical combination of quantifiers and qualitative predicates such as `Possessor(player,t)` or `Teammate(player1,player2,T`[18]`)`. The predicates which are extracted immediatly from the raw data stream and which are associated with single moments in discrete time describe spatial relationships between player and ball and organizational relationships between players. Based on their encapsulated rules the recognizers detect event occurrences with some accuracy.

Kaminka et al. admit that ambiguous interpretations can occur in their recognition approach when for instance the actual kicker of the ball cannot be determined with certainty among a set of candidates. Hence overlapping event occurrences may be recognized. The authors claim that in the considered application domain of simulated soccer the degree of uncertainty introduced in the recognition of atomic actions is moderate as it is the sequential combination of such atoms to multi-agent actions (dyads) gives rise to complexity and increased uncertainty due to the large space of combinations for the construction of sequences.

As for the handling of ambiguities in the recognition of atomic events the authors use manually assembled domain-dependent heuristic rules which prefer certain interpretations rather than other alternatives. Using the recognizers and the heuristics for disambiguation the stream of world state events is transformed into a single sequence of atomic events such as:

$\ldots$ `pass(player1,player2)` $\rightarrow$ `pass(player2,player3)` $\rightarrow$ `dribble(player3)` $\ldots$

For further analysis the complete event stream is segmented by team yielding two sets of uninterrupted event sequences of varying length. The assumption made by Kaminka, Veloso et al. is that within these event sequences team characteristic complex action or interaction patterns can be identified which give hints as to the strategic play of the respective team. A problem which arises due to the highly dynamic character of the application domain is the fact that while groups of agents try to carry out (pre-)planned coordinated activities the circumstances in the game may lead to reactive variations of the basic patterns. This factor must be acknowledged and handled in the statistic evaluation of the event sequences. Once the events are stored in a special trie data structure that allows for efficient data retrieval, two competing methods are used by a learning component to statistically evaluate the significance of event sequences or parts thereof. The first method is based on frequency counting, the second on statistical dependency checks.

Experiments showed that both methods are suitable to detect characteristic event sequences for a team of soccer agents. However the method build upon statistical dependency checks seems to model the causal relations among the actions within the sequences better than pure frequency counting.

The action recognition presented by Kaminka, Veloso et al. is quite limited with respect to expressiveness as solely action sequences are compiled which in the absence of further predicates describing the respective situational contexts are barely sufficient neither to understand the course of the game nor draw further inferences based on the provided foundation. Another drawback is that the only attributes which are maintained for each action are the participating agents. Thus the actions are only represented in terms of their most rudimentary features. While this modus operandi is suitable for the immediate learning task to be supported, alternative usage scenarios for the compiled data beyond simplistic statistical evaluations don't seem to be feasible with the presented special purpose recognition approach.

18 the capital `T` implies a time-invariant predicate

4.1.6 ISAAC, Aiding Humans in Understanding Team Behaviors

Raines and colleagues present their research efforts directed at the development novel effective means to help human designers of multi-agent systems to understand the complex behaviors by groups of agents in their respective domain [RTM00, NMMR04]. An automated team analyst (ISAAC) is proposed which is designed for a post-hoc, offline analysis of agent teams and adheres to three principal design constraints that are important for team analysis. First, ISAAC supports multiple analysis angles which differ in their level of granularity. While it is possible to concentrate on the analysis of critical actions performed by single agents it is also possible to analyze interaction patterns of agent groups as well as global team performance. Due to its focus on helping humans to understand agents ISAAC is designed not only to dig up helpful information from analyzed data but also to convey this information in an appropriate way which is distinct for each respective analysis angle. While analysis results for global team analysis are represented in natural language summaries, ISAAC employs a multi-media viewer for the remaining analysis angles. ISAAC uses a data-driven unobtrusive approach in its analysis using external behavior traces calculated from low-level sensor data as its starting point.

ISAAC was applied successfully in the domain of RoboCup *2D Soccer Simulation* both during the '97-'99 RoboCup World Championships and by a considerable number of simulation league teams as a valuable analysis tool in the preparatory phase before the respective RoboCup events. In 1999 ISAAC was awarded the RoboCup "*Scientific Challenge Award*". Even though it has deep roots in the domain of simulated soccer the analysis techniques developed for ISAAC have been shown to be applicable in other domains such as the analysis of communication behavior for collaborative software agents [NMMR04, pp.39].

For the development of ISAAC Raines et al. used a two-tier approach for the analysis problem at hand. The first stage addresses the acquisition of models which describe behaviors on the respective level of granularity in a compact and human-understandable way. The second stage is dedicated to the exploitation of the learned models in analyzing a certain team of soccer agents. In the following both stages are outlined briefly with respect to the three supported analysis angles.

On the lowest level dedicated to individual agent actions Raines et al. concentrate on so-called key actions whose execution leads to clear-cut immediate success or failure. In the work described in [NMMR04] only the shot at the adversary goal (in the intention to score) is supported as concrete key action. Once a set of features which are considered relevant with respect to success and failure such as `ball velocity`, `distance to goal` or `number of defenders`[19] has been identified decision tree learning (*C5.0*)[20] is used on a large set of sample situations from recorded soccer matches. Once the learning phase is completed the computed decision tree encodes a comprehensive set of rules specifying conditions the execution of the key action and either leading to success or failure. While the tree is not meant be used for prediction purposes it helps to identify the feature subset which indeed bears relevance for the success of a key action for a certain team. The rules can provide clues as to where a team's weaknesses with regard to the respective action are located. ISAAC allows for a so-called *perturbation analysis* where learned rules can be modified in a single rule condition[21]. Once the perturbation is specified ISAAC mines situation matching the new rule and evaluates their success. Thus it is possible to evaluate 'what-if'-scenarios.

On the level of agent interactions Raines et al. concentrate on key interactions, action sequences of a certain length such as (`opponent` → `shooter` → `goal`) which again yield immediate success or failure (a goal). On this level ISAAC learns a probabilistic

19 between agent trying to score and the goal

20 *C5.0* is an optimized implementation of the C4.5 decision tree learning algorithm written and distributed by Ross Quinlan/RuleQuest Research, http://www.rulequest.com (visited:16/12/2006)

21 The modification consists of a negation of the chosen condition

finite automaton. Based upon this data structure it is possible to analyze the way teams typically score goals. It is also possible to compare teams two at a time and determine a degree of resemblance in their interaction behavior.

On the level of global team model complete games are considered as input for another tree learning process (again C5.0) besides a set of features such as `ball possession time, caught in off-side trap`, etc. ISAAC learns and later uses seven classes ranging from `big win` (a victory by five goal or more) to `big loss`. Once again the decision tree identifies the important factors among the chosen feature set leading to the respective outcome. In order to analyze a new game ISAAC computes its game statistics and seeks to find a matching rule and generates a template-based natural-language report of the game where it also handles game results which are not in accordance with ISAAC's experience encoded in the decision tree.

Raines and colleagues claim that ISAAC has been intensively tested by the RoboCup soccer simulation community over the years and has shown without doubt its use for the development process of complex multi-agent systems for dynamic environments [NMMR04, p.5 & pp.32].

A drawback of the ISAAC system is the clear-cut constraint that it cannot be applied under real-time conditions due to its character as sophisticated tool to mine comprehensive information from log-file data for use by agent designers.

4.1.7 Probabilistic Analysis of Football Matches

Bobick and Intille present a probabilistic framework for the recognition of complex multi-agent actions in dynamic scenes which is used for the recognition of typical plays[22], executed by the offensive team, during American Football matches[23] [Int99, IB99, IB01] (cf. figure 4.5).

The approach advocated by Bobick and Intille comprises four representation elements. The first thereof is made up of a *temporal structure description* of the global team behavior for each distinct kind of offensive play. For each offensive player involved in the play a set of individual goals – actions to be executed as the offensive play unfolds – are specified with a suitable parameterization. A partial temporal ordering of the individual goals is defined using the `before` relation. Goals associated with different agents can also be brought in a global temporal relation stating that a pair of goals either needs to happen at `about` the same time. Description also allows alternative courses of actions as goals from distinct agents can be in an `XOR` relation.

The second representational element is a set of so-called *visual networks*, *Bayesian belief networks* which accept visual evidence obtained from the observation of the dynamic scene. Each belief network models a single goal or event used in the *temporal structure description*. These belief networks are constructed manually by a knowledge engineer. Due to their size (typically 15-25 nodes) and connectivity they allow for exact propagation algorithms to compute the probabilities for each node state given a certain evidence set. These belief networks feature two classes of nodes: *unobservable belief nodes* with a binary state (`true, false`), amongst them the *main goal node*, and *observable evidence nodes* which are directly dependent upon the raw input data. All *evidence nodes* feature symbolic states which are constituted either by binary or trinary[24] truth values or

22 Contrary to soccer, American Football is highly structured in distinct plays, short periods (measured in seconds) with a clearly defined task distinction between attacking and defending team. The situation at the start of each play is comparable to a standard situation in soccer. The attacking team plays according to well-defined tactical patterns.

23 A dedicated web site is still available at: http://vismod.media.mit.edu/vismod/demos/football/index.htm (visited:04/12/2006)

24 Besides `observed` and `notObserved` an additional state is introduced which represents uncertainty of observation: `maybeObserved`

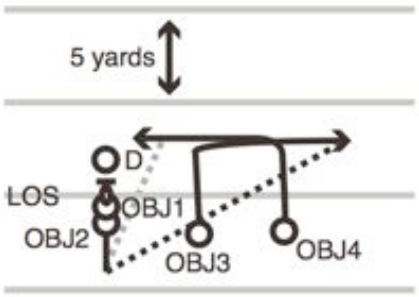


Figure 4.5: A typical play in American Football. [IB99, p.2]

specialized symbolic states for such input as distance between two objects. Such symbolic states are obtained by a quantization of continuous quantitative input values by dedicated *feature detectors*. The *main belief node* accepts parameters such as the acting agent. If a *visual network* is evaluated in the same way for consecutive time frames the emerging result is a *likelihood curve* over time. Within the probability-based formalism proposed by Bobick and Intille continuous-value curves as displayed in figure 4.6 correspond to binary 01-curves obtained when interpreting predicates with temporal dilation as used in the approaches by Miene (cf. section 4.1.1 on page 29) or the VITRA project (cf. section 4.1.3 on page 37) accordingly.

Temporal analysis functions constitute the third representational element. These functions act as evidence detectors for temporal relations (`before`,`around`) which are used in the *temporal structure descriptions* as well for goal occurrences. They rely upon heuristics which compare amongst others the activation levels of each goal over time, characteristics of each input curve and the temporal distance between certain curve features. The functions preserve the uncertainties in the output of the *visual networks*. Harsh thresholds are thus not encountered.

Finally the plays actually performed on the field by the offensive team are recognized using *multi-agent belief networks* which are assembled automatically on the basis of the respective *temporal structure descriptions*. A *multi-agent belief network* integrates the likelihood values obtained from the *temporal analysis functions* and calculates likelihood for the occurrence of a certain play. It is reported that the generated networks which recognize offense plays with eleven participating players typically contain at least 50 belief nodes and 40 evidence nodes. Still exact probability propagation remains feasible due to the binary state of all belief nodes and the network linking structure.

In their evaluation with 25 tracked plays using 10 *temporal play descriptions* the recognition accuracy is promising (21 out of 25 which corresponds to 84%). Most of the misclassifications that occur can be attributed to play descriptions which vary only in the motion of a single player. It should be noted that in general while the correct play is recognized as most probable option most of the time, alternative plays are still awarded high probabilities. Also the system has problems when faced with yet unknown plays where the desired result that none of the known plays is occurring is not obtained. Bobick and Intille claim that this problem is due to missing *visual networks* which still need to be constructed and as result a global model which is still 'to weak'. The recognition yields results soon after the start of a play. In terms of computational load the results are mixed as the system requires approximately 1 second of computation per frame per tested play yet on a 500 MHz Digital Alphastation. Even though faster hardware has become a standard in the meantime real-time application still seems critical.

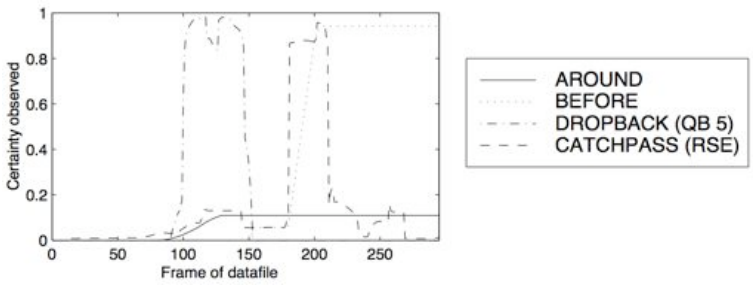


Figure 4.6: Sample likelihood curves returned visual belief networks for `dropback(QB5)` and `catchPass(RSE)` respectively superimposed with the temporal curves for `dropBack(QB5) before catchPass(RSE)` and `dropBack(QB5) around catchPass(RSE)` provided by dedicated temporal analysis functions. [IB99, p.5]

The approach by Bobick and Intille is interesting as it explicitly handles uncertainty throughout the complete recognition process from sensor input to high-level plays due to the utilization of Bayesian networks. Since facts are represented in terms of continuous-valued confidence time series, a transition to crisp, concrete statements that certain events occurred in a certain time interval within the observed dynamic scene seems difficult to obtain as appropriate confidence thresholds would need to be defined. For their own research Bobick and Intille can neglect such problems since in the end they are only interested in the identification of the most likely play.

4.1.8 FIPM: Football Interaction and Process Model

Beetz and colleagues are concerned with a computerized real-time analysis of human soccer games in order to support coaching activities. While comparable systems exist for offline game analysis, targeted primarily at strategy development and performance evaluation, such as [LW05, BCP+97] or ISAAC (cf. section 4.1.6 on page 45) the provision of automated online coach support that comprises real-time interpretation of sensor data, recognition and classification of ball actions and fast-action game analysis and assessment is seen as a new demanding challenge [BBG+06, BKL05, BSK+04, BKF04]. As a practical solution Beetz and colleagues introduce the *Football Interaction and Process Model* (FIPM) and associated software system. The stated long-term research goal is the applicability of the FIPM system for real-life soccer matches. Two strategies are discussed in order to obtain suitable input data regarding the whereabouts of both the players and the ball. First, the data could be accumulated using a microwave, real-time positioning system where wearable sensorics is attached to the players shin guards and the ball [BKL05]. Second, the data can be extracted from video footage of soccer matches by means of a dedicated image processing system. This approach has been outlined in [BBG+06]. Since both solutions are laborious pose noteworthy research problems in their own right, FIPM was developed in a first expansion stage using log records from matches of the RoboCup *2D Soccer Simulation League* played during the RoboCup 2003 championships as input.

While FIPM is meant to act as a real-time game analyzer, the system also comprises means to store analyzed games persistently in a dedicated game database and learn multiple game models by applying both data mining techniques and decision tree learning on subsets of the stored game database. The learned models are stored and can be used as background knowledge for the interpretation of agent/team behavior during the real-time analysis.

When a soccer match in analyzed by the real-time part of the system the first step in

the motion and action recognition process is an abstraction of the pre-processed sensor data in order to obtain a sufficiently concise motion representation that can be handled efficiently. A *motion interpreter* compiles *motion models* for each player on the soccer field and the ball. Each such model is comprised primarily of a sequence of *motion segments* $(m_1, m_2, \ldots, m_n)$. The idea is to segment the complete trajectory of movable objects obj over time into abstract representations of sub-trajectories, called *motion segments* defined as: $m_i \equiv (obj, t_1, t_2, p_1, p_2, f : \mathbb{T} \rightarrow \mathbb{R}^2)$ [BBG+06]. Each sub-trajectory is represented by its temporal extension $(t_1, t_2) \in \mathbb{T}$, the position of the respective object at these interval bounds $(p_1, p_2) \in \mathbb{R}^2$, and a linear function f which approximates the object position for each discrete moment $t : t_1 < t < t_2$. During a game the segmentation process iteratively checks whether the *motion segment* should be extended[25] or a new segment needs to be started. Thus, the *motion model* grows over the course of the game. Beyond the sequence of *motion segments* which is computed automatically, the *movement models* contain additional information about the occurrence of *instantaneous motion events* such as *ball contacts, ball out of bounds* and *referee whistles*. Beetz and colleague mention that these events are asserted manually by the system developer. This note implies that, while FIPM is capable of real-time processing, it is certainly not a *live* real-time game analyzer.

The compiled *motion models* are abstracted further into an *episode model* where a special focus lies upon episodes of ball movement. An episode is formally defined as a triple $(m_i, \ldots, m_j, se, fe)$ which describes a sequence of *motion sequences* and a *starting* as well as a *finishing event*. The recognition of *episode candidates* is performed with a finite automaton which initializes new candidate instances if it receives a new *motion segment* whose start coincides with the occurrence of a *ball-control event* for a certain player. The automaton then remains in an intermediate state accumulating new *motion segments* of the initializing player whenever necessary as long as it receives only further *ball-contact events* for the same player. Upon reception of a different event, it terminates and completes the *episode candidate* which now needs to be classified as one of the supported actions (*pass,shot* and *extended ball possession of a single player*). Beetz and colleagues claim that a complete classification based on manually crafted classification rules is error-prone which is for the most part due to the dynamics and complexity of soccer play leading to failed actions quite regularly. Thus only successful actions are recognized in a crisp fashion using rules. For failed actions the class membership is asserted with a subjective probability. In order to obtain the associated rules for the classification a decision tree learning is applied to derive classification rules automatically from manually labeled sets of pairs of episode features and episode class. It is stated that in order to apply the learning algorithm first the aforementioned feature set must be chosen appropriately. Beetz et al. use features which are encoded within the episode models such as the episode duration, the number of ball contacts within the episodes or whether or not the ball possession changed within the episode. That is, no additional information describing the episode context is used for the feature language. The second problem which is mentioned seems to be quite fundamental. While it is claimed that in principle in order to compile a suitable training set of feature-description to class associations the intentions of the acting agents needs to be known, in real life it is not. So intention is inferred manually by the developer during compilation of the training set.

FIPM classifies the episodes on the top-level into *pass. dribbling, shot* and *lost ball* where the latter only states that, something worked out not quite as intended by the action initiator. Using these four top-level classes the recognition can yield crisp results while making sure recognition is also both complete and un-ambiguous. If an action is recognized as a *lost ball* the learned decision tree can be used to yield an estimate for a more concrete classification such as *failed pass*. To conclude FIPM recognizes actions

25 which is true if the position prediction for the current moment yielded by $f(t)$ and the sensed position differ no more than a certain threshold values ϵ

with certainty on the top-level, can provide specialization estimates for the *lost ball* action based on a learned decision tree, and describes the action instances concisely in terms of the episode triples.

It is not mentioned whether or not the fact that the ball trajectory during recognized actions is maintained as sequence of *motion segments* is exactly exploited further in order to characterize the respective action better. However in [BSK+04] it is briefly mentioned that FIPM is capable of recognizing pass flavors such as short/long, deep/cross, fast/long. It also remains unclear in which way the *motion segments* of the players are used.

Having outlined FIPM's action recognition process a short treatment should be given to the game analysis capabilities. FIPM is comprised of five distinct layers building upon each other which each allow for distinct game analysis aspects. Those layers are: *position and motion, action, situation, tactical* and *assessment*.

With respect to game analysis, on the motion layer the data compiled during the action recognition allows for the calculation of movement profiles for each player. Based on a compilation of preferred player/team positions FIPM is eventually capable of an analysis of the teams' tactical lineup. On the action layer, statistics with regard to recognized, individual ball actions for each player allow for a compilation of player profiles and the identification of players with exceptional importance for the performance of a team. It is noted that FIPM is capable of a recognition of passing dyads. The further levels in the FIPM system are concerned with higher-order analysis. In the situation layer the identification of typical situations such as standard position-oriented attacks, counter attacks or kick and rush attacks is performed. Based on a representation of a game as a sequence of situation transitions FIPM can analyze situation specific losses of ball possession or the success rate of certain types of offensive play. On the highest level FIPM assesses global team tactics such as playing styles.

With regard to action models Beetz et al. state in [BKL05] that a "*key FIPM capability is the automatic acquisition of action models. Computer systems that analyze actions as complex and diverse as those in football must be equipped with rich action models*". Those action models come in several flavors. As an example *observation models* are employed to identify actions in the recognition process, *causal models* give clues as to the conditions under which distinct actions such as shots typically succeed, while *predictive models* can be used to determine with good accuracy if an action is likely to succeed or fail given a certain game situation and the experience introduced by the data set on which the learning was carried out. *Action Selection Models* can be used to predict the choice of action for soccer players given the respective situation. The bigger part of those models is used for statistical evaluation purposes in currently running games.

Beetz et al. claim potential use of their analysis system both for professional game analysis and broadcasting of sports events. The latter could be rendered individualized thus adopting the reports to the likes and interests of the respective user which for example can concentrate on the performance of certain favorite players and game situations. Moreover the continuous maintainance of a broad range of statistics as outline above provides a rich overall user experience.

4.2 Discussion

Now that the survey of related work has been compiled (cf. feature matrix in table 4.1 on page 53) the results thereof need to be discussed with regard to the motivation and demands that have been formulated in previous chapters of this thesis.

Considering the wide spectrum of approaches that were considered for the survey it seems obvious that, due to the respective concrete motivation to perform a flavor of spatio-temporal analysis, not all approaches are equally well suited for further contemplation

regarding applicability of proposed techniques in the context of the desired spatio-temporal real-time analysis of dynamic scenes from the RoboCup *3D Soccer Simulation League* by either agent type (coach or player) participating in simulated soccer games.

Approaches such as ISAAC (section 4.1.6) and FIPM (section 4.1.8) maintain a strong research focus on the compilation and maintainance of behavior/agent models and game statistics, thus enabling computer-aided game analysis by a human operator[26]. In principle, both above-mentioned approaches evaluate log records of completed soccer matches offline using data mining techniques. They focus on the deployment of efficient mechanisms for a retrieval of high-level information based on low-level sensor input. While ISAAC is completely dedicated to offline analysis due to its purpose as evaluator for agents acting and cooperating in an MAS for the respective human designer, FIPM, which is developed as analysis tool to support online coaching activities, is a hybrid system capable of an online interpretation of game input data based on the utilization of learned agent/game models to analyze an unfolding game.

Approaches as proposed by Miene (section 4.1.1 on page 29), Wendler (section 4.1.2 on page 33) and the Herzog et al. (section 4.1.3 on page 37) on the other hand are interested in the compilation of qualitative representations and action recognition. The goal of these approaches is a direct further use of the generated knowledge either by the observing agents within their simulation environment[27] or an observing agent such as an automated commentator. On that score these approaches bear noteworthy similarities with the approach developed in this thesis, since it has been stated that the soccer agents both on the field (players) and on the line (coach) should be equipped with an additional qualitative representation of their environment. The latter would lead to a comprehensive, hybrid knowledge base which incorporates both quantitative and qualitative aspects and whose increased amount of information can be exploited directly by the agents in order to adapt their style of play.

The approach by Miene is especially interesting due to its vertical integration as it discusses the whole work flow from the filtering of raw sensor values and time series transformation up to the recognition of complex multi-agent actions such as off-side traps. Miene's approach is comprehensive in another respect as well. It is not constrained to the recognition of actions. It rather distinguishes between situation properties, events and actions and recognizes the occurrence of all of these concepts based on the same formalism. Miene has found an intuitive way to describe complex concepts in terms constituent concepts and their temporal relations. While Wendler and Herzog et al. use similar descriptions as well, Miene's description is particularly interesting due to the fact, that no restriction is given for the constituent types. The approach by Miene bears some drawbacks too, as in its RoboCup related incarnation it has only been used in offline scenarios even though online-capability has been claimed. Also the detection of action concepts is exploited primarily as an a posteriori assessment thus limiting the approach's use for scenarios when the possibility to actually react is desirable.

Here, Wendler's approach offers an alternative as he suggests that actions should be attributed labels which indicate their state of execution. Wendler presents an incremental recognition strategy, borrowed from previous work by Herzog et al. which in principle allows for an early detection of actions and an delayed allocation of action attributes. The ability to deploy an incremental recognition strategy comes at a cost as the action specifications used by Wendler are less expressive than those used by Miene. This becomes clear, when the work of Herzog et al. is considered, where both formalisms (Miene and Wendler) are used side by side and it is shown how both relate to each other. Particularly, Herzog et al. state that both takes on the subject have their respective strengths, the formalism

26 where this operator adopts the position of either a MAS developer or coach

27 Both Miene and Wendler use the 2D online coach for the execution of the analysis. Miene's coach logs the course of the game incrementally, Wendler's coach automatically generates and deploys opponent models.

by Miene being better suited for further reasoning tasks, the formalism by Wendler for incremental recognition.

Brought forward to Miene's approach it is conceivable that a satisfactory approximation of quasi-incremental in-progress recognition is feasible, once the target concepts such as dribblings are decomposed into distinguishable constituents whose recognition fuels an incremental recognition of the target concept by means of recognition updates (i.e. action lengthening).

Wendler suggests an object-oriented implementation of the proposed action recognition approach which is custom-tailored to suit his respective requirements with regard to detection scheme, storage and retrieval. However the lecture of both [Wen03] and the publications of the VITRA group suggests that, at least to some degree, the expressiveness of the concept descriptions which are immediatly applied in the recognition process is traded in order to obtain real-time applicability. The research work by Gehrke and colleagues [Geh05] offers an effective alternative to avoid a decline of expressiveness a.) in the concept specification on the design level and b.) in the conveyance thereof into the respective implementation. Gehrke et al. propose the employment/adoption of a well-designed general purpose reasoning engine such as XSB-Prolog which, due to its support for declarative programming, allows for the use of the concept specification as actual part of the program leaving the procedural aspects of concept detections to the reasoning system as far as possible. As it is shown in [Geh05] that online applicability and the use of a dedicated reasoner is accordable in the domain of intelligent vehicles, an advancement in this direction seems promising for soccer simulation as well.

Wendler introduces the idea to recognize not only the basic action but also *concept-or particularly action characteristics* and thus arrives at a more comprehensive description of the observed dynamic scenes. While Wendler mixes quantitative parameters[30] into otherwise qualitative concepts which is not desired for this thesis, he provides an interesting way to exploit the given set of readily available qualitative ground data efficiently and to introduce increased diversification of the concepts also wrought in related approaches.

None of the three approaches mentioned above explicitly addresses the problem of sensor noise which is to be expected once imperfect vision data is used as starting point for a spatio-temporal analysis of dynamic scenes. Steinbauer et al. (section 4.1.4 on page 42) provided an interesting contribution how sensor noise can be dealt with at the level of abstraction to symbolic, qualitative values borrowing from the field of electrical engineering. The application of hysteresis is proposed and shown to work well under real-world conditions on real hardware. Since this thesis aims to deal with sensor noise such a system seems to be valuable.

Intille and Bobick (section 4.1.7 on page 46) propose an approach to action recognition on the agent and team level based upon Bayesian networks. The approach is interesting since it strives to recognize both situation properties and actions similar in principle to those recognized by Miene while uncertainty is handled explicitly throughout the whole recognition process. As a result the clear-cut validity intervals for recognized concepts which are common to the traditional approaches by Miene, Wendler and Herzog et al. are substituted with probability distributions over time. The concept is interesting as Bayesian networks are known to handle uncertainty and missing data. However the specification of the large amount of required belief networks which is critically discussed by the authors as possible development bottleneck seems to be intractable in the context of a diploma thesis. In [BBG+06, p.8] and [BKF04, p.14], Beetz et al. express their doubts as whether the approach proposed by Intille and Bobick can be transfered from the application domain of American football, which is a highly structured game of temporally distinct specific plays

28 Time Series Transformation
29 Predicate hystheresis was introduced in section 4.1.4 in the work of Steinbauer et al.
30 such as motion vectors

Approach	Application Domain	Usage Scenario (offline,online)	Data Model (categorical,fuzzy/bayes)	Noise-Handling	Detected Concepts	Concept Model	Concept Characteristics	Detection Method	Algorithms
Miene (4.1.1)	RoboCup , *2D Soccer Simulation*	✓, –	✓, –	tst[28]	facts, events, actions(s,m), act-sequences	taxo-nomic	success/failure	declarative (flexible time constraints)	custom: spatio-temporal logic reasoning, pattern-matching
Gehrke (4.1.1)	Traffic Sce-narios	✓,≈	✓, –	tst	facts, events, actions(s)	taxo-nomic	–	declarative (flexible time constraints)	Prolog-based spatio-temporal logic reasoning, pattern matching
Wendler (4.1.2)	RoboCup , *2D Soccer Simulation*	✓, ✓	✓, –	–	actions(s,m), act-sequences	flat	success/failure, action charac-teristics	procedural (strict time constraints)	custom: OO, managed incre-mental tracking
VITRA (4.1.3)	Soccer	✓, ≈	✓, –	–	event, ac-tions(s), act-sequences	flat	–	hybrid	temporal constraint solving (offline); custom: OO, man-aged incremental tracking (on-line,custom)
Kaminka (4.1.5)	RoboCup , *2D Soccer Simulation*	✓, –	✓, –	–	actions(s)	flat	–	procedural (strict time constraints)	custom: OO, managed incre-mental tracking
Intille (4.1.7)	American Football	✓,≈	–,✓	bayesian	actions(m)	flat	–	bayes nets	Bayesian updating/queries
FIPM (4.1.8)	Human Soc-cer	✓,≈	✓,✓	?	motion episodes, actions(s)	flat	success/failure	declarative	custom OO, incremental tracking; decision tree queries
Thesis	RoboCup , *3D Soccer Simulation*	✓,✓	✓, –	tst, hysthe-resis[29]	facts, events, actions(s,m), act-sequences	taxo-nomic, causal rela-tions	success/failure, action charac-teristics	declarative (flexible time constraints)	Prolog-based spatio-temporal logic reasoning, (incremental) pattern matching

Table 4.1: Feature Matrix of reviewed analysis approaches, Comparison with Thesis Approach

with highly-detailed player roles where the failure of plays is considered an exception, to the soccer domain. The baseline of the expressed criticism is that soccer is rather continuous and dynamic with respect to its game flow with considerable emphasis on reactive behavior traits. Plays are far less differentiated and actions fail on a regular basis.

5

A Conceptual Approach for 3D Soccer Simulation

The fifth chapter of this thesis discusses the conceptual foundation for the concrete implementation of a flexible framework for spatio-temporal analysis of dynamic scenes in the RoboCup *3D Soccer Simulation League*. From a high-level point of view, two fundamental scopes of duties can be distinguished for such an analysis. First, pre-compiled raw data from a readily available quantitative knowledge base is transformed by means of *qualitative abstraction* into a basic pool of atomic, qualitative facts. Second, the incremental compilation of an appropriate fact pool is succeeded by a data-driven detection of extended motion incidences spanning events, actions and action sequences by means of a matching process for spatio-temporal patterns.

Section 5.1 is concerned with the description of the qualitative abstraction. First, the scope of concrete qualitative predicates and their respective class codomains is worked out in section 5.1.1. Subsequently, section 5.1.3 outlines the conceptual implementation of a categorical classification with flexible bounds. Afterwards, section 5.1.4 comprises a short excursus into formalisms for the representation of time. Based on that foundation, in section 5.1.5 the qualitative predicates are associated with a temporal extension such that the qualitative abstraction yields spatio-temporal facts as a result. The field of qualitative abstraction is concluded with the introduction of a focus heuristic to constrain the pool of produced facts to a relevant subset.

Section 5.2 works out suitable formalisms for the representation of extensive motion situation based on suggestions from Allen's interval temporal logic [All84, All83, AF94]. Seizing ideas from Wendler an colleagues [Wen03, Mül02] it is shown how characteristic traits of event/actions can be represented formally thereby obtaining an increased expressiveness of the detection results.

Building upon the introduced specification means, section 5.3.1 to section 5.3.3 consequently develop the pool of concrete motion patterns for the RoboCup *3D Soccer Simulation League*, working from comparatively simple patterns located on the event level towards highly-structured action sequences.

Once the complete set of discernible motion patterns has been specified, section 5.4 describes the actual detection of concrete motion incidences and develops a selective and, as a consequence, less time-consuming detection workflow as compared with a naive brute

force alternative. The former thereby exploits incidence relationships amongst motion incidences.

5.1 Qualitative Abstraction

5.1.1 Identifying the Pool of Qualitative Ground Predicates

The first step in the process of qualitative abstraction is the identification of a suitable subset of possible qualitative predicates and their respective set of equivalence classes given the available set of time series data providing the input for segmentation and classification.

Description of required Raw Input Data

The principal fluent quantitative data for movable/moving objects on the simulated soccer pitch due to be fed into the qualitative abstraction as stated in section 3.3.2 on page 22 is specified as follows:

Object Locations :

$pos_{cart}(obj, t_i) = (x_t^{obj}, y_t^{obj}, z_t^{obj})$: the location the movable objects[1] (ball, players) in Cartesian coordinates ($\mathbb{R}^3$) within the simulation environment.

Object Motions :

$mot_{cart}(obj, t_i) = (x_t^{obj}, y_t^{obj}, z_t^{obj})$: the motion vector the movable objects (ball, players) in Cartesian coordinates ($\mathbb{R}^3$). For each movable object motion data is available directly due to the fact that the raw data is compiled by agents within the soccer simulation rather than extracted from log files [Mie04a, pp.47].

The motion data can be transformed immediatly to an alternate representation in 3D Polar coordinates as $mot_{pol}(obj, t_i) = (\theta_t^{obj}, \phi_t^{obj}, \rho_t^{obj})$ where the *distance* θ_t^{obj} refers to the motion velocity, the *azimuth* ϕ_t^{obj} refers to the motion direction in the xy-plane and, finally, the *elevation* ρ_t^{obj} refers to the motion inclination/declination.

Play Mode :

$playmode(t) = mode_t$: the momentary modus of play in the game being simulated, specified as scalar ($\mathbb{R}$).

The aforementioned principal data about respective location and motion of a single object provides a suitable starting point for the compilation of immediatly manifest zero- (denoted by $\mathbb{P}^0$) and univalent qualitative predicates[2] (denoted by $\mathbb{P}^1$). The compilation of bivalent predicates (denoted by $\mathbb{P}^2$) presupposes in addition the common consideration of location data for pairings of spatially related objects in a dynamic scene.

Spatial Orientation Relation :

$\Delta_{cart}^{pos}(obj_1, obj_2, t) = pos_{cart}(obj_2, t) - pos_{cart}(obj_1, t)$: the difference vector ($obj_1 \xrightarrow{to} obj_2$) of the location of two scene actors in Cartesian coordinates ($\mathbb{R}^3$).

1 The term 'movable object' constitutes a discrimination from static, immovable objects in a dynamic scene (environment → goal posts). The distinctive feature of 'movable objects' is the principle ability to change their location over time, either due to own initiative (agents → soccer players) or as a consequence of external action (passive objects → ball)

2 The method of counting that is applied the distinction of qualitative predicates in equivalence classes refers to the number of objects a predicates alludes to.

As for the velocity, $\Delta^{pos}_{cart}(obj_1, obj_2, t)$ can be transformed immediatly to a representation in Polar coordinates: $\Delta^{pos}_{pol}(obj_1, obj_2, t) = (\theta_{\Delta(1,2)}, \phi_{\Delta(1,2)}, \rho_{\Delta(1,2)})$ where the *distance* $\theta_{\Delta(t)}$ refers to the distance between the centers of obj_1, obj_2. The *azimuth* $\phi_{\Delta(t)}$ refers to the spatial orientation relation of obj_2 with respect to the reference obj_1 in the ground plane. Finally, the *elevation* $\rho_{\Delta(t)}$ refers to the spatial orientation relation of obj_2 with respect to obj_1 along the height dimension.

Beyond that, the combined contemplation of quantitative input data for two successive points of measurement provides a suitable input for the classification of value trends.

Motion Alteration & Acceleration :

$\Delta^{vel}_{cart}(obj, t_{i-1}, t_i) = vel_{cart}(obj, t_i) - vel_{cart}(obj, t_{i-1})$ describes the alteration of the motion of a scene actor in two consecutive points of measurement as difference vector in Cartesian coordinates ($\mathbb{R}^3$).

$\Delta^{vel}_{cart}(obj, t_{i-1}, t_i)$ can be transformed immediatly to a representation in polar coordinates $\Delta^{vel}_{pol}(obj, t_{i-1}, t_i) = (\theta_{\Delta(t)}, \phi_{\Delta(t)}, \rho_{\Delta(t)})$. The *distance* $\theta_{\Delta(t)}$ attracts particular interest as it represents the acceleration/deceleration of the scene actor obj in $\Delta(t_{i-1}, t_i)$.

Association of Time Series Data to Qualitative Predicates

For the concrete qualitative abstraction, implemented for this thesis, the following subset of the presented raw data is chosen as input for the process of qualitative abstraction. Due to the fact that the data is processed gradually in consecutive classification passes, a time series notation is used which is based on a formalization by Boronowski [Bor01, p.68].

Definition 5.1 (Time Series) *A time series is a finite tuple set* $Z = \{\langle t_1, y_1\rangle, \langle t_2, y_2\rangle, \ldots \langle t_n, y_n\rangle\}.n \in \mathbb{N}$ *whose elements* $\langle t_i, y_i\rangle \in Z$ *constitute a value run for a function* $f(t_i) = y_i$ *over a finite set of successive time points* $t_i \in T$. y_i *can comprise both a single scalar value or a collection of scalar values such as* a_i, b_i. *In the first case, the time series is* univariate, *in the latter* multivariate. □

Each family of time series – one for each single movable object or pairing of movable objects – is associated with its qualitative target predicate. For each such predicate, the codomain in terms of the set of equivalence classes $\mathbb{SYM}_p$ is specified.

X,Y-Position : $pos_{x,y}(obj) = \{\langle t_1, x^{obj}_1, y^{obj}_1\rangle, \ldots, \langle t_n, x^{obj}_n, y^{obj}_n\rangle\}$. $n \in \mathbb{N}$

These multivariate time series are used for the successive classification of movable object *residence regions* with the univalent predicate

$\mathbf{in\_region(obj, sym_{reg})} \in \mathbb{P}^1$. $sym_{reg} \in \mathbb{SYM}_{regions}$.

for the discrete time points $t_1, \ldots, t_n$.

Height $pos_z(obj) = \{\langle t_1, z^{obj}_1\rangle, \ldots, \langle t_n, z^{obj}_n\rangle\}$. $n \in \mathbb{N}$

These univariate time series are used for the successive classification of movable object's *height above ground level* with the univalent predicate

$\mathbf{z\_position(obj, sym_{height})} \in \mathbb{P}^1$. $sym_{height} \in \mathbb{SYM}_{height}$

for the discrete time points $t_1, \ldots, t_n$. As the soccer players within the simulation reside on ground level during the whole simulation this predicate is only meaningful for the soccer ball.

Velocity $vel(obj) = \{\langle t_1, vel_1^{obj}\rangle, \ldots, \langle t_n, vel_n^{obj}\rangle\} \,.\, n \in \mathbb{N}$
where $vel_t^{obj} \equiv \theta_t^{obj}$ in $mot_{pol}(obj, t)$

These univariate time series are used for the successive classification of movable object *velocities* with the univalent predicate

$\mathbf{velocity(obj, sym_{vel})} \in \mathbb{P}^1 \,.\, sym_{vel} \in \mathbb{SYM}_{velocity}$

for the discrete time points $t_1, \ldots, t_n$.

Motion Direction (X,Y)-plane :

$mot\_dir_{x,y}(obj) = \{\langle t_1, mot\_dir_1^{obj}\rangle, \ldots, \langle t_n, mot\_dir_n^{obj}\rangle\} \,.\, n \in \mathbb{N}$
where $mot\_dir_t^{obj} \equiv \phi_t^{obj}$ in $mot_{pol}(obj, t)$

These univariate time series are used for the successive classification of a movable object's *motion direction* in the ground plane with the univalent predicate

$\mathbf{motion\_dir(obj, sym_{dir})} \in \mathbb{P}^1 \,.\, sym_{dir} \in \mathbb{SYM}_{dirs}$

for the discrete time points $t_1, \ldots, t_n$.

Rise/Fall $mot\_dir_z(obj) = \{\langle t_1, mot\_dir_1^{obj}\rangle, \ldots, \langle t_n, mot\_dir_n^{obj}\rangle\} \,.\, n \in \mathbb{N}$
where $mot\_dir_t^{obj} \equiv \rho_t^{obj}$ in $mot_{pol}(obj, t)$

These univariate time series are used for the successive classification of the *inclination/declination* of the motion of a movable object with the univalent predicate

$\mathbf{z\_position\_trend(obj, sym_{rise})} \in \mathbb{P}^1 \,.\, sym_{rise} \in \mathbb{SYM}_{rise}$

for the discrete time points $t_1, \ldots, t_n$. Analogous to the state of affairs for $z\_position(obj, sym_{height})$, this predicate is also only meaningful for the ball.

Distance $dist(obj_1, obj_2) = \{\langle t_1, dist_1^{(obj_1, obj_2)}\rangle, \ldots, \langle t_n, dist_n^{(obj_1, obj_2)}\rangle\} \,.\, n \in \mathbb{N}$
where $dist_t^{(obj_1, obj_2)} \equiv \theta_{\Delta(1,2)}$ in $\Delta_{pol}^{pos}(obj_1, obj_2, t)$

These univariate time series are used for the successive classification of the symmetric distance between two movable objects with the bivalent predicate

$\mathbf{distance(obj_1, obj_2, sym_{dist})} \in \mathbb{P}^2 \,.\, sym_{dist} \in \mathbb{SYM}_{dist}$

for the discrete time points $t_1, \ldots, t_n$.

Spatial Orientation (X,Y-plane) :

$$s\_orientation(obj_1, obj_2 = \langle t_1, s\_orientation_1^{(obj_1, obj_2)}\rangle, \ldots \langle t_n, s\_orientation_n^{(obj_1, obj_2)}\rangle\}. \, n \in \mathbb{N}$$

where $s\_orientation_t^{(1,2)} \equiv \phi_{\Delta(1,2)}$ in $\Delta_{pol}^{pos}(obj_1, obj_2, t)$.

These univariate time series are used for the classification of the spatial orientation relation of a secondary movable object with respect to a reference object with the bivalent predicate

$\mathbf{s\_orientation(obj_1, obj_2, sym_{dir})} \in \mathbb{P}^2 \,.\, sym_{dir} \in \mathbb{SYM}_{dirs}$

for the discrete time points $t_1, \ldots, t_n$.

Acceleration $accel(obj) = \{\langle t_1, accel_1^{obj}\rangle, \ldots, \langle t_n, accel_n^{obj}\rangle\} \,.\, n \in \mathbb{N}$
where $accel_t^{obj} \equiv \theta_{\Delta(t)}$ in $\Delta_{pol}^{vel}(obj, t_{i-1}, t_i)$

These univariate time series are used for the classification of movable object *acceleration/deceleration* with the univalent predicate

$\mathbf{acceleration(obj, sym_{acc})} \in \mathbb{P}^1 \,.\, sym_{acc} \in \mathbb{SYM}_{acc}$

for the discrete time points $t_1, \ldots, t_n$.

Play Mode $playmode = \{\langle t_1, mode_1\rangle, \ldots, \langle t_n, mode_n\rangle\} \,.\, n \in \mathbb{N}$

Finally, this univariate time series whose values already constitute symbolic values provides the input for the compilation of a suitable play mode representation with the zerovalent predicate

$\mathbf{playmode(sym_{mode})} \in \mathbb{P}^0 \,.\, sym_{mode} \in \mathbb{SYM}_{mode}$

for the discrete time points $t_1, \ldots, t_n$.

Breakdown of the Codomains for the Chosen Qualitative Predicates

Now that the association from quantitative time series to qualitative predicates used for the remainder of this thesis has been outlined, the concrete sets of equivalence classes $\mathbb{SYM}_{type}$ for the respective predicates are elaborated. Moreover, the fundamental, concrete interval to class assignments are specified. It should be noted, that the implementation of a *top quality* choice of both classes and intervals associations for classification is a task worthy of further research efforts in its own right (e.g. [MW06, SFL06, Mus00, GLH04, HCDF95]). Thus, for the scope of the work presented here, it is satisfactory to compile a sufficient basis which supports the primarily focused analysis of dynamic scenes based upon the qualitative abstraction. In order to come up with plausible choices for the equivalence classes and interval partitions, domain expertise was exploited which has been accumulated in the artificial intelligence working group at the Center of Computing Sciences (TZI) at the University of Bremen in years of involvement in the RoboCup Simulation Leagues (both 2D and 3D).

$\mathbb{SYM}_{\mathbf{velo}}$ – Velocity The set of equivalence classes for the description of velocities is specified as $\mathbb{SYM}_{velo} = \{rest, very\_slow, slow, moderate, fast, very\_fast, beam\}$ following the graduation suggested by Miene for the RoboCup *2D Soccer Simulation* [Mie04a, p.76].

Two domain-specific variations from Miene's graduation proposal exist, however. First, the class *rest* entails both absolute motionlessness as well as virtually indiscernible motion[3]. Second, due to the peculiarity of the RoboCup 3D Soccer Server to beam scene actors to enforce the rapid compliance to the rules of the game, an additional equivalence class *beam* has been introduced in order to distinct pseudo-motion via the beam effect from very fast normal movement of scene actors. However, it should be noted, that it is not possible to safely distinguish every beam action but only long distance beams that cannot be possibly interpreted as effect of the natural motion of a movable object.

The concrete partition of the open codomain into intervals and the interval to class mapping (IC-mapping $\mathbb{CS}_{vel}$) is dependent on the actor type as players and the ball have different velocity ranges such that a ball which is moving $fast$ may be significantly faster absolutely than a player moving $fast$ as well. This distinction for classes of movable objects can presupposes a type sensitive comparison of qualitative velocities which is not required in the scope of this thesis. In [Geh05, p.79–82], Gehrke considers this problematic as well for actors in the traffic domain.

player $\mathbb{CS}_{vel} = \{\langle 0, 0.05, rest\rangle, \langle 0.05, 0.15, very\_slow\rangle, \langle 0.15, 0.3, slow\rangle, \langle 0.3, 0.6, moderate\rangle, \langle 0.6, 0.9, fast\rangle, \langle 0.9, 2.0, very\_fast\rangle, \langle 2.0, \infty, beam\rangle\}$

3 and thus essentially relaxes Miene's notion of *rest* due to the fact that in the RoboCup 3D Soccer Simulation both agents and particularly the ball seldom come to a complete stop or such stops may remain unobservable due to noise in the perception

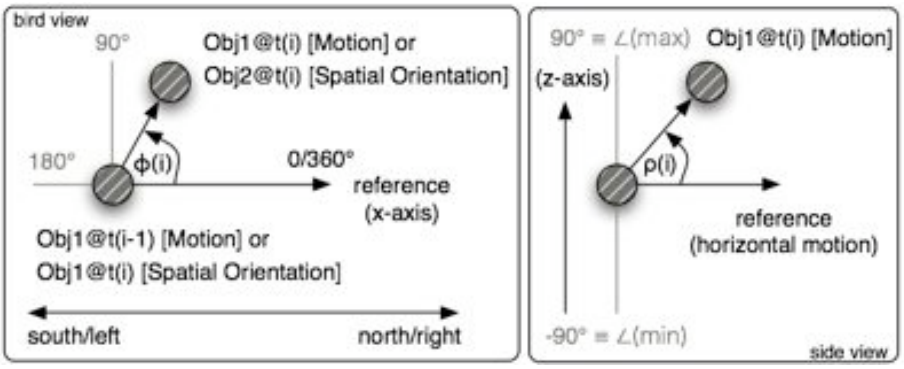


Figure 5.1: Determination of motion/spatial orientation- and inclination/declination raw data.

ball $\mathbb{CS}_{vel} = \{\langle 0, 0.07, rest\rangle, \langle 0.07, 0.25, very\_slow\rangle, \langle 0.25, 0.5, slow\rangle, \langle 0.5, 0.9, moderate\rangle, \langle 0.9, 1.3, fast\rangle, \langle 1.3, 5.0, very\_fast\rangle, \langle 5.0, \infty, beam\rangle\}$

For the $sym_{vel} \in \mathbb{SYM}_{velo}$ the total order relation $\leq$ is defined such that $rest \leq very\_slow \leq slow \leq moderate \leq fast \leq very\_fast \leq beam$.

$\mathbb{SYM}_{\mathbf{accel}}$ – Acceleration The set of equivalence classes for the description of acceleration is reduced to the minimum number of classes required to distinguish deceleration from constant velocity and acceleration. Analogous to the qualitative velocities pseudo acceleration due to beam effects is accommodated for with a dedicated additional equivalence class.

Thus, $\mathbb{SYM}_{accel} = \{decreasing, stable, increasing, beam\}$.

The concrete partition of the open codomain into intervals and the IC-mapping $\mathbb{CS}_{accel}$ is dependent on the actor type as players and the ball have a greater acceleration potential than players.

player $\mathbb{CS}_{accel} = \{\langle -\infty, -0.08, decreasing\rangle, \langle -0.08, 0.08, stable\rangle, \langle 0.08, 2.0, increasing\rangle, \langle 2.0, \infty, beam\rangle\}$

ball $\mathbb{CS}_{accel} = \{iv - \infty, -0.08, decreasing, \langle -0.08, 0.08, stable\rangle, \langle 0.08, 3.0, increasing\rangle, \langle 3.0, \infty, beam\rangle\}$

$\mathbb{SYM}_{\mathbf{dirs}}$ – Direction Qualitative directions are used to describe both the motion direction of a reference object and the spatial configuration of a second object with respect to the reference object from an egocentric point of view.

The description is dependent on the chosen frame of reference (FoR, cf. [Geh05, p.12] for an overview). With an intrinsic FoR, inherent properties of a scene actor such as its front side or line of sight define an object-specific reference axis. With respect to this reference, the 360° around the reference object can be partitioned in a set of equivalence classes and associated with linguistic terms such as *front, left, back*. The grain size of the division is dependent on the requirements of the respective application scenario.

In other scenarios, an extrinsic FoR is predetermined by external factors in the environment such as a fixed north pole such that a global object-independent reference axis exists. A common practice for specification of directions in an extrinsic FoR is via a compass rose aligned in a certain way with the global reference axis. Possible graduations comprise two (i.e. *north, south*), four (i.e. *north, east, south, west*, the

four primal points of the compass) and eight classes (i.e. *north, northeast, east, southeast, south, southwest, west and northwest*) [Her94, Geh05].

For the concrete application scenario, the RoboCup 3D Soccer Server predetermines a global, extrinsic FoR via its localization of the left/right half of the soccer pitch with the reference axis being parallel to the touch lines and showing towards the right goal. A symmetric, homogeneously partitioned compass rose with

$$\begin{aligned}\mathbb{SYM}_{dirs} &= \{north, northeast, east, southeast, south,\\ &\quad southwest, west, northwest\}\end{aligned}$$

is aligned with respect to the reference axis as shown in figure 5.1 and superimposed on the soccer region partonomy in figure 5.2 on page 64.

The choice of eight direction classes is motivated by the desire to determine the directedness of, especially forward-oriented, pass play with appropriate grain size such that diagonal pass play (which corresponds to steep passes when seen from the perspective of the acting team) can be distinguished from forward/backward- or sideways-oriented play.

The concrete partition of the cyclic codomain into intervals and the IC-mapping $\mathbb{CS}_{dirs}$ is specified as follows:

$$\begin{aligned}\mathbb{CS}_{dirs} &= \{\langle 22.5, 67.5, northwest\rangle, \langle 67.5, 112.5, west\rangle, \langle 112.5, 157.5, southwest\rangle,\\ &\quad \langle 157.5, 202.5, south\rangle, \langle 202.5, 247.5, southeast\rangle, \langle 247.5, 292.5, east\rangle,\\ &\quad \langle 292.5, 337.5, northeast\rangle, \langle 337.6, 22.5, north\rangle\}\end{aligned}$$

$\mathbb{SYM}_{dist}$ – Distance As for the directions, the description of the qualitative distance between two scene actors is also dependent on the FoR. As the distance is independent from the object properties such as the spatial extension due to the fact that the scene actors all possess spherical physical bodies and can be conceptually thought of as point-based, an absolute FoR is a given for the concrete application scenario.

The grain size of the equivalence classes for the distance is chosen to suit the particular requirements of the RoboCup 3D Soccer Simulation an can be specified as $\mathbb{SYM}_{dist} = \{touch, very\_close, close, medium, far\}$. The denotation of the first distance class as *touch* is a bit misleading as it stands for immediate proximity of two actors which entails immediate contact as well as extremely close proximity. Within touch distance, players can handle the ball or interact physically with other agents on the field. A distinct distance class $very\_far$ is neglected due to the attentional focus of the qualitative abstraction described in section 5.1.6 on page 82.

For the $sym_{dist} \in \mathbb{SYM}_{dist}$ the total order relation $\leq$ is defined such that $touch \leq very\_close \leq close \leq medium \leq far$.

The concrete partition of the open codomain into intervals and the IC-mapping $\mathbb{CS}_{dist}$ is specified as follows:

$$\begin{aligned}\mathbb{CS}_{dist} &= \{\langle 0, 0.55, touch\rangle, \langle 0.55, 1.15, very\_close\rangle, \langle 1.15, 2.35, close\rangle,\\ &\quad \langle 2.35, 5.05, medium\rangle, \langle 5.05, \infty, far\rangle\}\end{aligned}$$

The interval partition is compliant with cognitively motivated basic principles for the specification of distance systems that have been introduced by Hernández et al. in [Her94], namely the *monotonicity* principle[4] and the principle of *range restriction*[5] (cf. [Geh05, p.14–15] for an introduction to distance systems).

4 The monotonicity principle demands that the interval associated with a successive distance class must always be at least as big as the interval associated with the momentary distance class.

5 This principle demands that each interval associated with a distance class is at least as big as the distance from the origin to the lower bound of the interval

$\mathbb{SYM}_{\mathbf{rise}}$ – Inclination/Declination As the application scenario for the qualitative abstraction is the RoboCup 3D Soccer Simulation, a set of equivalence classes is required that describe the vertical motion trend of scene actors, in particular the ball, in terms of its inclination or declination behavior. As for the directions in the ground plane, the 3D Soccer Server provides an extrinsic FoR via the simulated gravitational forces the physical objects in the simulation environment are subjected to. The codomain of input values is $[-90^\circ, \ldots, 90^\circ]$ where -90° corresponds to a straight fall to the ground[6] and 0^{deg} corresponds to a motion parallel to the ground[7] (cf. figure 5.1 on page 60).

The set of equivalence classes is specified as:

$$\begin{aligned}\mathbb{SYM}_{rise} \quad = \quad & \{falling\_full, falling\_medium, falling\_light, stable,\\ & rising\_light, rising\_medium, rising\_full\}\end{aligned}$$

The grain size has been chosen such that through the inspection of the inclination of a kicked ball, it is possible to infer whether the ball has been kicked flat on the ground, low, medium or high. The distinct equivalence classes for falling motion have been introduced to retain symmetry.

The concrete partition of the open codomain into intervals and the IC-mapping $\mathbb{CS}_{rise}$ is specified as follows:

$$\begin{aligned}\mathbb{CS}_{rise} \quad = \quad & \{\langle -90.0, -60.0, falling\_full\rangle, \langle -60.0, -30.0, falling\_medium\rangle,\\ & \langle -30.0, -4.0, falling\_light\rangle, \langle -4.0, 4.0, stable\rangle, \langle 4.0, 30.0, rising\_light\rangle,\\ & \langle 30.0, 60.0, rising\_medium\rangle, \langle 60.0, 90.0, rising\_full\rangle\}\end{aligned}$$

$\mathbb{SYM}_{\mathbf{height}}$ – Height The set of equivalence classes for the description of a scene actor's height above ground level is specified as $\mathbb{SYM}_{height} = \{ground\_level, low, medium, high\}$. The grain size has been chosen as a compromise such that a suitable class distinction is a given which is not overly fine but still allows to coarsely reconstruct ball motion in terms of flight curves (i.e $ground\_level \rightarrow low \rightarrow medium \rightarrow low \rightarrow ground\_level \rightarrow \ldots$).

The concrete partition of the open codomain into intervals and the IC-mapping $\mathbb{CS}_{height}$ is specified as follows:

$$\begin{aligned}\mathbb{CS}_{height} \quad = \quad & \{\langle 0, 0.35, ground\_level\rangle, \langle 0.35, 1.5, low\rangle,\\ & \langle 1.5, 3.0, medium\rangle, \langle 3.0, \infty, high\rangle\}\end{aligned}$$

$\mathbb{SYM}_{\mathbf{region}}$ – Region The set of equivalence classes for the description of scene actor residence realizes a spatial partonomy of the soccer environment which leans both on regions and markings that determined by the rules of the game for the RoboCup 3D Soccer Simulation [FIF06, pp.6, law1, the field of play] and are influenced considerably by tactical evaluation interests.

On the coarsest level, the region classification must allow for a distinction between outer- and inner field. For tactical evaluation of the course of the game it must be further possible to distinguish between wing positions or center positions.

In [Mie04a, p.89], Miene suggests a twofold distinction of the respective wing (i.e. *westwing*) and, explicitly accommodating for the special relevance of the penalty area, a fourfold distinction of the central area confined by the wings (i.e. *center*) via a subdivision at the halfway line. For the wings the subdivision leads to distinctions

6 i.e. there is no horizontal motion part
7 i.e. there is no vertical motion part

such as ($westwing\_south$ vs. $westwing\_north$). For the center area the distinctions are ($penaltyArea\_south$, $center\_south$ vs. $center\_north$, $penaltyArea\_north$). Miene adds the goal areas which are located within the respective penalty area.

The region classification, implemented for this thesis, is based on the partonomy from Miene. However, the partonomy is refined such that a central area is defined not only along the field width but also along the field length as shown in figure 5.2 on the following page. The finer grain size in the center of the field was introduced due to the observation during recent international RoboCup championships that the game is often is the most active in the middle field as both teams try to bring forward the ball deeper into the field half of the adversary team. Due to the implemented set of detectable motion patterns (cf. section 5.3.2 on page 95) the goal area can be neglected.

The resulting set of equivalence classes for the residence regions can now be specified as:

$$\begin{aligned}\mathbb{SYM}_{region} = \{&westwing\_south, westwing\_center, westwing\_north,\\ &penaltyArea\_south, center\_south, center, center\_north,\\ &penaltyArea\_north, eastwing\_south, eastwing\_center,\\ &eastwing\_north, outer\_field\}\end{aligned}$$

The quantitative boundaries of the respective regions classes are shown in figure 5.2 on the following page. As for the classification of regions, a multivariate time series (x,y-positions) is processed which demands a two-tier classification process outlined as final part of section 5.1.3, rather than a direct mapping, the specification of a concrete $\mathbb{CS}_{region}$ is delayed for the time being.

$\mathbb{SYM}_{mode}$ – Play Mode The set of possible play modes that can occur within the RoboCup 3D Soccer Simulation is specified as:

$$\begin{aligned}\mathbb{SYM}_{mode} = \{&before\_kickoff, playon, gameover, kickoff\_left,\\ &kickoff\_right, kickin\_left, kickin\_right, goal\_left,\\ &goal\_right, goalkick\_left, goalkick\_right, cornerkick\_left,\\ &cornerkick\_right, offside\_left, offside\_right\}\end{aligned}$$

Before the time series data is fed into the classification process outlined in section 5.1.3 on the next page in each qualitative abstraction pass, suitable time series transformations [Bor01] can be applied in order to remove outliers in the value run of a time series or suppress noise seizing smoothing operators such as *weighted* or *exponential moving average* filters.

5.1.2 Derived Qualitative Ground Predicates

Based on the pool of qualitative ground predicates that has been introduced hitherto, further predicates can be derived which bear noteworthy importance for the compilation of the extensive motion patterns developed later in this chapter in section 5.3. These predicates are concerned with the ball control situation (viz. *ball free*, *exclusive ball control* and *struggle for ball supremacy*) and can be computed for a certain qualitative abstraction pass via an evaluation of the $z\_position(\ldots)$ and $distance(\ldots)$ ground predicates.

To begin with, the ball can only be controlled by the soccer players on the pitch whenever it is not airborne but rather located on ground level, expressed as $z\_position(\text{ball}, \text{sym}_{\text{height}})$. Whenever the ball is on the ground, the number of agents for which the condition $distance(\text{ball}, player, \text{touch})$ is satisfied determines the state of affairs with respect to

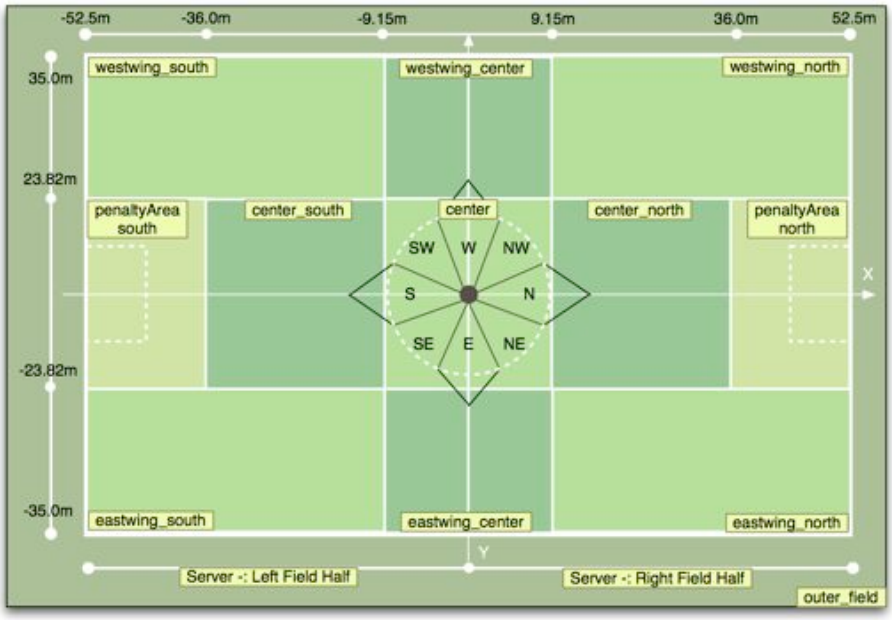


Figure 5.2: Schematic overview of the region partonomy which is used for the classification of residence regions with information of the respective region boundaries. Superimposed on the region overview is the global compass rose that is aligned with the x-coordinate axis that points from the left to the right side of the soccer pitch.

ball control. If no player is in touch distance or the ball is airborne, the control situation is described as $free\_ball()$. If a single player (pl) is in touch distance, it is described as $x\_ball\_control$(pl). Otherwise multiple players reside in touch distance and the ball control situation is described as $fight\_for\_ball()$.

5.1.3 Categorical Classification with Flexible Bounds

In the preceding section, the details of the concrete qualitative abstraction, the set of qualitative ground predicates and their respective set of equivalence classes have been worked out. Subsequently, the general classification modus operandi for the transformation from preprocessed quantitative time series ($\rightarrow$ values) to qualitative time series ($\rightarrow$ symbols, classes) is introduced.

Basic Ideas for a Stable Classification

In [SWW05, FSW04], Steinbauer et al. outline their approach for the compilation of a qualitative knowledge base for autonomous robots in the RoboCup *Middle-Size League*. The approach is geared to stability in the classification results in the face of slight environmental variations such as changing lighting conditions and the coarse, somewhat unreliable character of the perception of the environment based on an imperfect set of sensors.

The fundamental classification scenario wrought by Steinbauer and colleagues seems straight forward at first glance. For their qualitative world model Steinbauer et al. populate their knowledge base with a pool of simple, n-ary ground predicates $p(Obj^n)$ such as $inReach(Obj)$ [FSW04, p.4], each with a binary truth value. For any particular predicate,

these truth values can be understood as the two possible classes associated with the lower/upper half of the open codomain of the univariate time series used as classification input. Central for the classification is the fixed location of the threshold point where the codomain is split. The task of the classifier in each invocation is to determine on which side of the threshold the current input is residing.

Problems arise, once successive input values linger in immediate proximity to the fixed threshold such that repeated, rapid oscillations among the truth values occur. This scenario is encountered frequently, in particular under imperfect vision, even though only subtle change actually occurs in the environment/input data.

In [SWW05, FSW04], Steinbauer et al. seek for a pragmatic solution suitable to mitigate the oscillation problem, and resort to the principle of a *Schmitt trigger*[8] as a method of resolution borrowed from the domain of electrical engineering and signal processing which is based upon a simulation of the *hysteresis property*.

Generally speaking, *hysteresis* describes a property of, usually physical, systems which do not react immediatly to forces applied to them but rather exhibit a noteworthy aspiration to retain their momentary system state which depends on the respective, immediate history of the system.

A *Schmitt trigger* is a simple electronic circuit with two possible states which exhibits the hysteresis property in its switching behavior. Instead of implementing a fixed boundary between lower and upper state, two distinct thresholds are used. Which one of those is currently active is determined by the current state. Using a simple thermostat example, the switching behavior can be described as follows: A thermostat controlling a heater may turn the heater on when the temperature drops below A degrees, but not turn it off until the temperature rises above B degrees where A $<$ B. Thus the on/off output of the thermostat to the heater when the temperature is between A and B depends on the history of the temperature. This prevents rapid switching on and off as the temperature drifts around the set point.

Steinbauer et al. replace the fixed threshold value with a flexible threshold that is determined for each employment of the classifier by the hysteresis function fb. This function implements the principle of a Schmitt trigger with a flexible bound centered around an orientation threshold t_{base}, where the extend of the flexibility is expressed by the offset ϵ[9]. For each particular employment of the appropriate classification function, fb is evaluated in advance in order to obtain a concrete threshold as follows. Let sym^{fb}_{lower} denote the class associated with the interval below t_{base} and let sym_{in} denote the class found in the previous classification cycle.

$$fb(sym_{in}) = \begin{cases} t_{base} + \epsilon & : \quad if \quad sym_{in} = sym^{fb}_{lower} \\ t_{base} - \epsilon & : \quad otherwise \end{cases}$$

The classification based on flexible bounds for the predicate $inReach(Obj)$ as introduced by Steinbauer et al.[10] can now be defined as:

$$C(inReach(Obj), M, sym_{last}) = \begin{cases} true & : \quad if\ dist(Self, Obj) > fb(sym_{last}) \\ false & : \quad otherwise \end{cases}$$

In figure 5.3, the stabilization effect of applying predicate hysteresis is visualized. The choice of the offset value is a critical issue as its value determines the appreciation of values between desired state persistence and required change dynamics. The benefit of the introduction of predicate hysteresis was substantiated successfully via empirical test series as described in [SWW05, pp.3].

8 The concept of the Schmidt trigger was originally published in: Otto H. Schmitt, A Thermionic Trigger, Journal of Scientific Instruments 15 (January 1938): 24-26.

9 Setting $\epsilon = 0$ effectively turns *fb* into a common, fixed threshold.

10 in a slightly different notation [SWW05, p.3]

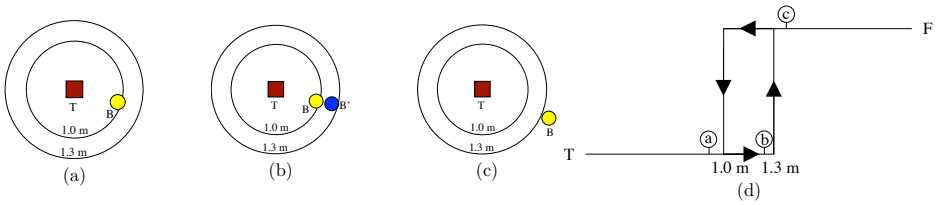


Figure 5.3: Evaluation of the predicate $inReach$ using a hysteresis function with $t_{base} - \epsilon = 1.0m$ and $t_{base} + \epsilon = 1.0m$. [FSW04, p.4]

In its standard form the offset value ϵ remains a fixed property of the associated function fb. However, in [SWW05, p.3], Steinbauer and colleagues point out that it is feasible as well to tie the offset value closely to the orientation threshold t_{base} in the following way:

$$fb(sym_{in}) = \begin{cases} t_{base} \cdot (1 + \epsilon_{rel}) & : \quad if \quad sym_{in} = sym^{fb}_{lower} \\ t_{base} \cdot (1 - \epsilon_{rel}) & : \quad otherwise \end{cases}$$

Thus, ϵ_{rel} is effectively tied to the threshold value t_{base} such that for high t_{base} the hysteresis effect is more pronounced.

This is motivated with the egocentric perception of the middle size robots which brings about a growth of noise in perception as the distance to observed objects grows as well. Considering that the threshold for a predicate such as $inReach(Obj)$ refers to the distance between observing agent and Obj the stabilizing effect of the predicate hysteresis should be adopted as well.

Classification of Univariate Time Series with Open Codomain

The concept of predicate hysteresis seems to lend itself for the implementation of a sufficiently stable qualitative abstraction and is thus adapted for general classification of univariate time series data where the respective input codomain is segmented into $n > 2$ parts, associated with an equal number of equivalence classes collected in a set $\mathbb{SYM}_i$, amongst which classification results may vary[11]. In the following, means to handle both open and cyclic codomains are worked out as both types of codomains are represented in the time series to be processed in the qualitative abstraction. While time series that encode velocities or distances feature open codomains, the cyclic counterpart is featured in time series referring to motion directions and spatial orientation relations.

The classification of univariate time series with open codomain is discussed first. From a formal point of view, a hysteresis-enabled classifier for an open codomain can be defined as a tuple:

$$Classifier^1_{open} = \langle \mathbb{T}, \mathbb{CS}_{open}, \mathbb{SYM}, \epsilon, ic_{prev}, ic_{def}, classify_{open} \rangle \tag{5.1}$$

In this tuple, $\mathbb{T}$ refers to a list of thresholds in ascending order which partition the complete codomain into intervals. This list must contain at least two pseudo-thresholds $t_{-\infty}$ and t_{∞}, such that in the default case[12] the whole codomain can be understood as a single interval bounded by the aforementioned pseudo-thresholds.

11 such as $\mathbb{SYM}_{dist} = \{touch, very\_close, close, medium, far\}$, (cf. section 5.1.1 on page 56)
12 No custom thresholds added in addition to the pseudo thresholds

Algorithm 1 $classify_{open}(val)$ returns a class

inputs: $\mathbb{CS}_{open}$, non-empty list of successive ic-mappings
$ic_{prev} \in \mathbb{CS}_{open}$, recent ic-mapping
ϵ, fixed hysteresis size
$val \in \mathbb{Q}$, numerical input for the classification procedure
local variables: $ic_{curr} = \langle t_{low}, ex_{low}, t_{up}, ex_{up}, sym \rangle$, current ic-mapping

```
1: ic_curr ← ic_prev
2: if ¬ entails(ic_curr, var) then
3:    if val < t_low then
4:       while ¬ entails(ic_curr, val) do
5:          ic_curr ← predecessor(ic_curr)
6:       end while
7:       contract(ic_prev)
8:       t_up ← t_up + ε ; ex_up = true
9:    else
10:      while ¬ entails(ic_curr, val) do
11:         ic_curr ← successor(ic_curr)
12:      end while
13:      contract(ic_prev)
14:      t_low ← t_low − ε ; ex_low = true
15:   end if
16: end if
17: ic_prev ← ic_curr ; return sym
```

Algorithm 2 contract(ic) returns void

inputs: $ic = \langle t_{low}, ex_{low}, t_{up}, ex_{up}, sym \rangle \in \mathbb{CS}_{open}$, ic-mapping to be contracted

```
1: if ex_up = true then
2:    t_up ← t_up − ε ; ex_up = false
3: end if
4: if ex_low = true then
5:    t_low ← t_low + ε ; ex_low = false
6: end if
```

$\mathbb{CS}_{open}$ is a non-empty list of unique extended variation of interval→class mappings (short: ic-mappings) that are formally defined as

$$ic = \langle t_{low}, ex_{low}, t_{up}, ex_{up}, sym \rangle \tag{5.2}$$

where $t_{low}, t_{up} \in \mathbb{T}$, $t_{low+1} = t_{up}$, $ex_{low}, ex_{up} \in \mathcal{B}ool$, $sym \in \mathbb{SYM}$. $\mathbb{SYM}$ denotes the collection of target classes from the $ic \in \mathbb{CS}_{open}$.

Successive ic-mappings $ic_i, ic_j \in \mathbb{CS}_{open}$ share a common threshold as upper/lower bound such that $t_n \in \mathbb{T}_{open} = t^i_{up} = t^j_{low}$. The inner thresholds in $\mathbb{T}$ are not fixed values. Rather, they can be displaced due to hysteresis-induced, temporary, one-side expansion of the current ic-mapping $ic_{curr} \in \mathbb{CS}_{open}$ by the fixed hysteresis size ϵ which retains the same meaning as in the preliminary approach by Steinbauer and colleagues. To ensure consistency with respect to the order of elements in $\mathbb{T}$, the hysteresis size ϵ is constrained such that $\epsilon \ll minargs_{ic_i \in \mathbb{CS}_{open}}(width(ic_i))$. The notation $\ll$ is to convey that the respective interval sizes should be *large* compared to the threshold flexibility.

The concrete policy of hysteresis-enabled classification for open codomains is explained subsequently. To begin with, $classify_{open} : \mathbb{Q} \times \mathbb{SYM} \to \mathbb{SYM}$ denotes the classification function. In dependence of a previous classification result stored in $ic_{prev} \in \mathbb{CS}_{open}$, in each employment this function determines the particular ic-mapping entailing the single

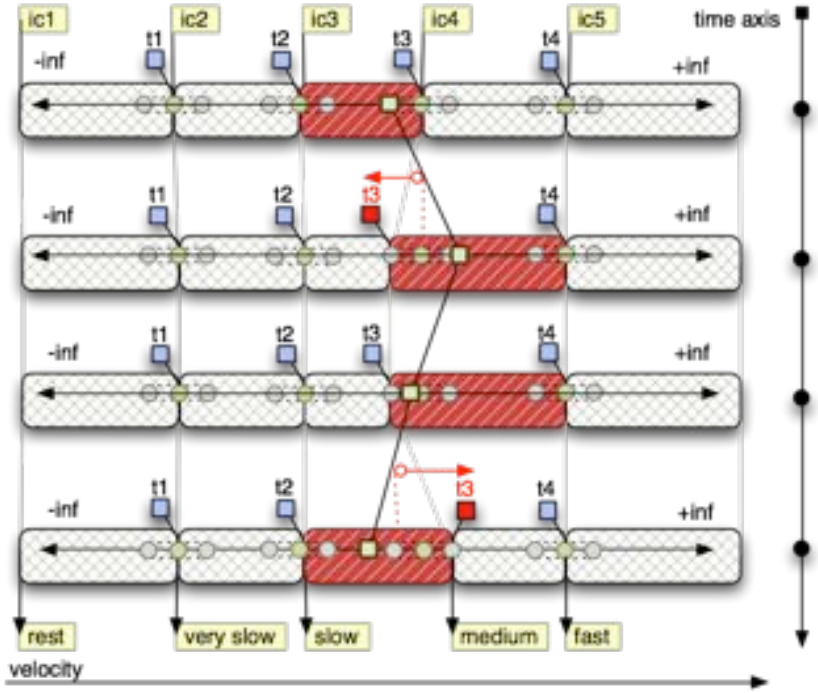


Figure 5.4: Schematic open codomain classification of velocities for four successive points in time. The stabilizing influence of the hysteresis effect becomes apparent in the third step where a premature interval hop is suppressed until a stronger value trend is found.

numerical input which is fed into the classifier, and thus determines the associated class. In the process, $classify_{open}$ implements the hysteresis functionality, seizing the ideas outlined in the previous section and lifts them to the general case. The mode of operation of $classify_{open}$ is outlined in algorithm 1.

The last element of the classifier tuple unmentioned so far, is the default IC for the classifier which is set upon initialization/reset.

Due to the implemented interval expansions and extractions, a persistence of the classifier to remain in its current state is established with respect to the direction from which the current interval was entered. Thus, oscillation between neighboring intervals can be mitigated to a certain extend which is determined by the choice of the hysteresis size ϵ. At the same time, no persistence is established in the direction of the most recent interval transition. Thus, successive iteration over neighboring intervals with a fixed transition direction is not constrained by the hysteresis effect. Figure 5.4 illustrates the generalized system of adaptable interval bounds in an schematic example, showing four successive employments of the classifier.

Classification of Univariate Time Series with Cyclic Codomain

Now that the classification for time series with open codomains has been introduced, the focus is shifted towards the treatment of classification for time series with cyclic codomains. The latter time series differ from the open equivalent in that possible values linger in a set interval $I = [val_{min}, val_{max}[$ (e.g. $I_{deg} = [0°, 360°[$) where $val_{max} \equiv val_{min}$. Cyclic codomains feature a discontinuity point t_{jump} where the lowest possible values are in immediate proximity to the highest possible values. The adaption of the methodology

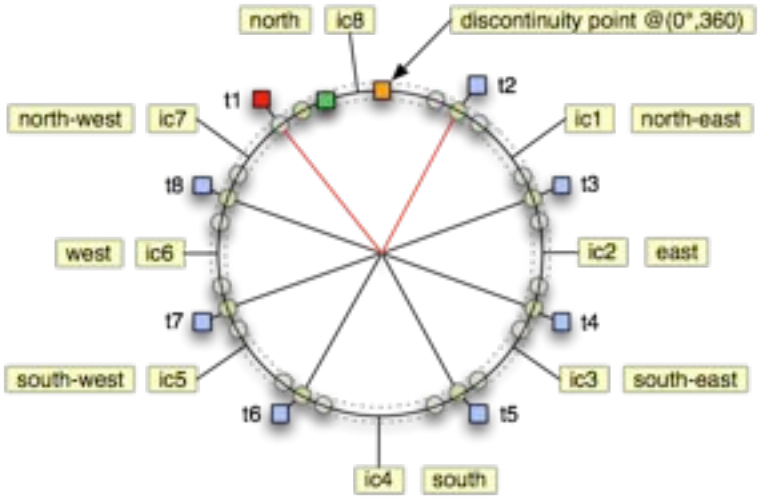


Figure 5.5: Schematic cyclic value range classification of (motion) directions with an eight-segment wind rose for a single point in time. ic_8 associated with the class *north* comprises the special interval which bypasses the discontinuity point t_{jump} found at 0/360 degrees.

for open codomains in order to obtain

$$Classifier^1_{ring} = \langle \mathbb{T}_{ring}, \mathbb{CS}_{ring}, \mathbb{SYM}, \epsilon, ic_{prev}, ic_{def}, classify_{ring} \rangle \tag{5.3}$$

comprises two major tasks. First, a means to deal with the discontinuity point in the specification of $\mathbb{CS}_{ring}$ is required. Second, a suitable classification function $classify_{ring}$ needs to be specified.

Due to the ring characteristic of cyclic codomains, $\mathbb{T}_{ring}$, unlike $\mathbb{T}_{open}$, does not contain pseudo-thresholds such as t_∞. Instead, the existence of a minimum of two regular thresholds is mandatory such that the codomain can be partitioned seamlessly in at least two intervals. To ensure consistency, it must hold that $\forall t_i \in \mathbb{T}_{ring} : dist(t_i, t_{jump}) > \epsilon$. Thus, in particular, thresholds cannot coincide with the discontinuity point.

$\mathbb{CS}_{ring}$ is a non-empty ring of successive ic-mappings as specified in Equation 5.2. The construction thereof differs from its sibling $\mathbb{CS}_{open}$ in a single, yet significant aspect. For $\mathbb{CS}_{open}$ with $\mathbb{T}_{open} = \langle t_1, \ldots, t_n \rangle$[13] pairs of successive thresholds $\langle t_i, t_{i+1} \rangle \; . \; i \in \{1, \ldots, n-1\}$ form the basis for the $ic_i \in \mathbb{CS}_{open}$.

For $\mathbb{CS}_{ring}$, an additional ic_n over the interval $\langle t_n, t_1 \rangle$ is required which bypasses the discontinuity point. Contrary to the normal bounds, for this special ic-mapping, it holds that the lower bound t_{low} is indeed bigger than the upper bound t_{up}. The entailment function used for both open and cyclic value ranges accommodates for this singularity as shown in algorithm 4.

The hysteresis-enabled classification for cyclic codomains is brought about by the classification function $classify_{ring} : \mathbb{Q} \times \mathbb{SYM} \to \mathbb{SYM}$, specified in algorithm 3. The function must handle a particularity of $classify_{ring}$ which is due to the ring structure of $\mathbb{CS}_{ring}$. In case of a transition amongst any two ic-mappings, an incremental iteration over neighboring intervals in either direction from the initial interval eventually succeeds in arriving at the new entailing target interval. Algorithm 3 accommodates for this fact, choosing the transition direction determined by the shorter route in order to apply the interval expansion at the appropriate bound.

13 where t_1, t_n coincide with the pseudo-thresholds $t_{-\infty}, t_\infty$

Algorithm 3 $classify_{ring}(val)$ returns a class

inputs: $\mathbb{CS}_{ring}$, non-empty ring of successive ic-mappings
$ic_{prev} \in \mathbb{CS}_{ring}$, recent ic-mapping
ϵ, fixed hysteresis size
$val \in \mathbb{Q}$, numerical input for the classification procedure
local variables: $ic_{curr} = \langle t_{low}, ex_{low}, t_{up}, ex_{up}, sym \rangle$, current ic-mapping
$steps = 0$, counter for interval hops

```
1: ic_curr ← ic_prev
2: if ¬ entails(ic_curr, val) then
3:    while ¬ entails(ic_curr, val) do
4:       ic_curr ← successor(ic_curr) ; steps ← step + 1
5:    end while
6:    contract(ic_prev)
7:    if size(CS) − steps ≤ size(CS) · 0.5 then
8:       t_up ← t_up + ε ; ex_up = true
9:    else
10:      t_low ← t_low − ε ; ex_low = true
11:   end if
12: end if
13: ic_prev ← ic_curr ; return sym
```

Algorithm 4 entails(ic_{in}, var) returns bool

inputs: $ic_{in} = \langle t_{low}, ex_{low}, t_{up}, ex_{up}, sym \rangle$, ic-mapping
var, value which is either entailed or not

```
1: if t_low < t_up then
2:    return (val > t_low && val ≤ t_up)
3: else
4:    return (val > t_low || val ≤ t_up)
5: end if
```

Two-tier Classification of Multivariate Time Series

Both classification scenarios described so far were laid out for the processing of univariate time series of quantitative input data. For the largest part, the desired qualitative abstraction for distinct aspects of motion as well as spatial configurations among objects can already be achieved, seizing the available methodologies for classification in open- and cyclic codomains. However, the qualitative abstraction of actor locations on the soccer pitch into *residence regions* motivates and presupposes the ability to process multivariate time series in classification as well. The implementation thereof should benefit from the enhanced stability due to hysteresis, and build immediatly on the means worked out for univariate classification. Moreover, bearing the classification of regions in mind as principal motivation, it should also provide enough flexibility to model heterogeneous regions in a natural way and focus classification to interesting regions of the classification space (i.e. the soccer pitch as portion of the simulation basement in the classification of residence regions).

In order to build upon existing one-dimensional classifiers, a two-tier strategy is employed for multivariate classification. The underlying idea is to divide the classification problem for multivariate time series into distinct classification problems for each dimension of the multivariate input and merge the intermediate results on a superordinate level.

To begin with, a classifier for the processing of multivariate time series can be formalized

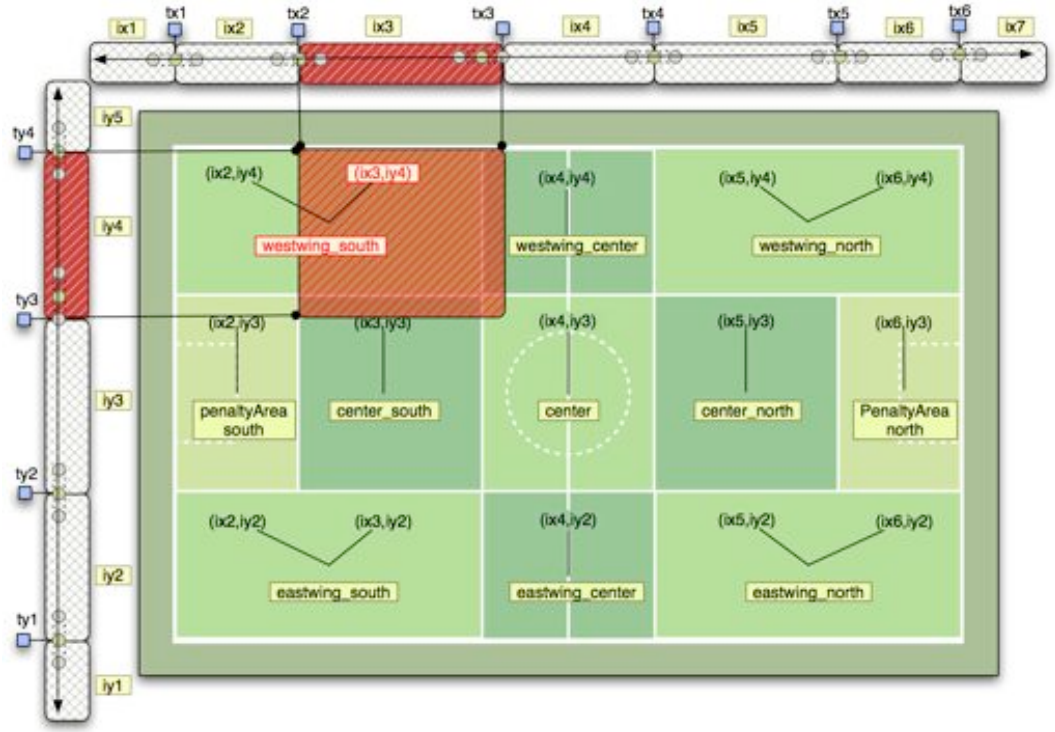


Figure 5.6: Application of multivariate classification of residence regions on the soccer pitch. Besides and above the pitch the subordinate, intermediate classifiers along the 2 spatial dimensions in the plane are shown. The target class $westWingSouth \in \mathbb{SYM}_{region}$, comprised by a homogeneous local neighborhood of size 2, is evaluated by a lookup operation of the intermediate classification result $\vec{inter} = \binom{ix_3}{iy_4} \in \mathbb{SYM}_x \times \mathbb{SYM}_y$.

as a tuple:

$$Classifier^n = \langle (Classifier^1_{\{open \| ring\}})^n, \mathbb{CS}^n, sym_{def}, classify^n \rangle \tag{5.4}$$

$(Classifier^1_{\{open \| ring\}})^n$ describes the vector of dedicated, univariate classifiers for either open or cyclic codomains, which are each concerned with the intermediate classification of a single dimension of the multivariate input. These classifiers, which operate independent of each other, already apply predicate hysteresis and thereby ensure a stable classification along each particular input dimension.

$\mathbb{CS}^n$ and sym_{def} are data structures that are used in a subsequent, merging evaluation of the intermediate classification results. $\mathbb{CS}^n$ is a set of pairs $\mathbb{SYM}^n \rightarrow \mathbb{SYM}$ where each concrete pair associates a possible intermediate result represented as a result vector with a fixed target class.

$\mathbb{CS}^n$ is laid out as a partial mapping such that it does not need to contain associations for all elements $\vec{sym} \in \mathbb{SYM}_1 \times \ldots \times \mathbb{SYM}_n$. Instead, whenever $\vec{inter} \notin \mathbb{SYM}_1 \times \ldots \times \mathbb{SYM}_n$, it is assumed that the default class specified by sym_{def} applies. This property of $\mathbb{CS}^n$ provides the possibility for partial classification of multivariate input.

Furthermore, the associations specified by $\mathbb{CS}^n$ need not be injective such that for each association target, there exists exactly a single association key. Instead it is allowed that *homogeneous local neighborhoods* in the mapping codomain are associated with the same target class.

Algorithm 5 $classify^n(val)$ returns a class

inputs: $(Classifier^1)^n$, vector of subordinate univariate classifiers
$val \in \mathbb{Q}^n$, multivariate input into classification procedure
local variables: $inter \in \mathbb{SYM}^n$, collector for intermediate classification results
$\mathbb{CS}$, list of mappings $\mathbb{SYM}^n \rightarrow \mathbb{SYM}$
$default \in \mathbb{SYM}$, default mapping
1: for $i = 1$ to n do
2: $inter_i \leftarrow Classifier_i : classify(val_i)$
3: end for
4: if $\exists \langle key, value \rangle \in \mathbb{CS} \,.\, key = inter$ then
5: return $value$
6: else
7: return $default$
8: end if

Definition 5.2 (Local Neighbors, Local Neighborhood) *Two symbolic vectors* $s\vec{y}m_1, s\vec{y}m_2 \in \mathbb{SYM}_1 \times \ldots \times \mathbb{SYM}_n$ *are* local neighbors *if* $s\vec{y}m_1$ *can be transformed to* $s\vec{y}m_2$ *with transition of the value* $s_i \in \mathbb{SYM}_i$ *of a single dimension to an immediatly neighboring class. A set of vectors* $s\vec{y}m_i$ *is then referred to as a* local neighborhood *if any two vectors in the set are connected via local neighbors that are contained in the same set as well. A* homogeneous local neighborhood *is a special local neighborhood which does not encircle completely another local neighborhood.* □

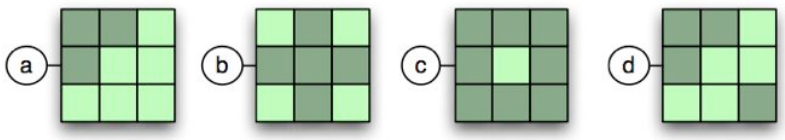


Figure 5.7: Examples clarifying the above definition: a. & b. *homogeneous local neighborhoods* (dark) in $\mathbb{SYM}^2$; c. non-homogeneous *local neighborhood* (dark) ; d. criteria for a local neighborhood not satisfied.

With a fixed set of univariate base classifiers a fine resolution of the classification can be achieved using 1:1-mappings in $\mathbb{CS}^n$. A coarser classification is achieved using n:1-mappings from homogeneous local neighborhoods to target classes as well. Mixing both mapping forms can model a varying resolution of the target classes such that spaces of higher relevance can be classified finer than spaces of lower relevance.

Both tiers of the mapping process for multivariate time series, the generation of an intermediate classification vector and the mapping to a target class is entailed in the classification function $classify^n$ as shown in algorithm 5.

The classification for multivariate time series has been introduced formally for arbitrary arity. The classification of residence regions which motivated the dimension extension from univariate classifiers in the first place can now be handled properly with a multivariate classifier of arity 2 that uses two univariate classifiers $Classifier_{open}$ which classify residence with respect to the x or y-axis of the plane. Figure 5.6 shows an example for the employment of a region classifier:

$$Classifier^{Region} = \langle (Classifier^1_{open})^2, \mathbb{CS}^2, sym_{def}, classify^2 \rangle \tag{5.5}$$

Now that means for both the classification of uni- and multivariate time series data have been worked out, an essential part of the problem of qualitative abstraction has

been covered. The set of ground predicates has been determined with their respective set of target classes and seizing the aforementioned ideas for the implementation of the classification for each classification cycle, predicates such as $dist(pl_1, ball, low)$ can be compiled. In the following those loose predicates obtain a temporal dimension as they are turned from spatio- to spatio-temporal entities.

5.1.4 Formalisms for the Representation of Time

In spatio-temporal analysis of dynamic scenes as opposed to pure spatial reasoning approaches the temporal dimension in reasoning is of central importance as dynamics is change through time.

To start with, a subset of relevant concepts from the interval-based temporal logic proposed by James Allen in a series of publications [All81, All83, All84, AH90, AF94] is introduced.The notion of time adopted in this thesis is founded and shown to be in accordance with Allen's notions. It is shown how first order predicates that describe categorical facts about the dynamic scene are associated with a validity interval via the dedicated logical predicates in order to describe scene continuities/properties. The initial discussion of Allen is followed by beneficial extension developed by Freksa in [Fre92] which seeks to mitigate deficiencies in Allen's logic and proposes effective coarse reasoning based on semi-intervals.

The concept of time which is adopted for this thesis is presented as a hybrid model which is based upon intervals as basis concepts, yet acknowledges the concept of time points as well where appropriate for the application of temporal relations on semi-intervals.

Allen's Interval Temporal Logic

For his interval temporal logic, Allen naturally adopts the understanding of time as a period structure and builds upon the time period as single primitive object of discourse and a primitive relation between two periods, called *Meets*, whose semantics is direct precedence such that, if a first intervals meets a second interval, there is no gap in between and the intervals touch. Allen assumes a linear and unbounded model of time. In the development of his theory, Allen is concerned with a subset of all conceivable time periods, namely those which feature a finite temporal extension (left-right-bounded periods) and thus adhere to his first of a set of five fundamental axioms forming the formal basis for his theory.

Let $I_{\|}$ the set of all such finite periods and let $i, j, k, l, m \in I_{\|}$, then according to [All83] the following basic axioms hold:

Axiom 5.1 *Time itself is unbounded, i.e. there is neither a start nor an end of time.*

$\forall i \in I_{\|} \exists j, k \in I_{\|} : Meets(j, i) \wedge Meets(i, k)$

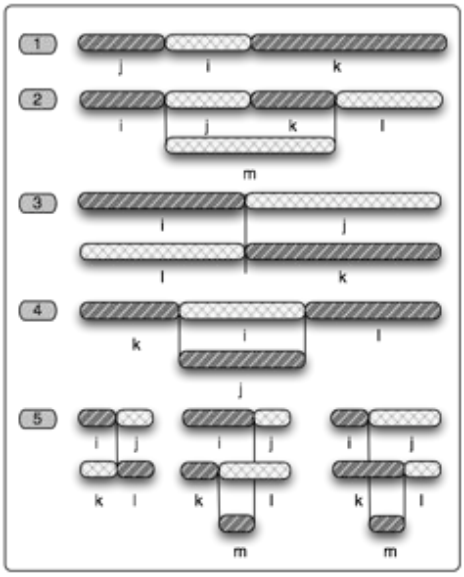
Axiom 5.2 *Succeeding time periods for which the Meets-relation holds can be combined into a larger period which is the composition of both its constituents.*

$\forall i, j, k, l : Meets(i, j) \wedge Meets(j, k) \wedge Meets(k, l) \Rightarrow \exists m : Meets(i, m) \wedge Meets(m, l)$

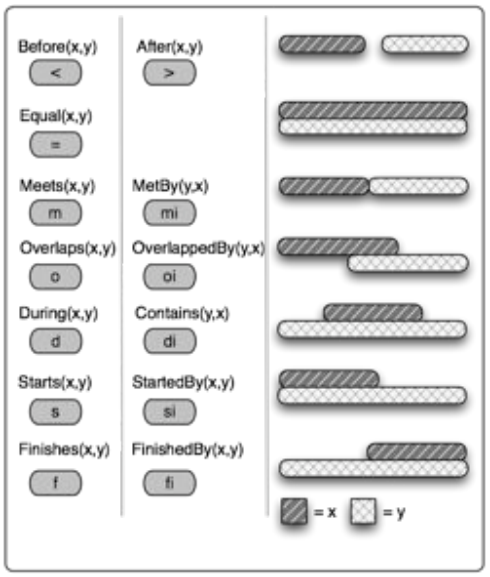
Allen writes that "*as a convenient notation we will often write $j + k$ to denote the interval that is the concatenation of the intervals j and k.*" [AF94, p.9]

Axiom 5.3 *Periods uniquely define an equivalence class for periods that meet them.*

$\forall i, j, k, l : Meets(i, j) \wedge Meets(i, k) \wedge Meets(l, j) \Rightarrow Meets(l, k)$



(a) Graphical visualization of Allen's five axioms for his interval temporal logic.



(b) Allen's thirteen possible qualitative relations between two periods with their respective inverses an mnemonic labels.

Axiom 5.4 *The equivalence classes referred to above also uniquely define the periods.*

$$\forall i,j,k,l : Meets(k,i) \wedge Meets(k,j) \wedge Meets(i,l) \wedge Meets(j,l) \Rightarrow i = j$$

Axiom 5.5 *Any two pairs of periods either meet at the same place or either the first or the second pair of intervals meets at an earlier place in time then its counterpart.*

$$\forall i,j,k,l : Meets(i,j) \wedge Meets(k,l) \Rightarrow Meets(i,l) \otimes (\exists m : Meets(k,m) \wedge Meets(m,j)) \otimes (\exists m : Meets(i,m) \wedge Meets(m,l))$$

Based on this compact set of axioms, that are typically visualized as depicted in figure 5.8a, Allen derives further properties such as the impossibility of any period meeting itself as theorems.

He also derives the complete range of 13 possible interval relations ($\mathbb{TR}_{Allen}$) which can hold between any two periods as a consequence of these axioms, all of which in figure 5.8b. As an example, $Before(i,j)$ is defined as follows:

Definition 5.3 (Before Relation) *Let $i,j,m \in I_{parallel}$, then the Before relation is defined as: $Before(i,j) \equiv \exists m : Meets(i,m) \wedge Meets(m,j)$.* □

Allen develops a dedicated constraint system, implemented as a composition table which allows for the following kind of temporal reasoning. Let $p_1, p_2, p_3 \in I_{\|}$ and let further $rel_a(p_1,p_2) \wedge rel_b(p_2,p_3) : rel_a, rel_b \in \mathbb{TR}_{Allen}$, than by means of the interval composition table, it is possible to infer the non-empty disjunction $\bigvee_1^n : (n \in \mathbb{N}) \geq 1$ of possible temporal relations that may hold between p_1 and p_3 given the knowledge provided by the initial relations.

Due to the fact, that Allen's 13 temporal relations are pairwise disjunct such that only one of them can hold for a pair of intervals, only one of the relations contained in the result can actually apply at a time, yet in order to determine exactly the correct candidate, additional information is required.

Allen's temporal logic may have been initially developed under the assumption of a continuous time domain it can be successfully applied to discrete time domains as well.

For the latter, the otherwise universal concept of a period as a structure which can be recursively, and infinitely decomposed into ever smaller sub-periods is abandoned and the notion of a smallest indivisible period, the *moment*, is introduced. Allen writes that "*a period can be classified by the relationships it can have with other periods. For Example, we call a period which has no sub-periods (i.e. no period is contained in it or overlaps it) a moment and a period which has sub-periods an* interval." [AF94, p.10].

The concept of a *moment* as indivisible atomic interval in a *discrete time domain* can be formally defined as:

Definition 5.4 (Atomic intervals (Moments)) *Let $j,k,i \in I_{\|}$, and let $I_{mom} \subset I_{\|}$ denote the set of indivisible, atomic intervals, then such intervals, referred to as* moments*, are defined as:*

$\forall i : \nexists j,k : Contains(i,j) \otimes Overlaps(k,i) \Rightarrow i \in I_{mom} \subset I_{\|}$. □

Once moments have been introduced, seizing Allen's composition axiom (5.2), time can be thought of as time line densely packed with consecutive moments. Consequently, each *interval* $i \in I_{\|} \setminus I_{mom}$ can be conceived as a composition of a sequence of moments on that time line.

Definition 5.5 (Interval Composition from Moments) *:*
$\forall i \in I_{\|} : \exists j_1, \ldots, j_n \subset I_{mom}, n \in \mathbb{N} \wedge Meets(j_{n-1}, j_n) : \sum_{c=1}^{n} j_c \equiv i$. □

In a discrete simulation environment such as the RoboCup *3D Soccer Server* a moment as introduced by Allen corresponds to the duration of a discrete simulation step. For physical simulations it is common to associate a certain fixed amount of continuous time (20ms in case of the soccer simulation) with each moment in order to couple discrete temporal progress in the simulator (in terms of sim-steps) with continuous temporal progress in the simulated physical environment.

While Allen's temporal logic insists that everything that is happening or holding true in a domain does so for extended intervals of time – however short these may be, thus negating the possibility of instantaneous action and events as for instance in the situation calculus, he also states that "*[...] we can define a notion of* time point *by a construction that defines the beginning and ending of periods. [...]* moments *and* [time-]points *cannot be collapsed. In particular* moments *are true periods and may meet other periods.*" [AF94, p.10].

It is stated that any semantic model which allows for these distinctions would be a possible model of time. Allen suggests a representation for moments and intervals in a discrete time model where periods are mapped to pairs of integers as in $\langle i,j \rangle : i,j \in \mathbb{Z} \wedge i < j$. In such a model moments correspond to pairs of the form $\langle i, i+1 \rangle$. Time points on the other hand are represented as simple scalar integers. This representational semantics has been commonly applied in (e.g. in [Mie04a, Geh05, Lat07]).

In a critical review of Allen's work, Freksa associates Allen's set of interval relations $\mathbb{TR}_{Allen}$ with corresponding relations between the beginnings and endings of the related pair of intervals by means of a comprehensive four-coordinate table as shown in Figure 5.8 [Fre92, pp.4]. A maximum of two such relations is required in order to uniquely distinct any of Allen's 13 interval relations from its siblings.

Freksa's Extension of Allen's Temporal Logic

Allen's interval logic was criticized as being too precise since complete knowledge about the considered intervals is required as a start which is often not feasible, especially in dynamic

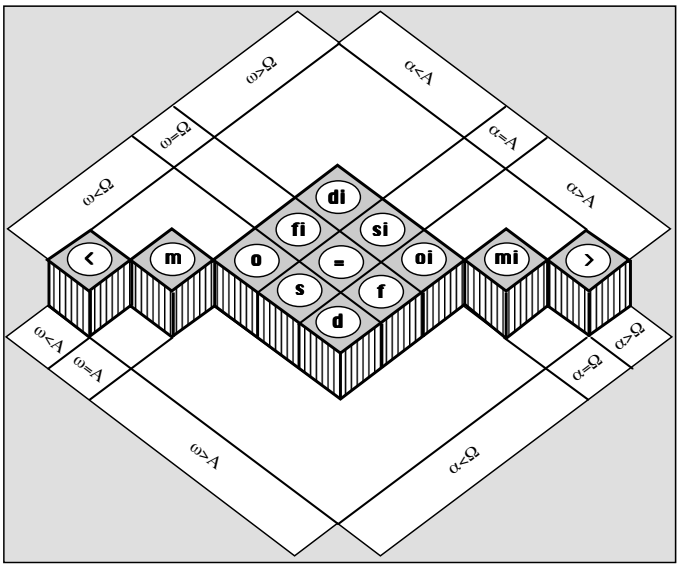


Figure 5.8: Alternative visualization of Allen's interval relationships characterized by relations between their beginnings α, A and endings ω, Ω. [Fre92, p.6.]

domains where for instance only the beginning of an event may be known while the event is still in progress. A means to reason upon such incomplete knowledge is, however, desirable. What is more, as the amount of knowledge about the relations of intervals in question decreases the representational overhead increases in a reciprocal fashion as uncertainty or missing knowledge about concrete temporal configurations is modeled as disjunction of possible relationships. "*The representation of incomplete knowledge [...] creates a cognitively awkward situation: the less we know, the more complex the representation of what we know becomes.*" [Fre92, p.4]. This is why Freksa refers to Allen's method for temporal reasoning as *fine reasoning*.

Freksa does not agree with Allen's proposal to use intervals as representational primitives for reasoning about events. As an alternative, he presents an extended concept for temporal reasoning where beginnings and endings of intervals, denoted as *semi-intervals* are used as primitives. In accordance with [Fre92, p.5], those semi-intervals can be equated with the aforementioned time points in a discrete time domain.

Once intervals are represented in terms of their enclosing semi-intervals as in $\langle \alpha_i, \omega_i \rangle$ it becomes clear that various temporal reasoning situation exists where a desired temporal relation between intervals can be expressed as relations between only a subset of available semi-intervals. "*In order to determine that Newton lived before Einstein it is sufficient to know that Newton's dead took place before Einstein's birth. It does not help if in addition we know when Newton was born or when Einstein died.*" [Fre92, p.7].

It is shown that it is feasible to perform coarse reasoning based upon a set of relaxed interval relations ($\mathbb{TR}_{Freksa}$). These relations correspond to disjunctions of Allen's interval relations and are thus in general less restrictive than those put forward by Allen (cf. figure 5.9). Central to Freksa's work is the concept of *conceptual neighbors* and *conceptual neighborhood* [Fre92, p.8]. For a certain kind of knowledge incompleteness which does not permit a fine resolution of closely related variants within a neighborhood, knowledge about temporal entanglement of two intervals is expressed directly via the relations in $\mathbb{TR}_{Freksa}$ on a broader level of granularity which corresponds to conceptual neighborhoods.

Freksa rearranged Allen's composition table such that conceptually neighboring relations

Relation (inverse Relation)	Semi-Interval Relationships	Disjunction of possible Allen relations
$Older(x,y)$ $(Younger(y,x))$	$\alpha < A$	Before, Meets, Overlaps, FinishedBy, Contains
$HeadToHeadWith(x,y)$ $(HeadToHeadWith(y,x))$	$\alpha = A$	StartedBy, Equal, Starts
$Survives(x,y)$ $(SurvidedBy(y,x))$	$\omega > \Omega$	Contains, StartedBy, OverlappedBy, MetBy, After
$TailToTailWith(x,y)$ $(TailToTailWith(y,x))$	$\omega = \Omega$	FinishedBy, Equal, Finishes
$Precedes(x,y)$ $(Succeeds(y,x))$	$\omega \leq A$	Before, Meets
$ContemporaryOf(x,y)$ $(ContemporaryOf(y,x))$	$\alpha < \Omega \wedge \omega < A$	Overlaps, FinishedBy, Contains, StartedBy, Equal, Starts, During, Finishes, OverlappedBy
$BornBeforeDeathOf(x,y)$ $(DiedAfterBirthOf(y,x))$	$\alpha < \Omega$	Before, Meets, Overlaps, FinishedBy, Contains, StartedBy, Equal, Starts, During, Finishes, OverlappedBy
$OlderAndSurvivedBy(x,y)$ $(YoungerAndSurvives(y,x))$	$\alpha < A \wedge \omega < \Omega$	Before, Meets, Overlaps
$OlderContemporaryOf(x,y)$ $(YoungerContemporaryOf(y,x))$	$\alpha < A \wedge \omega > A$	Overlaps, FinishedBy, Contains
$SurvivingContemporaryOf(x,y)$ $(SurvivedByAndContemporaryOf(y,x))$	$\alpha < \Omega \wedge \omega > \Omega$	Contains, StartedBy, OverlappedBy

Figure 5.9: Freksa's semi-interval relations with corresponding disjunction of possibly associated Allen-relations.

follow in succession. As it turns out the resulting interval disjunctions in the table always form conceptual neighborhoods which correspond either to Allen original relations or to (conjunctions of) Freksa's semi-interval relations. Freksa shows that *coarse reasoning* about temporal coherences based on conceptual neighborhoods is sound and sometimes more efficient compared to *fine reasoning* as favored by Allen. Composition tables are presented for that purpose [Fre92, p.24 & p.29].

It is possible to apply ideas from both Freksa and Allen in several ways. The respective formalisms can be used to derive new temporal coherences from a given set of initially known temporal relations of either flavor thus applying the temporal constraint systems introduced with the respective composition tables. It is also feasible to use only the representational formalisms in the context of temporal pattern rules as means to relate in a flexible way the time-dependent constituent concepts such as facts, events and actions. The latter is the way Allen's and Freksa's work finds their way into this thesis.

As the system for spatio-temporal real-time analysis of dynamic scenes under development here will be laid-out for concurrent, incremental application with regard to the normal execution of the soccer simulation, it is mandatory to allow for the use of right-open intervals whose validity, for a certain sim-cycle of a soccer game, has begun at a certain moment in the past and is still continuing.

Definition 5.6 (Left-bounded, right-open Intervals) *Intervals of the form* $i = \langle \alpha, \infty \rangle$ *whose*

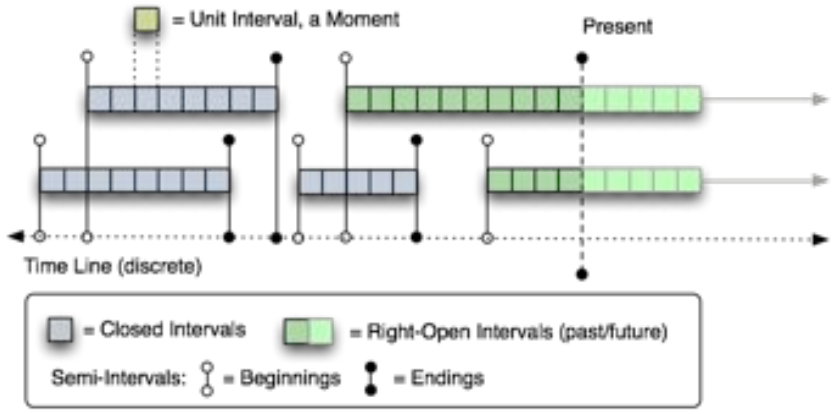


Figure 5.10: Conceptual visualization of the time model used throughout the remainder of this thesis with closed ($I_{\|}$) and right-open ($I_{\vdash}$) intervals combined.

beginning α is bound to a set point in time but whose duration is considered infinite such that no explicit ending ω can (yet) be specified, are denoted as left-bounded *and* right-open *interval $i \in I_{\vdash}$.*

For this class of intervals, it holds that:

$\forall i \in I_{\vdash} \exists j \in I_{\|} : Meets(j,i) \wedge \nexists j \in I_{\|} : Meets(i,j)$ □

Thus, the relations introduced by both Allen and Freksa will be applied on intervals from the broader interval domain $I_{\|\cup\vdash} \equiv I_{\|} \cup I_{\vdash}$ where the codomain for intervals is $dom(I_{\|\cup\vdash}) \stackrel{\text{def}}{=} \mathbb{Z} \times (\mathbb{Z} \cup \{\inf\})$. In order to use the interval relations $\mathbb{TR}_{Allen}$ and $\mathbb{TR}_{Freksa}$ appropriately under these conditions the set of semi-interval relations $<, =, >$ is adapted to reflect the extended codomain and it is further necessary to constrain both $Finishes(x,y)$ and $TailToTailWith(x,y)$ in such a way that they can only hold for $x, y \in I_{\|}$.

The resulting conceptualization of time used throughout the remainder of this thesis, can illustrated as in Figure 5.10.

5.1.5 Spatial Predicates with a Temporal Dimension

So far, the representational formalism for time used in this thesis has been introduced self-sufficiently. However for qualitative abstraction it is mandatory to associate validity intervals as defined so far with qualitative properties or continuity constraints expressed in first-order predicate logic such as $velocity(\text{ball}, \text{fast})$.

In order to achieve this association Allen introduces the *HOLDS* predicate in his temporal logic. The HOLDS predicate has both an assertional and an inferential aspect. From the assertional point of view, HOLDS allows for expressing assertions about the validity of time-variant facts such as $HOLDS(velocity(\text{pl}_5, \text{fast}), \text{i})$. $i \in I_{\|}$ in a very concise, intuitive manner. Furthermore, a conjunction of such HOLDS predicates can be viewed as fundamental pool of data of a spatio-temporal knowledge base from which further knowledge can be inferred. In its most simplistic form the temporal semantics of the HOLDS predicate itself can be used which Allen defines as follows:

Definition 5.7 (Homogeneity Property of the HOLDS Relation) *:*

Let $HOLDS(pred, i) : i \in I_{\|}$ and let further $In(i,j) \Leftrightarrow During(i,j) \vee Starts(i,j) \vee Finishes(i,j)$,

then the homogeneity property of the HOLDS relation is defined as:

$HOLDS(pred, i) \Leftrightarrow (\forall j.In(j, i) \Rightarrow HOLDS(pred, j))$ □

If a fact holds over a certain interval, it also holds during each sub-interval. Thus the HOLDS relation specifies a homogeneity constraint for the validity of the associated fact.

Thus if a HOLDS predicate has been asserted for a certain time interval it is possible to exploit the homogeneity criterion to infer that the same fact held during contained subintervals. This is the inferential view on the HOLDS-predicate.

In this thesis the dual view upon the HOLDS predicate is reflected explicitly by the introduction of the *FACT* predicate which is used for the assertion of new time dependent facts as a result of the qualitative abstraction. Thus the FACT-predicate covers the assertional aspect of Allen's HOLDS predicate. On the other hand, HOLDS is used for inferential purposes as both predicates are associated as outlined below.

Definition 5.8 (HOLDS Relation expressed in terms of the FACT relation) *:*

Let $i, j \in I_{\parallel} : HOLDS(pred, i) \Leftrightarrow \exists FACT(pred, j) : In(j, i) \vee Equals(j, i)$

Let $i, j \in I_{\vdash} : HOLDS(pred, i) \Leftrightarrow \exists FACT(pred, j) : Older(j, i)$ □

The reason why the FACT-relation is explicitly mentioned in this concept chapter and is not treated as an implementation detail is that both HOLDS and FACT can be used gainfully side by side in the specification of spatio-temporal patterns for high-level events, actions and action sequences due to their respective semantics. Using HOLDS, it can be determined simply whether or not a fact was valid over a certain period of time. Using FACT however it can be determined whether or not a fact was valid over exactly a certain period of time. Thus, while the HOLDS-relation refers to *general* continuity, FACT-relations refer to *maximal* continuity.

Interval Assembly Strategies

Via the process of qualitative abstraction which is applied incrementally for the complete course of simulated soccer games, quantitative time series are by and by transformed into accordant qualitative time series – a series of symbols for each cycle, using the classification facilities that have been outlined in the initial parts of section 5.1. It is then compulsory to decide for an interval assembly policy that controls how temporally extended qualitative ground facts are compiled, starting from loose sequences of symbol values.

A common strategy, applied exemplary in the approaches by Miene [Mie04a, pp.57] and Gehrke, is made up of an *explicit lengthening* of the validity intervals of existing predicates. Using this technique for each predicate, for each classification cycle, there is always the explicit requirement for exactly one interaction with the quantitative knowledge base which comprises either a lengthening operation (continuity) or the creation of a new fact (classification hop). Thus even if the circumstances of the case do not change, action is required nevertheless.

In this thesis a different policy for interval creation and maintainance is applied which was motivated in part by considerations by Gehrke with respect to possibilities to put notably less strain on the knowledge base interaction [Geh05, p.138]. Whenever new facts are created due to a classification hop, the fact is associated with an unbounded validity in that only the start of the validity interval is specified explicitly while the ending is left open. Having established such a new fact, no further interaction is required while the fact remains valid. Only when another classification hop occurs, is the open interval

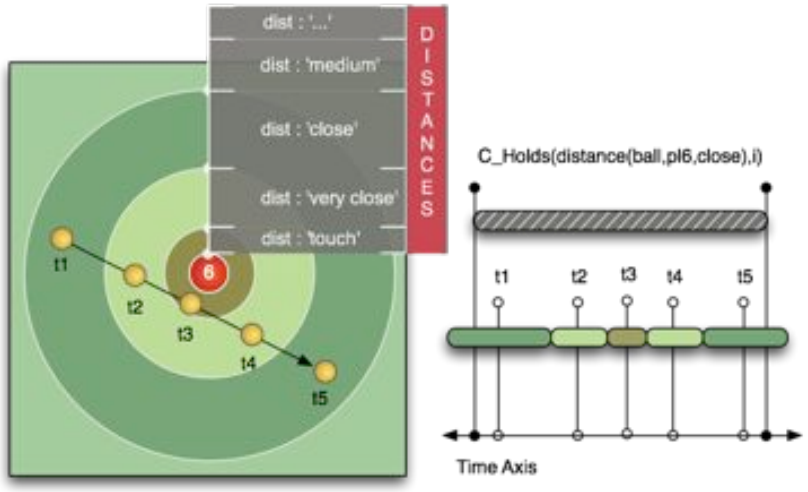


Figure 5.11: Example Scenario for use of the C_HOLDS predicate: The ball passes a resting player, being close, very close, in touch, very close and finally close again to the player in immediate succession. For the whole duration of the passage, the C_HOLDS relation holds as $touch \leq close$ and $very\_close \leq close$.

closed explicitly now that the ending has become known, and a new open interval is created.

It is assumed that classification hops are unlikely to occur with very high frequency. Thus, it is probably significantly cheaper to perform two knowledge base interactions only upon change than one interaction in every single cycle especially with regard to the real-time bias of the developed recognition module. Since this assumption drives the design decision for the *explicit closing*-policy, it is subject of evaluation in section 7.4.1 on page 137.

As a result of the fact creation policy for each particular moment, the qualitative knowledge base may contain two types of asserted facts in the fact pool $\mathbb{F}'$. First, facts that have been valid for a distinct duration in the past
(e.g. $FACT(velocity(\text{ball}, \text{fast}), \langle s_1, e_1 \rangle)$. $s_1, e_1 \in \mathbb{Z}$) and, second, those facts that have begun to hold sometime in the past have retained validity so far
(e.g. $FACT(playmode(\text{playon}), \langle s_2, \text{inf} \rangle)$. $s_2 \in \mathbb{Z}$) (cf. figure 5.10).

A relaxed HOLDS Predicate for Qualitative Subsumption

A closer examination of the capabilities of the standard HOLDS predicate, applied as introduced by Allen, reveals the following limitation which somewhat constrains its usefulness with respect to the formal specification of spatio-temporal patterns. An expression such as $HOLDS(distance(\text{ball}, \text{pl}_6, \text{close}), \text{i})$ is true if and only if the ball retains a position such that for the whole duration of the considered interval the distance becomes neither larger (e.g. $distance(\text{ball}, \text{pl}_6, \text{medium})$) nor lesser (e.g. $distance(\text{ball}, \text{pl}_6, \text{very\_close})$) but rather remains exactly close.

By all means situations exist where the desired semantics that should be expressed is covered through use of the holds-relation. In other situations however, it seems desirable being able to specify a slightly relaxed statement of affairs. If we consider the dribbling as a paradigmatic example, it makes sense to introduce the constraint that during each consecutive self assist executed by the dribbling player, the ball should retain in a distance

from that player which is no larger than close rather than exactly close[14].

In order to fulfill the demand for a somewhat relaxed interpretation of the HOLDS-predicate, a new predicate, named *containment holds* or short *C_HOLDS*, is introduced.

In order to simplify the specification the following temporal relation, let: $InOrEquals(i,j) \Leftrightarrow In(i,j) \vee Equals(i,j)$. The formal semantics for the C_HOLDS-relation can now be specified both for $i_1 \in I_{\|}$ and $i_2 \in I_{\vdash}$. Due to the fact that contrary to the normal HOLDS-relation, we are concerned with a contemplation about the dynamically changing value of a given predicate, the notation for predicates in the following definitions adheres to the following schema:

$$FACT(\overbrace{\underbrace{distance}_{\text{UID}}(\underbrace{\text{ball},\text{pl}_6}_{\text{reference}},\underbrace{\text{close}}_{\text{value}})}^{\text{qual. predicate}},\ i\ ).$$

Definition 5.9 (C_HOLDS for the Closed Interval Codomain $I_{\|}$) *:*

Let $pred(ref,arg) \in \mathbb{P}$ and let $arg, arg_1, \ldots, arg_n \in \mathbb{SYM}$ where $\mathbb{SYM}$ denotes a set of equivalence classes for which the total ordering relation $\leq$ is defined. Let further $i, i_1, \ldots, i_n \in I_{\|}$. Let $\bigwedge_{c=1}^{n}(FACT(\ldots))$ denote a conjunction of atomic facts $f \in \mathbb{F}'$. Then, the Containment Holds relation is defined as:

$$\begin{aligned}
&\forall i \in I_{\|} : C_HOLDS(pred(ref,arg),i) \Leftarrow \\
&\quad \exists \bigwedge_{c=1}^{n}(FACT(pred(ref,arg_c),i_c)) \,.\, c,n \in \mathbb{N} : \\
&\quad \forall c < n : Meets(i_c, i_{c+1}) \\
&\quad \wedge\, \forall e \in c,\ldots,n : \leq (arg, arg_e) \\
&\quad \wedge\, (InOrEquals(\sum_{c=1}^{n}(i_c), i)) \quad \square
\end{aligned}$$

Definition 5.10 (C_HOLDS for the Right-open Interval Codomain $I_{\vdash}$) *:*

Let $pred(ref,arg) \in \mathbb{P}$ and let $arg, arg_1, \ldots, arg_n \in \mathbb{SYM}$ where $\mathbb{SYM}$ denotes a set of equivalence classes for which the total ordering relation $\leq$ is defined. Let further $i_1, \ldots, i_n \in I_{\|}$ and $i, i_{n+1} \in I_{\vdash}$. Let $\bigwedge_{c=1}^{n}(FACT(\ldots))$ denote a conjunction of atomic facts $f \in \mathbb{F}'$. Then, the Containment Holds relation is defined as:

$$\begin{aligned}
&\forall i \in I_{\vdash} : C_HOLDS(pred(ref,arg),i) \Leftarrow \\
&\quad (\quad \exists\, (\bigwedge_{c=1}^{n}(FACT(pred(ref,arg_c),i_c)) \,.\, c,n \in \mathbb{N}, i_c \in I_{\|}) \\
&\qquad \wedge\, (FACT(pred(ref,arg_{n+1}),i_{n+1}) \,.\, i_{n+1} \in I_{\vdash}) : \\
&\qquad \forall c \leq n : Meets(i_c, i_{c+1}) \\
&\qquad \wedge\, \forall e \in c,\ldots,n+1 : \leq (arg, arg_e) \\
&\qquad \wedge\, (Older(\sum_{c=1}^{n+1}(i_c), i) \vee HeadToHeadWith(\sum_{c=1}^{n+1}(i_c), i)) \quad) \\
&\quad \vee \quad (\quad \exists\, FACT(pred(ref,arg_1), j) \,.\, j \in I_{\vdash} : \\
&\qquad \leq (arg, arg_1) \\
&\qquad \wedge\, (Older(j,i) \vee HeadToHeadWith(i,j)) \quad) \quad \square
\end{aligned}$$

14 Naturally, in order to successfully perform a self assist the distance eventually needs to fall short of close for kick/reception.

C_HOLDS is applicable only in the context of a subset of possible qualitative predicates as a total order relation ($\leq$) needs to be defined over the set of equivalence classes $\mathbb{C}_{pred}$ that forms the codomain of possible values for the predicate. $\leq$ is a reflexive, antisymmetric and transitive relation and it must hold that $\forall c_1, c_2 \in \mathbb{C}'_{pred}; . \, c_1 \leq c_2 \vee c_2 \leq c_1$. Among those sets of equivalence classes that posses a suitable order relation are qualitative distances (viz. $touch \leq very\_close \leq close \leq medium \leq far$) or velocities (viz. $rest \leq very\_slow \leq \ldots \leq very\_fast \leq beam$). On the contrary, qualitative predicates whose set of equivalence classes features a cyclic character such as motion directions cannot be used with the C_HOLDS predicate.

5.1.6 Focused Qualitative Abstraction

The previous sections outlined the process of qualitative abstraction of dynamic scenes with respect to the RoboCup *3D Soccer Simulation* without particular care for the large amount of facts that is created by and by given all possible uni- and especially bivalent relations introduced in section 5.1.1. Let n denote the number of univalent relations and m the number of bivalent relations. There are 23 movable objects actively participating in the game which are completely visible in each qualitative abstraction pass under the assumption of input data derived from perfect vision as featured by a coach agent observing a game in the RoboCup 3D Soccer Simulation league. What is more, due to *particle filter*-based motion estimation and efficient look-around [LRS+06, pp.22 & p33], it is possible for player agents such as those of the Virtual Werder team to maintain a less accurate but nevertheless nearly equally complete perception of the dynamic soccer scenes to be turned into a concise qualitative representation. Thus, for the worst case $num^{rels} = n \cdot 23 + m \cdot 22 \cdot 23$ qualitative relations exists which need to be tracked by the analysis system.

Experimental results presented by Gehrke [Geh05, pp.133] where the performance of a qualitative abstraction for traffic scenes with variable number of scene actors was evaluated, suggest, that a limitation of the actors considered in each qualitative abstraction pass is compulsory for efficient operation. Reetz-Schmidt notes in the description of the REPLAI-II plan recognition system (cf. section 4.1.3 and [Ret91]) that "*the observer has to concentrate on those agents he regards as interesting. A spectator of a soccer game would not try to observe all 22 players with equal attention, but would concentrate on interesting happenings on the field.*" [Ret91, p.177].

With regard to the pool of motion patterns that is developed for this thesis in section 5.3, a comparatively simple focusing heuristic can be employed. As the events and actions are ball-centered in character, a solution suggests itself where the ball is understood as single key object whose respective location on the field defines a dynamic region on the field which is considered as being immediatly interesting for the momentary course of the game. The region is defined by a certain radius (viz. 13m) around the ball. In order to mitigate the already somewhat familiar oscillation problematic which may occur when players are located in close proximity to the boundary of the relevance region, again a two-sided boundary with suitable flexibility (viz. 1m) can be implemented seizing once again the ideas by Steinbauer et al. described in section 5.1.3. Univalent relations are compiled for all objects within the relevance region, as should be expected. Bivalent relations are constrained further to those subset of relations from the ball as key object to the non-key objects in the relevance regions. Thus, as an example the relation $sorientation(\text{ball}, \text{pl}_1, \text{north})$ is calculated contrary to both $sorientation(\text{pl}_1, \text{ball}, \text{south})$ or in particular $sorientation(\text{pl}_1, \text{pl}_2, \text{northWest})$ for any two non-key objects in the relevance region. The feasibility of such a restrictive constraint on the relations compiled in each qualitative abstraction pass was substantiated empirically in preliminary experiments as

the detection quality of motion incidences for the patterns specified in section 5.3 is not decreased.

It must be noted however that this is due to the ball focus of specified patterns and less constraining heuristics would without doubt be required once additional patterns are compiled which evaluate for example (1) the potential menaces of players about to handle/kick the ball by approaching aggressors from the enemy team or (2) the coverage of players – supposedly in particular the player performing the spatio-temporal analysis in order to gain a better understanding of its current situation – by other players from the adversary team. Scenario 1 could be accommodated for with the possibility of temporary extension of the set of key objects as players are added to this set which are in control of the ball and thus susceptible to aggressive behavior by other players. Scenario 2 could be accommodated for with the establishment of further persistent key objects that each define their particular relevance region. In applying those extensions the focus strategy would evolve such that it converges with the focus strategies employed by human spectators while watching soccer games.

Thus the simple focus heuristic, employed for this thesis should be regarded as a first, coarse approximation which offers possibilities for later, gradual refinement which remains oriented at the pool of target motion patterns to be detected.

5.2 Formalisms for the Representation of Extensive Motion Situations

The previous section was concerned with the incremental compilation of the fundamental tier of a qualitative knowledge base comprised of spatial zero-, mono- and bivalent relational facts associated with their respective temporal extension ($\mathbb{F}'$). However, the pieces of knowledge compiled so far can hardly be considered relevant in their own right with regard to the requirements stipulated in section 3.4: to populate the knowledge base with extensive, domain-specific *motion instances* in terms of events, actions and action sequences. In fact, atomic qualitative building blocks have been provided.

Consequently, this section attends to the question, in which way motion situations in a dynamic soccer scene can be modeled formally by means of *motion patterns*. Each of those patterns comprises a blueprint, or common denominator, for a certain class of motion instances. Once those patterns that explicitly encode domain expertise due to their being manually crafted by a knowledge engineer, are available, the task to analyze a dynamic scene is reduced, for the scope of this thesis, to efficient detection of concrete motion instances by hierarchical matching of spatio-temporal patterns.

The formal specification of motion patterns for deployment in the scene analysis presupposes a well-founded understanding of the formal description of the desired target pattern types. In section 5.1.5 fundamental aspects of Allen's interval temporal logic have been studied that were necessary for the representation of simple *properties*[15]. Beyond that, Allen introduces *occurrences* as objects in his temporal logic and subdivides this class into two subclasses, processes and events which are distinguished from properties by "*the characteristics of the set of temporal intervals that they hold and occur over*" [All84, p.131].

5.2.1 Representation of Events

According to Allen, temporal facts that are expressed with the HOLDS-relation, exhibit the property of homogeneity as a distinguishing feature which unambiguously sets them apart

15 using the HOLDS- and – for the scope of this thesis – the derived FACT-relation

from events. While the HOLDS-relation is closed under the In-relation, events adhere to the converse semantics. As Allen puts it: "*[...] an event occurs over the smallest time possible for it to occur.*" [All84, p.131]. In Allen's logic, events are consequently represented by a new predicate, namely OCCUR, that is defined in terms of its anti-homogeneity property as follows:

Definition 5.11 (Homogeneity Property of the OCCUR Relation) *:*

Let $OCCUR(event, i)$ *denote the incidence of a specific event over the interval* $i \in I_{\|}$ *and let further* $j \in I_{\|}$*, then the* homogeneity property *the event incidence is defined as:*

$$OCCUR(event, i) \wedge In(j, i)\ .\ i, j \in I_{parallel} \Rightarrow\sim Occur(event, j)\ \square$$

Event classes (summarized in the set $\mathbb{E}$) can be defined as a first-order logic term such as $kick(shooter, \ldots)$[16] and thus resemble in their layout the facts produced in the qualitative abstraction. A concrete *event incidence* in a soccer game is expressed as $OCCUR(kick(\mathrm{pl}_1, \ldots), \mathrm{i}) \in \mathbb{E}'\ .\ \mathrm{i} \in I_{\|}$ where $\mathbb{E}'$ is the set of momentarily hitherto detected, concrete event incidences.

Each equivalence class of events can be described in terms of their particular necessary *incidence conditions* $P_{event}(t, e)\ .\ t \in I_{\|},\ e \in \mathbb{E}$ such that $OCCUR(e, t) \Rightarrow P_{event}(t, e)$. Furthermore, if $P_{event}(t, e)$ are both necessary and sufficient conditions for an event's occurrence, as it is the case in the application scenario considered for this thesis, the *event pattern* can be defined as follows:

Definition 5.12 (Event Pattern) *Let* $e \in \mathbb{E}$ *denote an equivalence class of events and let* $i, j \in I_{\|}$*. Then, the* event pattern *associated with* e *is defined as:*

$$OCCUR(e, i) \Leftrightarrow P_{event}(i, e) \wedge (\forall j\ .\ In(i, j) \supset\ \neg P_{event}(j, e))\ \square$$

The complicated specification is necessary in order to comply with definition 5.11. The term $P_{event}(i, e)$ is defined as a formula in first-order logic. The codomain of allowed constituent terms can be specified as follows:

$$dom(P_{event}(i, e)) = \mathbb{F}_{assert} \cup \mathbb{F}_{infer} \cup \mathbb{E} \cup \mathbb{TR}_{Allen} \cup \mathbb{TR}_{Freksa} \cup \mathbb{SR} \cup \mathbb{META}.$$

Thus, for the compilation of event patterns, valid building blocks are asserted qualitative facts ($\rightarrow FACT(\ldots, i) \in \mathbb{F}_{assert}$), inferred facts ($\rightarrow \{C\_\}HOLDS(\ldots, j) \in \mathbb{F}_{infer}$) and other events ($\rightarrow OCCUR(\ldots, k) \in \mathbb{E}$). In order to specify the temporal entanglement of the aforementioned constituents, the interval relations by Allen ($\mathbb{TR}_{Allen}$) and Freksa ($\mathbb{TR}_{Freksa}$) come in handy. An additional set of relations ($\mathbb{SR}$) can be used to enforce spatial constraints where required.

The $\mathbb{META}$ set contains two meta-logical functions which are adopted from the Prolog world, namely $setof(\ldots)$ [Bra01, pp.182] and $not_f(\ldots)$ [Bra01, pp.125]. Even though these functions tend to add impurities to the formalization, they provide both convenience and an extended expressiveness for the specification of event patterns that justify their inclusion into $dom(P_{event}(i, e))$.

Excursus: Meta-Logical Functions ($\mathbb{META}$) :

First, the $setof(\ldots)$ function acts as a kind of object collector. Consider the example: $setof(player, team\_member(player, \mathrm{SVWerderBremen}), squad)$. The $setof(\ldots)$ function accumulates all players that are known to be under contract with the SV Werder Bremen into a complete squad list[17].

16 The ellipsis '...' can be neglected for the time being. Its meaning is described in section 5.2.3, pp.88

17 Contemplate pattern 5.3 on page 92 in section 5.3.1 for a concrete application of $setof(\ldots)$

Second, the $not_f(\ldots)$ function has the semantics of *negation as failure*[18]. Consider the example $not_f(superior(Diego, \text{Ribéry}))$. This term holds once it is not explicitly known that Diego is superior to Ribéry. Thus, even if no statement is made in either direction as to the superiority of the aforementioned Bundesliga soccer players, the term $not_f(superior(Diego, \text{Ribéry}))$ is still true as the term $superior(Diego, \text{Ribéry})$ does not hold. Formally, $not_f(\ldots)$ is based on the *closed world assumption* [RN95, p.354–357]. According to this assumption, the world is closed in the sense that everything that holds true is stated explicitly in the knowledge base or can be derived thereof. This special semantics needs to be taken into account in the application of $not_f(\ldots)$ in the compilation of concrete event/action patterns[19]. •

Turning the focus of attention back to the compilation of $P_{event}(i,e)$, the terms $t \in dom(P_{event}(i,e))$ are turned into a valid formula of first-order logic using the common logical connectives (viz. $\wedge, \vee, \neg$). The composition of $P_{event}(i,e)$ is flexible as it allows to use subordinate event classes in the specification of superordinate event patterns.

As this thesis is concerned with the detection of event incidences via pattern matching of the respective $P_{event}(t,e)$, a simplified notation for event patterns is applied which conforms to $OCCUR(e,i) \Leftarrow P_{event}(i,e)$ (cf. section 5.3 on page 89 for example patterns). Thus, the notation of the event pattern contributes to a clarification of the pattern usage. As for the correctness of the notation, it should be noted, that Allen himself does likewise in [All84].

5.2.2 Representation of Actions and Action Sequences

Turning focus of attention from events towards actions, it is worthwhile, initially, to make a note of the fact that, in Allen's interpretation, events describe an activity which involves a certain product or outcome such as *"The ball has been kicked"*. Processes, on the other hand, refer to the proceeding of an activity and do not necessarily involve a culmination or desired result such as *"The player is running"*.

From a formal point of view, processes differ from events in their homogeneity property in that definition 5.11 does not hold. In contrast, if a process is occurring over an interval i, it must also be occurring over at least a single subinterval $j \subseteq i$. Processes are represented by another predicate in Allen's logic, namely OCCURRING, which is formally defined as follows:

Definition 5.13 (Homogeneity Property of the OCCURRING Relation) *:*

Let $OCCURRING(process, i)$ *denote the incidence of a specific process/action over the interval* $iI_{\|}$ *and let further* $j \in I_{\|}$*, then the* homogeneity property *of the process/action incidence is defined as:*

$$OCCURRING(process, i) \Rightarrow \exists i \; . \; In(j,i) \wedge OCCURRING(process, j) \; \square$$

It is worthwhile to contemplate the meaning of this definition with regard to the deployment of the OCCURRING-predicate for the specification of actions from the domain of simulated soccer. The latter has been proposed by Miene in [Mie04a, p.36 & pp.102] where *processes* are put on a level with *actions*, bearing to Allen as follows: *"[...] an action refers to something that a person or a robot might do. It is a way of classifying the different sorts of things that an agent can do to affect the world, thus it more resembles a sensory-motor program than an event. By performing an action, an agent causes an event to occur, which in turn may cause other desired events to also occur.[...] Some theories refer to the event that was caused as the action, but this is not what we intend*

18 This is why the 'f' for failure occurs as an index character in the function name.
19 Contemplate pattern 5.5 on page 93 in section 5.3.1 for a concrete application of $not_f(\ldots)$

here. Rather, we will draw on an analogy with the robot situation, and view actions as programs." [AF94, p.4].

It becomes apparent that Allen and Miene consistently understand actions as kind of *processes*, stressing the significance of the respective course character. In [AF94] however, Allen seems to be concerned primarily with such agent actions that are performed over the course of time intervals while a dedicated program is active on a single agent such as *walking to a position on the soccer pitch*. For this class of continuous actions, definition 5.13 appears to be feasible[20]. Miene, on the other hand, also considers actions that, although initialized by a single agent, require active participation of supporting agents. Examples comprise amongst others the *pass* and *fail_pass* actions. It is difficult to attribute a process character to such actions both from a common-sense- and from a formal-logical point with single-agent program in mind. Under such circumstances, if a pass action is specified using Allen's formalism (such as $OCCURRING(pass(\mathsf{pl}_1, \mathsf{pl}_4), \mathsf{i})$), it is immediatly obvious that definition 5.13 does not hold, since for each conceivable subinterval $j \subset i$, the pass action is not embraced completely. What is more, the pass initiator is involved in only a very short part of the whole pass course. Thus, in order to comply with definition 5.13, it is imperative to think of passes as – potentially unintended[21] – multi-agent programs and the constraint imposed by definition 5.13 referring to these programs.

The considerations made so far were aimed at a thorough comprehension of the process characteristic of actions in general and in the RoboCup *3D Soccer Simulation League* in particular. With respect to the analysis of dynamic scenes, and the detection of action incidences, it is sufficient to adopt the policy to treat actions as a sub-class of events. Actions, brought about by single/multi-agent programs, cause the occurrence of associated action events (e.g. the pass program causes the action event that *the ball has been passed from* pl_1 *to* pl_2 with the latter now controlling the ball as a result; cf. [AF94] as quoted above). Consequently, these events are subject of detection.

Thus, in correspondence with the event case, *action classes* (summarized in the set $\mathbb{A}_{seq}$) can be defined as first-order logic term such as $pass(shooter, receiver, \ldots) \in \mathbb{A}_{seq}$[22]. A concrete *action incidence* in a soccer game is expressed as $OCCURRING(pass(\mathsf{pl}_1, \mathsf{pl}_2, \ldots), \mathsf{i}) \in \mathbb{A}' \,.\, i \in I_{\|}$ where $\mathbb{A}'$ is the set of concrete action instances.

In accordance with the event case, action classes are described in terms of their respective *necessary* and *sufficient* incidence conditions $P_{act}(i, a)\ .\ \in I_{\|},\ a \in \mathbb{A}_{seq}$.

Definition 5.14 (Action Pattern for $a \in \mathbb{A}_{seq}$) *Let $a \in \mathbb{A}_{seq}$ denote an equivalence class of actions and let $i, j \in I_{\|}$. Then, the* action pattern *associated with a is defined as:* $OCCURRING(a, i) \Leftrightarrow P_{act}(i, a) \wedge (\forall j \,.\, In(i, j) \supset \neg P_{act}(j, a))$ □

This definition differs from its counterpart for events only in the codomain of allowed terms to be found in $P_{act}(i, a)$. It can be specified as follows:

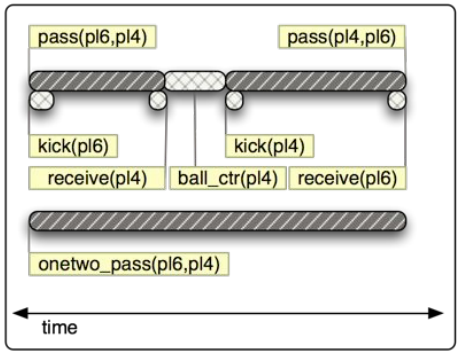
$$dom(P_{act}(i, a)) = \mathbb{F}_{assert} \cup \mathbb{F}_{infer} \cup \mathbb{E} \cup \mathbb{A} \cup \mathbb{TR}_{Allen} \cup \mathbb{TR}_{Freksa} \cup \mathbb{SR} \cup \mathbb{METR}$$

Thus, the codomain for $P_{act}(i, a)$ in a natural way comprises all building blocks that can be found already in $P_{event}(i, e)$ such that simple actions can be described in terms of constituent events (e.g. *kick*- and *reception*-events for the *pass*-action) as well as facts.

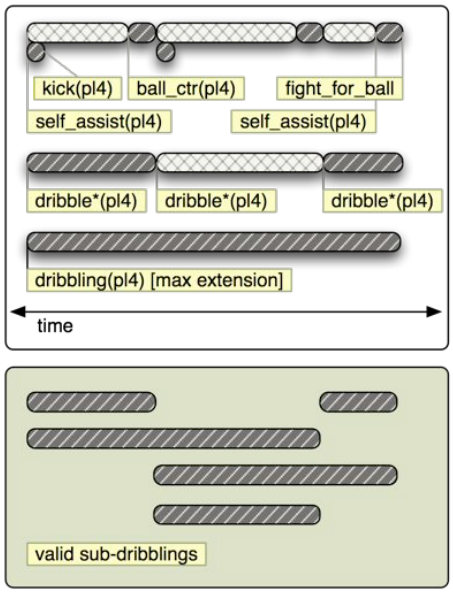
20 even though it is obviously appropriate only in continuous time domains due to the demand for continued existence of shorter sub-intervals for which the process, or the action program, still holds/is active

21 in case of fail passes due to the fact that agents from competing teams do not perform cooperative actions willingly

22 The ellipsis ... can be neglected for the time being. Its meaning is described in section 5.2.3, pp.88



(a) Onetwo-pass as an example for a type 1 action sequence with simple, sequential incidence course.



(b) Dribbling as an example for a type 2 action sequence as composition of single atomic dribblings.

However, $P_{act}(a, i)$ may also contain subordinate action classes, allowing for the following two scenarios: First, the specialization of basic actions (e.g. the *pass* specializes a *ball transfer*) and, second, the specification of *action sequences* with a sequential course of incidence. The *onetwo-pass*, conceived as concatenation of two consecutive *passes* back and forth between two players of the same team, is a well-suited example for such an action sequence (cf. figure 5.12a).

Using the aforementioned codomain for $P_{act}(i, a)$ accommodates the demand to specify actions of varying complexity in a concise and scalable manner. Seizing ideas from Miene [Mie04a, chapter 4], the natural hierarchical structure of the actions to be recognized is exploited. The more complex the action patterns, the more complex constituents may contribute to $P_{act}(i, a)$. Yet, at the same time, the incorporation of less complex constituents down to simple facts remains a valid option, whenever such a course of action is beneficial for a precise specification.

So far, only the first of two potential flavors of action sequences, those that feature a simple sequential course of incidence (type $1 \rightarrow a \in \mathbb{A}_{seq}$) have been covered. A closer examination of the space of conceivable action sequences to be found in the soccer domain, however, unveils the existence of action sequences comprised of a homogeneous, finite, yet in its length a priori indeterminate concatenation of a particular basic action (type $2 \rightarrow a \in \mathbb{A}_{loop}$). The dribbling is a paradigmatic example for the latter type of action sequences due to the fact that complete dribble sequences are composed of simple atomic dribblings where an agent forwards the ball only once (cf. figure 5.12b).

Let $OCCURRING(\mathrm{a}, i)\,.\,class(\mathrm{a}) \in \mathbb{A}_{loop}$, $i \in I_{\|}$ denote a concrete incidence of a type 2 action sequence. Then, sub-intervals $j\,.\,In(i, j)$ may exist, such that $OCCURRING(\mathrm{a}, j)$. This means, a longer action sequence may embrace shorter, yet congenial sub-sequences. In order to handle the specification type 2 action sequences, not covered so far by definition 5.14, an alternate definition is required which both reflects and exploits the compositional characteristics of $a \in \mathbb{A}_{loop}$ as follows:

Definition 5.15 (Action Pattern for $a \in \mathbb{A}_{loop}$) *Let $a \in \mathbb{A}_{loop}$ and $a_{supp} \in \mathbb{A} \equiv \mathbb{A}_{seq} \cup \mathbb{A}_{loop}$.*

Let further $i, j, k \in I_{\|}$. *Then, the* action pattern *associated with* a *is defined as:*

$$
\begin{aligned}
&OCCURRING(a,i) \Leftrightarrow \\
&\quad (OCCURRING(a_{supp}), i) \\
&\vee \quad (\quad OCCURRING(a, j) \wedge OCCURRING(a_{supp}, k)) \\
&\qquad \wedge Meets(j,k) \wedge Equals(i, j+k) \wedge P_{cont.}(a, a_{supp}) \quad) \quad \square
\end{aligned}
$$

This definition is to convey that self-similar action sequences can be both purely sequential, in which case they correspond immediatly with their respective constituent support action (e.g. atomic dribbling), or the concatenation of an existing sequence with a further support action. In the latter case, it is also possible to enforce certain spatial continuity constraints for the concatenated sequence by means of $P_{cont.}(a, a_{support}, i)$, a conjunction of conditions to be met by both a and $a_{support}$ with $dom(P_{cont.}(a, a_{support}, i)) = \mathbb{SR}$.

To conclude, relying on both definition 5.14 and definition 5.15, it is now possible to detect simple actions as well as both types of actions sequences. The resulting set of detectable action classes is now $\mathbb{A} = \mathbb{A}_{seq} \cup \mathbb{A}_{loop}$.

Concluding this subsection so far, both events and actions have been formally introduced including representational means in Allen's temporal logic with adjustments with respect to the application domain, their particular temporal semantics and the composition of their associated motion patterns.

5.2.3 Enhanced Representation via Exploitation of Incidence Context

However, the specification of both, the event- and action classes, is only partially complete. Revisiting the examples introduced so far – i.e. $kick(shooter, \ldots) \in \mathbb{E}$ for events, $pass(shooter, receiver, \ldots) \in \mathbb{A}$ for actions – both contain '...' as a placeholder. While an explanation will follow as to the actual content substituted by '...', it is worthwhile to appraise the composition of event/action classes so far:

Example $\rightarrow$ *pass* action : $\underbrace{pass}_{\text{UID}}(\overbrace{\underbrace{shooter, receiver}_{\text{reference}}, \underbrace{\ldots}_{\text{diversification}}}^{\text{characterization}})$

Fundamental for each concept class $c \in \mathbb{E} \cup \mathbb{A}$ is its *unique identifier* (UID). It already categorizes the event/action broadly, thereby setting it apart unambiguously from the remaining classes, available in $\mathbb{E} \cup \mathbb{A}$. Each class term may optionally comprise an argument tuple of fixed arity which is instrumental in the characterization of event/action incidences. The argument tuple is subdivided in two sections, denoted as reference and diversification. The *reference* sub-tuple accommodates for the fact that, for a well-founded understanding of motion situations in a dynamic scene, it is – generally speaking – of considerable significance, being able to associate the incidence of any particular motion situation with the involved scene actors. Taking up the pass example, the reference-tuple $\langle shooter, receiver \rangle$ specifies the player agents which are involved in the ball transfer.

Amongst the distinct approaches to spatio-temporal analysis of dynamic scenes that have been reviewed in chapter 4, several, such as Kaminka and colleagues (cf. section 4.1.5) or Beetz et al. (cf. section 4.1.8) exist that are satisfied constraining their respective detection granularity to unique identifier and reference alone. With respect to

the superordinate application context of the employed analysis approaches, such a restriction can be appropriate if it is sufficient to learn (1) which motion situations occurred and (2) which actors were involved therein.

Going one step further in order to obtain a comprehensive characterization for motion situations, however, in agreement with Wendler et al. (cf. section 4.1.2, [Wen03]) it is seen as desirable to bring about an extended diversification of those situations, such that characteristic traits are specified as arguments in the class term, summarized in the second characterization sub-tuple (diversification). The explanatory power of detected motion instances is increased. It is now possible to tell not only *that* a particular motion situation occurred but also *the particular characteristic* with which it occurred.

Coming back to the initial pass example, the primal description of the pass class could now be extended as follows:

$$pass(\overbrace{shooter, receiver}^{ref}) \longleftrightarrow pass(\overbrace{shooter, receiver}^{ref}, \overbrace{dir, height, force, success}^{div})$$

Now a concrete pass incidence could be characterized as forward-directed pass, played high and with considerable force which was completed successfully.

In [Mie04a, p.103], Miene tries to emulate the diversification approach with respect to the success of an action incidence, introducing distinct patterns for successful and failed passes. Yet, seizing the ideas presented by Wendler et al. seems more natural in comparison, representing the success of a pass as a specific character trait of a single action class.

Once the placeholder '...' used in the previous subsections is replaced with a concrete tuple of diversification arguments, thus extending the primal class signature, it is required to extend $P_{event}(c,i)$ or $P_{act}(c,i)$ respectively. This is done via appending further constituents from the codomain $dom(P_{event|act}(c,i))$ to the existing spatio-temporal patterns.

This modification on the specification comes along with an adopted view upon the task of the analysis of dynamic scenes as it is used in this thesis. As it was the case before, the primary task is still made up of a structured data-driven, i.e. bottom-up detection of the incidence of motion patterns in an unfolding dynamic scene. However, each successful detection of a new motion situation is now followed up with an non-stopping appraisal of the particular *incidence context* which is provided by the rich pool of facts that is compiled and maintained as result of the qualitative abstraction. Those facts could be treated solely as fuel for the pattern matching process in order to detect primal motion patterns. However doing so would barely exploit the full potential of the qualitative abstraction. Thus, for the scope of this thesis, the readily available pool of qualitative facts is understood as a valuable description of the situational context on the soccer pitch whose evaluation is mandatory for an efficient compilation of a rich, qualitative world model. The latter, supplemental analysis task is qualified as non-stopping as due to its implementation, the detection of motion situation is solely sophisticated but never revised.

5.3 Specification of Motion Patterns for the RoboCup 3D Soccer Simulation

Consequently, based on the formalisms that have been worked out in the previous section the concrete pool of motion patterns that will be subject to detection in concrete games of the RoboCup 3D Soccer Simulation League by the analysis developed in this thesis, is developed.

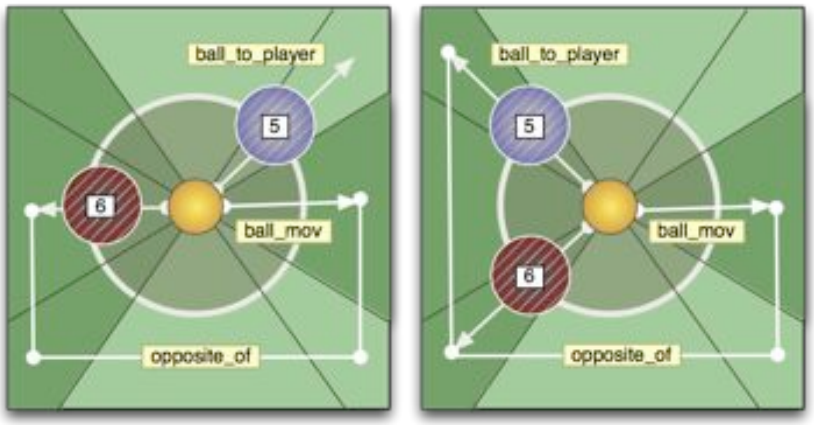


Figure 5.12: Two possible kick scenarios: The situation on the left can be disambiguated by the *exclusive kick* pattern as only agent 6 is *opposite of* the ball's motion direction dir_{mot} after the kick. The situation on the left cannot be disambiguated properly as both players are located *opposite of* the direction of movement due to the relaxed interpretation of $Opposite\_Of(dir_{mot}, dir_{b2p})$.

5.3.1 Specification of Event Patterns

In the first instance, the specification of event patterns is considered in the following ranging from several kick types, over ball reception and deflection to events that center around struggling for ball control and the retreat from the ball. In the compilation of the event patterns, special attention is put on the exploitation of the respective *incidence context* of events in order to obtain a comprehensive pattern diversification.

Kicking the Ball

A central concept in simulated as well as human soccer is the kick of the ball that initiates a ball transfer. In the RoboCup *3D Soccer Simulation League* three distinct kick scenarios can be identified which need to be handled adequately.

By far the most common kick is the *standard kick* which is to be unambiguously executed by a single player on the soccer pitch. This kick flavor corresponds to the kick which is commonly discussed in related work [Mie04a, p.97]. However, the event pattern for the standard kick which is compiled for this thesis as shown in motion pattern (5.1), p.91 entails a necessary means for a plausible disambiguation in kick situations where more than a single player is in ball control. A standard kick can only be attributed to a player if the motion direction dir_{mot} of the ball right after the kick is *opposite of* the direction from ball to player d_{b2p} at the beginning of the kick. In the kick pattern, the $Opposite\_of(dir_{b2p}, dir_{mot})$-relation holds if either dir_{mot} is directly opposite of dir_{b2p} in the wind rose or an immediate neighbor of the direction class that lies directly opposite of dir_{mot} (cf. figure 5.12). Thus, in situations where two players have approached the ball from coarsely opposing sides right before the kick incidence (cf. figure 5.12, scenario 1 on the left), a robust association with the correct kick candidate can be achieved.

Both the motion direction and the inclination behavior of the departing ball are evaluated as pieces of the incidence context and contribute to the threefold diversification for the kick pattern.

A second kick pattern is required due to the limited temporal resolution of world model

Motion Pattern 5.1 Event pattern for the *standard kick* by a single player

$$OCCUR(kick(\overbrace{shooter}^{ref}, \overbrace{dir^{glob:8}, height, \underbrace{\text{std}}_{type}}^{div}), \langle s, e\rangle) \Leftarrow$$

$$\begin{aligned}
& FACT(free_ball, \langle e, \text{inf}\rangle) \\
\wedge\ & FACT(acceleration(\text{ball}, \text{increasing}), \langle s, e_1\rangle) \\
\wedge\ & DurationAtMax(\langle s, e\rangle, 40) \\
\wedge\ & FACT(distance(\text{ball}, shooter, \text{touch}), \langle s_2, e_2\rangle) \\
\wedge\ & (Meets(\langle s_2, e_2\rangle, \langle s, e\rangle) \vee Contemporary_Of(\langle s_2, e_2\rangle, \langle s, e\rangle)) \\
\wedge\ & HOLDS(motion_Dir(\text{ball}, dir^{glob:8}), \langle e, e+1\rangle) \\
\wedge\ & HOLDS(sorientation(\text{ball}, shooter, dir_2), \langle s, s+1\rangle) \\
\wedge\ & Opposite_Of(dir, dir_2) \\
\wedge\ & HOLDS(zposition_trend(\text{ball}, trend), \langle e, e+1\rangle) \\
\wedge\ & Translate\uparrow^{height}_{ztrend}(trend, height)
\end{aligned}$$

Motion Pattern 5.2 Event pattern for the *volley kick* by a single player

$$OCCUR(kick(\overbrace{shooter}^{ref}, \overbrace{dir^{glob:8}, height, \underbrace{\text{volley}}_{type}}^{div}), \langle s, e\rangle) \Leftarrow$$

$$\begin{aligned}
& FACT(acceleration(\text{ball}, \text{increasing}), \langle s, \text{inf}\rangle) \\
\wedge\ & FACT(free_ball(), \langle s_1, \text{inf}\rangle) \\
\wedge\ & Older(\langle s_1, \text{inf}\rangle, \langle s, \text{inf}\rangle) \\
\wedge\ & FACT(motion_dir(\text{ball}, dir^{glob:8}), \langle s, \text{inf}\rangle) \\
\wedge\ & FACT(motion_dir(\text{ball}, dir_2^{glob:8}), \langle s_2, e_2\rangle) \\
\wedge\ & Meets(\langle s_2, e_2\rangle, \langle s, \text{inf}\rangle) \\
\wedge\ & (\ dir^{glob:8} \neq dir_2^{glob:8}\ ) \\
\wedge\ & FACT(distance(\text{ball}, shooter, \text{very_close}), \langle s_3, e_3\rangle) \\
\wedge\ & TailToTailWith(\langle s_2, e_2\rangle, \langle s_3, e_3\rangle) \\
\wedge\ & HOLDS(zposition_trend(\text{ball}, trend), \langle s, e\rangle) \\
\wedge\ & Translate\uparrow^{height}_{ztrend}(trend, height)
\end{aligned}$$

updates for both the agents playing on the soccer pitch and the online coach. As it turns out, it is possible for agents to kick the ball in between two consecutive updates such that it is never perceived that the agent comes into contact with the ball in which case the event pattern for the standard kick does not hold. Those kick incidences constitute the simulation equivalent to volley kicks in real soccer where the ball is forwarded immediatly by a certain player without time spent for ball control or orientation. The dedicated kick pattern for the *volley kick* is shown in motion pattern (5.2), p.91.

Motion Pattern 5.3 Disambiguation Event Pattern for a *collective kick* by a player group

$$OCCUR(kick(\overbrace{players}^{ref}, \overbrace{dir^{glob:8}, height, \text{co}}^{div}), \langle s, e\rangle) \Leftarrow$$

$$setof(player, OCCUR(kick(player, dir^{glob:8}, height, \text{std}), \langle s, e\rangle), players)$$

Motion Pattern 5.4 Event Pattern for *ball reception* by a single player.

$$OCCUR(receive(\overbrace{player}^{ref}), \langle s, s+1\rangle) \Leftarrow$$

$$\begin{aligned} & FACT(x\_ball\_control(player), \langle s, \text{inf}\rangle) \\ \wedge\ & C\_HOLDS(velocity(\text{ball}, \text{medium}), \langle s-1, s\rangle) \\ \wedge\ & FACT(free\_ball(), \langle s_2, e_2\rangle) \\ \wedge\ & Meets(\langle s_2, e_2\rangle, \langle s, s+1\rangle) \end{aligned}$$

It should be noted that both kick patterns introduced so far contribute to the complete specification of the kick scenario in the RoboCup *3D Soccer Simulation League*. Both constitute distinct characteristics of the same embracing basis concept. In the pattern specifications (pattern 5.1 on the previous page and pattern 5.2 on the preceding page) this is conveyed by the fixation of the type argument in the pattern diversification to a constant value (*standard* or *volley* respectively).

In section 3.4.2, a sound disambiguation for kick incidences was required which has been partially realized through the specification of the pattern for the standard kick. However, it is still possible for a particular type of ambiguous kick situation to occur which is depicted in scenario 2 on the right of figure 5.12 on page 90. In these situations two competing players are both located side-by-side such that it is hard or even impossible for a human to decide which of the players is involved in the resulting kick incidence. As a solution to this problem the statement made by the recognition system is relaxed such that a weaker, but still secure claim is made that the ball was shot by a group of competing agents. With respect to the otherwise categorical character of qualitative knowledge compiled and maintained in this thesis, this approach appears to be sound in comparison to an alternative approach proposed by Miene, where kick probabilities are introduced [Mie04a, pp.97]. Whenever in that approach more than one agent presumably kicks the ball at the same time, the probabilities for each kick candidate are equally distributed. This constitutes a transition from the recognition of the actual situation to assumptions about the course of events. The disambiguation pattern for the *collective kick* which is used in this thesis is shown in pattern 5.3. It is more a helper- than a fully grown pattern in its own right.

Ball Reception and Deflection

Now the distinguishable kick scenarios have been specified, the focus of attention is turned to the reception of the ball. In comparison, the dedicated reception pattern specified in motion pattern (5.4), p.92 is rather simple and does not provide a diversification. A ball is received unambiguously once a player gains exclusive control over a sufficiently slow

Motion Pattern 5.5 Event Pattern for *Ball deflection* by a single player.

$$
\begin{aligned}
&OCCUR(deflect(\overbrace{player}^{ref}), \langle s, e\rangle) \Leftarrow \\
&\quad FACT(free\_ball, \langle e, \mathsf{inf}\rangle) \\
&\wedge\ FACT(x\_ball\_control(player), \langle s, e\rangle) \\
&\wedge\ DurationAtMax(\langle s, e\rangle, 40) \\
&\wedge\ FACT(free\_ball, \langle s_1, s\rangle) \\
&\wedge\ HOLDS(motion\_dir(\mathsf{ball}, dir_1^{glob:8}), \langle s-1, s\rangle) \\
&\wedge\ HOLDS(motion\_dir(\mathsf{ball}, dir_2^{glob:8}), \langle e, e+1\rangle) \\
&\wedge\ (\, dir_1^{glob:8} \neq dir_2^{glob:8}\,) \\
&\wedge\ not_f(OCCUR(kick(player, \_, \_, \_), \langle s_2, e\rangle))
\end{aligned}
$$

Motion Pattern 5.6 Pattern for *fight engagement*, first variant.

$$
\begin{aligned}
&OCCUR(engage\_fight(\overbrace{aggressor}^{ref}, \overbrace{\mathsf{new}}^{div}), \langle s, s+1\rangle) \Leftarrow \\
&\quad FACT(distance(\mathsf{ball}, aggressor, \mathsf{touch}), \langle s, \mathsf{inf}\rangle) \\
&\wedge\ FACT(fight\_for\_ball(), \langle s, \mathsf{inf}\rangle) \\
&\wedge\ FACT(x\_ball\_control(player), \langle s_2, e_2\rangle) \\
&\wedge\ Meets(\langle s_2, e_2\rangle, \langle s, s+1\rangle)
\end{aligned}
$$

ball that was formerly free. The scenario where multiple players 'receive' the ball at the same time is covered by a dedicated event, namely the start_fight event, discussed below. With respect to the specification of the reception pattern it is worthwhile to note that containment holds is used to express that upon reception the ball should be *medium* or have a qualitative velocity that is entailed by slow such as *very slow* or *resting*.

Besides the normal reception of the ball, situations occur where the ball collides with a player and bounces off in a new direction as a result without the player issuing a kick. These deflections are described with the event pattern specified in motion pattern (5.5), p.93. The statement of the case, that the ball has provably not been kicked by the deflecting player during is expressed in the last line of the specification. This line states: Nothing is known about the player kicking the ball. The wildcards ('_') in the kick mean that it is of no interest, how the ball might have been kicked.

Struggle for Ball Control

In the following, two basic event patterns are treated which center around the fight for the ball. The first pattern thereof is the fight engagement which comes in two variants. First, a fight engagement occurs when an engaging adversary player comes in contact with the ball which was formerly controlled exclusively by another player. This pattern is specified in motion pattern (5.6), p.93.

A player can also engage in a fight for the ball as an additional competitor for exclusive

Motion Pattern 5.7 Pattern for *fight engagement*, second variant.

$$
\begin{aligned}
OCCUR(engage\_fight(\overbrace{aggressor}^{ref}, \overbrace{\text{existing}}^{div}), \langle s, s+1\rangle) \Leftarrow \\
& FACT(distance(\text{ball}, aggressor, \text{touch}), \langle s, \text{inf}\rangle) \\
\wedge\ & FACT(fight\_for\_ball(), \langle s_2, \text{inf}\rangle) \\
\wedge\ & Older(\langle s_2, \text{inf}\rangle, \langle s, \text{inf}\rangle)
\end{aligned}
$$

Motion Pattern 5.8 Event Pattern for the synchronous *start of a fight*, where multiple players reach the ball at the same time.

$$
\begin{aligned}
OCCUR(start\_fight(\overbrace{players}^{ref}), \langle s, s+1\rangle) \Leftarrow \\
& FACT(fight\_for\_ball(), \langle s, \text{inf}\rangle) \\
\wedge\ & FACT(free\_ball(), \langle s_2, e_2\rangle) \\
\wedge\ & Meets(\langle s_2, e_2\rangle, \langle s, \text{inf}\rangle) \\
\wedge\ & setof(player, FACT(distance(\text{ball}, player, \text{touch}), \langle s, \text{inf}\rangle), players)
\end{aligned}
$$

control of the ball. That is, a player engages in an already ongoing fight. The associated pattern is specified in motion pattern (5.7), p.94.

The other basic event pattern which centers around the struggle for ball control comprises situations where multiple players reach a previously free ball right a the same time leading to an immediate fight between the involved players. The *start fight* pattern specified in motion pattern (5.8), p.94 complements the controlled reception of the ball by a single player.

The recent event patterns were concerned with the emergence and evolutions of situation where multiple agents struggle for control of the ball. The *retreat pattern*, on the contrary, is concerned with the withdrawal of players from the ball. Two distinguishable scenarios of retreat exist. In the first scenario a player that exerts exclusive ball control withdraws from the ball without a shot. Such a behavior could be part of a strategy to leave the ball to be handled by a teammate which might be better suited to kick. The corresponding pattern is specified in pattern 5.9.

The second retreat scenario is associated with a fight for the ball. Sometimes, players retreat from a fight thereby leaving it to the remaining contestants to continue the struggle or ending the fight completely in favor of a single other agent which then exerts exclusive ball control. The retreat pattern associated with this scenario is specified in motion pattern (5.10), p.95.

Both retreat patterns introduced so far contribute to the complete specification of the retreat scenario. Analogous to the situation with the kick, both constitute distinct characteristics of the same embracing concept. Therefore, in the aforementioned pattern specifications the type argument is fixed in the pattern diversification to a constant value (*ball* or *fight* respectively).

The consideration of a player's withdrawal from the ball concludes the presentation of event patterns that have been worked out concretely in the scope of this thesis. The

Motion Pattern 5.9 Event pattern for a *retreat from the ball* such that the ball is free as a consequence.

$$OCCUR(retreat(\overbrace{player}^{ref}, \underbrace{\overbrace{\text{ball}}^{div}}_{type}), \langle s, s+1\rangle) \Leftarrow$$

$$\begin{aligned}
& FACT(distance(\text{ball}, player, \text{very_close}), \langle s, \text{inf}\rangle) \\
\wedge\ & FACT(distance(\text{ball}, player, \text{touch}), \langle s_1, e_1\rangle) \\
\wedge\ & Meets(\langle s_1, e_1\rangle, \langle s, s+1\rangle) \\
\wedge\ & FACT(x_ball_control(player), \langle s_2, e_1\rangle) \\
\wedge\ & FACT(free_ball(), \langle s, \text{inf}\rangle) \\
\wedge\ & not_f(OCCUR(kick(_,_,_,_), \langle s_3, e_3\rangle)) \wedge Meets(\langle s_3, e_3\rangle, \langle s, s+1\rangle) \\
\wedge\ & not_f(OCCUR(deflect(player), \langle s_4, e_4\rangle)) \wedge Meets(\langle s_4, e_4\rangle, \langle s, s+1\rangle)
\end{aligned}$$

Motion Pattern 5.10 Event pattern for a *retreat from a fight* such that at least a single other agent remains in ball control.

$$OCCUR(retreat(\overbrace{player}^{ref}, \underbrace{\overbrace{\text{fight}}^{div}}_{type}), \langle s, s+1\rangle) \Leftarrow$$

$$\begin{aligned}
& FACT(distance(\text{ball}, player, \text{very_close}), \langle s, \text{inf}\rangle) \\
\wedge\ & FACT(distance(\text{ball}, player, \text{touch}), \langle s_1, e_1\rangle) \\
\wedge\ & Meets(\langle s_1, e_1\rangle, \langle s, s+1\rangle) \\
\wedge\ & (\ (\ FACT(fight_for_ball(), \langle s_2, \text{inf}\rangle) \\
& \quad \wedge Older(\langle s_2, \text{inf}\rangle, \langle s, s+1\rangle)\ ) \\
& \vee (\ FACT(x_ball_control(player_2), \langle s, \text{inf}\rangle) \\
& \quad \wedge FACT(fight_for_ball(), \langle s_3, s\rangle)\ )\ ) \\
\wedge\ & not_f(OCCUR(kick(_,_,_,_), \langle s_3, e_3\rangle)) \wedge Meets(\langle s_3, e_3\rangle, \langle s, s+1\rangle) \\
\wedge\ & not_f(OCCUR(deflect(player), \langle s_4, e_4\rangle)) \wedge Meets(\langle s_3, e_3\rangle, \langle s, s+1\rangle)
\end{aligned}$$

selection process amongst possible event candidates was guided by two trailing thoughts. On the one hand, patterns such as *kick* and *reception* can be considered mandatory due to the fact that the largest part of the action patterns specified in the following section such as *passes* depend immediatly upon those basic events. On the other hand, patterns were selected such that a certain alignment with equivalent patterns developed by Miene for the RoboCup *2D Soccer Simulation* in [Mie04a] is feasible.

5.3.2 Specification of Action Patterns

The following section introduces a selection of action patterns that are relevant in the RoboCup *3D Soccer Simulation League* and specifies them based on the formalisms worked out in section 5.2.2.

Motion Pattern 5.11 Action pattern for a *ball transfer* concluded by a normal reception of the ball by the player acting as transfer target.

$$
\begin{aligned}
&OCCURRING(ball\_transfer(\overbrace{source, target}^{ref}, \overbrace{dir^{glob:8}, height, force}^{div}), \langle s, e\rangle) \Leftarrow \\
&\quad OCCUR(kick(source, dir^{glob:8}, height, type), \langle s, e_1\rangle) \\
&\wedge \quad (\ type = \text{std} \ \vee \ type = \text{volley}\ ) \\
&\wedge \quad OCCUR(receive(target), \langle s_2, e\rangle) \\
&\wedge \quad Older(\langle s, e_1\rangle, \langle s_2, e\rangle) \\
&\wedge \quad (\ HOLDS(free\_ball(), \langle s_3, e_3\rangle) \\
&\qquad \vee\ (\ OCCUR(deflect(\_), \langle s_6, e_6\rangle) \\
&\qquad\quad \wedge\ During(\langle s_6, e_6\rangle, \langle s_3, e_3\rangle) \\
&\qquad\quad \wedge\ HOLDS(free\_ball(), \langle s_3, s_6\rangle) \wedge HOLDS(free\_ball(), \langle e_6, e_3\rangle)\ )\ ) \\
&\wedge \quad Meets(\langle s, e_1\rangle, \langle s_3, e_3\rangle) \wedge Meets(\langle s_3, e_3\rangle, \langle s_2, e\rangle) \\
&\wedge \quad Virtually\_Playon(\langle s, e\rangle) \\
&\wedge \quad FACT(acceleration(\text{ball}, \text{increasing}), \langle s, e_4\rangle) \\
&\wedge \quad HOLDS(velocity(\text{ball}, vel_s), \langle s-1, s\rangle) \\
&\wedge \quad HOLDS(velocity(\text{ball}, vel_e), \langle e_4, e_4+1\rangle) \\
&\wedge \quad Translate\uparrow^{force:5}_{accel:6}(vel_s, vel_e, force)
\end{aligned}
$$

Ball Transfers

The undoubtedly most important action pattern in human- as well as in simulated soccer is the *ball transfer* which constitutes the basis pattern for a number of specialized concepts as it describes the general kick-induced transfer of the ball over a certain distance of the field and its subsequent reception or immediate forwarding respectively, either by the initiator of the transfer itself or another player on the soccer pitch. For this thesis, two ball transfer patterns are required contrary to related approaches such as [Wen03] due to the introduction of the volley kick. The latter provokes two distinct ball reception scenarios. Consequently, the pattern specified in motion pattern (5.11), p.96 refers to the scenario of a normal reception by the target player while the alternative pattern specified in motion pattern (5.12), p.97 refers to the scenario of a pseudo-reception via volley kick.

As the basic concept of the ball transfer possesses rather sparse explanatory power, a threefold diversification is introduced which specifies the respective direction of the ball transfer in the extrinsic global frame of reference imposed by the 3D Soccer Server[23], the kick-height and the force of the kick. Thus, a concrete ball transfer between two players could be characterized as *executed with full kick force, low above the ground in southern direction*. It should be noted that both the direction and the height argument are taken over immediatly from the diversification of the kick event initiating the ball transfer.

This paradigmatic example points at the fact, that diversification arguments encode knowledge which can be propagated in a meaningful way from subordinated to more complex motion patterns. Such a propagation may be 1:1. However, the specification of the patterns that specialize the ball transfer further will show that propagation may involve a transformation such as a change of reference frame as well.

23 due to the fixation of what is the *left* and *right* half field of the soccer pitch

Motion Pattern 5.12 Alternative action pattern for a *ball transfer* concluded by a direct forwarding of the ball via volley kick performed by the player acting as transfer target.

$$
\begin{aligned}
&OCCURRING(ball\_transfer(\overbrace{source, target}^{ref}, \overbrace{dir^{glob:8}, height, force}^{div}), \langle s, e\rangle) \Leftarrow \\
&\quad OCCUR(kick(source, dir^{glob:8}, height, type), \langle s, e_1\rangle) \\
&\wedge\ (\ type = \text{std} \ \vee\ type = \text{volley}\ ) \\
&\wedge\ OCCUR(kick(target, \_, \_, \text{volley}), \langle s_2, e\rangle) \\
&\wedge\ Older(\langle s, e_1\rangle, \langle s_2, e\rangle) \\
&\wedge\ Meets(\langle s, e_1\rangle, \langle s_3, e_3\rangle) \wedge Meets(\langle s_3, e_3\rangle, \langle s_2, e\rangle) \\
&\wedge\ (\ HOLDS(free\_ball(), \langle s_3, e_3\rangle) \\
&\quad \vee\ (\ OCCUR(deflect(\_), \langle s_6, e_6\rangle) \\
&\quad\ \wedge During(\langle s_6, e_6\rangle, \langle s_3, e_3\rangle) \\
&\quad\ \wedge HOLDS(free\_ball(), \langle s_3, s_6\rangle) \wedge HOLDS(free\_ball(), \langle e_6, e_3\rangle)\ )\ ) \\
&\wedge\ Virtually\_Playon(\langle s, e\rangle) \\
&\wedge\ FACT(acceleration(\text{ball}, \text{increasing}), \langle s, e_4\rangle) \\
&\wedge\ HOLDS(velocity(\text{ball}, vel_s), \langle s-1, s\rangle) \\
&\wedge\ HOLDS(velocity(\text{ball}, vel_e), \langle e_4, e_4+1\rangle) \\
&\wedge\ Translate\uparrow^{force:5}_{accel:6}(vel_s, vel_e, force)
\end{aligned}
$$

The third diversification argument, the kick force is the result of a direct evaluation of the pattern incidence context. Based on the qualitative velocity of the ball right at the start (vel_s) and end (vel_e) of the acceleration phase that caused by the kick that starts the ball transfer, $Translate\uparrow^{force:5}_{accel:6}(\ldots)$ evaluates[24] a qualitative force measure based upon the qualitative distance between vel_s and vel_e as shown in figure 5.13.

Motion Pattern 5.13 Pattern for Virtual Playon

$$
\begin{aligned}
&Virtually\_Playon(\langle s, e\rangle) \Leftarrow \\
&\quad HOLDS(playmode(\text{playon}), \langle s, e\rangle) \\
&\vee\ (\ HOLDS(playmode(mode), \langle s, e_1\rangle) \wedge HOLDS(playmode(\text{playon}), \langle s_2, e\rangle) \\
&\quad \wedge (mode \neq \text{playon}) \\
&\quad \wedge Meets(\langle s, e_1\rangle, \langle s_2, e\rangle)\ )
\end{aligned}
$$

The specification of the ball transfer pattern also utilizes $Virtually\_Playon(\langle s, e\rangle)$ as extra sub-pattern which enforces the constraint that ball transfers may start in a standard situation (i.e. $\rightarrow$ *kick-in*, *corner-kick*, *kick-off*) as well as during the normal course of a game ($\rightarrow$ *playon*). However, once the action has been started via a kick such that the playmode has definitely changed to playon it needs to retain in this state for the remainder of the ball transfer. Thus, only those transfers are allowed which evolve as

24 This method is rather simplistic in its approach and more sophisticated solutions may exist. Yet, the simple approach suffices for the time being as the force argument of the diversification is not evaluated further in higher-level patterns.

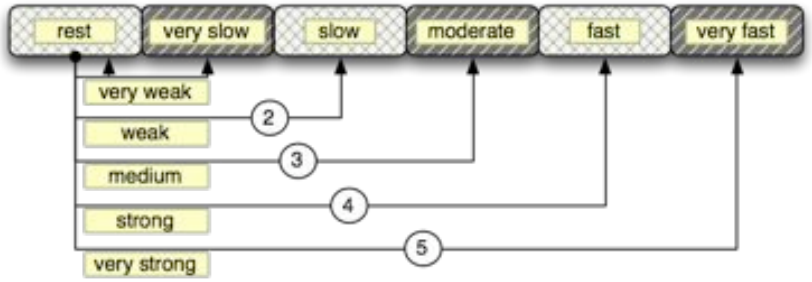


Figure 5.13: Schematic representation for the interpretation of the delta between start/end velocity ($vel_s \longrightarrow vel_e \;.\; vel_s \leq vel_e$) of a kick-induced acceleration phase as kick force. Note that the total order relation $\leq$ applies for the set of equivalence classes for the kick force.

Motion Pattern 5.14 Action pattern for *successful passes* between fellow players of the same team

$$OCCURRING(pass(\overbrace{kicker, receiver}^{ref}, \overbrace{dir^{team:6}, height, force, \underbrace{success}_{succ}}^{div}), \langle s, e\rangle) \Leftarrow$$

$$\begin{aligned}
& OCCURRING(ball_transfer(kicker, receiver, dir^{glob:8}, height, force), \langle s, e\rangle) \\
\wedge\ & (kicker \neq receiver) \\
\wedge\ & (team(kicker) = team(receiver)) \\
\wedge\ & Translate_FoR\uparrow^{team:6}_{glob:8}(dir^{glob:8}, team(kicker), dir^{team:6})
\end{aligned}$$

part of the normal course of the game. $Virtually\_Playon(\ldots)$ was first suggested by Wendler in [Wen03, p.50] (cf. figure 4.3 on page 34 in chapter 4) and is implemented for this thesis as shown in motion pattern (5.13), p.97. With respect to the formalization of action patterns in section 5.2.2, $Virtually\_Playon(\langle s, e\rangle)$ should be considered as an abbreviation which ties together elements from both $\mathbb{TR}_{Allen}$ and $\mathbb{F}_{infer}$.

The fundamental ball transfer pattern can be specialized by a further investigation of the players that act as source/target with respect to their identity and their team membership. In doing so, it is possible to distinguish passes between two distinct players from self assists by single players. Passes can be further classified as successful and thus probably intended passes, or as failed passes depending on whether source and target belong to the same team. As the success of a pass is a characteristic trait, an additional argument is introduced in the diversification of the pass pattern besides those taken over from the ball transfer. Both pass scenarios are specified in distinct patterns. Motion pattern 5.14 specifies successful passes, motion pattern (5.15), p.99 specifies failed passes. Finally, motion pattern (5.16), p.100 specifies the self assist pattern.

For passes as well as for the self assist, the frame of reference of the kick direction is adapted from an intrinsic global perspective imposed by the 3D Soccer Server used for the basic ball transfer pattern to another intrinsic perspective determined by the origin of the team[25] to whom the action is attributed [26]. This change of the frame

25 The origin is associated with a teams penalty area/goal which is either in the northern or souther half field viewed from the center point in the global frame of reference

26 determined by the agent that initiates the respective pass or self assist incidence with a kick

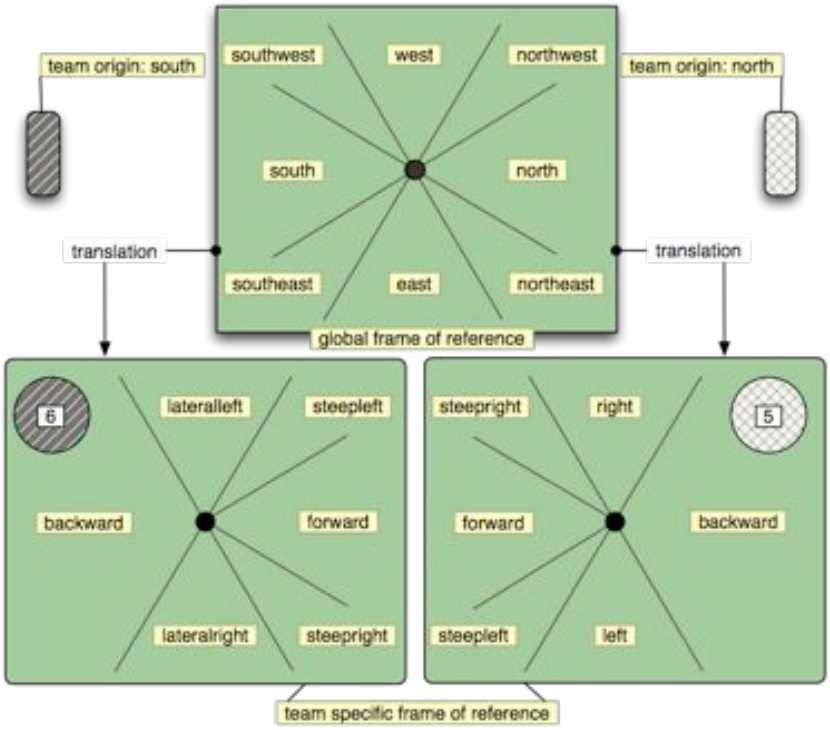


Figure 5.14: Schematic visualization of $Translate\_FoR\uparrow^{team:6}_{glob:8}(\ldots)$. Both set of equivalence classes for the global- and team-oriented perspective are shown and how they are transformed in dependence of the respective team origin.

Motion Pattern 5.15 Action pattern for undesired *failed passes*

$$OCCURRING(pass(\overbrace{kicker, receiver}^{ref}, \overbrace{dir^{team:6}, height, force, \underbrace{\text{failure}}_{succ}}^{div}), \langle s, e\rangle) \Leftarrow$$

$$\begin{aligned} & OCCURRING(ball\_transfer(kicker, receiver, dir^{glob:8}, height, force), \langle s, e\rangle) \\ \wedge\ & (kicker \neq receiver) \\ \wedge\ & (team(kicker) \neq team(receiver)) \\ \wedge\ & Translate\_FoR\uparrow^{team:6}_{glob:8}(dir^{glob:8}, team(kicker), dir^{team:6}) \end{aligned}$$

of reference for motion directions has been suggested[27] by Miene in [Mie04a, pp.91]. $Translate\_FoR\uparrow^{team:6}_{glob:8}(\ldots)$ is used to translate the eight directions of the global wind rose (cf. figure 5.14) into six directions which are tied more closely to the application domain of simulated soccer and thus contribute to a more intuitive understanding of the characteristics of particular pass and self assist incidences. Figure 5.14 also shows the spectrum of target classes and how they correspond to the input classes. The translation comprises a partial coarsening through the aggregation of three direction classes to a new

27 Miene reduces eight extrinsic, global direction classes to two extrinsic, yet team-depended motion classes *forward* and *backward*.

Motion Pattern 5.16 Action pattern for *Self Assist* by a single player

$$OCCURRING(self\_assist(\overbrace{player}^{ref}, \overbrace{dir^{team:6}, height, force}^{div}), \langle s, e\rangle)) \Leftarrow$$

$$\begin{aligned} & OCCURRING(ball\_transfer(kicker, receiver, dir^{glob:8}, height, force), \langle s, e\rangle) \\ \wedge\ & (kicker = receiver) \\ \wedge\ & Translate\_FoR\uparrow^{team:6}_{glob:8}(dir^{glob:8}, team(kicker), dir^{team:6}) \end{aligned}$$

Motion Pattern 5.17 Action pattern for *successful atomic dribbling*

$$OCCURRING(dribble\_atom(\overbrace{player}^{ref}, \overbrace{dir^{team:6}, \underbrace{\text{success}}_{succ}}^{div}), \langle s, e\rangle) \Leftarrow$$

$$\begin{aligned} & OCCURRING(self\_assist(player, dir^{team:6}), \langle s, e_1\rangle) \\ \wedge\ & FACT(x\_ball\_control(player), \langle s_2, e_2\rangle) \\ \wedge\ & Meets(\langle s, e_1\rangle, \langle s_2, e_2\rangle) \\ \wedge\ & OCCUR(kick(player, \_, \_, \text{standard}), \langle s_3, e_3\rangle) \\ \wedge\ & (\ Meets(\langle s_2, e_2\rangle, \langle s_3, e_3\rangle) \vee BornBeforeDeathOf(\langle s_2, e_2\rangle, \langle s_3, e_3\rangle)\ ) \\ \wedge\ & Meets(\langle s, e\rangle, \langle s_3, e_3\rangle) \\ \wedge\ & C\_HOLDS(distance(\text{ball}, player, \text{close}), \langle s, e\rangle) \end{aligned}$$

entailing class *back*. The coarsening is motivated by the fact that the soccer game is commonly developed forward, towards the goal line of the enemy, using passes and self assists such that a lesser resolution of backward-directed ball movement is sufficient and corresponds to human understanding of the game[28].

Atomic Dribbling

The next action pattern, the *atomic dribbling* is not so much an interesting pattern in its own right but rather a mandatory support pattern a_{supp} (cf. definition (5.15)) for a subordinated type 2 action sequence *dribbling* which will be described later in this chapter.

The atomic dribbling builds upon a self assist which is followed by a short period of exclusive ball control. Two possible scenarios exist for the conclusion of an atomic dribbling which are treated with dedicated patterns. First, a successful atomic dribbling, specified in motion pattern (5.17), p.100, ends with a kick. Second, an unsuccessful atomic dribbling, specified in motion pattern (5.18), p.101, leads to a fight for the ball as a player from the opposing team engages the dribbling player before the ball can be forwarded again. The success of an atomic dribbling is a characteristic trait which accounted for in the pattern diversification.

In both patterns for the atomic dribbling, the C_HOLDS-relation is employed to express

28 Consider live commentaries as an example, such as can be found in the description of possible game scenarios in section 3.1

Motion Pattern 5.18 Action pattern for *unsuccessful atomic dribbling*

$$OCCURRING(dribble\_atom(\overbrace{player}^{ref}, \overbrace{dir^{team:6}}^{div}, \underbrace{\text{failure}}_{succ}), \langle s, e\rangle) \Leftarrow$$

$$\begin{aligned}
& OCCURRING(self_assist(player, dir^{team:6}), \langle s, e_1\rangle) \\
\wedge\ & FACT(x_ball_control(player), \langle s_2, e\rangle) \\
\wedge\ & Meets(\langle s, e_1\rangle, \langle s_2, e\rangle) \\
\wedge\ & OCCUR(engage_fight(aggressor, type), \langle s_3, e_3\rangle) \\
\wedge\ & Meets(\langle s_2, e\rangle, \langle s_3, e_3\rangle) \\
\wedge\ & C_HOLDS(distance(\text{ball}, player, \text{close}), \langle s, e\rangle)
\end{aligned}$$

Motion Pattern 5.19 Action pattern for the *ball taming*, a special action pattern which documents deficiencies in the basic ball handling skills of the Virtual Werder team.

$$OCCURRING(ball\_taming(\overbrace{player}^{ref}), \langle s, e\rangle)) \Leftarrow$$

$$\begin{aligned}
& OCCUR(receive(player), \langle s, e_1\rangle) \\
\wedge\ & FACT(x_ball_control(player), \langle s, e_2\rangle) \\
\wedge\ & OCCUR(retreat(player, \text{ball}), \langle s_3, e_3\rangle) \\
\wedge\ & Meets(\langle s, e_2\rangle, \langle s_3, e_3\rangle) \\
\wedge\ & FACT(free_ball(), \langle s_3, e_4\rangle) \\
\wedge\ & OCCUR(receive(player), \langle s_5, e\rangle)) \\
\wedge\ & Meets(\langle s_3, e_4\rangle, \langle s_5, e\rangle)
\end{aligned}$$

the spatial constraint that during the complete course of a dribbling, the player needs to retain close to the ball which entails being very close or even in touch for certain sub-intervals. In fact, these patterns motivated the introduction of the C_HOLDS-relation in section 5.1.5 on page 78 in the first place.

A Virtual Werder Special Action – Ball Taming

The final action pattern which is specified within the scope of this thesis is a special pattern whose conception was motivated by a idiosyncrasy in the ball handling of the Virtual Werder agents. Due to lack of sophistication in the low-level skills, the players commonly have to struggle considerably to gain effective control over a ball approaching with considerable residual speed. Consequently, a player attains control over the ball, yet looses it during repositioning in order to forward the ball and needs to attain control again before any ball-centered action can be performed. The associated pattern is fittingly denoted as *ball taming* and specified in pattern 5.19.

The *ball taming* documents that it is possible to compile action patterns that bear particular relevance for a certain team, besides the general purpose action patterns that have been discussed before.

Motion Pattern 5.20 Action sequence pattern of type(1) for the *onetwo pass* amongst two fellow players

$$
\begin{aligned}
&OCCURRING(onetwo\_pass(\overbrace{intiator, partner}^{ref}), \langle s, e\rangle) \Leftarrow \\
&\quad OCCURRING(pass(initiator, partner, dir_1^{team:6}, \_, \_, \text{success}), \langle s, e_1\rangle) \\
&\wedge \quad Translate\uparrow_{team:6}^{team:2}(dir^{team:6}, \text{forward}) \\
&\wedge \quad FACT(x\_ball\_control(partner), \langle s_2, e_2\rangle) \\
&\wedge \quad Overlaps(\langle s, e_1\rangle, \langle s_2, e_2\rangle) \\
&\wedge \quad OCCURRING(pass(partner, initiator, dir_2^{team:6}, \_, \_, \text{success}), \langle s_3, e\rangle) \\
&\wedge \quad Translate\uparrow_{team:6}^{team:2}(dir_2^{team:6}, \text{forward}) \\
&\wedge \quad (Meets(\langle s_2, e_2\rangle, \langle s_3, e\rangle) \vee BornBeforeDeathOf(\langle s_3, e\rangle, \langle s_2, e_2\rangle))
\end{aligned}
$$

5.3.3 Specification of Action Sequence Patterns

Based on the set of action patterns that has been worked out in the previous section, two paradigmatic examples for action sequences, one for each type introduced in section 5.2.2, are specified. They are to show that it is possible to specify complex composite concepts concisely seizing the same formalism which have already been deployed for simple actions. This is due to the fact that the subordinate actions are available as building blocks for the specification of high level motion patterns such that the specification is performed on a growing level of abstraction while retaining the possibility to fall back on simple concepts/facts.

Onetwo-Pass

First, the *give'n'go-* or *onetwo pass* is specified in motion pattern (5.20), p.102 as an example for a simple multi-player action sequence. A coarsely forward-directed pass is played by an initiating player to a supporting team member which,after a short phase of exclusive ball control, forwards the ball back to the initiator again via a coarsely forward-directed pass. The constraint that both passes should be coarsely forward-directed is due to the characterization of the onetwo pass as a means of developing the game often outsmarting a particular defending player. $Translate\uparrow_{team:6}^{team:2}(\ldots)$ is used to coarsen the six team-specific direction classes shown in figure 5.14 on page 99 used in the pass diversification, the five possible direction classes except *backward* into the single new class *forward*. Then, due to the use of forward as a constant, the term $Translate\uparrow_{team:6}^{team:2}(dir^{team:6}, \text{forward})$ in the pattern specification expresses precisely the desired constraint.

Extended Dribbling

As a second action sequence pattern of type 2, the pattern for extended dribblings is specified in motion pattern (5.21), p.103. As already suggested in the formal description in section 5.2.2 on page 85 of patterns for type 2 action sequences, those patterns are primarily concerned with the proper composition of already recognized support patterns. This is reflected in the specification of the extended dribbling, which says that either a new dribbling has begun in which case the dribbling immediatly corresponds to the basis action. Alternatively, an existing, successful dribbling can be continued via another atomic

Motion Pattern 5.21 Action sequence pattern of type(2) for the *extended dribbling* composed of a sequence of atomic dribblings (cf. motion pattern (5.17), p.100 and motion pattern (5.18), p.101).

$$
\begin{aligned}
&OCCURRING(dribbling(\overbrace{player}^{ref}, \overbrace{dir^{team}}^{div}), \langle s, e\rangle) \Leftarrow \\
&\quad OCCURRING(dribble\_atom(player, dir^{team:6}, \text{success}), \langle s_1, e\rangle) \\
&\wedge \quad (\quad (\quad OCCURRING(dribbling(player, dir_1^{team:6}, \text{success}), \langle s_2, e_2\rangle) \\
&\qquad \wedge Meets(\langle s_2, e_2\rangle, \langle s_1, e\rangle) \\
&\qquad \wedge (dir_1^{team:6} = dir^{team:6}) \\
&\qquad \wedge HeadToHeadWith(\langle s, e\rangle, \langle s_2, e_2\rangle) \quad) \\
&\qquad \vee Equals(\langle s, e\rangle, \langle s_1, e\rangle) \quad)
\end{aligned}
$$

dribbling.

As an additional spatial constraint for extended dribblings, it is required that a dribbling that is a composition of multiple atomic dribblings retains a fixed direction.

The specification of the extended dribbling concludes the specification of event-, action- and finally action sequence patterns that are to be employed in the detection of concrete motion situations. Not all of the actions called for in section 3.4 have been seized in the development of concrete action patterns for the detection process such as *scoring*. As a consequence, the extension of the pattern pool will be an issue in the outlook at the end of this thesis.

5.4 Detection of Extensive Motion Instances

Over the course of the current chapter, it has hitherto been shown in section 5.1, how the fundamental tier of a qualitative knowledge base comprised of spatio-temporal atomic facts can be compiled incrementally during the term of games of simulated soccer (viz. $f \in \mathbb{F}' \overset{inst}{\leftarrow} \mathbb{F}_{assert}$). Basic means were introduced for simple inferences upon the existing fact pool $\mathbb{F}'$ via the HOLDS and C_HOLDS predicates ($\mathbb{F}_{infer}$). Subsequently, in section 5.3, a knowledge engineering process contributed another tier of qualitative knowledge in the shape of motion patterns ($\mathbb{E} \cup \mathbb{A}$) which explicitly encode domain expertise with respect to the composition of events, action and action sequences.

The detection of concrete motion incidences, which is considered synonymous with the term *spatio-temporal analysis of dynamic scenes* with respect to the scope of this thesis, is to be performed interchangeably with successive passes of qualitative abstraction. It can be understood as a matching task of the spatio-temporal patterns ($\mathbb{E} \cup \mathbb{A}$) against the momentary fluent content ($\mathbb{F}' \cup \mathbb{E}' \cup \mathbb{A}'$) of the qualitative knowledge base which comprises the qualitative description of a particular game so far. Lattner notes that "*the process of pattern matching assigns objects to variables of a pattern in a way that all conditions of the pattern are satisfied.*" [Lat07, p.24]. Successful matchings against the specified motion patterns lead to new motion incidences which in return broaden the fluent content of the knowledge base and thus allow for the finding of successive matches (i.e. $e \in \mathbb{E}' \overset{inst}{\leftarrow} \mathbb{E}$ or $a \in \mathbb{A}' \overset{inst}{\leftarrow} \mathbb{A}$ respectively).

In the following, a methodology for the concrete implementation of the pattern matching task is presented.

5.4.1 Basic Detection Methodology

First of, with respect to the concrete implementation of the matching methodology in a functional technical system, the adaption of general purpose means for patterns matching has been considered a superior alternative compared to the development of a self-grown matching solution based on incremental event/action tracking as proposed by [Wen03, Mül02]. This decision is clearly substantiated by preliminary research by Jan Gehrke who demonstrated in [Geh05] the feasibility of a successful employment of a general purpose, logic-based reasoning system (XSB Prolog [SSW+06b, SSW+06a]) for the detection of event incidences under real-time constraints for a limited set of actors in simulated urban and highway traffic scenarios. The approach proved that logic programming is in principle well suited for generalized pattern matching and can be extended to handle the special case of *spatio-temporal pattern matching* as well. At the same time, the use of a logic-based reasoning system is attractive due to the possibility to profit from declarative programming, and still bears the promise of sufficient efficiency for real-time employment. Alternative approaches such as [NM06] which are concerned with scene interpretation using description logics have been studied as well. However the published results in that area suggest, that as of writing this thesis, logic programming is still better suited for a pragmatic system design, intended for this thesis.

In logic programming, which is the most widely used form of automatic reasoning according to [RN95], logical inference is realized via backward chaining strategies. As the motion patterns compiled in section 5.3 correspond to *first-order definite clauses* where the *premise* (body) specifies the inner composition of the respective pattern while the *consequence* (head) specifies the pattern signature, the formalization of the motion patterns worked out so far can be transfered with minimal change into a concrete logic programming language such as Prolog[29]. In fact, the formalization of motion patterns in section 5.3 already seized two meta-logical functions $(not_f(\ldots), setof(\ldots) \in \mathbb{META} \subset dom(P_{act|event}(c,i)))$ which have been introduced with Prolog in order to enhance expressiveness for pattern specification. Once the spatio-temporal predicates/functions $\mathbb{SR} \cup \mathbb{TR}_{Freksa} \cup \mathbb{TR}_{Allen}$ are expressed in a suitable way, the spatio-temporal pattern matching can handled via standard logical inference that relies upon sophisticated backward chaining algorithms whose basic principles are outlined in AI textbooks such as [RN95, pp.287]. Consequently, the remainder of this section will focus on how to employ logical inference for pattern matching effectively.

A first step in the implementation of an efficient, incremental detection strategy involves finding a course order of detection for the set of available motion patterns ($\mathbb{E} \cup \mathbb{A}$) to be applied beneficially in each consecutive detection pass. For that matter, it is possible to arrange the motion patterns in a hierarchy with respect to the maximum level of complexity of the substantial constituents ($\mathbb{F}_{assert} \cup \mathbb{F}_{infer} \cup \mathbb{E}\{\cup\mathbb{A}\}$) in their respective pattern body as follows: First, atomic ($\mathbb{F}_{assert}$) as well as inferred atomic facts ($\mathbb{F}_{infer}$) constitute a fundamental level of complexity (level 0). The hierarchical level of the extensive motion patterns, both events and actions alike, evaluates to the level immediatly above the highest level of any substantial constituent appearing in the respective pattern body. Thus, the patterns for the *exclusive kick* or the *reception of the ball* reside on level 1, while the *ball transfer* whose specification comprises both of the aforementioned patterns besides simple facts resides on level 2.

Each consecutive detection pass d_i can then be implemented as a bottom-up search for new motion incidences with respect to the aforementioned pattern hierarchy. Each pass begins with patterns of level 1 whose body can be matched against the momentary pool of atomic facts $\mathbb{F}'_i$ alone, either directly in case of FACT(...) constituents or indirectly

29 Prolog (PROgramming in LOGic) is mentioned here as it is undoubtedly the most widely employed exponent of logic programming languages with considerable employment in both research and economy.

via subordinate matching of patterns associated with HOLDS(...) and C_HOLDS(...). As new motion incidences are detected (viz. assignments of the respective pattern head-/signature not yet entailed in $\mathbb{E}'_i \cup \mathbb{A}'_i$ are found) the results are fed back immediatly into the pool of momentary fluent content ($\mathbb{F}'_i \cup \mathbb{E}'_i \cup \mathbb{A}'_i$). This is crucial as it allows for an immediate intra-pass reuse of the partial detection yield attained so far. Higher-order patterns whose body comprises subordinated events and/or actions can be matched subsequently against a freshly updated pool of fluent content. In particular OCCUR(...) and OCCURRING(...) constituents in the respective pattern body can be matched directly against $\mathbb{E}'_i \cup \mathbb{A}'_i$[30] due to the bottom-up detection which makes sure, that pattern constituents are always detected beforehand.

Thus, in order to detect new *ball transfer* incidences (ball transfer → level 2), a direct matching is performed against already asserted *kick-* and *reception* incidences (kick,reception → level 1). In order to detect new *pass/self assist* incidences (pass/self assist → level 3), a direct matching is performed against already asserted *ball transfers*. And finally, in order to detect new *onetwo pass* incidences (onetwo pass → level 4), a direct matching is performed against already asserted *passes*.

5.4.2 Balancing the Amount of Atomic Facts as the Game Evolves

The detection process outlined so far features a signification Achilles' Heel in the shape of the charging level of the qualitative knowledge base with respect to the fluent contents. The problem grows in significance as the number of passes for both qualitative abstraction increases. The underlying problem is that the performance of the pattern matching task with respect to execution speed is reciprocal to the size of asserted low-level facts and – to a far lesser extent – the motion incidences that constitute the pool of match candidates.

With respect to a balancing of $\mathbb{F}'$, it is instructive to review the spectrum of atomic qualitative facts which are compiled relentlessly in the successive passes of the qualitative abstraction with respect to the stated goal of the spatio-temporal analysis at hand which is to compile an expressive, qualitative description of the situation on the soccer pitch such that the agents' understanding of the game is increased and behavior/skill development becomes feasible on a higher, more intuitive basis (cf. section 1.2.1 on page 4 for reference).

Contemplating the growing $\mathbb{F}'$ with respect to this statement of affairs, it must be noted that for the largest part of asserted facts such as:
$HOLDS(distance(\text{ball}, \text{pl}_1, \text{medium}), \langle 330, 520 \rangle)$
it holds that they posses no immediate informational value for a coach or player in their own right. Rather, they act solely as building blocks for higher-level concepts. Thus, once it becomes clear, that those facts will no longer contribute any more to the matching of new motion instances, as those are detected based on facts that reside in a certain time frame of maximal extend measured from the momentary present of detection, they can be safely removed from $\mathbb{F}'$ without losings in the relevant knowledge stored in the fluent content of the knowledge base or constraints on the detection process. To be more precise, those facts whose validity has expired such that the whole validity interval of the particular fact resides completely beyond the relevant time frame in the past can be subjected to oblivion.

For a concrete implementation of a fact oblivion strategy it is mandatory to have a suitable heuristics that determines the extend of the oblivion time frame. A simple, yet sufficient approach is to empirically determine the maximum length l of any motion incidence in the concrete application scenario whose associated motion pattern still seizes constituents $\mathbb{F}_{assert} \cup \mathbb{F}_{infer}$ in its pattern body. It is then safe enough to assume that for

30 $\mathbb{E}'_i \cup \mathbb{A}'_i$ entails both detection results from completed previous detection passes d_0 to d_{i-1} as well as the partial detection yield obtained so far within the momentary detection pass d_i.

any practical means, facts whose validity expired before $now - l$ can be removed.

The employment of an oblivion strategy should balance the size of $\mathbb{F}'$ effectively. The subject of fact oblivion is addressed again in the description of the concrete system implementation in chapter 6 and in the evaluation in section 7.4.3.

5.4.3 Guidance for Detection

With respect to the real-time efficiency demands on the detection of new motion incidences in successive passes d_i, it is compulsory to employ a detection strategy which is suitable to selectively guide the course of the recognition, rather than resorting to a brute force strategy that tries and detect new incidences for each available motion pattern in $\mathbb{E} \cup \mathbb{A}$, regardless of the particular situation on the simulated soccer pitch and the detection yield obtained already in an existing cycle. The goal is to try and perform the matching for a certain pattern only if the current course of the detection pass has hitherto already provided necessary and sufficient indicators for either a certain detection success or the well founded possibility thereof. Appropriate decision heuristics for or against detection attempts can be derived of the observation that the motion incidences are not necessarily independent entities but may also be related via certain relationships to subordinate incidences such that the detection of the latter during a detection pass *triggers* the detection of superordinate motion incidences in the first place. In the following paragraphs, two *trigger scenarios* are considered, namely *specialization triggering* and *tail_to_tail triggering*.

First, in section 5.3.2 it has been shown with the family of ball transfer actions, that motion patterns can sometimes be *direct specializations* of subordinate patterns (such as *pass*,*self_assist* – *ball_transfer*). Due to the fact that specialized patterns are detected subsequent to their respective generalization during the course of a detection pass as they are in a natural way located higher in the pattern hierarchy, it is evident that a detection attempt is only promising once the detection of the generalized pattern has already yielded a concrete result. In this case, the general motion incidence is due to be further specialized within the same detection pass. Otherwise, it is sound to skip the detection attempt for specialized concepts as they are bound to fail. Thus, *specialization triggering* is a valid first means of guidance for the detection process.

However, with respect to the concrete pool of motion patterns developed in section 5.3 it becomes evident that *specialization triggering* alone can provide only moderate savings in the number of detection attempts per pass due to the fact that direct pattern specialization is used sparsely. However, a second relationship can be derived from the internal composition of the specified motion patterns which leads to the means of *tail_to_tail triggering* which is applicable for a greater subset of patterns. The concept can be exemplified via a closer inspection of *ball transfer* incidences which reveals that the ending of those incidences coincides necessarily with the ending of a particular subordinate motion incidence, namely regular *ball receptions* or a direct forwarding of the ball via *volley kick*. This effect is a direct consequence of the concrete specification of the associated *ball transfer* motion pattern(s) as introduced in section 5.3.2.

Generalizing from the particular example to the general circumstance it can be said, that for a particular motion pattern, one or more closing events/actions may exist, which are denoted here as *atomic tail_to_tail triggers*, paired with a distinct fulfillment policy for the special case of multiple *atomic tail_to_tail triggers*. Within a detection pass d_i the detection of a pattern associated with a single *tail_to_tail trigger* is attempted once a trigger incidence has been added already to the momentary, partial detection yield. For the detection of a pattern associated with multiple *tail_to_tail triggers*, the *fulfillment policy* determines that a detection is attempted either if instances for at least a single *tail_to_tail trigger* ($\rightarrow$ *existential-policy*, mnemonic: $\exists$) or all specified *tail_to_tail*

Level	Target Pattern	Trigger Type	Trigger Pattern
Events			
1	$kick(\ldots, \text{std})$	TTT	$\overset{starts}{\rightarrow} ball\_free()$
2	$kick(\ldots, \text{co})$	SPE	$kick(\ldots, \text{ex})$
1	$receive(\ldots)$	TTT	$\overset{starts}{\rightarrow} ball\_free()$
1	$start\_fight()$	TTT	$\overset{starts}{\rightarrow} fight\_for\_ball()$
1	$deflect(\ldots)$	TTT	$\overset{starts}{\rightarrow} ball\_free()$
Actions			
2	$ball\_taming(\ldots)$	TTT	$receive(\ldots)$
2	$ball\_transfer(\ldots)$	TTT($\exists$)	$receive(\ldots)$; $kick(\ldots, \text{volley})$
3	$pass(\ldots)$	SPE	$ball\_transfer(\ldots)$
3	$self\_assist(\ldots)$	SPE	$ball\_transfer(\ldots)$
2	$dribble\_atom(\ldots)$	TTT($\exists$)	$kick(\ldots)$; $engage\_fight(\ldots)$
Action Sequences			
3	$dribbling(\ldots)$	TTT	$dribble\_atom(\ldots)$
4	$onetwo\_pass(\ldots)$	TTT	$pass(\ldots)$

Table 5.1: *Specialization triggering*- (SPE) and *tail_to_tail triggering* (TTT{$\exists\|\forall$}) relations for the event patterns (cf. section 5.3.1, pp.90) and action patterns (cf. section 5.3.2, pp.95) specified for the RoboCup *3D Soccer Simulation*.

triggers ($\rightarrow$ *universal-policy*, mnemonic: $\forall$) have been added to the detection yield.

Both means, the *specialization triggering* as well as the *tail_to_tail triggering* can be used in an XOR-fashion for a considerable subset of all specified motion patterns thus leading to a noteworthy decrease in the number of actually attempted detections with respect to the maximum number of possible attempts, especially for those detection passes when the ball is traveling freely after a kick. The association of appropriate triggers is part of the knowledge engineering task and succeeds the formal specification of the motion patterns. The trigger associations which have been identified for the pattern pool developed in section 5.3 are specified in table 5.1. The table embraces only a subset of specified patterns as it is not always feasible to find suitable triggers of the given kind.

The influence of the detection strategy is evaluated in section 7.4.2 on page 139 where the fundamental brute force detection strategy is compared performance-wise with the guided detection approach outlined above.

A possible road for further investigation with respect to an enhanced detection guidance would comprise the introduction of additional triggers based on certain properties of the momentary situational context that do or do not hold for particular detection passes as well as events/occurrences/general change in the situational context. It is expected that further futile detection attempts could be avoided with such an extension.

6 The Realized Analysis Framework

Subsequent to the conceptual design of the spatio-temporal analysis system for dynamic scenes with a focus on real-time aptitude, the chapter at hand is concerned with the realization of a working prototype of the proposed analysis approach.

Section 6.1 takes up the thesis' leitmotif with an overview of the implemented analysis framework, in the following referred to as *SCD* which is an abbreviation for: *Segmentation, Classification & Detection*[1]. The overview describes the components that have been implemented as part of this thesis and how they integrate into the RoboCup *3D Soccer Simulation* environment.

Based on this introduction, section 6.2 on page 111 delves deeper in the modular design and the implemented functionality of the *SCD Core*. The course of action within each analysis cycle is detailed where section 6.2.1 is concerned with the qualitative abstraction while the detection of extensive motion incidences is discussed subsequently in section 6.2.2.

In succession, section 6.3 on page 120 is concerned with the integration and deployment of the *SCD Core Library* both into agents residing in the RoboCup 3D Soccer Simulation and standalone applications. The former deployment is denoted as *LiveSCD* and was implemented exemplary for an advanced *SCD Coach* agent[2]. The latter deployment was realized for evaluation purposes in the *SCD Simulator*.

The chapter concentrates on providing a high level overview of the implemented functionality. For a deeper, code-level description, it is referred to the doxygen-documentations (in `html`) that are included in the DVD distributed with this thesis (cf. appendix A.1).

6.1 Overview of the SCD Framework

In order to provide the reader with a general overview of the implemented *SCD framework* in the first instance, a suitable starting point is the description of the distinct components whose interplay renders possible the desired SCD analysis operation and the way the SCD

1 The synonym *SCD* was chosen as the *SC* part reflects the task area of qualitative abstraction (cf. section 5.1) while the *D* part reflects the detection of extensive motion incidences (cf. section 5.4).

2 derived from the standard *Virtual Werder 3D Head Coach*, a fusion of a *UTUtd 3D coach agent* [ABD+06, p.2] and a standard RoboCup *3D Soccer Monitor* [O+06]

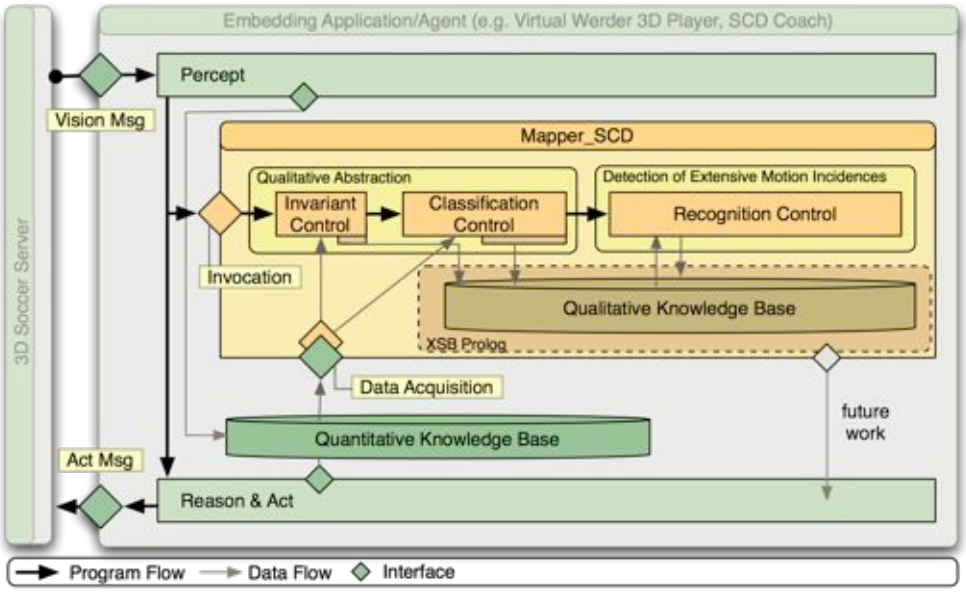


Figure 6.1: High-Level overview of the SCD software module (in yellow/brown tones) paradigmatically embedded into a RoboCup *3D Soccer Simulation League* agent (in green/grey tones).

framework can be embedded and conducted by applications.

The SCD framework was designed as a self-contained, platform-independent software module which can be integrated easily into existing applications such as Virtual Werder 3D soccer agents (both coach and players) acting in the RoboCup *3D Soccer Simulation* environment[3]. The claim about platform-independence is substantiated, since the SCD framework was developed and tested primarily on an Apple Powerbook (1.25 GHz G4 processor → *PowerPC* architecture) running Mac OS 10.4 'Tiger', with additional development and testing being performed on standard PC hardware (Intel P4 3.0 GHz processor → *x86* architecture) running Ubuntu 7.04 'Feisty Fawn'. Thereby, both architectures are supported by all SCD framework components without restrictions. In order to induce low integration costs on the embedding client[4], the SCD module exposes a concise application interface to its clients. It is basically comprised of a.) an abstract interface that needs to be implemented for the retrieval of elements of the momentary quantitative world state perception (cf. figure 6.1, 'data acquisition'), and b.) a management module that encapsulates the analysis functionality and exposes solely a single invocation method that allows clients to activate the execution of a complete analysis cycle (cf. figure 6.1, 'invocation'). This type of minimal interface to the implemented SCD software module was chosen in order to achieve a broad applicability. Irrespective of the particular implementation of the quantitative knowledge base of a SCD client, once a bridge that implements SCD's abstract input interface exists, the SCD software-module can be embedded immediatly.

Over the complete course of its employment, the SCD module pursues the incremental compilation and maintainance of a comprehensive qualitative knowledge base. Therefore, in each percept-reason-act cycle, the analysis is to be called subsequent to the processing and integration of a fresh incoming vision message into a quantitative world model and

3 Figure 6.1 outlines schematically how the *SCD framework* is embedded in an RoboCup soccer agent

4 In the context at hand, the term 'client' refers to the respective application which integrates/uses the functionality provided by the *SCD software module*. This denotation is distinct from the understanding of 'client' as in the client/server distinction.

conceptually before the execution of an agent behavior[5].

A working SCD module that performs its two primary tasks, the qualitative abstraction of dynamic scenes provided through the global input interface and in succession a detection of extensive motion incidences, is a hybrid software entity with a C++ and a Prolog aspect. While the interface to client applications, the control logic for the SCD operations and the qualitative abstraction are implemented in C++[6], the qualitative knowledge base and important parts of the detectors for extensive motion situations rely upon an XSB[7] Prolog backend. The connection of the Prolog system is managed transparently for the respective embedding application by the SCD control.

Due to its hybrid character and a software design which emphasizes in particular comprehensive configuration capabilities of both qualitative abstraction and detection, the SCD framework is comprised of three main constituents: first, the SCD Core library. Second, the XML-configuration which determines both scope and style of the qualitative abstraction and the scope of the detection. And finally, a set of Prolog rules.

6.2 SCD Core Functionality

The preceding section detailed that from the point of view of an application embedding the SCD system, its functionality is hidden behind a particular control module which can be asked explicitly to perform a complete analysis pass of the dynamic scene, based on an updated momentary world state perception accessible via the input interface (denoted further on as *Extractor_Worldstate*). Internally, this main SCD control module, called *Mapper_SCD*, integrates the complete SCD functionality which is provided by three specialized submodules: a.) the mapping of invariant facts controlled by the *Invariant_Control*, b.) the mapping of fluent, atomic facts controlled by the *Classification_Control* and finally c.) the detection of motion incidences controlled by the *Recognition_Control*. The first two submodules jointly realize the qualitative abstraction. That is, they transform the quantitative world state data provided by the *Extractor_Worldstate* into a fundamental pool of qualitative facts that is stored in the Prolog knowledge base. The detection of motion incidences is performed subsequent to the qualitative abstraction. At the time of writing, the detection relies on the available qualitative data alone. Thus, it is detached from the quantitative input provided by the embedding application. The outlined statement of affairs is shown concisely in figure 6.1 on the facing page.

6.2.1 Qualitative Abstraction

The first aspect of qualitative abstraction, performed by the *Invariant_Control* bears particular relevance during the initial phase of the analysis of a simulated soccer match. It comprises the compilation of *invariant facts* that are dependent on the concrete game at hand such as the *team and role association* of each player and the *team origins*. Conceptually, *invariant facts* differ from atomic facts $\mathbb{F}'$ as introduced in the concept chapter in that they are not fluent in character, i.e. bound to a certain validity interval[8]. Rather, once asserted in the scope of an analysis run which retains a fixed game context, those

5 Due to the hitherto missing interface for an exploitation of the compiled qualitative knowledge, analysis results are not yet fed back to the embedding application for further exploitation.

6 relying heavily upon object-oriented software design techniques such as advanced polymorphism and template programming and the excellent *Boost C++ Libraries* [Kar05] http://www.boost.org (visited:28/09/2007)

7 The most recent stable versions XSB 3.0.1 'Sagres' and XSB 3.1 'Incognito' both available at http://www.xsb.sourceforge.net (visited:28/09/2007) were used

8 Consequently they are expressed as simple predicates $p \in \mathbb{P}'$ rather than via $FACT(\text{pred}, \text{i}) \in \mathbb{F}' \,.\, i \in I_{\parallel} \cup I_{+}$ (cf. section 5.1.5 on page 78)

facts can be considered unchangeable background information which can be exploited in motion patterns[9] besides regular atomic facts.

The second, comparatively extensive aspect in the qualitative abstraction, performed by the *Classification_Control* module, comprises the abstraction of the world state data into the qualitative ground predicates specified in section 5.1.1 on page 56 using the classification concepts that have been proposed in section 5.1.3 on page 64, and thereupon the compilation of atomic facts with temporal dimension using the 'explicit-lengthening' fact assembly strategy proposed in section 5.1.1 on page 56.

Both modules, *Invariant_Control* and *Classification_Control*, are called in immediate succession by the supervising *Mapper_SCD* instance at the beginning of each external analysis invocation. Subsequent to a successful invocation of both abstraction-related management modules, the qualitative knowledge base has been updated such that based here-upon, a new detection cycle for extensive motion incidences can be initiated.

Compilation of Invariant Facts

Following the order of invocation in the analysis cycle, the *Invariant_Control* is described first in more detail.

When the *Invariant_Control* module is instantiated on startup, based on the specification in the SCD *Configuration* the desired subset of specialized invariant mappers, each responsible for a single type of invariant facts is loaded into the system. The small set of implemented mappers comprises:

Mapper_Invariant_Role $\rightarrow role(player, sym_{role})\ .\ sym_{role} \in \{\text{goalie}, \text{field\_player}\}$

Mapper_Invariant_Team $\rightarrow team(player, team\_name)$

Mapper_Invariant_Origin $\rightarrow origin(team\_name, sym_{orig})\ .\ sym_{orig} \in \{\text{south}, \text{north}\}$

When the *Invariant_Control* is called by the *Mapper_SCD*, the loaded mappers are called sequentially. However due to their function to compile invariant facts, the mappers need not be called in each successive analysis pass over the whole course of the game. Typically, once the *Extractor_WorldInfo* provides sufficient raw data, the mappers can compile the full set of invariant facts, such as the role and team association for each player participating in the soccer match at hand, in a single analysis cycle and immediatly store those facts in the qualitative knowledge base for further use via a specialized messenger class (*XSB_Messenger_Invariant*), that provides a concise interface for the generic assertion of invariants. This messenger is built upon the efficient XSB low-level C-interface[10]. Once a particular mapper has thus fulfilled its intended purpose, it informs the *Invariant_Control* of its success which consequently removes that particular mapper from its call map.

Thereby, soon after the beginning of an analysis run, the *Invariant_Control* runs out of work and subsequent invocations thereof preceding the call to the *Classification_Control* return immediatly without the *Invariant_Control* contributing further to the qualitative abstraction.

Compilation of Atomic Fluent Facts

Once called by the *Mapper_SCD*, the scope of duties of the *Classification_Control*, whose design is shown schematically in figure 6.2 on the next page, can be distinguished in a.)

9 Confer to the pass patterns (cf. patterns 5.11, 5.12) specified in section 5.3.1 as they seize the team association of source/target of a ball transfer which is an invariant fact in order to determine the success characteristic.

10 documented in (→ http://xsb.sourceforge.net/manual2/node34.html (visited:27/09/2007)).

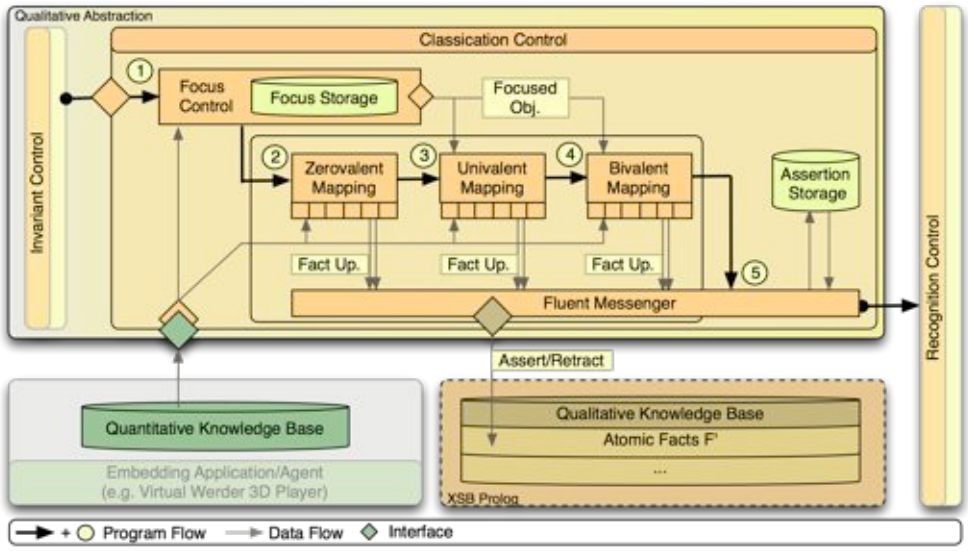


Figure 6.2: Schematic overview of the *Classification_Control* module which is responsible for the qualitative abstraction of fluent atomic facts ($\mathbb{F}'$).

a pre-processing step which comprises the application of a focus strategy (cf. figure 6.2, step (1)), b.) the execution of the actual mapping operation (cf. figure 6.2, steps (2)–(4)) and c.) a post-processing step which comprises the application of the implemented oblivion strategy for atomic facts (cf. figure 6.2, step (5)).

Step 1: Pre-Processing or Applying the Focus Strategy In section 5.1.6 on page 82, it was argued that with regard to an efficient implementation of the qualitative abstraction, it is essential to determine a subset of all objects and object pairings that, according a dedicated heuristic, is considered 'relevant'[11] such that only those qualitative predicates which refer to 'relevant' objects/object relations need to be tracked by the qualitative abstraction. The *Focus_Control* implements the heuristic proposed in section 5.1.6. When the module is called in an abstraction cycle, it determines the sets of objects and object pairs which reside within the momentary focus of attention. Subsequently, a reconciliation is performed with the stored sets of focused objects/object pairs compiled in the previous call. Thus it is possible to determine additionally the respective sets of those objects/object pairs that either just left or entered the focus of attention. All of these pieces of focus information are stored internally and can be retrieved by the *Classification_Control* while conducting the three stages of actual quantitative-to-qualitative mapping.

Before continuing with a high-level description of the latter process, it should be mentioned that the focus heuristic proposed in this thesis has been hard-coded within the *Focus_Control* module for the prototype implementation. However, as the *Focus_Control* module effectively encapsulates the heuristic implementation, a more flexible, configurable replacement thereof is conceivable since necessary changes are localized and thus do not affect other parts of the SCD framework.

Steps 2-4: Mapping of Atomic Facts Once the pre-processing task is completed, the actual incremental compilation of atomic facts $\mathbb{F}'$ can be performed. As figure 6.2 indicates,

11 The scope of what is 'relevant' is essentially preset by the employed pool of motion classes whose concrete occurrences are to be detected.

the process is decomposed in three successive tiers. In each thereof, atomic facts that encapsulate predicates of a certain *reference arity*[12] are compiled. The particular mapping steps, denoted as *Zerovalent Mapping*, *Univalent Mapping*, and *Bivalent Mapping* in figure 6.2, are not especially encapsulated in particular management modules. Rather, the *Classification_Control* manages three sets of *mapping engines*. Each of those can be conceived as a self-contained *classification unit* that bears responsibility for the incremental compilation of facts associated with a distinct predicate equivalence class[13].

Specialized versions of these mapping engines exist that handle predicates of the aforementioned reference arities. The respective instantiations thereof which are embraced in one of the sets managed by the *Classification_Control* share a common invocation interface.

In a default deployment of the SCD framework as implemented for this thesis, the first managed set (associated with *Zerovalent Mapping*) entails a single mapping engine for the *playmode*. The second set (associated with *Univalent Mapping*) entails mapping engines for *velocity*, *acceleration*, *z_position*, *z_position_trend*, *motion_dir* and the incidence region *in_region* of movable objects. The third set (associated with *Bivalent Mapping*) finally entails mapping engines for *distance* and *s_orientation* of pairs of movable objects. Thus, the mapping engines span the complete set of qualitative predicates specified in section 5.1.1, pp.57. It is possible to customize the respective sets of mapping engines for an analysis run as each can be deactivated independently in the SCD configuration.

During a complete cycle of qualitative abstraction, the supervising *Classification_Control* iterates sequentially over the three managed sets of mapping engines, thereby calling each with their respective subset of relevant focus data, provided by the *Focus_Control*. Then, internally each mapping engine handles the details of the compilation of new facts associated with its predicate equivalence class without requiring further interplay with the *Classification_Control*. It should be noted that due to the mutual independence of the mapping engines, the particular mapping order constitutes only a convention and that furthermore, from a conceptual point of view, a concurrent implementation is conceivable for future performance optimization of the mapping process[14].

In order to achieve a flexible implementation that can be easily extended with additional mapping engines, the latter have been designed as generic control modules which rely upon means provided by the *Classification_Control* on creation for a.) the individually suitable acquisition of quantitative input data and b.) the insertion of compiled facts into the qualitative knowledge base (cf. figure 6.3 on the facing page).

For data acquisition, the *Classification_Control* manages a comprehensive set of specialized *extraction adaptors* that have access to the global *Extractor_WorldInfo* and thus the full momentary world state information (cf. section 5.1.1, pp.56). These adaptors extract and pre-process subsets of the available raw input for immediate further use in the classification process conducted by the respective mapping engines. Conceptually, each of the implemented adaptors can thereby be thought of as access to one of the uni- or multivariate time series specified in section 5.1.1, pp.57.

The *Classification_Control* also holds a specialized messenger module, the *Messenger_Fluent* that allows for performance-optimized assertion/retraction of arbitrary atomic

12 The concept of predicate's *reference* was introduced in section 5.2.3 on page 88. The *reference arity* denotes the amount of reference attributes which can be 0 ($class(pred) \in \mathbb{P}^0$), 1↔single object ($class(pred) \in \mathbb{P}^1$) or 2↔object pair ($class(pred) \in \mathbb{P}^2$).

13 With respect to the predicates considered here, the term 'equivalence class' denotes the set of concrete predicates that share the same signature, i.e. a.) the denominator and b.) the attribute list such as $velocity(obj, sym_{vel}) \in \mathbb{P}^1 \,.\, obj \in \mathcal{MO},\ sym_{vel} \in \mathbb{SYM}_{velocity}$ or $distance(obj_1, obj_2, sym_{dist}) \in \mathbb{P}^2 \,.\, obj_i \in \mathcal{MO},\ sym_{dist} \in \mathbb{SYM}_{dist}$

14 This assessment is based on the conceived software design for the *Classification_Control* and the XSB Prolog support for multi-threaded programming introduced with the 3.0.1 'Sagres' release [SSW+06b, chapter 7, p.173–184].

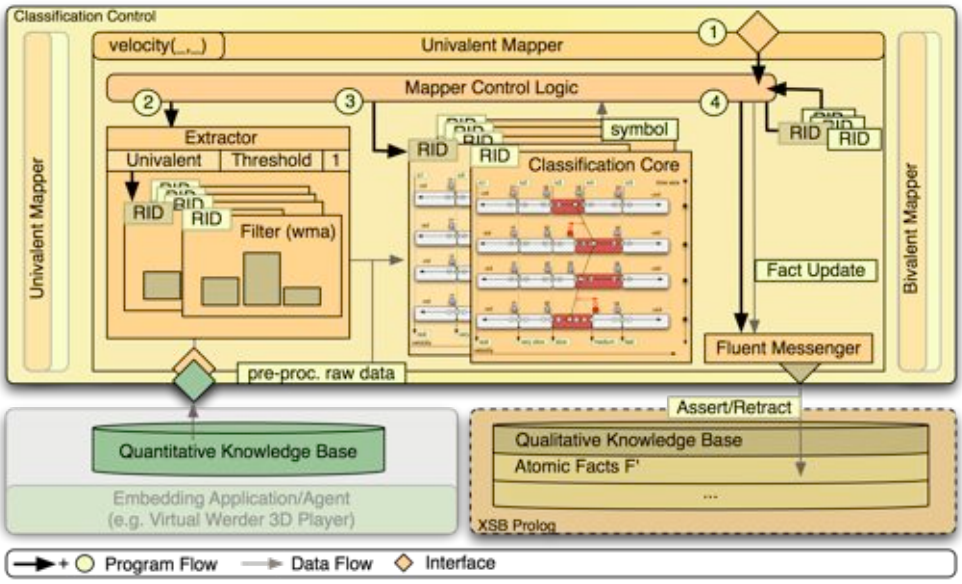


Figure 6.3: Detailed overview of the inner composition of a single mapping engine (→ *Mapper_Unary*) and its internal program flow.

facts – as well as detected motion incidences (both events and actions) – into/from the Prolog knowledge base. The implementation of the *Messenger_Fluent* does not rely on the common XSB C-interface documented in [SSW+06a, p.54–62], but rather seizes the underlying 'low-level C-interface'[15]. Thus, it allows for a C++/Prolog communication with a minimum of overhead[16]. A single *Messenger_Fluent* instance handles the complete interaction with the Prolog knowledge base. Looking forward, this particular statement of affairs rendered it possible to assign the additional task of fact oblivion to the *Messenger_Fluent* (cf. paragraph 6.2.1 on page 117).

Turning the focus of attention back to the composition of the mapping engines, while the I/O of the mapping engines has been outlined, another compulsory element still requires introduction, namely the fundamental *classification core* that builds the bridge from quantitative input[17] to symbolic values (cf. section 5.1.1, pp.57). The set of generic classification cores that has been implemented for this thesis corresponds to the classifier types that have been formally proposed in section 5.1.1 on page 56[18]. Each mapping engine obtains its desired classification core from a dedicated concrete factory which constructs the desired type of core[19] based on a comprehensive specification in the configuration (cf. figure A.1 on page 165) of the mapping engine whose scope corresponds to the interval-to-class associations detailed in section 5.1.1, pp. 57.

Each mapping engine is responsible for the compilation of spatio-temporal atomic facts associated with a distinct *predicate equivalence class*. Zerovalent mapping engines constitute the most simple case. When called by the *Classification_Control* the engine uses its *extraction adaptor* to acquire a single piece of input data which is consequently fed

15 A dedicated web site with online documentation is available at: http://xsb.sourceforge.net/manual2/node34.html (visited:03/11/2007)
16 Such as message processing/parsing
17 preprocessed time series data provided by an extractor adaptor
18 and an additional simple classification core for symbol to symbol translation required for the mapping of the playmode
19 i.e. an implementation of either $Classifier^{1}_{open}$, $Classifier^{1}_{ring}$, $Classifier^{Region}$ or a symbol-translation classifier

into its single *classification core*. The core determines the symbol associated with the input data. If the mapping engine is called for the first time, the representation of a new atomic fact with a right-open validity interval is compiled, and the *Messenger_Fluent* is used to enter this new fact into the qualitative knowledge base. In subsequent calls, the mapping engine determines whether the momentary classification result is d'accord with its immediate predecessor. If so, the atomic fact most recently entered into the knowledge base still reflects the current state of affairs and the mapping operation is finished early[20]. Otherwise, the most recently entered atomic fact is revised, replacing its right-open variant in the knowledge base with a closed variant, now that the end of the respective validity interval is known. Immediatly thereafter, another atomic fact with right-open validity interval that reflects the new situation is issued to the knowledge base. This incremental fact compilation scheme is repeated over the complete course of the analysis run.

Vis-à-vis the zerovalent case, multivariate mapping engines feature a notably more complex internal composition and application flow, reflected in figure 6.3 on the preceding page. This is due to the fact that the latter types of mapping engines essentially operate as *dynamic multivariate classifiers* as they are responsible for the classification (and subsequent fact assembly) of a dynamically changing subset of conceivable 'variations' of a particular predicate equivalence class such as:

$$distance(obj_1, obj_2, sym_{dist}) \in \mathbb{P}^2 \,.\, obj_i \in \mathcal{MO},\ sym_{dist} \in \mathbb{SYM}_{dist}$$

A particular variation is thereby characterized by a unique, complete instantiation of the predicate class' reference attributes:

$$distance(\mathsf{ball}, \mathsf{vw3d\_6}, sym_{dist}) \in \mathbb{P}^0 \,.\, \{\mathsf{ball}, \mathsf{vw3d\_6}\} \subset \mathcal{MO},\ sym_{dist} \in \mathbb{SYM}_{dist}$$

Thus, with respect to the mapping task at hand, each predicate variation can be treated like a zerovalent predicate. The additional complexity in multivalent mapping is thus broken down to the management of a temporally fluctuating set of zerovalent mapping tasks.

When a multivalent mapping engine is called by the *Classification_Control*, the momentary focus data (objects/ordered objects pairs (RID[21]) in/entering/leaving focus of attention) is handed over. Based on this data base the mapping engine can update its internal data structures in a two-tier pre-processing ahead of the actual mapping process. Based on the set of RIDs that have entered the focus of attention in the momentary call cycle, the mapping engine can determine whether or not it has both already created a dedicated copy of its classification core and prepared the extraction adaptor (i.e. created a dedicated copy of the respective time series preprocessing filter[22]) for each particular RID. If for any RID the answer to both questions is no, this means that the associated movable object or ordered pair of movable objects enters the focus of attention for the first time in the analysis run. Following a least commitment strategy, the required resources are provided now that a concrete demand has been detected. Once the list of entering RIDs has been processed, it is guaranteed that the necessary means for a classification of the associated predicate variations exist. In the second preprocessing tier, the set of RIDs that have left the focus of attention is processed. The classification cores and filters in the extractor adaptor associated with the elements in the set are reset to their default mode, such that a fresh classification can begin once the particular RID re-enters the focus.

Having successfully updated the internal data structures, a multivalent mapping engine begins the actual mapping operations for each predicate variation specified by the set of

20 Without fruitless interaction overhead between the mapping engine and the Prolog knowledge base via the *Messenger_Fluent*

21 Subsequently RID is used as an abbreviation for the respective ordered tuple of reference attributes, a single moving object in the univalent case and an ordered pair thereof in the bivalent case.

22 Available filters comprise a Weighted Moving Average (WMA, (cf. figure 6.3 on the previous page)) and an Exponential Moving Average (EMA) implementation

momentary RIDs in focus. The distinct mapping operations are performed sequentially. The procedure conforms to the zerovalent case described earlier with the mapping engine calling the extraction adapter in each step parameterized with the current RID and using the correct associated classification core. The whole application flow can be followed conceptually in figure 6.3 on page 115.

Step 5: Post-Processing or Applying the Fact Oblivion Strategy When the qualitative mapping has been performed by the *Classification_Control*, as shown in figure 6.2 on page 113, the final task which is initiated in the scope of the qualitative abstraction is the oblivion of old atomic facts, outlined in section 5.4.2 on page 105. As already mentioned earlier, the *Messenger_Fluent* takes charge of this task rather than a distinct, dedicated component in the SCD framework. From a design point of view, this decision seems like it spoils the otherwise clear task assignment of other modules in the SCD framework. However from a pragmatic perspective, the *Messenger_Fluent* is the module of choice due to the fact that all fact-related interaction with the qualitative knowledge base is handled by the messenger. Thus each time it is ordered to issue a new atomic fact with a closed validity interval that is therefore subject to aging to the knowledge base, the messenger stores the associated fact representation in an internal storage structure (cf. figure 6.2 on page 113). When the *Messenger_Fluent* is then called to perform fact oblivion, it can easily resort to this storage and retract those atomic facts from the knowledge base which a.) have been entered by mapping engines that registered explicitly for consideration in the oblivion process on startup and b.) whose validity has expired a certain amount of time ago. The concrete amount of time during which atomic facts remains relevant after the closure of their associated validity interval is specified explicitly by the mapping engine upon registration for fact oblivion. It is subject to individual customization via the configuration (cf. figure A.1 on page 165).

With the consolidation of the pool of atomic facts stored in the qualitative knowledge base, the process of qualitative abstraction is concluded for the momentary analysis cycle. Subsequent analysis steps, i.e. the detection of extensive motion incidences described in the following section, build exclusively upon the qualitative data base that has been maintained so far, rather than pursuing a hybrid strategy which also exploits qualitative low-level data.

6.2.2 Detection of Extensive Motion Incidences

As shown in the coarse structural overview presented in figure 6.1 on page 110, the detection of extensive motion incidences based upon the momentary contents of the qualitative knowledge base is performed by the *Recognition_Control* module. It is called by the superordinate *Mapper_SCD*, once the qualitative abstraction is completed. The internal composition of the *Recognition_Control* is shown conceptually in figure 6.4 on the following page. With regard to the concrete implementation, the detection comprises two distinct sub-tasks that are handled in succession. First, the compilation of derived atomic facts (cf. section 5.1.2 on page 63) and second, the detection of concrete events, actions and action sequences.

The compilation of the derived atomic facts is handled by the *Recognition_Control* rather than the *Classification_Control* due to the fact that these facts are not acquired immediatly as a consequence of qualitative mapping. Their compilation requires the incremental tracking of the evolution of the dynamic scene described by sets of fundamental atomic facts which is conceptually similar to the detection of high-level motion incidences.

Essentially, the *Recognition_Control* is a control module that manages a set of detectors that are called in sequence. Thereby, each detector is a self-contained unit that bears responsibility for the detection of a.) derived atomic facts (cf. figure 6.4, step (1)) or b.)

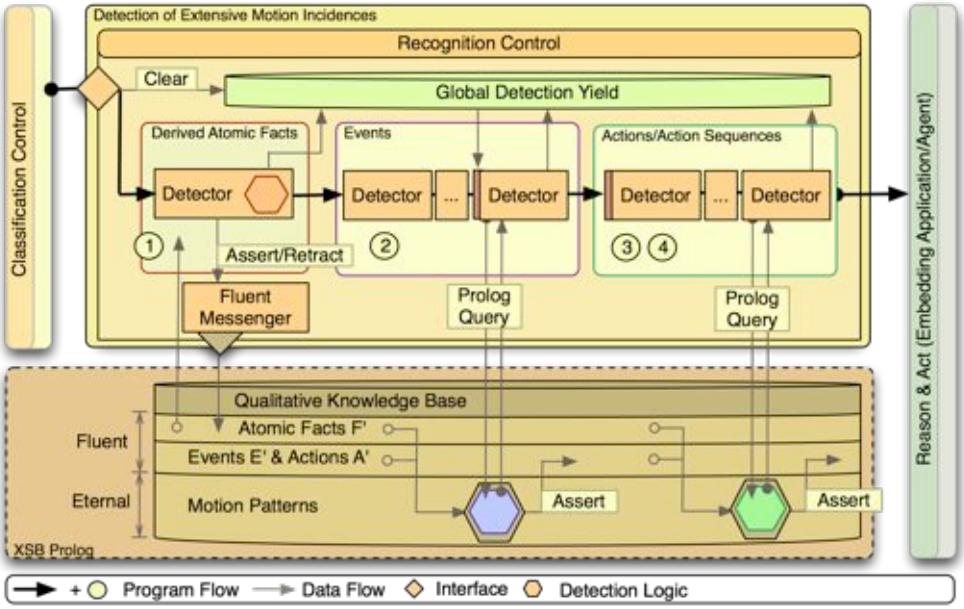


Figure 6.4: Schematic overview of the *Recognition_Control* module which is responsible for the detection of extensive motion incidences.

motion incidences that are associated with a distinct set motion classes (cf. figure 6.4, steps (2)-(4)). Two distinct types of detectors have been implemented.

The first type constitutes an *integrated detection unit* where both the detection management and the detection logic are implemented jointly as a conventional C++ class. Concrete detectors of this type are derived from a common base class that already provides the means to issue detection results to the qualitative knowledge base, using an own instance of the *Messenger Fluent* that has already been introduced in the description of the qualitative abstraction. In the implementation of concrete detectors of this integrated type, the developer is given complete freedom of choice as to the applied detection methodology. Qualitative input data that is required for the detection must be retrieved from the knowledge base using direct Prolog queries[23]. The integrated approach has been used exclusively for a single detector that is responsible for the incremental compilation of all derived atomic facts referring to the ball control predicates specified in section 5.1.2.

The second type of detector, employed for the detection of all extensive motion incidences, differs from the integrated variant as it constitutes a *compound detection unit* which relies on a generic detection management implemented as a general-purpose C++ wrapper class on the one hand and an individual, externalized detection core realized as a Prolog rule on the other hand.

This prolog rule can be thought of a short program that can perform spatio-temporal pattern matching of the facts/motion incidences stored in the qualitative knowledge base against an entailed motion pattern that is the direct implementation of a formal pattern specification as introduced in section 5.3, pp.89. Successfully detected motion incidences that are not yet contained in the knowledge base are asserted immediatly and the detection yield is returned as result of the Prolog query. Examples for such externalized detection cores are shown in figure 6.6 on page 121 (→ detection of ball transfer occurrences) and figure 6.7 (→ detection of pass occurrences). These figures constitute excerpts from the actual Prolog implementation. Both have been overlaid

23 That is, so far no specialized extraction interface for qualitative facts has been implemented.

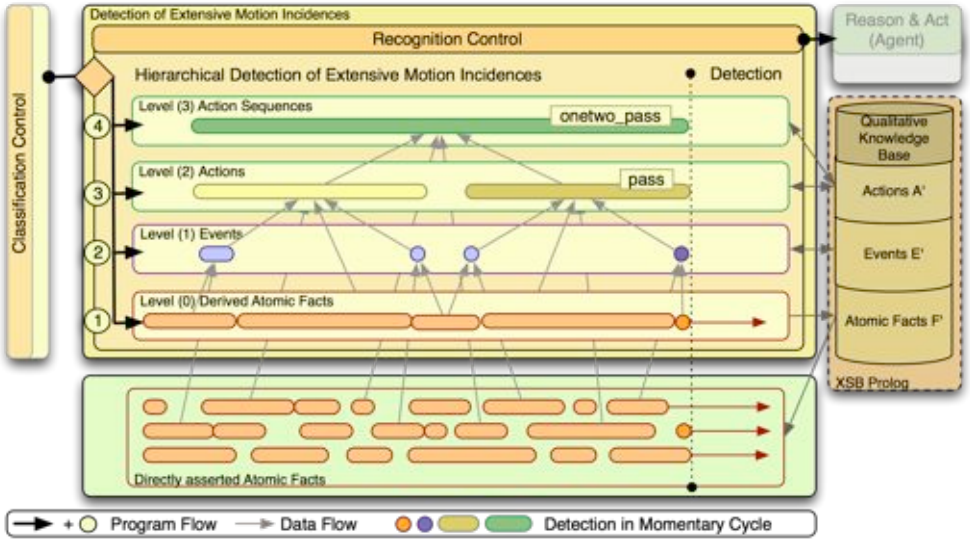


Figure 6.5: Alternative schematic overview of the course of action of a detection cycle performed by the *Recognition_Control*. It it to show the hierarchical detection style. Starting with the assessment of new derived atomic facts the detection proceeds to extensive motion incidences thereby working towards top-level incidences. Within the same detection cycle intermediate results collected hitherto can already provide the basis for the detection of superordinate motion incidences.

with additional information such that it is easy distinct the rules' respective goal[24] used in the query, the control code with the standardized knowledge base interaction and most importantly the contained motion pattern. The ball transfer and pass were chosen as paradigmatic examples as they illustrate the distinct query styles that are used for normal detection ($\rightarrow$ *occurring*_**after**(bound, ...)) and specialization detection ($\rightarrow$ *occurring*_**at**(start, end, ...)). In a specialization query the exact temporal extent of the potential new incidence is already preset while in a normal query, only the end of the last detected incidence is specified, an information which can then be used to optimize the internal pattern matching using expert knowledge about the temporal entanglement of motion incidences of the considered motion class[25].

The particular implementations of the motion patterns in Prolog deserve extra attention as they constitute a strong reason why Prolog was chosen as for the implementation of the pattern matching process in the first place. A direct comparison of specification and implementation (e.g. figure 6.6 on page 121 $\leftrightarrow$ motion pattern (5.11), p.96) is instructive and points up that the formalization translates virtually unchanged directly to executable code, thus rendering the compound approach an excellent base for rapid prototype development of detectors responsible for new/additional motion classes. For the scope of this thesis, Prolog detection cores have been implemented for each of the event and action classes specified in section 5.3.

Focusing on the course of action of a single detection cycle conducted by the *Recognition_Control*, the set of both integrated and compound detectors is called in ascending order of complexity of the motion class associated with the particular detector, seizing the proposal from section 5.4.1, pp.104 (cf. figure 6.5). This call order is employed as it is a

24 The term 'goal' or 'head' is a synonym for the implication of the Prolog rule.

25 For instance, for ball transfers in soccer it holds that concrete occurrences can only happen in sequence. This constraint is consequently enforced in the motion pattern in figure 6.6 on page 121.

prerequisite for the adherence to the guided detection strategy proposed in section 5.4.3, pp.106. Even though an automatic determination of the call order based on the constituents contained in the particular motion pattern implementations is conceivable and a feasible addition in future revisions of the *Recognition_Control*, for the time being the order of calls is determined by hand.

In section 5.4.3, a concept of guided detection has been proposed which allows for a conditional execution of the respective detection attempts by the specialized detectors, based on whether or not the detection yield obtained so far in the momentary detection cycle provides sufficient hints for a detection success. The concrete implementation of this concept is based upon two pillars. First, a data structure, referred to as the global detection yield, managed by the *Recognition_Control* where each detector that has been successful in digging up new derived facts/motion incidences in the momentary detection cycle can post its local detection yield. Second, specialized triggers which can be assigned to each detector. These triggers can be fed with the hitherto compiled global detection yield and based upon its contents can decide whether the detector should actually attempt finding new motion incidences or skip the detection cycle. They are provided by a special trigger factory and can be built generically based on the individual configuration of a certain detector. The configuration also defines which of the two available detection strategies, that is either the brute force or the guided approach are preset globally and thus relayed by the *Recognition_Control* to the particular detectors in each call. If the guided detection strategy is set globally, the subset of detectors, no matter if of integrated or compound flavor, that supports a selective detection behavior, adheres to the global hint. Thus, it is possible to employ detectors with dumb/selective call behavior side by side.

Once the whole set of detectors that is managed by the *Recognition_Control* has been called, it is guaranteed that all extensive motion incidences that could possibly be detected up to the momentary present have indeed been detected and stored in qualitative knowledge base. The analysis cycle is completed.

6.3 Implemented Use Case Scenarios

For the scope of this thesis, the SCD module that has been outlined in the preceding section has been paradigmatically employed in the following use case scenarios: First, the SCD module was embedded in the *SCD Coach* agent, which actively participates in the simulation of soccer matches in the RoboCup 3D Soccer Server. Second, the SCD module was embedded in the *SCD Simulator*, a standalone software application.

6.3.1 SCD Deployment in RoboCup 3D Soccer Simulation

It was stated as a primary goal of this thesis that the spatio-temporal analysis of dynamic scenes developed herein must be suitable for real-time deployment. In the application domain at hand, this means essentially it must be feasible for an agent participating in the considered simulated soccer matches to embed and conduct the analysis module. As for the evaluation of this thesis, a special variant of the RoboCup 3D Soccer Server created by the Iranian UTUtd team was used [ABD+06], two integration scenarios were possible: a.) integration of the SCD module in the common soccer agents and b.) integration in a coach agent which was introduced as part of the UTUtd 3D Soccer Server.

Due to the fact, that the SCD framework was to be evaluated first using a perfect, i.e. noise-free vision, and it was not yet clear which fraction of the available reason-act cycles would be consumed by the additional analysis functionality, the choice fell upon the coach agent. For this purpose, the existing Virtual Werder 3D coach/monitor hybrid,

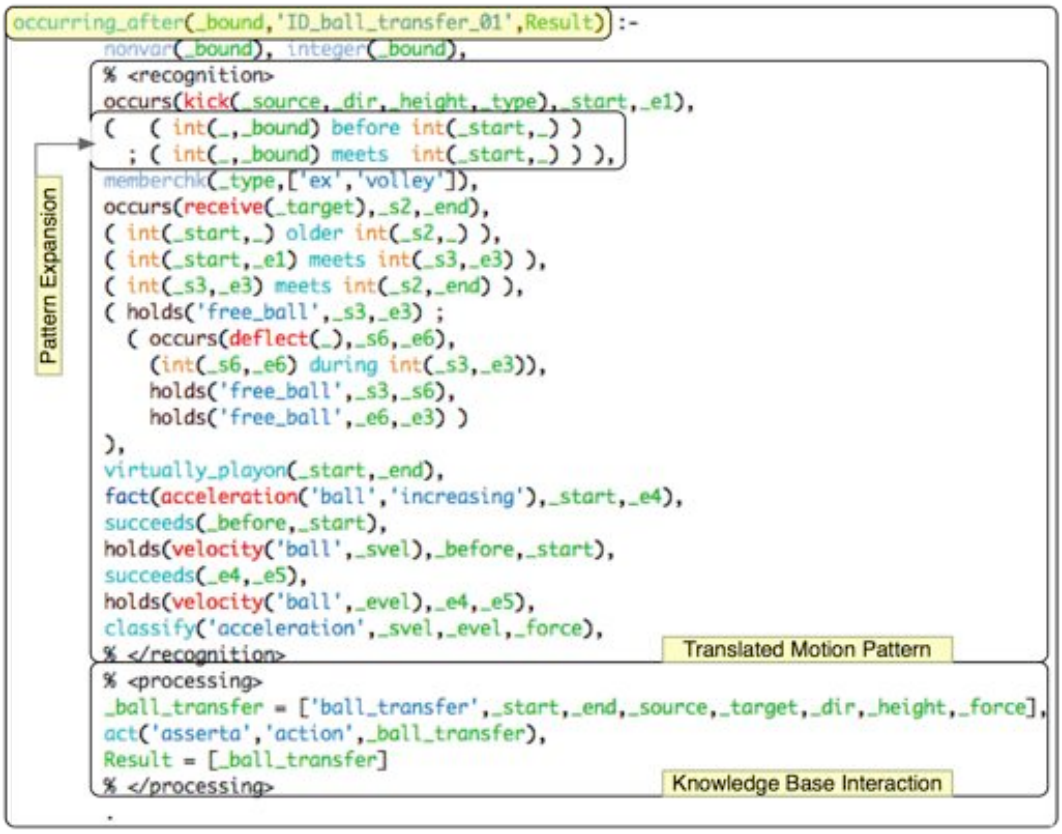


Figure 6.6: Implementation of the Prolog detection core for the *ball_transfer* motion class. Via the *_bound* variable and suitable additions in the motion pattern, the pattern matching process is streamlined.

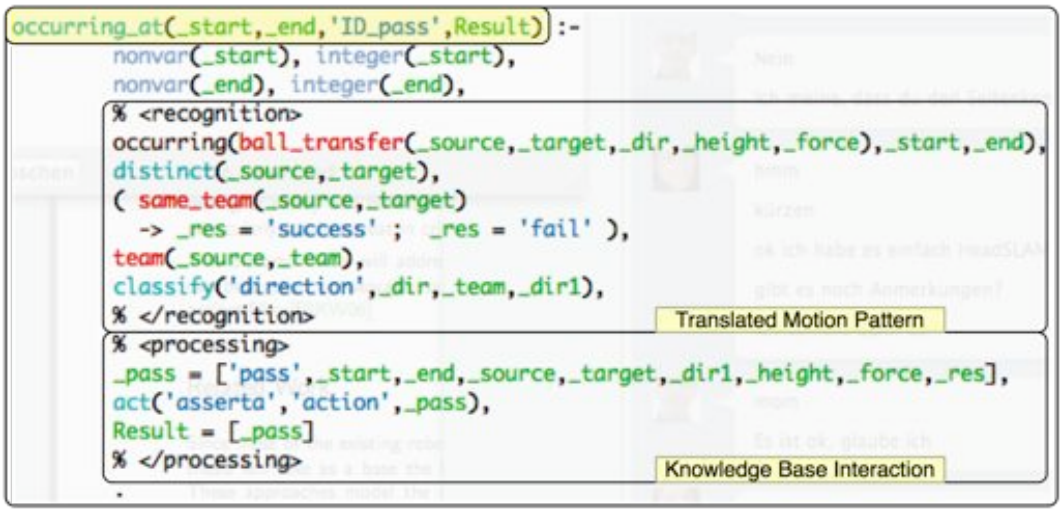


Figure 6.7: Implementation of the Prolog detection core for the motion classes of both *failed and successful passes*. The fact that a successful pass is a direct specialization of a ball transfer is reflected in the detection management code.

developed originally by the author to conduct reinforcement learning scenarios in the RoboCup *3D Soccer Simulation*, has been adapted. Seizing the design principles established by the Virtual Werder 3D team in the development of its extensive code base [LRS+06], the SCD module was integrated as an additional world state mapper that is called in succession to existing mappers responsible for the compilation and maintainance of the quantitative world model. The abstract interface defined by the SCD framework for the access to quantitative input data was implemented as a new extractor that interfaces with the coach knowledge base. The methodology used for the integration in the coach agent is immediatly applicable for the normal Virtual Werder 3D player agents and demonstrates the simplicity of an integration of the SCD module into an existing code base.

As a side-node, it should be mentioned that the design of the SCD framework outlined in the preceding sections enabled the following development feature with respect to the *SCD Coach*. Once the binary of the *SCD Coach* with the embedded SCD module is available, seizing the means of compound detectors described in section 6.2.2, existing detectors can be modified and new ones created without recompilation of the coach binary. While a modification of existing detectors is possible by a modification of the associated Prolog rule, the addition of new detectors also requires an entry in the XML configuration file which can be compiled in about a minute. In both cases, the modified analysis behavior is available in the next employment of the *SCD Coach* in the RoboCup *3D Soccer Simulation* environment.

6.3.2 SCD Deployment in Standalone Analysis

The previous section broached the issue of the embedding of the SCD module into an agent that resides inside the RoboCup *3D Soccer Simulation* and which is thereby enabled to perform a *LiveSCD* analysis of the momentary game while also pursuing its original duties defined by its role association.

However, both the development process of the SCD framework and a large part of the evaluation[26] called for the possibility to perform repeated analysis runs based on a single, fixed simulated soccer match.

This reproducibility requirement was accommodated with the implementation of game-recording capabilities for both the *Virtual Werder 3D* players and the *SCD Coach*. Both have received mappers that write the agents' respective perception of the momentary world state to log files using a common format based on Lisp-like s-expressions which is further detailed in section A.3 on page 165. The logs that are by and by compiled that way during the simulation of soccer matches conducted by the RoboCup *3D Soccer Server* can subsequently be used as input for a standalone application, the *SCD Simulator*. This application can parse a log file at a time into an internal memory structure. It integrates the SCD module and invokes its analysis once for each recorded agent perception. Again, the abstract input interface defined by the SCD framework for access to quantitative input data has been implemented as a new extractor which interfaces with the internal representation of the agent's conserved momentary worldstate perception.

Thus, using the *SCD Simulator* it is possible to analyze a certain fixed game repeatedly. It is also possible to compare the analysis results based upon a constrained, noisy perception (player agents) with the noise-free perception of the *SCD Coach* without having to embed the SCD module directly in the Virtual Werder 3D agents.

26 namely the fraction centered around the evaluation of performance variations under changing parameterizations of employed real-time optimizations (cf. section 7.4 on page 137)

7 Evaluation

The seventh chapter of this thesis is devoted to the evaluation of the approach for spatio-temporal analysis which has been outlined in chapter 5 and implemented in the concrete software system introduced in chapter 6. The evaluation is structured as follows:

First, section 7.1 briefly describes the general setup of the test environment. In succession, section 7.2 evaluates the recognition quality with respect to *precision* and *recall*[1] of a subset of extensive motion incidences, both events and actions, for which motion patterns have been developed throughout section 5.3 in the concept chapter. Primarily, recorded world states originating from the *SCD Coach* (cf. section 6.3.1) are used as input for the analysis. However, in a comparative analysis, it is also evaluated to what extend the implemented analysis system can already handle degraded input from regular Virtual Werder 3D agents.

As the real-time efficiency is considered a crucial factor for a feasible application of the implemented system during regular games of the RoboCup 3D Soccer Simulation League, rather than as a post-processing tool, the following sections evaluate the runtime performance of the developed system within the RoboCup 3D Soccer Simulator (section 7.3) and thus the targeted simulation environment with respect to both consumed simulation time within a reason-act cycle and consumed real time for qualitative abstraction and detection of extensive motion incidences respectively. Section 7.4 evaluates the variation of the runtime performance caused by several optimizations such as the application of a guided detection strategy (cf. section 5.4.3 on page 106), the applied fact assembly strategy (cf. section 5.1.5 on page 78) and the oblivion of aged atomic facts (cf. section 5.4.1 on page 104).

The chapter is concluded in section 7.5 with a discussion of the results compiled throughout the evaluation.

7.1 General Setup of the Test Environment

The evaluation for this thesis has been performed on a single simulation machine located in the Virtual Werder Lab with the following specification: 32bit Intel Pentium 4 CPU

1 The definition for the terms *precision* and *recall* for this thesis can be found in definition 7.1 on page 125.

(1 Core, Hyperthreading deactivated) with 3.0 GHz clock speed and 2GB RAM. The system runs Ubuntu Linux (7.04, Feisty Fawn) with a modified Ubuntu-Kernel[2]. For the generation of test games an extended version of the standard RoboCup 3D Soccer Server 0.5.3, developed by the Iranian UTUtd 3D Soccer Simulation Team[3] for the 3D Development Competition at the RoboCup 2006 in Bremen [ABD+06], referred to as UTUtd server, was used. This server featured amongst other improvements support for coach agents with precise, undisturbed vision. It was modified by the author for use in this thesis such that team binaries from past RoboCup events (RoboCup 2006 and China Open '06) can be deployed unchanged as adversaries for the Virtual Werder team.

A series of 15 complete test games was run with the latest Virtual Werder 3D sphere team competing three times respectively against five of the top teams of the RoboCup 3D Soccer Simulation League, namely Fantasia[4], SEU[5], WrightEagle[6], FC Portugal[7] and Aeolus. The games were conducted under the same conditions that are accepted standard for national and international RoboCup competitions with respect to the configuration of the simulator. However, due to limited resources all games were conducted in a non-distributed setting on the aforementioned test machine running the server and both agent teams. Each game was attended by the *SCD Coach* (cf. section 6.3.1 on page 120) compiled for this thesis.

Both the players of the Virtual Werder team as well as the *SCD Coach* compiled log files which encode the perception of the game in a common format, suitable as input data for the offline application of the analysis system developed in this thesis via the dedicated log-based *SCD Simulator*. Moreover the coach performed online analyses for all 15 conducted test games, thereby compiling performance statistics to be evaluated in section 7.3.

The set of fifteen test games is included on the DVD that comes along with this thesis and constitutes the basis for all evaluation steps described hereafter.

7.2 Detection Quality: Precision & Recall

As outlined in the introduction, the issue of the first section of the thesis evaluation is the practical test of the implemented analysis system with regard to the *detection quality* for extensive motion incidences under real-world conditions. That is, the system is applied in the context of regular, RoboCup tournament compliant 3D Soccer League matches as geared up in the preparation of the evaluation rather than in restricted, well-tuned lab scenarios. Thus, a demand specified in section 3.2.2 on page 19 is accommodated for.

7.2.1 Description of the Test Setup

In the given context the term quality refers to two performance criteria borrowed from the field of *information retrieval* (IR), namely *precision* and *recall* [RN95, p.840–844]. In using these key measures the evaluation at hand is geared to its equivalents in related research such as in [Wen03, pp.149] and [Mie04a, p.133]. The semantic of both *precision* and *recall* is defined as follows for the scope of this thesis:

2 to be precise, a 2.6.20-based kernel with additional kernel modules (perfctr performance counter) required for the operation of the RoboCup 3D Soccer Server
3 Information about the UTUtd team can be found on a dedicated web site: http://www.fos.ut.ac.ir/robocup/ (visited:02/09/2007)
4 Dalian University of Technology, China. 1^{st} rank China Open 2006
5 Southeast University, China. 2^{nd} rank, China Open 2006
6 University of Science and Technology of China, China.
7 University of Aveiro/University of Porto, Portugal.

Definition 7.1 (Precision and Recall) *Let* $P_{System} = \mathbb{A}'_{Sytem} \cup \mathbb{E}'_{System}$ *denote the set of automatically detected motion incidences and let* $P_{Truth} = \mathbb{A}'_{Truth} \cup \mathbb{E}'_{Truth}$ *denote the* ground truth *set of actually occurred motion incidences.*

Then, the precision *measure* m_{prec} *is defined as the proportion:*
$m_{prec} = \mid P_{System} \cap_s P_{Truth} \mid \; / \mid P_{System} \mid$.

The recall *measure* m_{rec} *is defined as the proportion:*
$m_{rec} = \mid P_{System} \cap_s P_{Truth} \mid \; / \mid P_{Truth} \mid$. □

Before actual evaluation results are presented, it is compulsory to clarify the correlation of both P_{System} and P_{Truth} with the maximum detection yield $P_{max} = \mathbb{E}' \cup \mathbb{A}'$ and, consequently the way in which *ground truth* detection results are acquired.

To the best knowledge of the author at the time of writing, no immediatly comparative approach for the real-time spatio-temporal analysis of dynamic scenes has yet been developed and published for the considered application domain of the RoboCup 3D Soccer Simulation League as documented in the survey of related work in chapter 4 on page 29. Thus, no eligible candidate for a direct comparison of detection results is at hand. As a consequence thereof, the implemented analysis is tested against a manually crafted ground truth, rather than a competing system/approach.

The ground truth is thereby obtained as follows: A human spectator tracks a subset of the available test games mentioned in the previous section and in the process tries and detects incidences for a subset of the motion classes whose automatic detection has been enabled in the implemented analysis system via associated motion patterns. A compulsory prerequisite for an unbiased manual detection is the ignorance of the automatic detection results. The results of a manual detection qualify as ground truth under the given circumstances for two reasons: First, the motion patterns that are the backbone of the automatic detection of motion incidences are the result of a knowledge engineering process where the human expert tries and formalizes his intuitive understanding of the composition of motion classes. Even though this process is done thoroughly, it is expected that only a subset of the engineer's expertise can be captured, especially with respect to exceptions from the norm. Second, the standard 3D Soccer Monitor bundled with the server features a sufficiently high frequency of vision updates such that in conclusion the manual detection should be a superset of those motion incidences detectable by the implemented analysis system.

As it is difficult for a human to *unambiguously* detect all the motion classes known to the analysis system, certain restrictions need to be introduced for the sake of comparability of the respective detection results. Thus, the following quality assessment will concentrate on the detection of *ball reception* (cf. pattern (5.4)), *kicks of the ball* (cf. patterns (5.1) (5.2) (5.3)), the *ball transfers* (cf. patterns (5.11) (5.12)) and their respective *specializations* (cf. patterns (5.14) (5.15) (5.16)).

Concrete incidences of the associated action classes in $\mathbb{E} \cup \mathbb{A}$ constitute the reduced incidence sets P_{System} and P_{Human} respectively. The larger part of the motion classes for which incidences are to be detected are recognized with an expressive diversification such as height and direction for the kick event. These diversification attributes are not considered in the evaluation as they are rather difficult to compile by a human spectator provided the standard 3D Log Monitor[8]. Rather the evaluation concentrates on the detection of the basis concepts[9].

8 The 3D Log Monitor, referred to as rcssmonitor-lite, is distributed routinely with the RoboCup 3D Soccer Server both in the standard server and the derived UTUtd server used for this thesis. No functional modifications have been made on this piece of standard software.

9 Concommittant testing during the development phase of the detection aspect of the SCD analysis showed that once the basic incidences are detected properly, the association of the diversification arguments works as expected as well.

```
[SIM][sim:32679][game:322.774]
  asserta(event(receive(fcportugal_11),32678,32679)).
  asserta(action(ball_transfer(fcportugal_9,fcportugal_11,southEast,...),32358,32679)).
  asserta(action(pass(fcportugal_9,fcportugal_11,steep_left,...,success),32358,32679)).
[SIM][sim:32899][game:322.974]
  asserta(event(retreat(fcportugal_11,ball),32699,32700)).
[SIM][sim:32899][game:324.976]
  asserta(event(receive(fcportugal_11),32898,32899)).
  asserta(action(ball_taming(fcportugal_11),32678,32899)).
[SIM][sim:32959][game:325.576]
  asserta(event(kick(fcportugal_11,south,even,ex),32939,32959)).
[SIM][sim:33059][game:326.577]
  asserta(event(receive(fcportugal_11),33058,33059)).
  asserta(action(ball_transfer(fcportugal_11,fcportugal_11,south,...),32939,33059)).
  asserta(action(self_assist(fcportugal_11,forward),32939,33059)).
[SIM][sim:33099][game:326.978]
  asserta(event(kick(fcportugal_11,south,even,ex),33079,33099)).
  asserta(action(dribble_atom(fcportugal_11,forward,success),32939,33079)).
  asserta(action(dribbling(fcportugal_11,forward,success),32939,33079)).
[SIM][sim:33239][game:328.379]
  asserta(event(receive(fcportugal_11),33238,33239)).
  asserta(action(ball_transfer(fcportugal_11,fcportugal_11,south,...),33079,33239)).
  asserta(action(self_assist(fcportugal_11,forward),33079,33239)).
[SIM][sim:33299][game:328.98]
  asserta(event(kick(fcportugal_11,south,even,ex),33259,33299)).
  asserta(action(dribble_atom(fcportugal_11,forward,success),33079,33259)).
   retract(action(dribbling(fcportugal_11,forward,success),32939,33079)).
  asserta(action(dribbling(fcportugal_11,forward,success),32939,33259)).
[SIM][sim:33319][game:329.18]
  asserta(event(receive(fcportugal_10),33318,33319)).
  asserta(action(ball_transfer(fcportugal_11,fcportugal_10,south,...),33259,33319)).
  asserta(action(pass(fcportugal_11,fcportugal_10,forward,...,success),33259,33319)).
```

Figure 7.1: Excerpt from the detection output from the match *Virtual Werder 3D vs. FC Portugal* which documents the successful detection of incidences for *ball_taming*, *extended dribbling*, *self_assists* and a *successful passes*. The dribble detection is of special interest as the incremental lengthening of an ongoing dribble sequence is demonstrated to work successfully (cf. section 5.2.2 on page 85).

Ball control events that indicate the start of a struggle for ball control, the changing or the ending thereof are neglected in the evaluation, even though they are all fully implemented in the analysis system. This is due to the fact that the latter class of motion incidences are to a larger degree subject to individual interpretation such that the results from two distinct detection systems cannot be compared easily as varying interpretations are not necessarily indicators of failure but may rather qualify as alternative observations of the same statement of affairs in a dynamically evolving soccer scene.

7.2.2 Detection Quality based on Precise Worldstate Perception

The first part of the concrete quality assessment is based on the precise worldstate data provided by the *SCD Coach* which is complete in the perception of all movable objects in the simulation environment and free of server-induced vision noisification. Successive worldstate updates are available every 20 sim-steps (corresponding to 200ms of simulated temporal progress in the physical simulation).

Two half times played against distinct teams in two games of the above mentioned evaluation test series are exploited. First, the second half time in the second match *Virtual Werder 3D vs. FC Portugal*. Second, the first half time of the first match *Virtual Werder 3D vs. SEU*. Both adversary teams are among the top RoboCup 3D Soccer Simulation

Motion Class	Type	$\#P_{System}$	$\#P_{Truth}$	$\#\cap$	m_{prec}	m_{rec}
Virtual Werder 3D vs. FC Portugal (Game 1, 1^{st}Half)				1500 analysis cycles		
kick	event	62	67	62	1.0	0.925
collective kick	event	4	3	2	0.5	0.667
receive	event	78	76	76	0.974	1.0
ball_transfer	action	59	65	57	0.966	0.877
pass (success)	action	17	19	16	0.941	0.842
pass (failure)	action	23	28	23	1.0	0.821
self_assist	action	18	18	18	1.0	1.0
Virtual Werder 3D vs. SEU (Game 2, 2^{nd}Half)				1500 analysis cycles		
kick	event	63	71	63	1	0.887
collective kick	event	1	1	1	1.0	1.0
receive	event	83	83	79	0.951	0.951
ball_transfer	action	60	69	57	0.95	0.826
pass (success)	action	36	41	36	1.0	0.878
pass (failure)	action	16	20	14	0.875	0.7
self_assist	action	8	8	7	0.875	0.875
Accumulated Results (Corresponds to Full Game)				3000 analysis cycles		
kick	event	125	138	125	1.0	0.905
collective kick	event	5	4	3	0.6	0.75
receive	event	161	159	155	0.963	0.974
ball_transfer	action	119	134	114	0.957	0.850
pass (success)	action	53	60	52	0.981	0.866
pass (failure)	action	39	48	37	0.948	0.771
self_assist	action	26	26	25	0.961	0.961
header notation: m_{prec}: precision, m_{rec}: recall, $\#P_{xy}$: detections by '*xy*', $\#\cap$: mutual detections						

Table 7.1: Evaluation of the *precision* and *recall* key measures for two distinct half times of matches *Virtual Werder 3D* vs. *FC Portugal* and *SEU* respectively for a selected subset of implemented event/action classes. Accumulated results are presented as well.

League teams[10]. FC Portugal won the RoboCup championships in 2006[11] and the RoboCup German Open 2007[12]. SEU reached rank 3 in the RoboCup championships 2006 and was runner-up in the RoboCup China Open 2006[13]. Thus, the choice of adversary teams set value to the expected dexterities such that games with high quality standard are considered.

The automated analysis whose results are presented in a condensed form in table 7.1 was performed offline in the *SCD Simulator* (cf. section 6.3.2 on page 122) based on the recorded game observations compiled by the *SCD Coach* during the initial simulation of the particular games[14]. An excerpt of the output that was generated by the analysis is shown in figure 7.1 on the preceding page. The remainder of this section is structured such that the cumulative detection results are considered which are expressed in the respective *precision* and *recall* key measures. Subsequently, limitations of the implemented analysis system are critically discussed and hints are presented that point at possible solutions/mitigation of the observed problems.

In order to begin the interpretation of the results condensed in table 7.1 the *recall* key measures m_{rec} are considered first. As definition 7.1 on page 125 states, the recall

10 with the distinction that this statement of affairs refers to the classic 3D Soccer Simulation with sphere-based agents, that has been used until spring/summer 2007
11 A dedicated web site is available at: http://www.robocup2006.org/start?lang=en (visited:17/9/2007)
12 A dedicated web site is available at: http://www.robocup-german-open.de/en (visited:17/9/2007)
13 A dedicated web site is available at: http://ai.ustc.edu.cn/rco/rco06/ (visited:17/9/2007)
14 Due to the fact, that online and offline analysis build upon the exactly same input data and thus yield the same results, this course of action is justified

Action Class	Analysis Miene (2D Sim)		Implemented Analysis (3D Sim)	
	m_{prec}	m_{rec}	m_{prec}	m_{rec}
pass (success)	0.955	0.933	0.981	0.866
pass (failure)	0.898	0.928	0.948	0.771
self_assist	1.000	0.955	0.961	0.961
cumulative/Δ	0.938/–	0.932/–	0.963/+0.025	0.866/-0.066

Table 7.2: Comparison of detection quality for selected action incidences.

characterizes the ability of the analysis system to detect the considered motion incidences that '*actually occur*' in the soccer matches[15]. Considering the results for the match Virtual Werder 3D vs. FC Portugal, the recall values for all kick and reception incidences are better than 90% with the exception of the collective kick which occurred too infrequent such that the recall value is considered too weak in its significance. In direct comparison, the kick detection ($m_{rec} = 0.925$) does not perform as well as the reception detection ($m_{rec} = 1.0$) which is due to the higher complexity of the kick patterns for the standard and volley kick introduced in section 5.3.1. In the game against the SEU team both the kick detection ($m_{rec} = 0.887$) and the reception detection ($m_{rec} = 0.951$) lie below their equivalents in the game against the FC Portugal.

The visual inspection of both matches that was performed for the compilation of the ground truth data offers an explanation. The style of play that is favored by the SEU team is highly dynamic and features a tendency towards immediate forwarding of the ball over multiple stations during the build-up of the offensive game. In particular, an airborne ball can be forwarded volley without touching the ground and with the agent being barely in ball contact. This rapid style of play, adapted by other teams such as WrightEagle as well, is more difficult to capture by the implemented kick patterns such that misclassifications (kick interpreted as deflection/retreat from ball) and complete failure of detection occur with increased frequency. However, even under those circumstances the detection of the considered event patterns yields acceptable if not excellent results. Since the kick patterns feature a comparatively high complexity compared to those patterns exempted from direct evaluation, it is expected that acceptable results can be achieved for those as well.

With respect to the action patterns related to the transfer of the ball the results, the recall values range between 82.1% (failed passes) and 100.0% (self assists) for the match against FC Portugal and between 70.0% (failed passes) and 87.8% (successful passes) for the match against SEU. So, in general a recall value of more than 80% is achieved on average for the considered action patterns. The lower recall compared to that achieved for the considered subordinated events is thereby due to the hierarchical layout of the motion patterns which induces a direct dependency[16] of detection quality for high-level actions from its equivalent for subordinated events. For instance, the misclassification of a kick event as a deflection of the ball necessarily prohibits the detection of the resulting ball transfer (and consequently its particular specialization) once the free ball is received again. Concluding the evaluation of the recall results, a *loose*[17] comparison with similar results acquired for the analysis approach by Miene in [Mie04a, p.133] in the RoboCup 2D Soccer Simulation League in table 7.2 documents that the recall performance of the analysis at hand is only slightly inferior (6.6%) and can thus still be judged as competitive.

In succession, the *precision* key measures are considered. As specified in definition 7.1 on page 125, the precision characterizes the extent to which the analysis system yields true results with respect to the ground truth and does not imagine false motion incidences. The data in table 7.1 states that the performance of the analysis with respect to precision

15 where the manually crafted ground truth acts as the reference measure
16 Consider the specified action patterns in section 5.3.2 on page 95
17 Loose in the sense that the two RoboCup Soccer Simulation Leagues each have their own characteristics such that a comparison can only yield hints rather than hard data.

Figure 7.2: Example Situation which illustrates a premature ball transfer detection in a match *Virtual Werder 3D vs. SEU*

is notably better than the associated recall with results between 94.1% and 100% for the match against FC Portugal[18] and between 87.5% and 100% for the match against SEU. Considering the cumulative recall in comparison with the neighbored approach by Miene, table 7.2 shows that the results that could be achieved by the implemented analysis are on par with preliminary, yet offline-only approaches. These results are encouraging as they document that the knowledge engineering process that let to the pool of motion patterns specified in section 5.3 succeeded in creating patterns that are robust against incorrect positive detections. Rather, the primary shortcoming attributable to the implemented patterns is a suboptimal coverage of the space of possible, distinct event and action characteristics that can occur in the application domain.

It is thus expected that a thorough pattern revision based on the examination of failure situations and a more involved knowledge engineering process can mitigate a notable fraction of encountered mishaps in the detection of motion incidences. Figure 7.2 provides a series of snapshots from the match Virtual Werder 3D vs. SEU which paradigmatically illustrates the misclassification of a ball transfer. At the beginning of the sequence the ball is kicked by Virtual Werder's player 5 (*vw5*) and approaches both SEU's player 5 (*seu5*) and *vw10*. Once the ball enters the influence sphere of *vw10* with only moderate remaining speed a premature detection of a reception incidence by *vw5* occurs which immediatly leads to the dependent detection of a ball transfer and thus a successful pass between *vw5* and *vw10* in the same analysis cycle. However the situation continues

18 the value for the collective kick is not considered due to a lack of significance

and it is rendered obvious that the ball only passes the influence sphere of *vw10* without physical interaction an continues rolling towards *seu2*, the true ball recipient. The situation suggests, that a more observant reception pattern, as well as a pattern for ball passage might be required and that the dependent ball transfer pattern should be modified such that ball passages are not allowed to conclude a valid ball transfer.

However, the revision of existing and implementation of new motion patterns cannot solve all classification problems as certain ones are a consequence of the decision to perform a qualitative abstraction as a basic step and consequently to perform the detection solely based on qualitative atomic facts on the one hand and the comparatively low sampling rate for the vision updates that are available to the soccer agents (20sim-steps/200ms compared to 15 sim-steps/150ms for the monitor and thus the human spectator). Both issues lead to a coarsening in the perception of the dynamic scene that causes misclassifications.

Thus, with respect to future work it would be beneficial to perform further experiments that are suited to verify the coarsening hypothesis. First, to increase the sampling rate of vision messages artificially from a 20 sim-step update interval to 10 or 5 ms intervals. While this connotes an departure from the default RoboCup 3D Soccer Simulation guidelines such experiments are technically realizable via modifications of the *SCD Coach* such that it is provided with log monitor vision updates that feature a configurable frequency. Second, as stated in section 5.1 on page 56, the implemented qualitative abstraction is considered of decent, but not superior quality. Optimizations in that field could improve the expressiveness of the available atomic facts and enable the construction of more sophisticated motion patterns.

7.2.3 Detection Quality based on Imprecise Worldstate Perception

In the previous section, the evaluation of the detection quality was based on the precise worldstate data provided by the *SCD Coach*. Based here-upon, the following second part of the quality assessment examines the extent of the degradation of analysis quality once an imprecise worldstate perception, provided by the *Virtual Werder 3D* soccer agents, substitutes the hitherto used input.

The technical manual [O+06] which is distributed with the RoboCup 3D Soccer Server specifies the official noise parameters for the restricted agent vision perceptor which features a 180° field of vision as follows: First, "*a small* calibration error *is added to the camera position. For each axis, the error is uniformly distributed between -0.005m and 0.005m. This error is calculated once and remains constant during the complete match*" [O+06, p.7]. Second and more important, *dynamic noise* normally distributed around 0.0 is added to the polar sight of the objects on the soccer pitch: a.) a distance error: $\sigma_{dist} = 0.0965$, b.) an angular error in the XY-plane: $\sigma_{xy} = 0.1225$ and c.) an angular latitudinal error $\sigma_{lat} = 0.1480$. The error increases with the distance of sighted objects. As outlined in the team documentation, the *Virtual Werder 3D* agents use a combination of a ball-oriented poke-around strategy [LRS+06, p.33], integrate distinct vision messages with their current observation of the world and use a particle filtering technique to deduct the noise produced by the server [LRS+06, pp.22]. When noisy perception by the agent is mentioned in this section, this consequently refers to the best possible estimation of the real state of the world, which is maintained by the agent using the aforementioned techniques. The noise parameters used by the server remained untouched for the evaluation due to the demands for application in a real-world scenario put forward in section 3.2.2.

In the following, the analysis results that are achieved given both types of input are evaluated face to face for a single half time of a fixed match *Virtual Werder vs. Aeolus*. The basic evaluation methodology introduced in section 7.2.2 is applied again in order to retain consistency. Thus, the quality variation induced by the alternate input data is

Motion Class	Type	$\#P_{System}$	$\#P_{Truth}$	$\#\cap$	m_{prec}	m_{rec}
Coach Data	Virtual Werder 3D vs. Aeolus (Game 4, 1^{st}Half)				1500 analysis cycles	
kick	event	50	53	47	0.94	0.887
collective kick	event	4	2	2	0.5	1.0
receive	event	64	64	60	0.938	0.938
ball_transfer	action	44	49	39	0.886	0.796
pass (success)	action	20	21	19	0.950	0.905
pass (failure)	action	20	23	17	0.850	0.739
self_assist	action	5	5	3	0.600	0.600
Agent Data	Virtual Werder 3D vs. Aeolus (Game 4, 1^{st}Half)				1500 analysis cycles	
kick	event	43	53	43	1.0	0.811
collective kick	event	5	2	1	0.2	0.5
receive	event	62	64	60	0.967	0.938
ball_transfer	action	38	49	35	0.921	0.714
pass (success)	action	20	21	17	0.850	0.810
pass (failure)	action	13	23	13	1.000	0.565
self_assist	action	6	5	5	0.833	1.000
header notation: m_{prec}: precision, m_{rec}: recall, $\#P_{xy}$: detections by '*xy*', $\#\cap$: mutual detections						

Table 7.3: Evaluation of the *precision* and *recall* key measures for the first half time of the match *Virtual Werder 3D* vs. *Aeolus* using perceptions from the *SCD Coach* and a *Virtual Werder 3D* player (no.6, midfielder).

expressed in terms of comparative *precision* and *recall* measurements shown in table 7.3.

The *precision* and *recall* key measures for the detection of the considered motion incidences for the coach perception reside within the expectations given the results of the previous section with the constraint that due to the characteristic of the match *Virtual Werder 3D vs. Aeolus* the recall for the kick detection and as a consequence the dependent ball transfer and specializations reside below the norm. However with regard to the comparative evaluation, this can be accepted.

Considering the *recall* key measure for the kick and reception event incidences based on the game perception of the Virtual Werder 3D player 6, a midfielder, table 7.3 shows that even under noisy perception the detection quality retains a surprisingly high level with 93.8% for the reception (coach: 93.8%) and 81.1% for the standard kick (coach: 88.7%). Thus while the recall does not degrade under agent vision, the kick recall degrades mildly about 7.6% where a fraction of this value can be attributed to the analysis' obviously heightened inclination towards a detection of collective kicks. The remainder of the 7.6% decrease in recall seems due to the missing timely recognition of ball acceleration right at the start of a kick such that misclassifications as ball deflection and retreat from ball occur with higher frequency. However, it can be stated that the detection of basic event incidences features a noteworthy robustness against noise and does not become increasingly susceptible to phantom detections. The latter is conveyed clearly via the associated *precision* key measures that are on par or even better than their equivalents under coach perception.

The above-mentioned degradation of the recall performance for standard kick incidences under agent perception influences the recall values for the considered action incidences. Table 7.3 lists a recall value of 71.4% (coach: 79.6%) for the general ball transfer which corresponds to a 8.2% decrease. In the considered game the recall for the detection of derived actions is rather unevenly distributed with a recall of only 56.5% for failed passes (coach: 73.9%) and 81.0% for successful passes (coach: 90.5%). It is expected that this is a particularity of the considered game and that statistically the degradation is similar for both pass characteristics. Another interesting result could be achieved for the self

assists where the recall (100.0%) is better than its coach equivalent (60.0%).

Without interpreting too much into the results obtained in the particular, considered game, it remains to make a note of the fact that indeed the implemented analysis approach can yield serviceable results when employed based upon constrained, noisy perception over extended periods of simulated soccer matches. This holds, even though the distance of the observing agent to the area where the relevant action occurs can naturally grow big as the observing agent is primarily integrated into the strategic overall concept of its team (player role, dynamic positioning strategy). So far however, the implemented analysis system is ignorant of the observer's distance from the focus area.

With respect to improving the detection quality once the implemented spatio-temporal real-time analysis approach is performed by the soccer agents rather than their coach, besides optimized preprocessing of the vision data on the agent side a possible future development path would be a task sharing approach. As all agents of a RoboCup 3D Soccer Simulation League team are homogeneous in their setup, it would be possible to give each of them the ability to perform a continued *LiveSCD* analysis of an ongoing game. In order to reduce computational load and exploit the agent's communication capability the detection of extensive motion incidences could be dispatched dynamically to the agent which is closest to the focus area. A new observer agent could be determined every time the agent responsible so far finds that its distance to the focused area has grown such that it's sight is clouded too much. In order for this dynamic dispatching scheme to work the responsible agent would continuously need to communicate successfully detected motion incidences such that all fellow agents can keep their own qualitative knowledge bases up to date. If such a knowledge broadcasting can be guaranteed to work as desired given the communication constraints imposed by the RoboCup 3D Soccer Server a distributed analysis is worth contemplation for future work. It would be interesting to investigate which fraction of the quality degradation found in this section when the analysis is performed by a single agent alone can be counterbalanced by a shared approach.

7.3 Real-Time Analysis in the RoboCup 3D Soccer Simulation

As mentioned in section 7.1, the *SCD Coach* performed a full real-time spatio-temporal analysis of the dynamic soccer scenes at hand with its integrated *LiveSCD* analysis[19] for each of the fifteen matches that constitute the evaluation basis for this thesis. In the process, statistics with regard to performance, knowledge base charging level with the distinguishable 'fact' types (atomic facts $\mathbb{F}'$, events $\mathbb{E}'$ and actions $\mathbb{A}'$) and detection throughput have been compiled on a per-pass basis for a total of 3000[20] analysis passes respectively. The recorded performance data for each pass comprises the duration for the complete spatio-temporal analysis, measured both in simulation- and real computation time. Moreover, the distribution of consumed time between the two primary tasks of qualitative abstraction and the detection of motion incidences is further broken down for the real computation time.

During qualitative abstraction, the full set of nine qualitative relations $p \in \mathbb{P}^0 \cup \mathbb{P}^1 \cup \mathbb{P}^2$ specified in section 5.1.1 is considered in the compilation of atomic facts $\mathbb{F}'$, applying the *explicit closing* assembly strategy introduced in section 5.1.1 for all objects/object pairs that reside within the focus of attention (cf. section 5.1.6). The oblivion of old atomic facts was applied as introduced in section 5.4.1 in order to balance the size of $\mathbb{F}'$ over the

19 based upon the release version XSB 3.0.1 '*Sagres*' as Prolog backend

20 Soccer agents an the coach within the RoboCup 3D Soccer Simulation receive successive vision updates every 20 sim-steps. A regular game in the 3D Soccer Simulation League has a duration of 600 (simulated) seconds. Each discrete sim-step is associated with a real-time interval of 0.01s, 600s correspond to 60000 sim-steps. With 20 sim-steps per reason-act cycle, this in return corresponds to 3000 analysis passes.

VW3D vs.	Aeolus	Fantasia	WrightEagle	SEU	FC Portugal
Qualitative Abstraction					
Median	1.176	1.026	1.304	1.198	1.13
Q_5/Q_{95}	0.43/1.914	0.34/1.826	0.532/2.328	0.516/2.99	0.362/2.335
Min/Max	0.137/7.227	0.131/5.544	0.167/6.137	0.18/5.181	0.122/6.553
Mean±SD	1.297±0.692	1.12±0.57	1.395±0.652	1.275±0.562	1.242±0.677
Actors/cycle	8.147±2.732	6.871±2.52	8.564±2.62	7.77±2.11	8.339±2.81
Relations/cycle	49.88±16.39	42.2±15.1	52.38±15.73	47.61±12.65	51.03±16.83
Facts in KB	653.3±194.5	588.64±165.4	719.54±180.16	687.07±169.94	644.17±221.08
Recognition of Motion Incidences					
Median	0.99	0.958	1.115	1.341	1.013
Q_5/Q_{95}	0.308/1.914	0.324/1.862	0.369/2.328	0.407/2.99	0.181/2.335
Min/Max	0.11/5.505	0.110/5.412	0.11/15.093	0.111/8.0	0.107/7.793
Mean±SD	1.027±0.599	1.0±0.553	1.218±0.786	1.478±0.924	1.137±0.717
Events/game	581.0	581.0	655.3	738.0	642.0
Events/cycle	0.194±0.6	0.194±0.6	0.218±0.63	0.246±0.66	0.214±0.632
Actions/game	109.0	200.3	157.3	247.3	133.3
Actions/cycle	0.036±0.239	0.07±0.354	0.052±0.316	0.0824±0.41	0.04±0.27
Complete Spatio-Temporal Analysis (QA&DE)					
Median	2.211	2.035	2.491	2.626	2.264
Q_5/Q_{95}	1.018/4.072	0.992/3.612	1.204/4.423	1.284/4.701	0.754/4.27
Min/Max	0.413/8.709	0.302/7.467	0.351/16.262	0.376/9.277	0.298/8.859
Mean±SD	2.324±0.957	2.152±0.812	2.612±1.056	2.754±1.113	2.379±1.083

Table 7.4: Evaluation of (1) Online Performance (real-time, in milliseconds) of the implemented system for the analysis of dynamic soccer scenes within the RoboCup 3D Soccer Simulator, subdivided into qualitative abstraction & recognition, as well as for both primal tasks of the analysis combined (2) Character of qualitative abstraction and detection (for both events and actions).

VW3D vs.	Aeolus	Fantasia	WrightEagle	SEU	FC Portugal
0 SimSteps	2847.0	2919.7	2757.7	2530.0	2729.3
1 SimSteps	153.0	80.3	242.3	470.0	270.7

Table 7.5: Distribution of consumed sim-steps for the complete analysis over the course of complete simulated soccer matches. Note that 0 sim-steps should be read as 'the analysis could be completed in less than the duration of a complete sim-step'.

course of the simulated games.

The detection of extensive motion incidences involved a guided pattern matching against the complete pool of specified event and action/action sequence patterns $\mathbb{A} \cup \mathbb{E}$ that have been formally specified in section 5.3. Thus, seven event classes $e \in \mathbb{E}$ and seven action classes $a \in \mathbb{A} \equiv \mathbb{A}_{seq} \cup \mathbb{A}_{loop}$ are detected.

Table 7.4 provides a comprehensive statistical evaluation of the recorded test data with common location and variance key measures. The particular columns of the table refer to games between Virtual Werder 3D and one of the five considered adversary simulation league teams. The entries in the respective table cells constitute averages over the three complete games per pairing.

7.3.1 Runtime Performance in the Soccer Simulator

With respect to the time consumed on average for the complete analysis of the dynamic scenes per pass, the median computation time runs from 2.035ms to 2.626ms. This is a first hint that the implemented analysis approach is suited for real-time use. A more detailed picture can be acquired via a contemplation of the associated lower and upper

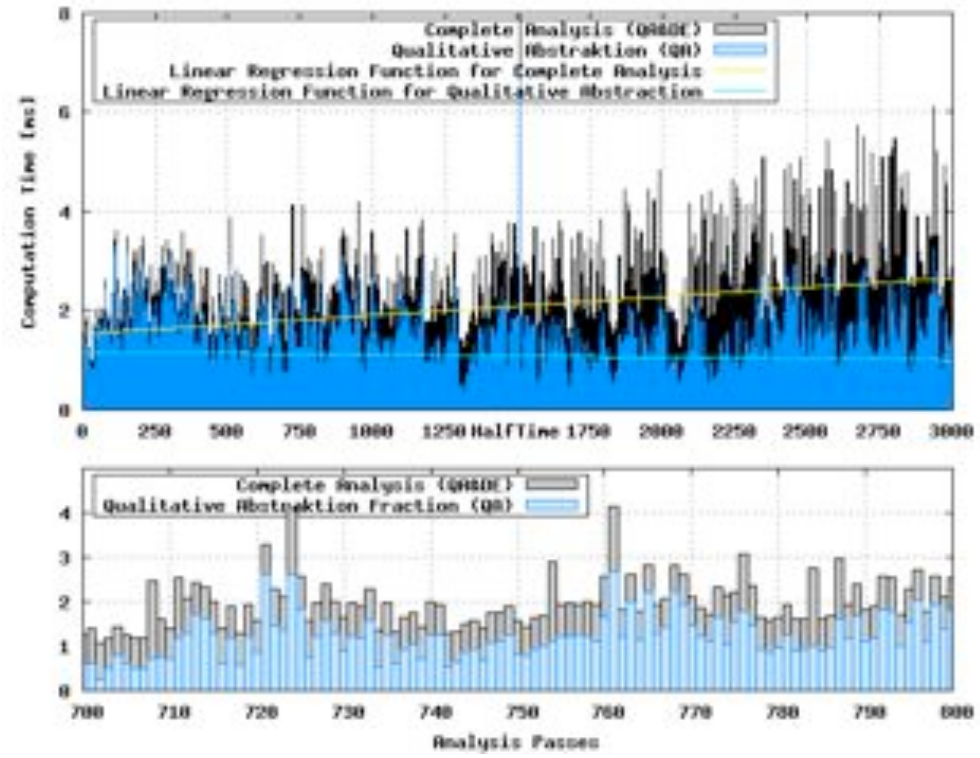


Figure 7.3: Exemplary plot of the performance distribution between qualitative abstraction and detection of extensive motion incidences over the complete course of a test match ('*Fantasia vs. VW3D*', game one of three) and a detailed excerpt of taken from the same game.

quantile values (Q_5, Q_{95}) and the min/max values for the required computation time within a single analysis pass. The Q_{95} values state that for 95% of all analysis cycles, no more that 5ms are required for computation. While the maxima for the respective teams are notably higher as they run between 7.467ms for the matches '*Fantasia vs. VW3D*' and 16.262ms for the matches '*WrightEagle vs. VW3D*', the associated upper quantiles suggest that isolated outliers are responsible for the high maxima while generally speaking the data supports the statement of affairs that the analysis is indeed real-time capable (i.e. for all pairings of teams the median computation time is located roughly halfway between the Q_5 and Q_{95} quantile). The outliers-hypothesis is substantiated by figure 7.3 which shows exemplary a plot of the required computation time for the complete analysis over the course of a single match '*Fantasia vs. VW3D*'. The plot indicates that outliers as captured in the maxima occur, yet they are extremely seldom and thus do not constrain the general real-time aptitude of the implemented analysis as a whole.

This statement is substantiated further by additional data illustrated in table 7.5 where the averaged distribution of consumed sim-steps for the complete analysis is listed for the recorded test matches used in the evaluation. Given the fact that a full reason-act cycle for agents, both the coach and soccer players, within the RoboCup *3D Soccer Simulation League* has a duration of 20 sim-steps, the consumption of at most one complete sim-step is definite proof for the already supposed real-time aptitude of the analysis.

Moreover, the data is highly promising with regard to further utilization of the incrementally compiled qualitative knowledge, as the whole analysis is fast enough to consume only a minimum of valuable time within each successive reason-act cycle such that the applica-

tion of further super-ordinated high-level techniques such as statistical analyses, opponent modelling or plan recognition (cf. chapter 4 on page 29) is a valid option. Briefly put, the approach is fast enough, not only to compile comprehensive qualitative knowledge but also to allow dependent modules to exploit the provided knowledge as starting point.

Shifting the focus of attention back to the data captured in table 7.4 on page 133, the fact that the standard derivation from the mean for the complete computation time as well as for the qualitative abstraction and recognition fractions is comparatively high (roughly one third/ one half the mean) seems conspicuous at first. However, this distinctive feature in the compiled test data can be explained as a characteristic trait of the implemented analysis system. With respect to the qualitative abstraction, passes with higher computational load (i.e. when the ball is moving swift such that the considered spatial relations such as $\mathit{distance}(\text{ball}, \mathit{player})$ and $\mathit{sorientation}(\text{ball}, \mathit{player})$ change rapidly, or large amounts of atomic facts are subjected to oblivion) alternate with passes of relative calmness where the spatial relations remain largely unaltered and the fact oblivion has only a mild effect.

Furthermore, due to the guided recognition approach (cf. section 5.4.3), the detection fraction of the analysis is comparatively expensive in passes where sequences of extensive motion incidences are detected in succession, bottom-up, building upon each other. For instance, when a previously free ball is received by player pl_1 the following chain of successful detections may occur:

$\mathit{receive}(\text{pl}_1) \rightarrow \mathit{ball\_transfer}(\text{pl}_2, \text{pl}_1, \_, \_, \_) \rightarrow \mathit{pass}(\text{pl}_2, \text{pl}_1, \_, \_, \_, \text{success}) \rightarrow \mathit{onetwo\_pass}(\text{pl}_1, \text{pl}_2)$.

In other passes, no motion incidences are detected at all as for example the ball is flying or rolling freely and only a small subset of pattern matchings are attempted (viz. for those patterns whose detection is not triggered via a preceding, successful detection of subordinated motion incidences in the same detection pass). Figure 7.3 on the facing page exemplary shows the oscillation of required computation time for distinct analysis passes.

Concluding the temporal aspects of the evaluation, the respective size of the qualitative abstraction and the detection fractions of the time used for complete analysis passes is roughly equal with light prevalence for the qualitative abstraction. Thus, the detection seems to be well-tuned with regard to performance efficiency. Section 7.4.2 on page 139 will investigate further the influence of the applied guided detection strategy on the detection performance.

7.3.2 Characteristic Traits of the Implemented Analysis System

Besides the performance oriented data discussed so far table 7.4 on page 133 also provides data about the modus operandi and the detection results of the implemented analysis system which should be considered in combination with the plot in figure 7.4 on the next page.

With respect to the qualitative abstraction, table 7.4 denotes data about the mean number of actors inside the focus of attention, that is in the circular region with a radius of 13m around the ball as key object (cf. section 5.1.6 on page 82), and the associated standard derivation. The means differ slightly between 6.87 and 8.14 actors with a standard derivation coarsely located around 2.5. These differences probably mean that teams such as *Aeolus* and *WrightEagle* employ a comparatively compact style of play where more players are attracted to the ball compared to *Fantasia*. Larger numbers of actors in the focus of attention lead to an increased amount of relations that have to be tracked in the qualitative abstraction passes and a higher mean of atomic facts $\mathbb{F}'$ in the knowledge base. Consequently, the time consumed by the qualitative abstraction is increased as well.

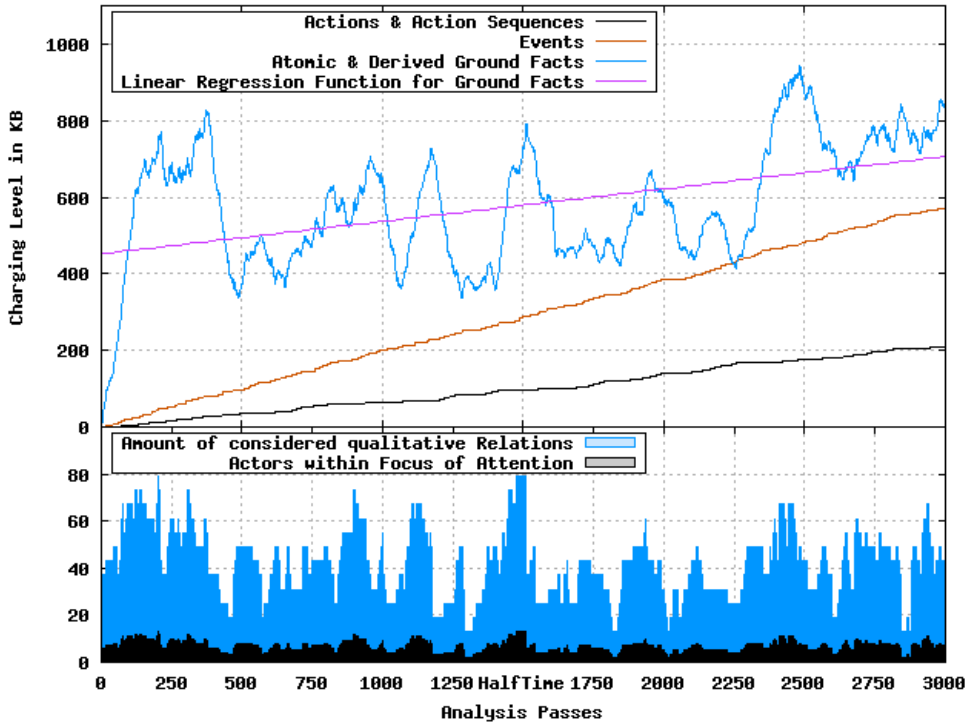


Figure 7.4: Exemplary plot of the charging Level of the Prolog knowledge base with ground facts, events and actions/action sequences respectively over the complete course of a test match ('*Fantasia vs. VW3D*', game one of three) and the development of both numbers of actors in the focus of attention and the resulting number of qualitative relations that need to be tracked.

The large standard derivation from the mean for the charging level of ground facts in the knowledge base can be explained as a consequence of the chosen implementation for fact oblivion strategy introduced in section 5.4.1 on page 104. The upper plot in figure 7.4 illustrates that the charging level of atomic facts oscillates considerably around a slightly increasing linear regression function $f(x)$. Considering both plots shown in figure 7.4, the data seems to suggest a correlation between the number of *relevant* scene actors within the focus of attention (and, as a consequence, the amount of qualitative relations that need to be tracked) and the net growth of atomic facts for over the course of a game. Although the oblivion of old atomic facts, as introduced in section 5.4.3 on page 106, is applied uniformly over the whole course of a game, retracting from the knowledge base in each successive analysis pass those atomic facts whose validity expired more than 2000 sim steps in the past, the net amount of facts increases for those periods with a high density of scene actors in focus and decreases in others. So, while a part of the oscillation can be attributed to the alternating dynamics of the soccer game with phases of relative calmness (ball freewheeling slowly before reception by a player) followed by phases of rapid changes (hard kick of the ball in between a group of players), another part of the oscillation seems to be funded in the fact that the saturation level for atomic facts is correlated to the momentary number of scene actors in focus. Thus both the size of the focus of attention and the age threshold for oblivion influences the balancing efforts that have been introduced in the analysis system as real-time optimization. Section 7.4.3 on page 141 will delve further into the influence of the age threshold on the net charging level of atomic facts over the course of a complete soccer match. The general light net

growth of the charging level of atomic facts towards the end of a game which is suggested by the linear regression function $f(x)$ in the upper plot of figure 7.4 is owing to the matter of fact that a subset of atomic facts, namely those that relate to the play mode and the ball control situation, are exempted from oblivion, as they are considered relevant pieces of information in their own right.

The total number of concrete event/action incidences detected on average over the course of a game runs from 581/109 to 738/243. The style of play of the competing teams as well as the respective sophistication in the treatment of the ball can be considered important factors. For instance, the *SEU* players handle the ball very well and favor rapid forwarding of the ball in their offensive play which is reflected in the associated top detection yield.

7.4 Performance Variation via Real-Time Optimization

The performance contemplation applied in the previous section referred to a single possible configuration for the SCD analysis system developed for this thesis, which was considered the *best choice* for application in a real-time scenario. This best choice assumption shall be substantiated in the remainder of this section via a continuative contemplation where, based on a single, fixed game chosen from the pool of fifteen available test games, namely game one of three that the Virtual Werder 3D team played against the Fantasia, the performance variations induced by three important conceptual choices for the implementation of the qualitative abstraction (the *fact assembly* strategy) and the detection of extensive motion incidences (the *fact oblivion* strategy and the driving *detection strategy*) are investigated.

Due to the fact that the standalone *SCD Simulator* allows for the execution of repeated analyses of the course of a single, fixed soccer match, encoded in an associated log file incrementally recorded by the *SCD Coach* during the original simulation of the game in the preparation of this evaluation, an immediate comparability of game analyses with configurations that differ in a single aspect in order to evaluate the consequences of one of the strategy choices mentioned above, is warranted. The results which are presented in the following sections bear relevance for the analysis approach as a whole and should be credited as indicators for the feasibility of the default configuration used in the aforementioned test game series rather than as a full, statistically grounded dissection of the surveyed features which remains subject to future work.

With regard to the performance results obtained for the implemented analysis of dynamic scenes within the *SCD Simulator*, the results presented in the following sections should not be directly compared with the results obtained on average in the target scenario where the analysis was performed repeatedly by the *SCD Coach* within the RoboCup *3D Soccer Environment*. Rather, the results are upraised as immediate offline comparison measurements only[21].

7.4.1 Influence of the Fact Assembly Strategy

Due to the fact that the qualitative abstractions is the first of two primary task areas that comprise the complete analysis of the dynamic soccer scene at hand in each analysis pass, two possible strategies for the assembly of atomic facts, namely '*explicit closing*' and '*explicit lengthening*', that have both been conceptually introduced in section 5.1.5, p.79, are surveyed with respect to the alteration in performance for the complete process of qualitative abstraction over the course of a game of simulated soccer.

21 Unlike the *LiveSCD* analysis performed by the *SCD Coach*, the *OfflineSCD* analysis in the *SCD Simulator* is based upon the release version XSB 3.1 '*Incognito*' as Prolog backend

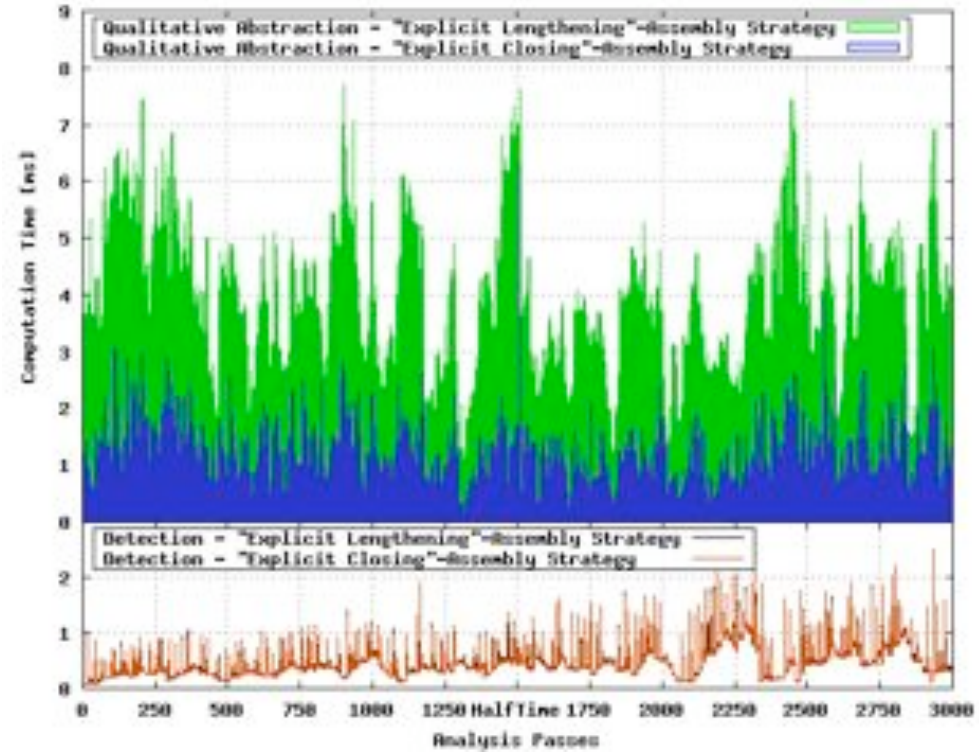


Figure 7.5: Performance comparison of a simulated '*explicit lengthening*' assembly strategy for atomic facts with the default '*explicit closing*' assembly strategy over the course of a fixed game.

The '*explicit closing*' assembly strategy presupposes the utilization of right-open intervals $I_{\vdash}$ and assumes that facts that are asserted as valid from a certain fixed time point forward retain their validity until at a later point in time the contrary is asserted explicitly (i.e. the validity interval for the particular fact is terminated). It is an alternative approach to fact assembly as those advocated by Miene [Mie04a] and Gehrke [Geh05] that get by with closed intervals $I_{\|}$ alone. The equivalent to a fact with a right-open validity interval is a fact whose validity interval features the same beginning, but whose ending always conforms to the momentary present. The potential price to be paid in utilizing this approach is the need to explicitly and repeatedly elongate the validity interval of such facts that retain valid in successive analysis passes[22]. In section 5.1.5, p.79, the assumption was expressed that the latter approach might be inferior for the RoboCup 3D Soccer Simulation application domain.

In order to perform a comparative evaluation, the behavior of the '*explicit lengthening*' assembly strategy was emulated in order to retain the functionality of the remainder of the implemented analysis system[23]: Instead of abandoning the use of right-open intervals $I_{\vdash}$ completely and fully reimplementing the fact assembly for the '*explicit lengthening*' approach, atomic facts with a right-open validity interval that are found to retain their validity for yet another qualitative abstraction pass are explicitly retracted from the knowledge base and reasserted in the same form immediatly afterwards. With respect to the

22 Thus the lengthening cost are most pronounced for those facts that are most resistant to change. For the concrete application domain of simulated soccer, consider play mode facts as an example

23 Several of the motion patterns that have been compiled in section 5.3.1 on page 90 rely on the existence of facts $f \in \mathbb{F}'$ with validity intervals $i \in I_{\vdash}$.

	Explicit Closing	Explicit Lengthening
Qualitative Abstraction		
Median	0.768	3.625
Q_5/Q_{95}	0.175/1.822	1.620/5.927
Min/Max	0.053/6.327	0.560/7.725
Mean±SD	0.859±0.525	3.634±1.274

Table 7.6: Evaluation of the variation in runtime performance (in milliseconds) for the qualitative abstractions using distinct fact assembly strategies over the course of a single, fixed soccer match. The statistical key measures presented here complement the plot in figure 7.5 on the preceding page.

computational expenses for the interaction with the Prolog knowledge base, this course of action is equivalent to a real explicit elongation of a closed validity interval $\langle s, e\rangle \in I_{\|}$ to $\langle s, e+1\rangle$ which also requires both a retract and re-assertion of a certain fact.

The experimental results of the utilization of either of the above-mentioned fact assembly strategies for the same game analyzed in the *SCD Simulator* are presented graphically in figure 7.5 on the preceding page where the respective distribution of consumed computation time is plotted over the course of the test game. In addition, table 7.6 presents a selection of statistical key measures that summarize the qualitative abstraction performance and thus complement the results encoded in the graph.

To start with, a visual inspection of the plotted data in figure 7.5 indicates a superior performance of the *explicit closing* assembly strategy that is also reflected in the key measures. With 0.768ms compared to 3.625ms, the median time is roughly 21% of the *explicit lengthening* case. What is more, the upper Q_{95} quantile for *explicit closing*, with a value of 1.822ms, amounts to only half the median for *explicit lengthening* The lower Q_5 quantile for *explicit lengthening* (1.620ms) is located only marginally below the aforementioned upper Q_{95} quantile for *explicit closing*. However, the performance degradation induced by the utilization of the explicit lengthening fact assembly strategy is not only characterized through the statistical location parameters alone. It is also reflected in variation parameters such as the respective size of the standard derivation and the interquantile distance. Both key measures are far more pronounced in the explicit lengthening case. The greater amount of oscillation in the performance value run is also clearly visible in the performance graphs in figure 7.5.

Even though the general statement of affairs revealed by the evaluation was expected in the run-up, the actual extent of the performance variation for the computation of the qualitative abstraction came as a surprise. The assumption made by Gehrke in [Geh05, p.138] that the *explicit lengthening* fact assembly strategy can yield noteworthy performance gains has been confirmed without doubt in the given experimental setup for the considered application domain of simulated soccer. Beyond that, it is expected that the obtained results are conferrable to other application domains such that the explicit closing strategy should be considered there as well.

7.4.2 Influence of Detection Strategy

While the previous section was concerned with the evaluation of an important aspect in the process of qualitative abstraction, the current section is devoted to the evaluation of the performance gains that can be achieved via the utilization of the *guided, bottom-up detection* approach for extensive motion incidences outlined in section 5.4.3 on page 106, compared with a *brute force, bottom-up detection* approach as a bottom line.

The *guided detection* was developed in the conceptualization done for this thesis as a

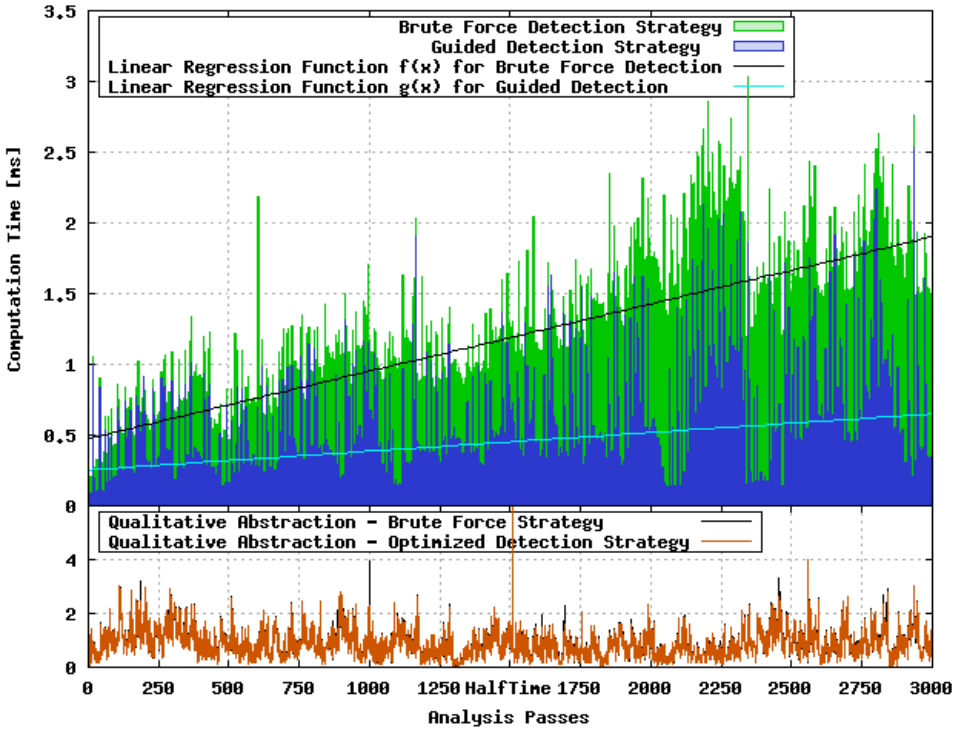


Figure 7.6: Upper Plot: Performance comparison of a brute force motion incidence detection with the guided detection strategy proposed in the concept chapter (section 5.4.3 on page 106) for a fixed game. Lower Plot: The performance for the qualitative abstraction remains unaltered.

means to effectively reduce the computation time for the detection of motion incidences such that in every successive detection pass over the course of a game, the detection of new concrete incidences is only attempted for those equivalence classes of events $e \in \mathbb{E}$ and actions $a \in \mathbb{A} \equiv \mathbb{A}_{seq} \cup \mathbb{A}_{loop}$ where the intermediate yield produced to date in a particular detection pass already provides ample prospects for further detection success. In contrast, the *brute force detection* unalterably tries and detect new incidences for each event or action class regardless of the momentary situation in the dynamic scene.

The experimental results of the utilization of the brute fore as well as the guided detection strategy for the same game analyzed in the *SCD Simulator* are presented graphically side by side in figure 7.6. The graphs illustrate the respective distribution of consumed computation time over the complete course of the test game. In addition, again, table 7.7 on the facing page presents complementary statistical key measures.

An initial contemplation of the results conveyed by the plotted data in figure 7.6 allows for multiple observations. The most obvious of course is the noteworthy performance advantage of the guided bottom-up detection over the brute force equivalent which is reflected clearly in the key measures as well. With 0.371ms, the median computation time for the guided detection amounts to roughly one third of the 1.117ms required for brute force detection. Furthermore, the quantile values indicate that for 95% of all detection passes the time consumed when applying guided detection lies below both the median of 1.117ms and the mean of 1.189ms for the brute force case. Thus, the implemented guided detection succeeded in achieving the desired performance gains required for real-time aptitude of the detection approach.

	Brute Force Bottom-Up	Guided Bottom-Up
Recognition of Motion Incidences		
Median	1.117	0.371
Q_5/Q_{95}	0.476/2.050	0.138/1.004
Min/Max	0.191/3.033	0.079/2.528
Mean±SD	1.189±0.483	0.452±0.289

Table 7.7: Evaluation of the variation in runtime performance (in milliseconds) for the recognition/detection of extensive motion incidences (event, actions and action sequences) using distinct detection strategies over the course of a single, fixed soccer match. The statistical key measures presented here complement the plot in figure 7.6 on the facing page.

The graphs in figure 7.6 suggest that the savings in computation time are most pronounced in those passes where the computation time for the guided detection are low, that is where due to the guidance only a minor subset of detections for distinct event/actions incidences are actually attempted. On the other hand, even in passes where a comparatively large number of new motion incidences is detected the guided detection naturally remains more efficient than its simpler sibling as it never has to try and detect the complete set of specified event and actions classes $\mathbb{E} \cup \mathbb{A}$.

The graph for the brute force strategy also reveals a trend in the development of consumed detection time over the course of the game, in that the mean detection duration increases monotonously regardless of the common oscillations in detection time induced by the varying number of movable objects in the focus of attention (cf. section 7.3.2 on page 135). This trend is reflected in the inclination of the linear regression function $f(x)$ that has been superimposed on the performance graph for the brute force strategy in figure 7.6.

While a similar, yet less pronounced trend can be identified via $g(x)$ for the guided detection, viewed in isolation the significance of the latter slight inclination is unascertained. However, the ascending course of $f(x)$ suggests that, generally speaking, the detection performance of the implemented analysis approach is contingent on the global charging level of the Prolog knowledge base. In order to get a general idea, it is instructive reconsider figure 7.4 on page 136 that has been compiled for the performance evaluation in section 7.3 as that figure refers to the same game used in this evaluation section and illustrates the growth of the knowledge base for that game that is due to a slight growth of the number of atomic facts $\mathbb{F}'$ due to exemptions in the fact oblivion strategy and the accumulation of high level motion incidences $\mathbb{E}' \cup \mathbb{A}'$.

The following section 7.4.3 delves deeper into the correlation between detection performance and knowledge base charging level. Concluding the comparison of detection approaches targeted in this chapter, it remains to notice that the brute force detection is to a larger extent susceptible to the observed creeping performance degradation that burdens the implemented analysis system to an increasing degree with longer application terms.

7.4.3 Influence of the Fact Oblivion Strategy

The concluding paragraph of the previous section put forward the correlation between detection performance for extensive motion incidences and the global knowledge base charging level as subject for evaluation. Consequently, this section is concerned with one aspect thereof in that the correlation of the atomic facts charging level ($|\,\mathbb{F}'\,|$) and the detection performance of the implemented analysis system is evaluated more closely. The choice of a detailed evaluation of this particular aspect of the charge/performance

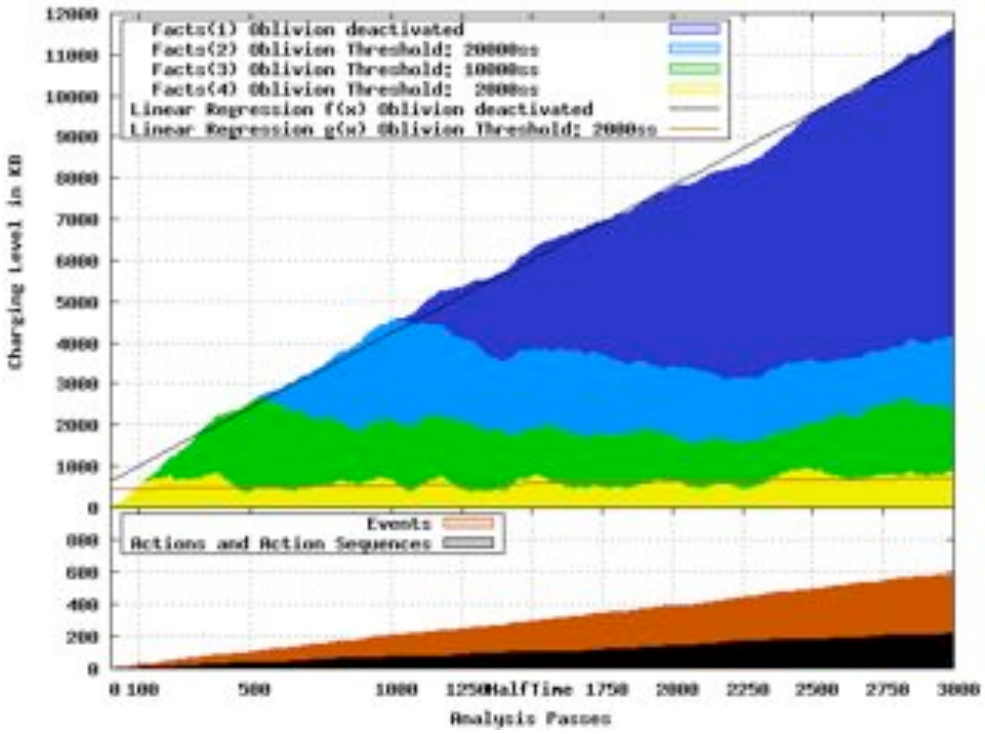


Figure 7.7: Upper Plot: Development of the knowledge base charging level for atomic facts $\mathbb{F}'$ over the course of a fixed game with four distinct parameterizations for the fact oblivion process. The oblivion threshold of $t_{obl}^{def} = 2000$ sim-steps constitutes the default configuration for fact oblivion which was used for the LiveSCD analyses performed for this evaluation. Lower plot: The charging level for events $\mathbb{E}'$ and facts $\mathbb{A}'$ remains unaltered regardless of the oblivion parameterization.

correlation is due to $\mathbb{F}'$ constituting the largest fraction of asserted factual data in the qualitative knowledge base in the analysis application scenario at hand[24] (cf. table 7.4 and the charge graph in figure 7.4, both to be found in section 7.3 on page 132). The concrete evaluation in this section is performed by means of a variation of the oblivion strategy for atomic facts $\mathbb{F}'$, introduced in section 5.4.1, pp.105.

When the oblivion of atomic facts is activated over the course of a complete game, by and by old facts whose validity has expired before a certain oblivion threshold t_{obl} measured in simulation steps, are subjected to oblivion and consequently retracted from the knowledge base. In the online analyses that have been performed by the *SCD coach* during the games of the evaluation test series which constituted the foundation for the online performance evaluation of the implemented analysis approach in section 7.3, a default threshold of $t_{obl}^{def} = 2000$ sim steps was applied. For the evaluation at hand, two more relaxed oblivion thresholds, namely $t_{obl}^{1} = 10000$ sim steps and $t_{obl}^{2} = 20000$ sim steps were applied in distinct offline analyses of a fixed game performed in the *SCD Simulator*. An additional analysis run was performed where the oblivion of atomic facts was deactivated. The intermediate analysis runs with t_{obl}^{1}, t_{obl}^{2} were performed to obtain data for compromise cases in between a very strict oblivion policy on the one hand and the complete lack of an oblivion strategy to constrain the growth of the charging level of

24 comprised of about 3000 analysis passes in total, predetermined by the default duration of games in the RoboCup 3D Soccer Simulation League

atomic facts on the other hand.

The results of the four distinct offline analyses are summarized in two plots. First, figure 7.7 on the facing page illustrates the development of the fact charging level for each oblivion configuration over the complete course of a fixed, simulated soccer match. Second, figure 7.8 on the next page illustrates the respective development of the computation time for the detection of extensive motion incidences. The common contemplation of both graphs is to demonstrate the extent of the correlation of between knowledge base charging level and detection performance.

However, figure 7.7 can be considered in isolation first, as it yields results as to the effect of growing oblivion thresholds t^{i}_{obj}. The graph illustrates how the charging level of atomic facts starts to grow monotonously as indicated by the linear regression function $f(x)$ immediatly after the start of the game/analysis. If the fact oblivion is deactivated, that initial growth continues for the complete course of the game with a stable linear rate of about 6000 asserted atomic facts per half time[25] (1500 analysis passes). If the oblivion of atomic facts is activated, the unbounded growth of the fact charging level is eventually brought to a halt. The moment of the growth cut-off is determined by the respective oblivion threshold. As the oblivion thresholds $t^{def}_{obl} = 2000, t^{1}_{obl} = 10000$ and $t^{2}_{obl} = 2000$ are measured in simulation steps, the associated analysis passes are: $c^{def}_{obl} = 100, c^{1}_{obl} = 500$ and $c^{2}_{obl} = 1000$. The graph illustrates how the cut-off occurs exactly after the calculated amount of analysis passes measured from the start of operations. Once the unbounded growth has subsided, the charging level is adjusted to retain in a narrow value range centered around a certain mean level. This is pointed out paradigmatically for the charge graph associated with the default oblivion threshold via the linear regression function $g(x)$[26]. The graph also illustrates that the maximum derivation from the anticipated mean grows with the size of the oblivion threshold. Thus, for smaller oblivion thresholds the intended regulation effect brought about by the fact oblivion is achieved in a more controlled, sustainable fashion. A deeper evaluation of the exact fact regulation characteristics would be beneficial to substantiate with necessary statistical significance the observations that have been made based on the single considered value run. However, such continuative considerations are deferred to future work.

For the scope of this thesis the observations related to the knowledge base charge alone are considered sufficient. Consequently, the evaluation continues to relate the fact charging level with the detection performance shown in figure 7.8. The four distinct performance graphs for the oblivion configurations discussed earlier yield informative and in part unexpected results which provide valuable insights into the interplay of the fact oblivion process and the concrete implementation of the motion patterns introduced in section 5.3.

Concentrating on the results pertaining the analysis characteristics first, the performance graph for the default oblivion threshold $t^{def}_{obl} = 2000$ sim-steps oscillates in a narrow value range around a mean performance. Over the whole course of the game, the performance does not degrade considerably. These results confirm the data acquired in section 7.3.1 on page 133.

If, on the other hand, the amount of atomic facts keeps growing linearly, unconstrained by oblivion, as illustrated in the particular graph for the knowledge base charging levels, the mean computation time required for each successive detection pass also features a persistent growth. What is more, the data indicates that unlike the constant growth rate observed for the fact charging level, the growth of mean computation time per detection

25 It is important to bear in mind, that the concrete numbers are to be understood as exemplary, both game- and team-dependent benchmarks as they lack statistical backup. Thus, it is more important to concentrate on the conveyed fundamental statement of affairs/trends.

26 g(x) features a light inclination due to the fact that a subset of atomic facts that refer to the ball control state and the play mode are exempted from oblivion.

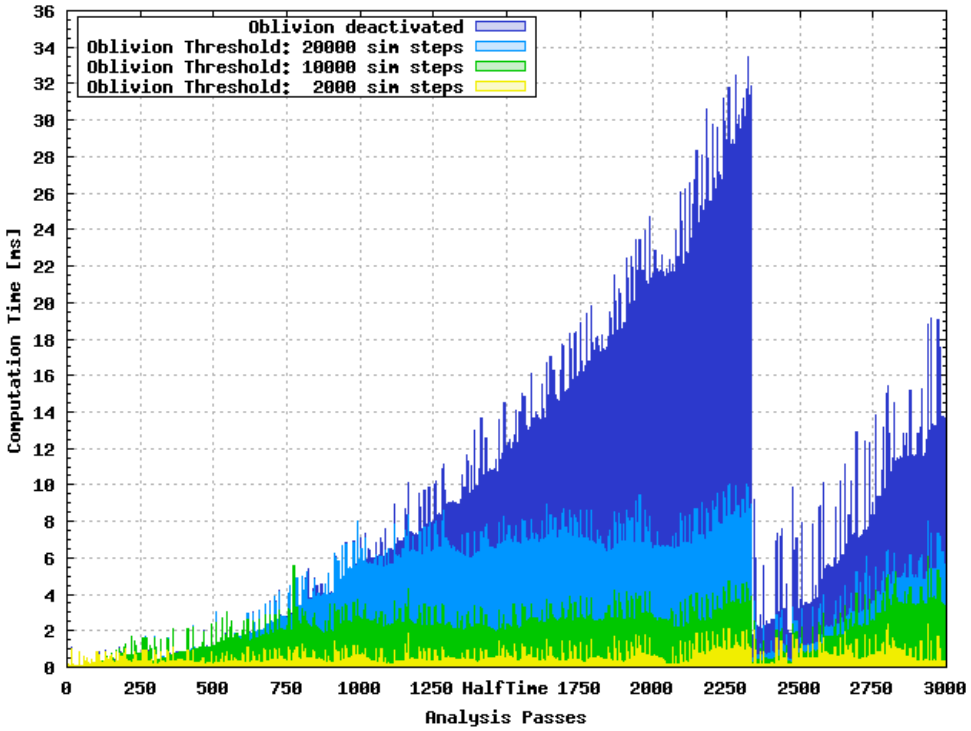


Figure 7.8: Development of the performance for the detection of extensive motion incidences over the course of a fixed game with four distinct parameterizations for the fact oblivion process. Starting from a near-constant base performance for the complete game, a swift performance degradation occurs as the oblivion parameterizations allow for ever higher charging levels of atomic facts $\mathbb{F}'$ as shown in figure 7.7.

accelerates over the course of the game as more and more atomic facts bloat the knowledge base. The growth is pronounced in such a way that for the considered game after 2200 analysis passes more than 30 milliseconds are routinely required per detection pass which is more than 10 times the computation time consumed in the initial phase of the game. The assumption that the detection performance must suffer from unbounded growth of the knowledge base is thus confirmed.

The performance graphs for the intermediate oblivion thresholds $t^1_{obj} = 10000$ and $t^2_{obl} = 20000$ sim-steps conform to the expectations that are based on the contemplation of the respective graphs for the facts charging level and the performance development for the extreme cases discussed above. Just like the amount of atomic facts gets balanced eventually on a notably higher mean level as compared with the default oblivion case, the performance initially degrades for a while before it stabilizes at about 3.5ms for t^1_{obj} and about 9ms for t^2_{obj}. It is interesting that there is a notable delay of roughly 250 analysis passes between the cut-off time points for the respective oblivion thresholds in figure 7.7 and the transition from degrading to stable detection performance in figure 7.8. At the time of writing, the exact reason for this delay is not yet determined and thus needs to be investigated further in order to come up with founded conclusions.

In the consideration of figure 7.8, the sudden collapse of the respective performance graphs after 2334 analysis steps catches the reader's eye immediatly as it does not match with the expectation that the graphs would continue to develop in a rather monotonous way as observed during the initial phase of the game. At first, the sudden dramatic

improvement of overall detection performance followed by another phase of degradation similar to that encountered earlier and thus the forming of a sawtooth-like value run was attributed to internal problems in the XSB Prolog backend. However, this assessment was proven incorrect through a deeper investigation of the detection performance for each particular employed motion pattern, both for event and action patterns.

It turned out that the observed performance degradation is a result of the growing pool of atomic facts that have to be considered in each analysis pass, especially for those patterns that, due to their construction, feature no special connection with the momentary present (e.g. no constituents with right-open intervals) and whose detection is attempted in each single analysis pass due to the lack of a suitable trigger condition. If the concrete occurrence of such a pattern is comparatively rare over the course of the whole analysis run, fruitless detections with ever increasing costs are performed for large number of successive analysis passes until, eventually, a first motion incidence is detected. This successful detection leads to a new situation in that depending on the character of the pattern, in successive analysis passes, only asserted facts in the knowledge base that lie beyond the time point of the most recent incidence are considered in further matchings. This is implemented for patterns such as ball deflection (cf. section 6.2.2, p.119) and causes a very efficient resizing of relevant facts that need to be considered in the pattern matching immediatly in succession to each successful incidence detection. From this point forward however, the performance degradation starts again as observed before.

Thus, the sawtooth-characteristic of the performance graphs can be explained not as an extrinsic effect but as a consequence of a resizing of the set of relevant facts as compared to all facts contained in the knowledge base. With these results in mind, the influence of the implemented oblivion strategy can be assessed from a better founded perspective. As figure 7.8 on the preceding page indicates, the oblivion of atomic facts goes along well with the automatic resizing of the time frame for relevant pattern constituents. It effectively causes the amount of relevant facts to dwell below a certain mean level (dependent on the respective t_{obl}) in-between successive successful detections of incidences for particular motion patterns. Thus, particularly with the use of restrictive oblivion thresholds such as the default of $t_{obl}^{def} = 2000$ sim-steps, the performance for the matching of critical patterns is warranted to remain acceptably high, regardless of whether or not the incidences associated with a certain motion pattern occur frequently or only now and then during the course of an analysis run.

To conclude, based on the performed observations, it is rendered obvious that the implementation of an effective oblivion strategy is crucial for the efficient utilization of the whole analysis system, especially in an environment with hard real-time demands. Via the configuration of the oblivion threshold parameter which determines the total amount of atomic facts that are kept in the working memory, the system's detection performance is controlled immediatly. Furthermore, it is desirable only to keep a minimum of facts in the knowledge base which is required such that the pattern matching performed in the detection of extensive motion incidences is not corrupted due to prematurely forgotten facts. Doing so, the real-time performance is optimized while at the same time the ability of the detection is preserved.

With respect to the observed unbounded growth in the size of relevant atomic facts in-between detections that is mitigated but not conceptually treated via the fact oblivion strategy, the results suggest the development of a more sophisticated approach for the resizing of the time frame for relevant pattern constituents that is not bound to the successful detection of motion incidences such that the efficiency of the resizing is not correlated with the frequency of incidences. A possible alternative which might be suited, is the introduction of a fixed window size for relevant facts which is determined via expert knowledge about the characteristics of the respective motion pattern similar to the oblivion threshold. Following this line of thought the scope of duties would be rendered more precise as a means to globally countervail the bloating of the knowledge base with

unserviceable qualitative low-level knowledge while the respective pattern matching processes themselves are enabled to regulate the time frame where globally available atomic facts bear relevance to them.

7.5 Discussion

Concluding the evaluation, the results which have been obtained throughout the chapter are briefly summarized and discussed. The evaluation was concerned with two primary criteria which determine whether or not the conceptual approach to real-time spatio-temporal analysis of dynamic scenes in the RoboCup 3D Soccer Simulation League, outlined in chapter 5 can be considered a success. First, the quality of the analysis results, investigated in section 7.2, and second the runtime performance, investigated in section 7.3 and section 7.4.

With regard to the analysis quality, it could be shown that both the qualitative abstraction and the detection of extensive motion incidences building upon the former foundation yield satisfying results under precise, noise-free perception with respect to precision and recall even though the pool of motion patterns developed in the knowledge engineering process for this thesis and implemented for the concrete analysis system prototype achieve only a partial covering of all conceivable motion situations that occur in the chosen application domain. The analysis was shown to work in principle under noisy, partial perception as well with a graceful degradation of analysis quality.

With regard to the analysis performance, the implemented analysis system was shown to be suitable for unconstrained real-time application in the targeted application domain of the RoboCup 3D Soccer Simulation and beyond. In fact, due to the implemented performance optimizations, it is feasible to consider a noteworthy broadening of the detectable motion classes via additions to the existing pool of motion patterns without compromising the real-time capability. Thus, once the revision of the pattern pool, suggested as a solution to some of the detection deficiencies, shows the need of pattern diversification (e.g. more kick patterns) and a more complicated inner composition of those patterns, these changes can be integrated without fear of introducing too much inefficiency.

Concluding, the implemented analysis system seems to provide a promising basis for future development and builds a foundation for dependent research, for instance in the fields of intention and plan recognition.

8 Conclusion

In the preceding chapters, an approach for a comprehensive, fundamental[1] spatio-temporal analysis of dynamic scenes with special emphasis on real-time aptitude that constitutes a fusion of precursory research efforts and own amendments has been developed and tested in the application domain of the RoboCup *3D Soccer Simulation League*. The chapter at hand is devoted to a critical examination and assessment of the achievements worked out throughout this thesis (cf. section 8.1). Subsequently, possible paths for future work both with respect to enhancements of the implemented analysis framework and follow-up research issues are addressed (cf. section 8.2).

8.1 Critical Assessment of the Thesis Achievements & Results

In the incipient motivation of this thesis, constraints of the hitherto available RoboCup *3D Soccer Simulation League* agents developed by the Virtual Werder 3D team, with respect to their purely quantitative world model and associated knowledge base, were pinpointed. It was argued that the existing world model was inadequate with regard to the highly dynamic simulation environment as it did not provide the agents with the ability both to perceive and understand extended motion incidences (events, actions and action sequences) that are characteristic for (simulated) soccer games. Spatio-temporal analysis of dynamic scenes was identified as a suitable means to improve the agents' grounding situation with an incremental compilation of a comprehensive qualitative knowledge base that builds upon expert knowledge about the motion classes that occur in the domain of soccer.

Subsequent to the motivation, a review of the RoboCup challenge was presented, and it was shown to what extent the ambition to outfit software agents with a qualitative representation of their particular task environment can be conceived as a contribution towards artificial soccer players that equal their human archetype not only in terms of

1 The designation of the developed analysis as 'fundamental' is meant as a discrimination as the developed analysis functionality spans solely the detection of extensive motion incidences in dynamic scenes. That is, the knowledge that is gathered is not yet used to derive further conclusions, using e.g. deductive or abductive reasoning techniques.

locomotive skills but also in terms of 'soccer IQ'[2].

Afterwards, several paradigmatic scenarios from the RoboCup *3D Soccer Simulation League* were presented in order to identify the scope of qualitative concepts that should be compiled by the analysis. Starting from these concrete examples, a catalog of requirements was put together which defined, besides the desired target concepts to be represented in the qualitative knowledge base, demands with regard to computational efficiency and the ability to accept and process noisy, potentially incomplete input data.

In the subsequent chapter, a representative selection of relevant approaches to spatio-temporal analysis was reviewed and compared with respect to immediate aptitude as starting point for the development of an own analysis approach. Based on the results of that review, it was decided to build upon seminal work by Miene et al. and both, render the proposed approach suitable for unconstrained real-time deployment and at the same time extend the expressiveness of the detectable motion incidences, enabling the support for a detection not only of basic concepts but also concept characteristics/traits.

Consequently, the development of a vertically integrated analysis approach that comprises the task areas of both qualitative abstraction and the detection of a wide set of extensive motion incidences, is among the main contributions of this thesis. From a critical point of view, it must be restrictively noted, that the qualitative abstraction was handled pragmatically to a large extent as it was the primary goal to obtain a pool of qualitative atomic facts that is 'sufficient' rather than 'optimal' as a starting point for the detection of extensive motion incidences. Thus, the thesis does not allege to build upon a particularly well-grounded and cognitively appropriate qualitative abstraction. It is therefore expected that further follow-up work in this direction is necessary and can contribute to improved detection results.

The analysis approach has been formalized extensively, concentrating on both representational and procedural aspects. It has been fully implemented in the *SCD framework* to an extent that every single conceptual building block that is outlined in the concept chapter is reflected/implemented in the framework.

Seizing the implementation that was produced for this thesis, it was rendered possible to perform a thorough evaluation of the developed analysis approach with regard to analysis quality and execution performance in realistic application scenarios. For the latter, comprehensive benchmarking tests were performed which tested both the overall performance of the *SCD framework* under optimized deployment conditions and performance variations using alternating subsets of available performance optimizations.

It could be shown that the analysis yields promising results with respect to the 'precision' and 'recall' key measures, if it is based upon very precise quantitative input data[3]. In addition, it could be shown that the analysis remains usable to an acceptable degree if degraded input data is used that is noisy to an extent encountered in real application scenarios in the RoboCup *3D Soccer Simulation League*. Although the analysis quality decreases under these conditions, it does so with a graceful degradation. This latter result is promising with regard to potential follow-up research as it demonstrates for the first time the aptitude of the fundamental analysis approach originally developed by Miene under noisy perception, even when a.) the analysis system is not particularly adapted for this modus operandi and b.) the perceiving agent is in part rather distant from the momentary focus of attention due to its normal role assignment in the soccer match.

With respect to execution performance, it could be shown that the developed approach to spatio-temporal analysis of dynamic scenes and its concrete implementation are suitable for non-restricted real-time employment. In particular, the analysis is performed fast

2 The term 'soccer IQ' seems to originate from the US American language area and refers to a player's ability to 'understand' or 'read' the game.

3 Worldstate perceptions obtained from the *SCD Coach* at a relatively low frequency with regard to the high dynamics of the analyzed soccer scenes.

enough that the employment of other high level approaches that exploit the maintained comprehensive qualitative knowledge is rendered feasible.

Concluding, it can be reported that analysis results compiled via *LiveSCD* performed by the *SCD Coach* in extensive series of test games are already successfully used as input data for a loosely related, upcoming diploma thesis by Carsten Elfers that is concerned with the '*Prediction of [Agent] Actions by Means of Relational Hidden Markov Models on the Basis of a Spatio-Temporal Representation*'[4] [Elf07].

8.2 Paths for Future Development

Both the proposed methodology for real-time spatio-temporal analysis of dynamic scenes and the concrete implementation in the *SCD framework* constitute nameable steps towards a comprehensive solution to the thesis' underlying research problem to "*[...]build a soccer (multi-)agent system in which each robot knows it is playing soccer and understands all important elements of the game [...]*" [SW04, p.p.758]. However, along the way several paths for future research and development became apparent that are considered worth pursuing.

8.2.1 Revision of Existing Motion Patterns

To start with, the evaluation of the analysis quality in section 7.2, pp.124 suggested that a thorough revision of the existing motion pattern specifications that were proposed in section 5.3 is compulsory in order to achieve a better coverage of actually occurring variations of the considered motion classes. This applies in particular for the kick events whose correct[5] detection is a critical prerequisite for the detection of dependent motion incidences. In addition, certain motion classes that have been enumerated in the catalog of requirements compiled for this thesis such as 'clearing of the ball' as well as 'scoring' have not been implemented yet. An addition of those classes, especially the 'scoring' or rather 'scoring attempt' was interesting as it would have consequences for the types of objects that are to be considered for the qualitative abstraction and detection. In addition to movable objects (players, ball), static objects (the goal posts) would need to be taken into account. This would necessitate on the long run the modeling of an object taxonomy (cf. [Mie04a, pp.87]) and the integration of explicit role restrictions in the formalizations of the motion patterns[6]. What is more, it would be interesting to investigate to what extent scoring attempts can actually be formalized reliably and which additional spatial-relations would need to be introduced.

Another extension of the existing motion pattern modeling would involve the exploitation of the incidence region classification that is already supported by the *SCD framework*. Seizing these pieces of data, it is possible to introduce additional characterizations to the ball transfer patterns (such as '*opening pass to the wing*' or '*cross pass (into the penalty area)*') and to extend the *SCD system* towards the detection of strategic plays that involve multiple active players such as '*wing attacks*' or '*changeovers*'.

In section 6.2.2, it was shown that the detection of extensive motion incidences in its current implementation is based solely on the pool of qualitative atomic facts that have been compiled hitherto by means of qualitative abstraction. That is, no further use is made of the quantitative low-level input. It might be an interesting research problem to find ways to meaningfully encode the low-level data such that at a later point in time,

4 title translated from German
5 i.e. high precision and recall values
6 For instance, a ball can be deflected by players on the soccer pitch and a goal post while on the other hand only the former can kick or receive the ball.

the detection of high-level concepts can actually benefit from that additional data. A hint how such a hybrid data base may look like was in parts provided in the review of the research by Beetz and colleagues [BBG+06, Sect. 5] who propose a segmentation of player/ball movements into piecewise linear segments. The concept is loosely related to *qualitative motion vectors* (QMVs) introduced by Musto [Mus00, pp.32]. Both formalisms aim at a concise representation of movements (where the approach by Beetz et al. remains basically quantitative in character) which can be used for an interpretation of movement characteristics. For instance, in [Wen03] Wendler uses a the movement vector of pass receivers as a distinct attribute that characterizes a particular pass instance, e.g. 'pass into a receiver's route'[7].

8.2.2 From Handcrafted to towards Learned Motion Patterns

The occupation with the explicit modelling of soccer expert knowledge as formalized motion patterns during the knowledge engineering process, that would have to be continued if the revision/addition of motion classes is implemented, already led to doubts whether optimal results in the pattern design can be achieved seizing only the design skills of the developer. In [LH04, LMVH06, Lat07], Lattner develops a methodology for '*Temporal Pattern Mining in Dynamic Environments*' (MiTemP), that has already been tested with sample data from the RoboCup *3D Soccer Simulation League*. Using this approach it is possible to automatically assemble motion patterns in an unsupervised learning process. It would be interesting to investigate a.) if MiTemP is capable to come up with extensive motion patterns that have equivalents in the pattern pool used so far and thus can provide optimized substitutions for the handcrafted patterns, b.) if MiTemP can be used to refine existing basic motion patterns, thereby extending their coverage of actual motion incidences, and c.) if any of the above questions can be answered positively, whether it is feasible from a technical point of view to adapt the *SCD framework* such that the patterns created by MiTemP can be integrated into externalized detection cores used by compound detectors (cf. section 6.2.2). If a successful integration could be achieved, it was possible to detect concrete incidences of these newly integrated motion classes in matches of the RoboCup *3D Soccer Simulation League*.

8.2.3 Application Scenarios and Further Use

Another important aspect that needs to be addressed in future work is the exploitation of the compiled qualitative knowledge. So far, the information provided by the *SCD framework* over the course of a particular analysis run is not yet exploited, neither for a statistical evaluation of the hitherto perceived course of the game as proposed in section 1.2.3 which constitutes a first, manifest application scenario, nor as a basis for dependent high-level approaches in the field of plan/intention recognition, behavior prediction or opponent modelling. To some extent, preliminary research in the aforementioned fields has already been conducted within the application domain of robotic/simulated soccer (e.g. in the RoboCup *2D Soccer Simulation League* or in the *Four-legged Robot League*) in or with participation of the artificial intelligence working group at the Center for Computing Technologies (TZI). Those approaches could be reviewed with regard to their aptitude as follow-up techniques that seize the preliminary results of the *SCD analysis* in order to improve the competitiveness of the Virtual Werder 3D team. In order to properly support dependent high-level approaches the *SCD framework* must provide efficient means for knowledge retrieval that abstract the underlying XSB Prolog backend. In doing so, the *SCD framework* would develop towards a service provider, granting clearly defined access to qualitative knowledge while at the same time keeping the concrete knowledge base implementation transparent for the clients.

7 translated from German: 'Pass in den Lauf'

8.2.4 Beyond RoboCup: Alternative Application Domains

For the scope of this thesis the analysis methodology was implemented solely for the application domain of simulated soccer in the RoboCup *3D Soccer Simulation League*. As the evaluation yielded acceptable results in this primary domain, it would be interesting to try and port the implemented analysis system to other application domains. Before leaving the sports domain and seeking for applications in general sports analysis, (experimental) biology (e.g. cell tracking or the study of social insect colonies [BKV01]) or automated monitoring (e.g. observance and analysis of traffic scenes (crossroads) or analysis of pedestrian motion patterns in animate public areas), as an intermediate step, an adoption from simulated to real human soccer seems to suggest itself. An application in this field presupposes the availability of raw input data provided by a motion tracking system such as [BGB+07], which can pre-process visual data from video recordings of soccer matches. As a matter of fact, the resulting data of a motion tracking process performed for the final match of the FIFA world championships 2006 between Italy and France are available. These were used as input for various analysis by a variety of research groups as part of a video installation by Harun Farocki[8], titled '*Deep Play*', that was presented to the public at the Documenta XII exhibition in Kassel, Germany in 2007[9]. With regard to the current implementation of the *SCD framework*, a prerequisite for such an transition to other application domains is the introduction of a clear separation of fundamental core analysis functionality and dedicated modules for a scenario-specific adoption of the *SCD framework*.

8 Amongst others, Andrea Miene was involved in the project and the motion captured data was obtained in direct correspondence with Farocki's film crew.

9 A dedicated web site is available at: http://www.documenta12.de/uebersichtsdetails.html?L=0&gk=A&level=&knr=9 (visited:09/10/2007)

Bibliography

[ABD+06] Nima Aghaeepour, Meysam Bastani, Fatemeh Miri Disfani, et al. Ututd2006-3D Team Description Paper for 3D Development Competition. Technical report, University of Teheran, 2006.

[AF94] James.F. Allen and George Ferguson. Actions and Events in Interval Temporal Logic. *Journal of Logic and Computation*, 4(5)(521), 1994.

[AH90] James F. Allen and Patrick J. Hayes. Moments and Points in an Interval-based Temporal Logic. *Comput. Intell.*, 5(4):225–238, 1990.

[AHR88] G. André, G. Herzog, and T. Rist. On the Simultaneous Interpretation of Real World Image Sequences and their Natural Language Description: The System SOCCER. In B. Radig, editor, *Proceedings of the 8th European Conference on Artificial Intelligence (ECAI)*, pages 449–454, Munich, Germany, 1988.

[All81] James F. Allen. An Interval-based Representation of Temporal Knowledge. In P.J. Hayes, editor, *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI-81)*, volume 2, pages 221–226, 1981.

[All83] James F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[All84] James F. Allen. Towards a General Theory of Action and Time. *Artificial Intelligence*, 23(2):123–154, 1984.

[BBG+06] Michael Beetz, Jan Bandouch, Suat Gedikli, Bernhard Kirchlechner, et al. Camera-based Observation of Football Games for Analyzing Multi-Agent Activities. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, 2006.

[BCP+97] Inderpal S. Bhandari, Edward Colet, Jennifer Parker, Zachary Pines, Rajiv Pratap, and Krishnakumar Ramanujam. Advanced Scout: Data Mining and Knowledge Discovery in NBA Data. *Data Mining and Knowledge Discovery*, 1(1):121–125, 1997.

[BGB+07] Michael Beetz, Suat Gedikli, Jan Bandouch, Bernhard Kirchlechner, Nico von Hoyningen-Huene, and Alexander Perzylo. Visually Tracking Football Games Based on TV Broadcasts. In *IICAI*, 2007.

[BKF04] Michael Beetz, Bernhard Kirchlechner, and Fischer F. Interpretation and Processing of Position Data for the Empirical Study of the Behavior of Simulation League Robot Teams. In *KI 2004 Workshop*, 2004.

[BKL05] Michael Beetz, Bernhard Kirchlechner, and Martin Lames. Computerized Real-Time Analysis of Football Games. *IEEE Pervasive Computing*, 4(3):33–39, 2005.

[BKN+04] Tjorben Bogon, Mirco Kuhlmann, Cord Niehaus, Steffen Planthaber, Carsten Rachuy, Arne Stahlbock, Ubbo Visser, and Tobias Warden. Description of the Team Virtual Werder 3D 2004, June 2004.

[BKV01] Tucker Balch, Zia Khan, and Manuela Veloso. Automatically Tracking and Analyzing the Behavior of Live Insect Colonies. In Jörg P. Müller, Elisabeth Andre, Sandip Sen, and Claude Frasson, editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 521–528, Montreal, Canada, 2001. ACM Press.

[Bor01] Michael Boronowsky. *Diskretisierung reellwertiger Attribute mit gemischten kontinuierlichen gleichverteilungen und ihre Anwendung bei der zeitreihenbasierten Entscheidungsbauminduktion*. Number 246 in DISKI. infix Verlag, 2001.

[Bra01] Ivan Bratko. *Prolog Programming for Artificial Intelligence*. Pearson/Addison Wesley, 3 edition, 2001.

[BSK+04] Michael Beetz, Flossmann Sven, Bernhard Kirchlechner, et al. Watching Football with the Eyes of Experts: Integrated Intelligent Systems for the Automatic Analysis of (Simulated) Football Games. In *5th Annual Conference dvs-Section Computer Science in Sport*, 2004.

[Cav00] Marc Cavazza. High-Level Interpretation in Virtual Environments. In *Applied Artificial Intelligence*, volume 14, pages 125–144. Taylor & Francis, 2000.

[CH01] Anthonny Cohn and Shyamanta M. Hazarika. Qualitative Spatial Representation and Reasoning: An Overview. In *Fundamenta Informaticae*, volume 46, pages 1–29, 2001.

[CNO+02] Mao Chen, Itsuki Noda, Oliver Obst, Pat Riley, Timo Steffens, et al. *RoboCup Soccer Server - for Soccer Server Version 7.07 and later*, August 2002.

[DFL03] F. Dylla, A Ferrein, and G. Lakemeyer. Acting and Deliberating using Golog in Robotic Soccer. In *The Third International Workshop on Cognitive Robotics*, AAAI, 2003.

[DHS+01] Christian Drücker, Sebastian Hübner, Esko Schmidt, Ubbo Visser, and Hans-Georg Weland. Virtual Werder. In Peter Stone, Tucker Balch, and G. Kraetzschmar, editors, *RoboCup 2000: Robot Soccer World Cup IV*, volume 1 of *Lecture Notes in Arificial Intelligence*, pages 421–424, Berlin, Heidelberg, Germany, 2001. Springer-Verlag.

[DLM+05] F. Dylla, G. Lakemeyer, Jan Murray, Oliver Obst, Ubbo Visser, et al. Towards a League-Independent Qualitative Soccer Theory for Robocup. In *8th International Workshop on RoboCup 2004 (Robocup World Cup Soccer Games and Conferences)*, 2005.

[Elf07] Carsten Elfers. Aktionsvorhersage durch relationale Hidden Markov Modelle auf der Basis einer qualitativen raumzeitlichen Repräsentation. Master's thesis, Universität Bremen, 2007.

[FIF06] FIFA. *The Laws of the Game 2006*. Fédération Intaernationale de Football Association, FIFA-Strasse 20, 8044 Zurich, Switzerland, July 2006.

[Fre92] Christian Freksa. Temporal Reasoning Based on Semi-Intervals. In *Artificial Intelligence*, volume 54, pages 199–227, 1992.

[FSW04] Gordon Fraser, Gerald Steinbauer, and Franz Wotawa. Application of Qualitative Reasoning to Robotic Soccer. In *18th International Workshop on Qualitative Reasoning*, Illinois, USA, 2004.

[Geh05] Jan D. Gehrke. Qualitative Szenenrepräsentation für intelligente Fahrzeuge. Master's thesis, Universität Bremen, 2005.

[GLH04] Jan D. Gehrke, Andreas D. Lattner, and Otthein Herzog. Qualitative Mapping of Sensory Data for Intelligent Vehicles. Technical report, TZI - Center for Computing Technologies, Universität Bremen, 2004.

[HBG+96] Gerd Herzog, Anselm Blocher, Klaus-Peter Gapp, Eva Stopp, and Wolfgang Wahlster. VITRA: Verbalisierung visueller Information. In *Informatik - Forschung und Entwicklung*, volume 11, pages 12–19. Berlin, Heidelberg: Springer, February 1996.

[HCDF95] Daniel Hernández, E. Clementini, and P. Di Felice. Qualitative Distances. In *Proceedings of COSIT'95, LNCS 988*, Semmering, Austria, 1995. Springer.

[Her94] Daniel Hernández. *Qualitative Representation of Spatial Knowledge*. Number 804 in Lecture Notes in Arificial Intelligence. Springer Verlag, Berlin, 1994.

[Her95a] Gerd Herzog. From Visual Input to Verbal Output in the Visual Translator. In *Proceedings of the AAAI Fall Symposium on Computational Models for Integrating Language and Vision*. Cambridge University Press, July 1995.

[Her95b] Gerd Herzog. Utilizing Interval-Based Event Representations for Incremental High Level Scene Analysis. Technical Report 91, Universität des Saarlandes, 1995.

[HG89] Gerd Herzog and Retz-Schmidt Gudula. Das System SOCCER: Simultane Interpretation und natürlichsprachliche Beschreibung zeitveränderlicher Szenen. In J. Perl, editor, *Sport & Informatik*, pages 95–119, 1989.

[HM05] Tony Hoare and Robin Milner. Grand Challenges in Computing Research. In *Conference on Grand Challenges in Computing Research & Computing Education*, 2005.

[Hsu02] Feng-Hsiung Hsu. *Behind Deep Blue: Building the Computer that Defeated the World Chess Champion*, volume 1. Princeton University Press, 2002.

[IB99] Stephen S. Intille and Aaron F. Bobick. A Framework for Recognizing Multi-Agent Action from Visual Evidence. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 1999.

[IB01] Stephen S. Intille and Aaron F. Bobick. Recognizing Planned, Multiperson Action. *Computer Vision and Image Understanding: CVIU*, 81(3):414–445, 2001.

[Int99] Stephen Sean Intille. *Visual Recognition of Multi-Agent Action*. PhD thesis, Massachusetts Institute of Technology, 1999.

[KA00] Hiroaki Kitano and Minoru Asada. The RoboCup Humanoid Challenge as the Millenium Challenge for advanced Robotics. In *Advanced Robotics*, volume 13, pages 723–736, 2000.

[KAK+95] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa. RoboCup: The Robot World Cup Initiative. In Proceedings of the IJCAI-95 Workshop on Entertainment and AI/ALife, 1995.

[Kar05] Björn Karlsson. *Beyond the C++ Standard Library - An Introduction to Boost*. Addison Wesley, 1 edition, 2005.

[KFCV03] Gal Kaminka, Mehmet Fidanboylu, Allen Chang, and Manuela Veloso. Learning the Sequential Coordinated Behavior of Teams from Observation. In Gal Kaminka, Pedro Lima, and Raul Rojas, editors, *RoboCup 2002: Robot Soccer World Cup VII*, number 2752 in Lecture Notes in Arificial Intelligence, pages 111–125. Springer Verlag, Berlin, 2003.

[KLP+05] Mirco Kuhlmann, Andreas D. Lattner, Steffen Planthaber, Carsten Rachuy, Arne Stahlbock, Ubbo Visser, and Tobias Warden. Virtual Werder 3D 2005 Team Description, June 2005.

[KO04] Marco Kögler and Oliver Obst. Simulation League: The Next Generation. In Polani et al. [Kög03], pages 458 – 469.

[Kög03] Marco Kögler. Simulation and Visualization of Agents in 3d Environments. Master's thesis, Universität Koblenz/Landau, 2003.

[KTS+97] H. Kitano, M. Tambe, P. Stone, M. Veloso, S. Coradeschi, E. Osawa, H. Matsubara, I. Noda, and M. Asada. The RoboCup Synthetic Agent Challenge. In *International Joint Conference on Artificial Intelligence (IJCAI97)*, 1997.

[Lat07] Andreas D. Lattner. *Temporal Pattern Mining in Dynamic Environments*. PhD thesis, Universität Bremen, Bremen, Germany, May 2007.

[LH04] Andreas D. Lattner and Otthein Herzog. Unsupervised Learning of Sequential Patterns. In *ICDM 2004 Workshop on Temporal Data Mining: Algorithms, Theory and Applications (TDM'04)*, Brighton, UK, November 1st 2004.

[LMVH06] Andreas D. Lattner, Andrea Miene, Ubbo Visser, and Otthein Herzog. Sequential Pattern Mining for Situation and Behavior Prediction in Simulated Robotic Soccer. In Ansgar Bredenfeld, Adam Jacoff, Itsuki Noda, and Yasutake Takahashi, editors, *RoboCup-2005: Robot Soccer World Cup VIII*, volume 4020 of *Lecture Notes in Computer Science*, pages 118–129, Osaka, Japan, 2006. Springer Verlag, Berlin.

[LPR+06] Andreas D. Lattner, Steffen Planthaber, Carsten Rachuy, Arne Stahlbock, Ubbo Visser, and Tobias Warden. Virtual Werder 3D 2006 Team Description, June 2006.

[LRS+06] Andreas D. Lattner, Carsten Rachuy, Arne Stahlbock, Ubbo Visser, and Tobias Warden. Virtual Werder 3D Team Documentation. Technical Report 36, TZI - Center for Computing Technologies, Universität Bremen, September 2006.

[LW05] Fahong Li and Robert J. Woodham. Analysis of Player Actions in Selected Hockey Game Situations. Technical report, Department of Computer Science, University of British Columbia, Vancouver, Canada, 2005.

[MBdSG+06] Norbert M. Mayer, Joschka Boececker, Rodrigo da Silva Guerra, Oliver Obst, and Minoru Asada. 3D2Real: Simulation League Finals in Real Robots. In Gerhard Lakemeyer, Elizabeth Sklar, Domenico G. Sorrenti, and Tomoichi Takahashi, editors, *RoboCup 2006: Robot Soccer World Cup X Preproceedings*, Lecture Notes in Artificial Intelligence, 2006.

[Mie04a] Andrea Miene. *Räumlich-zeitliche Analyse von dynamischen Szenen*. PhD thesis, Universität Bremen, 2004.

[Mie04b] Andrea Miene. *Räumlich-zeitliche Analyse von dynamischen Szenen, Appendix*. PhD thesis, Universität Bremen, 2004.

[MLVH04] Andrea Miene, Andreas D. Lattner, Ubbo Visser, and Otthein Herzog. Dynamic-Preserving Qualitative Motion Description for Intelligent Vehicles. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV '04)*, pages 642–646, 2004.

[Mül02] Uwe Thomas Müller. Beschreiben und Erkennen von Verhaltensmustern beim simulierten Fußballspiel. Master's thesis, Humboldt-Universität zu Berlin, Berlin, Germany, 2002.

[Mus00] Alexandra Musto. *Qualitative Repräsentation von Bewegungsverläufen*. PhD thesis, Technische Universität München, 2000.

[MV02] Andrea Miene and Ubbo Visser. Interpretation of Spatio-Temporal Relations in Real-Time and Dynamic Environments. In A. Birk, S. Coradeschi, and S. Tadokoro, editors, *RoboCup 2001: Robot Soccer World Cup V*, volume 2377 of *Lecture Notes in Computer Science*, pages 441–447. Springer, 2002.

[MVH04] Andrea Miene, Ubbo Visser, and Otthein Herzog. Recognition and Prediction of Motion Situations Based on a Qualitative Motion Description. In D. Polani, B. Browning, A. Bonarini, and K. Yoshida, editors, *RoboCup 2003: Robot Soccer World Cup VII*, volume 3020 of *Lecture Notes in Computer Science*, pages 77–88. Springer, 2004.

[MW06] A. Miene and T. Wagner. Static and Dynamic Qualitative Spatial Knowledge Representation for Physical Domains. *KI - Kunstliche Intelligenz*, Vol. 2/06:pp. 109–116, 2006.

[New02] Monty Newborn. *Deep Blue: An Artificial Intelligence Milestone*. Springer, 2002.

[NM06] B. Neumann and R. Möller. On Scene Interpretation with Description Logics. In H.I. Christensen and Nagel H.-H., editors, *Cognitive Vision Systems: Sampling the Spectrum of Approaches*, number 3948 in Lecture Notes in Computer Science, pages 247–248. Springer, 2006.

[NMHF97] Itsuki Noda, Hitoshi Matsubara, Kazuo Hiraki, and Ian Frank. Soccer Server: a Tool for Research on Multi-Agent Systems. To appear in Applied Artificial Intelligence, 1997.

[NMMR04] Ranjit Nair, Tambe Milind, Stacy Marsella, and Taylor Raines. Automated Assistants for Analyzing Team Behaviors. *Journal of Autonomous Agents and Multi-Agent Systems*, 8(1):69–111, 2004.

[Nod95] Itsuki Noda. Soccer Server : a Simulator of Robocup. In *Proceedings of AI Symposium 1995*, pages 29–34. Japanese Society for Artificial Intelligence, December 1995.

[NSH+98] Itsuki Noda, Suzuki Sho'ji, Matsubara Hitoshi, Minoru Asas, and Hiroaki Kitano. RoboCup-97: The First Robot World Cup Soccer Games and Conferences. *AI Magazine*, 19(3):49–59, 1998.

[O+06] Oliver Obst et al. *RoboCup Soccer Server 3D Manual*, July 2006.

[OR05] Oliver Obst and Markus Rollmann. SPARK – A Generic Simulator for Physical Multiagent Simulations. *Computer Systems Science and Engineering*, 20(5), September 2005. To appear.

[Ret91] Gudula Retz-Schmidt. Recognizing Intentions, Interactions, and Causes of Plan Failures. In *User Modeling and User-Adapted Interaction*, volume 1, pages 173–202, The Netherlands, 1991. Kluwer Academic Publishers.

[Ret92] Gudula Retz-Schmidt. *Die Interpretation des Verhaltens mehrerer Akteure in Szenenfolgen*. Informatik-Fachberichte; Sub-Series: Artificial Intelligence; Thesis;. Berlin [a.o]: Springer, 1992.

[RHA87] Thomas Rist, Gerd Herzog, and Elisabeth André. Ereignismodellierung zur inkrementellen High-level Bildfolgenanalyse. In E. Buchberger and J. Retti, editors, *3. Österreichische Artificial-Intelligence-Tagung*, pages 1–11. Berlin, Heidelberg: Springer, 1987.

[RN95] Stuart Russel and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall International, Inc., New Jersey, USA, 1995.

[Roj06] Raul Rojas. The Book - Robots Playing Soccer: A forever growing Documentation of the FU-Fighters. http://robocup.mi.fu-berlin.de/pmwiki/Main/TheBook, August 2006.

[RS88] Gudula Retz-Schmidt. A REPLAI of SOCCER: Recognizing Intentions in the Domain of Soccer Games. In *Proceedings of the 8th European Conference on Artificial Intelligence (ECAI)*, pages 455–457, 1988.

[RTM00] Taylor Raines, Milind Tambe, and Stacy Marsella. Automated Assistants to Aid Humans in Understanding Team Behaviors. In *AGENTS '00: Proceedings of the fourth international conference on Autonomous agents*, pages 419–426, New York, NY, USA, 2000. ACM Press.

[SFL06] Stefan Schiffer, Alexander Ferrein, and Gerhard Lakemeyer. Qualitative World Models for Soccer Robots. In Stefan Wölfl and Till Mossakowski, editors, *Qualitative Constraint Calculi, Workshop at KI 2006*, pages 3–14, 2006.

[SSW+06a] Konstantinos Sagonas, Terrance Swift, David S. Warren, et al. *The XSB System (Version 3.0) Volume 2: Libraries, Interfaces and Packages*, July 2006.

[SSW+06b] Konstantinos Sagonas, Terrance Swift, David S. Warren, et al. *The XSB System (Version 3.0) Volume 1: Programmer's Manual*, July 2006.

[SW04] Christopher Stanton and Mary-Anne Williams. Grounding Robot Sensory and Symbolic Information Using the Semantic Web. In Daniel Polani et al., editors, *RoboCup 2003: Robot Soccer World Cup VII*, pages 757–764. Berlin, Heidelberg: Springer, 2004.

[SWW05] Gerald Steinbauer, Jörg Weber, and Franz Wotawa. From Real-world to its Qualitative Representation – Practical Lessons Learned. In *18th International Workshop on Qualitative Reasoning*, pages 186–191, Graz, 2005.

[TNF+98] Kumiko Tanaka-Ishii, Itsuki Noda, Ian Frank, Hideyuki Nakashima, Koiti Hasida, and Hitoshi Matsubara. MIKE: An Automatic Commentary System for Soccer. In Yves Demazeau, editor, *Proc. of Third International Conference on Multi-Agent Systems*, pages 285–292, July 1998.

[VAHR99] Dirk Voelz, Elisabeth Andre, Gerd Herzog, and Thomas Rist. Rocco: A RoboCup Soccer Commentator System. In Minoru Asadan and Hiroaki Kitano, editors, *RoboCup 1998: Robot Soccer World Cup II*, number 1604 in Lecture Notes in Arificial Intelligence, pages 50–60. Springer Verlag, Berlin, 1999.

[Wel94] Daniel S. Weld. An Introduction to Least Commitment Planning. *AI Magazine*, 15(4):27–61, 1994.

[Wen03] Jan Wendler. *Automatisches Modellieren von Agenten-Verhalten: Erkennen, Verstehen und Vorhersagen von Verhalten in komplexen Multi-Agenten-Systemen*. PhD thesis, Humboldt-Universität zu Berlin, 2003.

[WH05] Thomas Wagner and Kai Hübner. An Egocentric Qualitative Spatial Knowledge Representation Based on Ordering Information for Physical Robot Navigation. In D. Nardi, Martin Riedmiller, and C. Sammut, editors, *RoboCup 2004: Robot Soccer World Cup VIII*, volume 3276 of *Lecture Notes in Arificial Intelligence*, pages 134–149, Heidelberg, Germany, 2005. Springer-Verlag.

Appendix

A.1 Contents of the DVD

The following register outlines the contents of the DVD that is distributed with the paperback version of the thesis at hand.

〉 thesis 〉

This directory contains both the LaTeX source for the diploma thesis at hand as well as two distinct versions thereof for print and screen.

... 〉 `thesis_print.pdf`

The standard version of the thesis which is identical with the paperback version.

... 〉 `thesis_screen.pdf`

Screen version of the thesis with more comprehensive colorization. In particular the constituents of the event and action patterns introduced in section 5.2, p.84 and p.86 are associated with a color which is consequently employed for the concrete motion patterns introduced in section 5.3, pp.89. This visual aids may assist in understanding more deeply the composition of the particular motion patterns.

〉 code 〉

This directory contains the concrete implementation of the *SCD framework*.

... 〉 `requirements.sh`

This is a bash script which determines whether or not a particular target system satisfies the requirements for third party software in order to build, install and use SCD software such as the *SCD Simulator*. It lists relevant dependencies and consequently performs automated checks.

... 〉 `Makefile`

This is a master makefile which controls the installation of SCD applications. It allows the installation/deinstallation of the *SCD Simulator* using the '`make simulator/simulator_uninstall`' targets. The *SCD Simulator* will be installed in 〉 ${HOME} 〉 bin 〉, libraries in 〉 ${HOME} 〉 lib 〉. As this is a development installation, the library headers are also installed in 〉 ${HOME} 〉 include 〉. Throughout the build process, more detailed information is provided.

... 〉 project_scd 〉

This directory contains the source code for the *SCD Core Library* which implements the main analysis functionality. A doxygen-generated documentation is available in html-format (see below).

... 〉 build 〉

... ⟩ core ⟩
... ⟩ doc ⟩ html ⟩ index.html
... ⟩ eval ⟩
... ⟩ knowledgebase ⟩
... ⟩ xml ⟩
... ⟩ configure
... ⟩ Makefile

... ⟩ project_scd_simulation ⟩

This directory contains the source code for the *SCD Simulator* that was used both for development and evaluation purposes. A doxygen-generated documentation is available in html-format (see below).

... ⟩ build ⟩
... ⟩ config ⟩

This directory contains the comprehensive xml-configuration of the *SCD analysis* that was referred to in the implementation chapter (`config_scd.xml`).

... ⟩ config_scd.xml
... ⟩ config_scd.dtd
... ⟩ config_sim.xml
... ⟩ config_sim.dtd

... ⟩ core ⟩
... ⟩ doc ⟩ html ⟩ index.html
... ⟩ knowledgebase ⟩
... ⟩ prolog ⟩

This directory contains the Prolog implementation that was done for the scope of this thesis. The file `rules.P` is of special importance as it contains the detection cores and thus the concretely implemented motion patterns. The files `interval_[...].P` contain the Prolog implementation of the respective interval relations.

... ⟩ configure
... ⟩ Makefile

... ⟩ project_scd_coach_lib ⟩

This directory contains the source code for the *SCD Coach Lib* that constitutes a branch from the *Virtual Werder 3D Coach Lib*. It contains all additions to the coach that were necessary for the integration of the *SCD analysis*. A doxygen-generated documentation is available in html-format (see below).

... ⟩ behaviors ⟩
... ⟩ build ⟩
... ⟩ communication ⟩
... ⟩ core ⟩
... ⟩ doc ⟩ html ⟩ index.html
... ⟩ effectors ⟩
... ⟩ knowledgebase ⟩
... ⟩ skills ⟩
... ⟩ configure
... ⟩ Makefile

... ⟩ project_scd_coach ⟩

This directory contains the source code for the *SCD Coach* executable. This is really only a slim launcher that integrates the *SCD Coach Lib*. Therefore, no special doxygen-documentation is provided.

... ⟩ build ⟩
... ⟩ config ⟩
... ⟩ core ⟩
... ⟩ prolog ⟩
... ⟩ configure
... ⟩ Makefile

... ⟩ project_vw3d_base_lib ⟩

This directory contains the source code for the *Virtual Werder 3D Base Lib* which provides functionality used by the *SCD Core Lib*, the *SCD Simulator* and the *SCD Coach Lib*.

⟩ tools ⟩

... ⟩ rcssserver3d-mod ⟩

This directory contains a version UTUtd 3D Soccer Server that was patched for use in the diploma thesis such that standard binaries from regular RoboCup competitions can be used (which required in essence a minor modification in the agent initialization process). This server was used during development and evaluation.

... ⟩ agent_binaries ⟩

This directory contains agent binaries from other RoboCup teams. In particular 32-bit binaries of the agents used in the evaluation are provided.

... ⟩ xsb_incognito_3.1 ⟩
... ⟩ xsb_sagres_3.0.1 ⟩

⟩ evaluation ⟩

... ⟩ evaluation_basis_logfiles ⟩

This directory contains log files of the series of test matches that constitutes the basis for the largest part of the evaluation performed in chapter 6.

... ⟩ aeolus_vw3d_0{1|2|3} ⟩
... ⟩ fantasia_vw3d_0{1|2|3} ⟩
... ⟩ fcportugal_vw3d_0{1|2|3} ⟩
... ⟩ seu_vw3d_0{1|2|3} ⟩
... ⟩ wrighteagle_vw3d_0{1|2|3} ⟩

Each folder contains the following files:

1. coach.log

   The coach log compiled over the course of the game for use with the *SCD Simulator*

2. monitor.log

   Monitor log for replay of the game in the *3D Soccer Monitor* (rcssmonitor3d-lite)

... ⟩ eval_optimizations ⟩

Coach logs copied from the raw input folders, basis for performance evaluation.

... ⟩ evaluation_basis_logfiles_extra ⟩

This folder contains log files of a series of additional test matches where besides the *SCD Coach*, the Virtual Werder 3D agents also wrote agent logs over the course of the games that are suited as input for the *SCD Simulator*.

... ⟩ aeolus_vw3d_01 ⟩
... ⟩ fantasia_vw3d_01 ⟩
... ⟩ seu_vw3d_01 ⟩

Each folder contains:

1. `coach.log`

   The coach log compiled during the game for use with the *SCD Simulator*
2. `agent11-stdout.log` – `agent21-stdout.log`

   Agent logs compiled during the game for use with the *SCD Simulator*
3. `monitor.log`

   Monitor log for replay of the game in the 3D Soccer Monitor (rcssmonitor3d-lite)

... ⟩ quality_evaluation_reports ⟩

... ⟩ [...].log

Debug output generated during offline analysis runs in the *SCD Simulator* which contains amongst others data about all detected motion incidences.

... ⟩ quality_evaluation_reports_extra ⟩

... ⟩ [...].log

Debug output generated during offline analysis runs in the *SCD Simulator* for the comparison of analysis result under coach- and agent perception.

... ⟩ quality_evaluation_gamesequences ⟩

... ⟩ ball_passage ⟩

Snapshot sequence for scene where the ball passes the influence sphere of an uninvolved player which leads to a false ball transfer detection.

... ⟩ performance_evaluation_data ⟩

... ⟩ [...].dat

Input data for statistical evaluation and the compilation of plots, produced during offline analysis runs in the *SCD Simulator*.

... ⟩ [...].plt

GnuPlot scripts that were used to produce plot shown in chapter 7.

... ⟩ performance_evaluation_plots ⟩

Plots that were produced during the performance evaluation phase.

... ⟩ performance_evaluation_statistics ⟩

... ⟩ eval_strategy_assembly.ods

Statistical evaluation for '*Explicit Closing*' vs. '*Explicit Lengthening*' fact assembly strategy (→ Open Office format)

... ⟩ `eval_strategy_detection.ods`

Statistical evaluation for '*Brute Force*' vs '*Guided*' motion incidence detection strategy (→ Open Office format)

... ⟩ `eval_strategy_oblivion.ods`

Statistical evaluation for different parameterization of the fact oblivion strategy (→ Open Office format)

... ⟩ `eval_runtime_performance.ods`

Statistical evaluation of the integrated runtime performance of the implemented analysis system (→ Open Office format)

⟩ `literature` ⟩

This directory contains the literature database compiled over the course of this thesis as well as a repository of all referenced documents that could be directly obtained. All documents in the repository are in pdf-format.

A.2 Excerpts from the SCD Configuration

```
<scd:mapping_engine_unary id="classifier_velocity_ball"
                          status="&on;"><!--on-->
 <!-- + <config_details> -->
 <scd:data_source value="extractor_velocity"/>
 <scd:predicate id="velocity"/>
 <scd:supported_types value="ball"/>
 <scd:bound_flexibility value="0.05"/>
 <!-- + <classification_core> -->
 <scd:class_core id="&open;"
                 assembly_strategy="&expl_closing;">
  <scd:classification_type_open>
   <!-- Thresholds Open Range Axis -->
   <scd:thresholds value="0.07,0.25,0.5,0.9,1.3,5.0"/>
    <!-- Concrete Classes -->
    <scd:classes value="rest,very_slow,slow,moderate,fast,very_fast,beam"/>
  </scd:classification_type_open>
 </scd:class_core>
 <!-- - </classification_core> -->
 <scd:managed_oblivion validity_duration="2000"/>
 <!-- - </config_details> -->
</scd:mapping_engine_unary>
```

Figure A.1: Excerpt from the *SCD configuration* which is provided as part of the thesis DVD (⟩ `code` ⟩ `project_scd_simulation` ⟩ `config` ⟩ `config_scd.xml`). The excerpt shows the configuration for the unary mapping engine which is responsible for the *velocity* of the ball. In particular the complete choice/parameterization of the appropriate classification core is shown.

A.3 EBNF of Agent Worldstate Messages

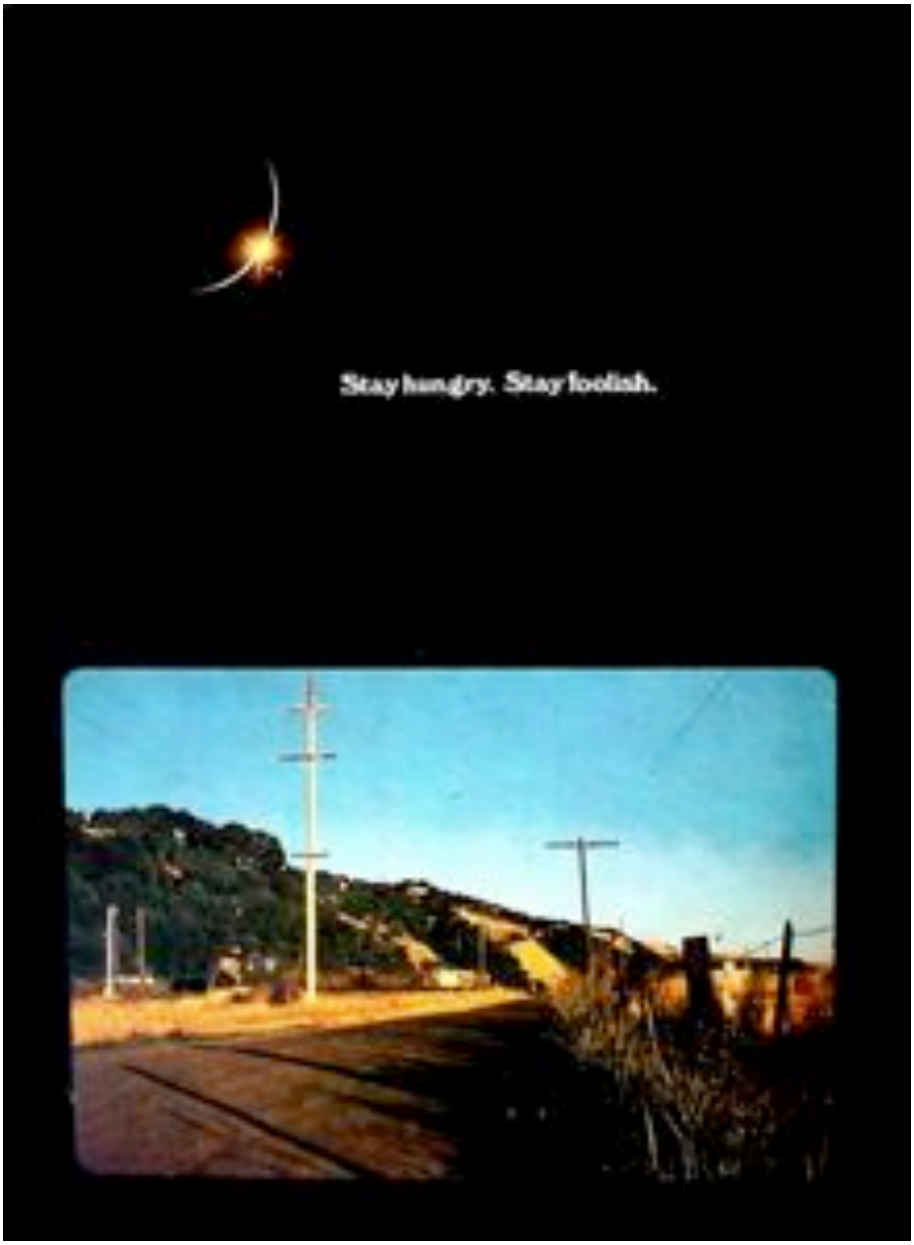
The format of the worldstate messages which are compiled by both *Virtual Werder 3D* players and the *SCD Coach* in order to obtain a complete game-record from the respective agent's point of view, is expressed in EBNF-notation as follows:

$$
\begin{aligned}
\mathit{worldstate} &:= \text{(eval } \mathit{time\_info}\ \mathit{kb\_info} \text{)} \\
\mathit{time\_info} &:= \text{(time (sim } \mathit{stime} \text{) (game } \mathit{gtime} \text{) (playmode } \mathit{mode} \text{))} \\
\mathit{kb\_info} &:= \text{(kb (team } \mathit{side}\ \mathit{name} \text{) (} \mathit{moving\_objects}\ \mathit{moving\_object}* \text{))} \\
\mathit{objects} &:= \mathit{player} \mid \mathit{myself} \mid \mathit{ball} \\
\mathit{player} &:= \text{(mo (team } \mathit{side}\ \mathit{name} \text{) (unum } \mathit{no} \text{) } \mathit{position}\ \mathit{velocity} \text{)} \\
\mathit{myself} &:= \text{(mo (team } \mathit{side}\ \mathit{name} \text{) (unum } \mathit{no} \text{) } \mathit{position}\ \mathit{velocity} \text{ (myself))} \\
\mathit{ball} &:= \text{(mo (id BALL) } \mathit{position}\ \mathit{velocity} \text{)} \\
\mathit{position} &:= \text{(pos } x\ y\ z \text{)} \\
\mathit{velocity} &:= \text{(vel } x\ y\ z \text{)} \\
x, y, z, \mathit{gtime} &:= \text{float} \\
\mathit{side}, \mathit{no}, \mathit{stime}, \mathit{mode} &:= \text{int} \\
\mathit{name} &:= \text{string}
\end{aligned}
$$

Seizing the style of messages used in the two-way communication between soccer agents and the RoboCup 3D Soccer Server, the worldstate messages are valid Lisp-like *S-expressions* [O+06].

List of Figures

List of Tables



from The Last Whole Earth Catalogue, Menlo Park (CA) 1971, editor: Steward Brand et al.