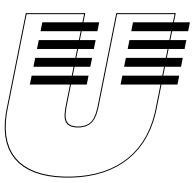

UNIVERSITÄT
KOBLENZ · LANDAU

Qualitative Velocity and Ball Interception

Frieder Stolzenburg,
Oliver Obst, Jan Murray

4/2002

Fachberichte
INFORMATIK

Universität Koblenz-Landau
Institut für Informatik, Rheinau 1, D-56075 Koblenz

E-mail: researchreports@infko.uni-koblenz.de,
WWW: http://www.uni-koblenz.de/fb4/

Qualitative Velocity and Ball Interception*

Frieder Stolzenburg, Oliver Obst, Jan Murray†

Universität Koblenz-Landau, AI research group
Universitätsstr. 1, D–56070 Koblenz, GERMANY
{stolzen,fruit,murray}@uni-koblenz.de

Abstract

In many approaches for qualitative spatial reasoning, navigation of an agent in a more or less static environment is considered (e.g. in the double-cross calculus [13]). However, in general, the environment is dynamic, which means that both the agent itself and also other objects and agents in the environment may move. Thus, in order to perform spatial reasoning, not only (qualitative) distance and orientation information is needed (as e.g. in [1]), but also information about (relative) velocity of objects (see e.g. [2]). Therefore, we will introduce concepts for qualitative and relative velocity: (quick) to left, neutral, (quick) to right. We investigate the usefulness of this approach in a case study, namely ball interception of simulated soccer agents in the RoboCup [11]. We compare a numerical approach where the interception point is computed exactly, a strategy based on reinforcement learning, a method with qualitative velocities developed in this paper, and the naïve method where the agent simply goes directly to the actual ball position.

1 Motivation

There is some literature on qualitative spatial reasoning where the subject of consideration is motion observation and representation (see e.g. [6]). Few references can be found dealing with objects moving in the environment of the subject. It also seems so far that velocity of objects is not considered in the qualitative reasoning community. However, in robotics it is necessary to deal with velocity of both the robot and objects in the environment. Taking robotic soccer in the RoboCup as an example, we have to reason about velocities of the ball, team mates and adversaries.

*This paper will be presented at *Spatial Cognition III* (Tutzing, Germany, May 2002) as a poster.

†The authors are partially supported by the grants *Fu 263/6-1* and *Fu 263/8-1* from the German research foundation *DFG*.

1.1 Robotic Soccer as Example Scenario

In order to evaluate different approaches and compare them with each other, we make use of the RoboCup soccer simulator. The RoboCup [11] is an international research and education initiative, attempting to foster artificial intelligence and intelligent robotics research by providing a standard problem where a wide range of technologies can be integrated and examined. There are annual tournaments in different leagues with real robots of different sizes or virtual, i.e. simulated robots. The authors of this paper are involved in the simulation league (see e.g. [5, 10]) with simulated soccer agents using the Soccer Server [3, 7], a physical soccer simulation system.

The Soccer Server is a system that enables autonomous agents consisting of programs written in various programming languages to play a match of soccer (association football) against each other. A match is carried out in a client/server style: A server provides a virtual field and simulates all movements of a ball and players. Each client controls movements of one player. Communication between the server and each client is done via UDP/IP sockets. The Soccer Server consists of two programs, the server itself and a monitor. The server program simulates the ball and players movements, communicates with clients, and controls a game according to the rules. All games are visualized by displaying the field of the simulator by the soccer monitor on a computer screen.

In general, reasoning about velocities can be of interest in dynamic environments to estimate when and if at all one agent can reach another one. In the case of physical soccer robots, accurate formulæ for object movements may not be available or difficult to obtain. The RoboCup simulation league, where these formulæ are known to the robots, seems to be an ideal environment to test whether qualitative processing of velocity is possible at all and how a qualitative computation performs in contrast to a numeric one.

1.2 Overview of the Paper

In the sequel, we first introduce several quantitative methods devoted to solve the example problem, namely ball interception (Section 2). After this, we discuss qualitative approaches from the literature and introduce a new method that makes use of qualitative direction and velocity information (Section 3). Then, we evaluate all methods introduced in this paper, showing that the qualitative method does not perform too badly in contrast to the other approaches (Section 4). Finally, we give some conclusions (Section 5).

2 Ball Interception with Numerical Methods

Information about velocity is especially important in applications, where spatial agents are situated in a highly dynamic environment. This means, not only the agent moves,

but also objects or other agents in the environment move and change their position, possibly with high or varying speed. In the following, we consider robotic soccer as application scenario, which has the desired properties. In particular, we will address the problem of ball interception. Thus, we have the situation that the soccer ball is rolling with a certain velocity towards a certain direction. Now it is the task of the soccer agent to reach the ball as fast as possible.

2.1 Soccer Simulation in the RoboCup

The optimal method for ball interception obviously is to compute the interception point exactly, considering relative position, orientation and velocity of the agent and the ball, and then let the agent go to this point as quickly as possible. We must take into account the physical laws for the ball and player movement. This means, that the ball becomes slower and slower, the longer it rolls, and every player certainly has a maximal speed for running. Regarding all this might lead to a fairly complicated procedure for computing the interception point, depending on how precise and realistic the modeling of physical laws such as friction etc. is done.

Fortunately, the ball movement model and also the player dash model that is implemented in the Soccer Server is relatively simple, so that by a straightforward iterative procedure, the interception point can be computed effectively. We will develop the respective formulæ further down. Unfortunately, in this model, it is assumed that the ball acceleration $\vec{a}$—that is the derivative of its velocity $\vec{v}$ wrt. the time t—is negatively proportional to $\vec{v}$, i.e. $\vec{a} \sim -\vec{v}$. However, this does not correspond to any standard physical phenomena such as friction, where $\vec{a}$ is constant, or air resistance, where we have $\vec{a} \sim -\vec{v}^2$.

In consequence, this means that the derivation for computing the interception point (in Section 2.2) is only valid in the context of the movement model in the Soccer Server. This is certainly one reason, why we should address the problem with a qualitative approach (see Section 3.3). What are in general the advantages of a qualitative approach?

- Qualitative approaches are likely to be more robust wrt. changes in the world model, because they abstract or approximate the physical reality, which is far more complicated and unreliable than the Soccer Server world. Each change in the world model requires an adaption of a hard-coded quantitative method.

- It is an interesting question on its own to compare the performance of qualitative and quantitative approaches. Even if the exact computation of the interception point can be done (as for the Soccer Server see below), a qualitative method may be simpler and not worse than a quantitative method.

- Another motivation to try it with a qualitative approach is that probably real (human) soccer players do not solve differential equations while chasing the ball.

2.2 Computing the Interception Point

Let us now derive the formulæ for computing the exact interception point. It is used in our reference method for the comparison with other approaches. In the Soccer Server, the ball speed at time t is calculated as $v(t) = \mu \cdot v(t-1)$ with $\mu < 1$. Therefore, if the velocity of the ball is v_0 at time $t = 0$, we have $v(t) = v_0 \cdot \mu^t$. For the distance s, that the ball has moved after t steps, it holds:

$$s(t) = \sum_{i=1}^{t} v_0 \cdot \mu^{i-1} = v_0 \cdot \frac{1-\mu^t}{1-\mu}$$

In the sequel, we make the assumption that an agent can move in any direction with a fixed (maximal) velocity v_1. At time $t = 0$, the ball is at position $\vec{a}$ in the Cartesian coordinate system with the agent at its origin. Let $\vec{b}$ (with $\|\vec{b}\| = v_0$) be the velocity vector of the ball in this coordinate system. Then, after t steps the position of the ball is:

$$P(t) = \vec{a} + s(t) \cdot \vec{b}$$

Clearly, the agent can reach the ball at any time t with $\|P(t)\| \leq v_1 \cdot t$. Since there is no closed form for t, we apply *Newton's method* in order to find the zeros of $f(t) = \|P(t)\| - v_1 \cdot t$. Therefore, we need the derivative $f'(t) = \|P\|'(t) - v_1$ with

$$\|P\|'(t) = \frac{P'(t) \cdot P(t)}{\|P(t)\|}$$

where in this context $\cdot$ denotes the inner product of vectors. We have to compute the following sequence t_n, until $|f(t_n)| < \varepsilon$ for some small threshold $\varepsilon > 0$:

$$t_n = \begin{cases} 0, & \text{if } n = 0 \\ t_{n-1} - \frac{f(t_{n-1})}{f'(t_{n-1})}, & \text{if } n > 0 \text{ and } f'(t_{n-1}) < 0 \\ 999, & \text{otherwise} \end{cases}$$

This procedure eventually yields the first of at most three zeros $t > 0$. There exists at least one zero; it is found at latest after t_n has been set to 999, which avoids oscillation. If there are three zeros, then Newton's method will find the smallest one. This follows from the fact that the acceleration a of the ball (the derivative of v) is negatively proportional to v. A similar (but different) method for computing the interception time has been described in [12]. Figure 1 sketches the situation for ball interception and also gives an example for the function f with three zeros. We see that, after a phase where the ball can be reached by the player, there is a phase where the ball is out of reach, and only at the end, when the ball has slowed down sufficiently, the ball can be reached again.

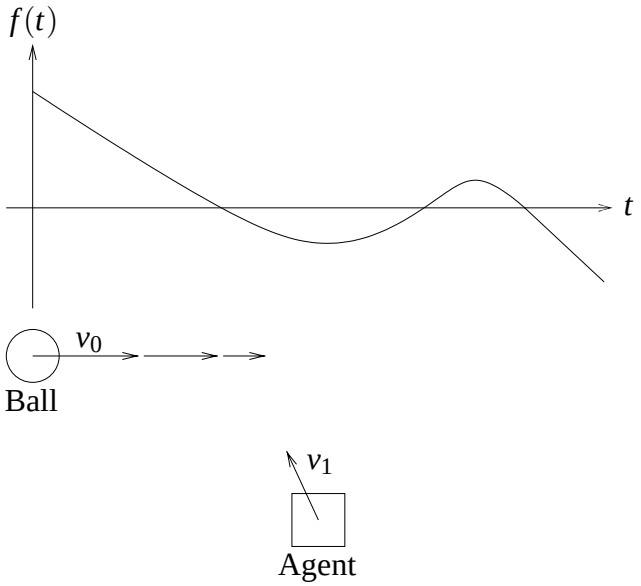

Figure 1: Reaching the ball.

2.3 Applying Reinforcement Learning

An analytical approach to ball interception relies on exact knowledge of system behavior. For a qualitative approach, this might not be necessary, but at least some control knowledge (what action leads to what behavior) must be available. The reinforcement learning approach described in [8] assumes neither control knowledge nor system knowledge in advance (i.e. at the time of programming). A controller learns a policy for a given goal, and at the time of application the learned policy is used to achieve the goal. However, like in the analytical approach, the assumption is that the underlying model of the world does not change during execution. Reinforcement learning has successfully been used for different problems in simulated robotic soccer [9], and in the subsequent sections we are going to compare a ball interception method trained with reinforcement learning and our methods (see Section 4).

3 Ball Interception with Qualitative Velocity

For a brief overview about qualitative reasoning in general, refer [4]. The article comes up with a motivation and some short historical remarks about the field of qualitative reasoning. The author gives some real world application examples as well as an introduction into basic qualitative reasoning techniques.

3.1 Approaches with Qualitative Velocity or Trajectories

Representation of motion in a qualitative manner is the subject of the paper [6]. The goal is to represent the trajectory of a moving object and abstract from irrelevant data at the same time. The representation in this paper is based on sequences of qualitative motion vectors, which consist of two components, that is a qualitative direction and a qualitative distance. The qualitative direction component is representing the direction of the movement, while the qualitative distance component depicts the distance the object moved into the given direction. These two components alone are not sufficient to represent the speed of the object. Though it is not in the focus of the paper, it can be found that for a qualitative representation of motion speed one has to use the ratio of time elapsed between two directional changes and the distance. However, for the qualitative representation of motion tracks, the speed is not used during the rest of the paper.

An approach to modeling behavior of physical devices qualitatively is described in [2]. Qualitative values are assigned for numerical values and derivatives. Qualitative variables are used to explain different states of a physical system. The authors explain rules that have to be valid for qualitative variables in order to get a correct description of a system. In this approach, qualitative velocity and also its derivative, the acceleration is considered. However, these values are reduced to one dimension with range $+$, $-$ or 0. By this abstraction, some problems can be described quite adequately, e.g. a pressure regulator and a mass-spring-friction system. However, spatial information is not present in this approach, that is needed for robotic soccer. For further details, the reader is referred to the paper [2].

The double-cross calculus [13] has been invented to navigate using qualitative spatial information. The double-cross calculus uses a set of three points and a set of 15 base relations to achieve this. The three points (the observer, the point where the observer is looking to and a reference object) are related by one of the relations. It is also possible to represent incomplete information by using unions of relations. Like in the above-mentioned approaches, a static environment is assumed. Hence, this approach is also not sufficient for highly dynamic environments. Nevertheless, the double-cross calculus has applications in geographical information systems (GIS).

The qualitative approaches mentioned here, first and foremost, are dedicated to the cognitively adequate description of physical reality. But in this context, we want to apply qualitative information to the control of agent behavior. We will see that only qualitative direction and velocity is sufficient for the task of ball interception. One major observation is that the direction to the interception point need not be given too exact (see Section 3.2). This is one of the main ingredients for the qualitative approach for ball interception (see Section 3.3).

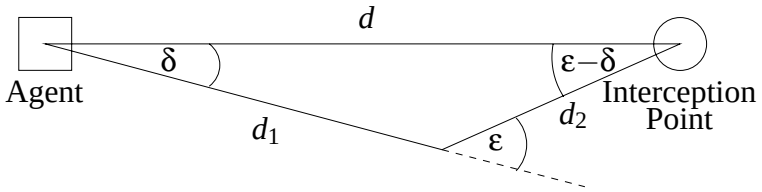

Figure 2: Correction of direction.

3.2 Qualitative Correction of Direction

The optimal strategy of an agent for ball interception is to first turn its body towards the interception point and then head to this point directly. If the interception point can be computed exactly, then the agent just has to move the distance d to this point, after turning its body once by the angle δ, that is the difference between the current orientation of the agent and the direction to the ball. See also Figure 2. Thus, the agent could obey the following rule: if the interception point is straight ahead, continue the movement into this direction; otherwise, correct the orientation accordingly.

If there is no inaccuracy in either the sensor data about directions or the actions performed by the agent, then this rule yields an optimal strategy for the agent. In this case, the agent turns only once, namely at the beginning, and then moves straight forward to the interception point. However, in practice (i.e. for real or simulated robots), the direction often cannot be determined exactly. Hence the rule stated above leads to the undesirable behavior that the agent corrects its orientation most of the time, and does not come closer to the interception point.

Therefore, the rule should be relaxed and a correction should only be done, if the angle for correcting the direction exceeds a certain threshold ε. This strategy is also illustrated in Figure 2. The distance $d_1 + d_2$ the agent has to move in this case clearly is longer than d (because of the triangle inequality). Of course, this is a disadvantage, but as the evaluation (in Section 4) reveals, the agent does not turn too often and therefore reaches the goal faster.

The interesting question now is clearly: how much does the distance increase, if the correction of direction is done only after a certain threshold is exceeded? For this, we apply *Mollweide's formula* for triangles with sides a, b and c, and corresponding angles α, β and γ:

$$\frac{a+b}{c} = \frac{\cos\frac{\alpha-\beta}{2}}{\sin\frac{\gamma}{2}}$$

If we apply this equality to the triangle in Figure 2, we get the ratio

$$\frac{d_2+d_1}{d} = \frac{\cos\frac{\delta-(\varepsilon-\delta)}{2}}{\sin\frac{180^\circ-\varepsilon}{2}} = \frac{\cos(\delta-\frac{\varepsilon}{2})}{\cos\frac{\varepsilon}{2}}$$

which becomes maximal for a given (fixed) threshold angle ε, if $\cos(\delta-\frac{\varepsilon}{2}) = 1$, i.e.

$\delta = \frac{\varepsilon}{2}$. Thus in the worst case the ratio is $1/\cos\frac{\varepsilon}{2}$. As the following table shows, the overhead wrt. the distance can be neglected for angles up to about 30°.

threshold angle ε	0°	5°	10°	20°	30°	45°	60°	90°
overhead of $1/\cos\frac{\varepsilon}{2}$	0.0%	0.1%	0.4%	1.5%	3.5%	8.2%	15.5%	41.4%

Therefore, the threshold can be chosen quite high. This does not decrease the performance of the agent for ball interception significantly. Hence, an approximating, i.e. a qualitative method seems to be very appropriate in this context. We will state a qualitative method for ball interception in Section 3.3. Its evaluation is given in Section 4.

3.3 A Method with Qualitative Velocity

A naïve method for ball interception would be just to go straight ahead to the ball. This certainly can be seen as a non-numerical, qualitative approach. But since the ball moves, this strategy obviously can be improved. It is better to go to the (earliest) interception point directly. However, since it often cannot be computed exactly, a qualitative approach is preferable. This can be done by making use of qualitative velocity and directions. If the agent looks at the (possibly) moving ball, the agent can distinguish whether the ball moves (fast) to the left or right (from the point of view of the agent), or there is no clear bias to one side. This means we consider the projection of the ball velocity that is orthogonal to the line from the agent to the ball. Note that we abstract from the component of the velocity which is parallel to this line.

As a measure for qualitative velocity we take the following model, which makes the simplifying assumption that the ball moves with a constant velocity. The ball velocity v_0 is normalized wrt. the constant (maximal) velocity v_1 of the agent. Let us now have a look at Figure 3. The ball moves with velocity v_0 to the point where the agent can intercept it with velocity v_1. The angle β is the angle from the ball between the agent and the (approximated) interception point. The qualitative velocity is now determined by the angle α, that is the angle from the agent between ball and interception point. The agent and the ball must arrive after the same time t at the interception point. Therefore, by applying the law of sine we get

$$\frac{\sin\alpha}{\sin\beta} = \frac{v_0 \cdot t}{v_1 \cdot t}$$

and hence

$$\sin\alpha = \frac{v_0}{v_1} \cdot \sin\beta$$

In order to determine the velocity in a qualitative manner, the agent considers a finite number of sectors around the agent. Sectors are also used for qualitative orientation information in [1]. Usually the number of sectors is a power of 2. Let us first investigate the case with $n = 8$ sectors. Then each sector has the size $\phi = 360°/8 = 45°$.

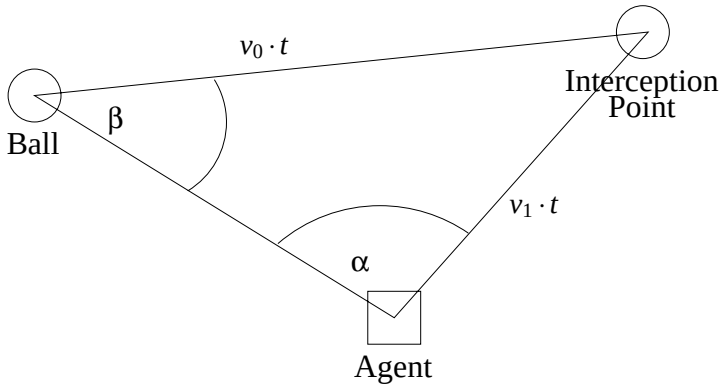

Figure 3: Determining the velocity qualitatively.

In this context, we map the angle α from above to one of the sectors (as shown in Figure 4), i.e. to a positive or negative multiple of ϕ. But since we only approximate the actual movement, we only consider 5 different values for α, namely quick to left ($-90°$), to left ($-45°$), neutral ($0°$), to right ($+45°$), quick to right ($+90°$). Note, that we consider velocity only in one dimension. As before, let δ be the difference angle between the current orientation of the agent and the direction to the ball. Then, the agent just behaves according to the following rule:

If $\delta + \alpha \geq \phi$, then turn by the angle $\delta + \alpha$; otherwise, go straight ahead.

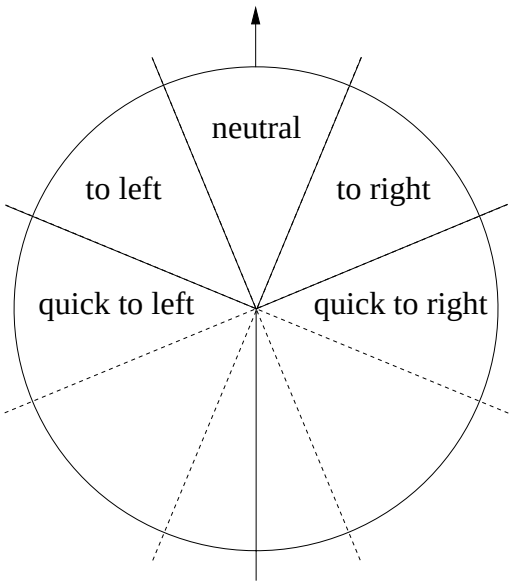

Figure 4: Qualitative velocity and orientation.

4 Evaluation

We conducted several experiments in order to compare the performances of a variety of different approaches to ball interception. Those different methods are the following.

NAI and NAI5 – the naïve methods

If the ball is approximately in front of the player, it runs to the current position of the ball. Otherwise it turns towards the ball. The notion of *approximately in front of* is realized with the help of a *threshold angle* ε. If the absolute value of the angle δ between the agent's orientation and the current ball position is less than ε, the agent just runs forward (cf. Figure 5 left). If, however, $|\delta| > \varepsilon$, the agent turns by δ, in order to face the current ball position (Figure 5 right). Please note, that this method completely ignores the speed and direction of the ball movement.

Following the considerations in Section 3.2, we chose 22.5° as value for ε in NAI. The method NAI5 differs from NAI in the value of the threshold angle ε only. In this method the value of ε is set to 5°. Choosing between turning and running is necessary because the Soccer Server treats those actions as mutually exclusive, i.e. in one simulation step a player can either run (dash) or turn.

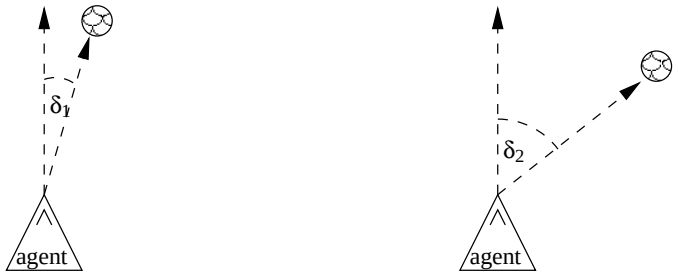

Figure 5: Naive ball interception. $\delta_1 \leq \varepsilon \rightarrow$ run. $\delta_2 \geq \varepsilon \rightarrow$turn

NWT – Newton's method

The Newton's method calculates the point, where the agent can intercept the ball directly. This method has already been described in Section 2.2.

LRN – the learned method

This is an approach that uses reinforcement learning to train an agent to intercept the ball. The actual behavior stems from the *Karlsruhe Brainstormers* RoboCup team [9] and has been introduced in Section 2.3.

Q8 and Q16 – the qualitative methods

These are two instances of the qualitative method introduced in Section 3.3. Q8 uses eight different qualitative directions, while Q16 makes use of sixteen sectors.

4.1 The Setting

We used an automated coach agent—called *evalcoach*—to run the evaluation sessions. A coach is a special type of agent that can connect to the Soccer Server. In contrast to the usual agents (the players) a coach is able to move the ball and players on the field and to act as a referee, e.g. by changing the play mode of the simulator, e.g. from *play_on* to *free_kick*. See [3] for further details. As this client is intended for evaluation or learning scenarios, it receives data about all the objects on the field in external coordinates, in contrast to the players, that only receive incomplete information in an egocentric coordinate system. In addition to that no noise is added to the data received by the coach client.

The basic setup of the evaluation is shown in Figure 6. A coach and one player are connected to the Soccer Server.

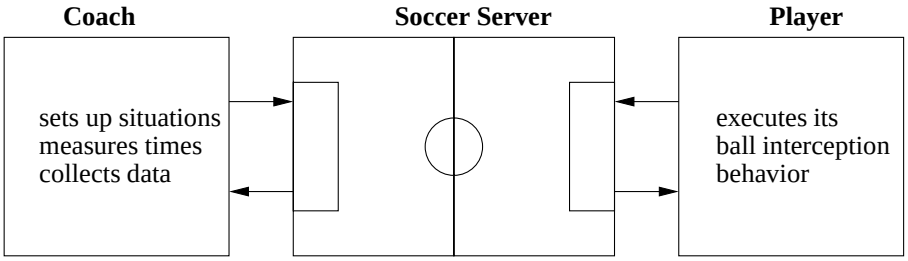

Figure 6: The evaluation setup.

The player is only executing its ball interception behavior. For each of the interception methods described above such a specialized agent exists. In order to reduce the effects of noise and randomization as far as possible, the players made use of a special type of information sent by the Soccer Server, the so called *fullstate*. The fullstate mode allows an agent to get precise information about the position of the ball even if the ball is currently not seen. Similarly the player's position can be determined. In addition to that, noise in the movement of both ball and player have been set to zero.

Each evaluation session consists of 250 different *scenarios*. Each scenario is specified by the initial position and velocity of the ball and the player, which are selected according to the following rules.

- Set the agent in the center of the field, with random orientation and speed.
- Place the ball at an arbitrary position within a $40 \times 20\,m^2$ rectangle in the middle of the field (see Figure 7). Assign a motion with random direction and speed to it.

An *episode* within such a scenario corresponds to the player trying to intercept the ball. It ends, if either the agent has successfully intercepted the ball (i.e. the ball can be kicked by the agent), or the ball goes out of bounds, or a timeout has been reached.

The latter two outcomes of an episode are counted as failures. In order to compensate for network problems and process load on the computer, we ran 100 episodes for each scenario.

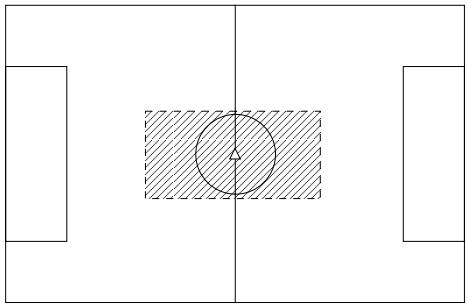

Figure 7: Initial setup of the evaluation. The agent is in the center of the field. The ball is randomly placed within the hatched rectangle.

The actual control of a session was done by the evalcoach with this algorithm:

while there is another scenario **do**
  read next scenario
  for i := 1 to 100 **do**
    setup scenario
    run episode *i*
    save duration of episode *i*
    note *Success or Failure*
  end for
  Record average and variance of duration
  Record percentage of successes
end while

4.2 Analysis of the Data

For each of the six interception methods described above an evaluation session was done. Thus, we obtained tables containing the the average interception time for each scenario, the variances of the times and the percentage of successful episodes per scenario for each method.

Impossible scenarios, i.e. scenarios in which *no* player succeeded in intercepting the ball even once, were removed from the table. After this cleaning step, 234 scenarios were left, in which at least one interception method was successful. They built the basis for comparing the different approaches.

Let us present some general figures first. The success rate for each scenario lay by either 100% or 0%. As we eliminated randomness from the evaluation sessions,

this means, that no significant disturbances due to network or process load appeared. Nevertheless, influences by the hardware can be observed, as the variances of several scenarios are larger than zero throughout all six sessions.

All interception methods succeeded in almost all scenarios, which can be seen on the left in Figure 8. On the right side of this figure the percentage of sessions that were "won" by a single method is shown. By "won" we mean, that one method was faster than every other method in the scenario. The slice labeled *rest* summarizes all scenarios that had no unique winner, i.e. where two or more methods were equally good.

Approach	No. of Successes
NAI	232
NAI5	231
LRN	233
Q8	232
Q16	232
NWT	234

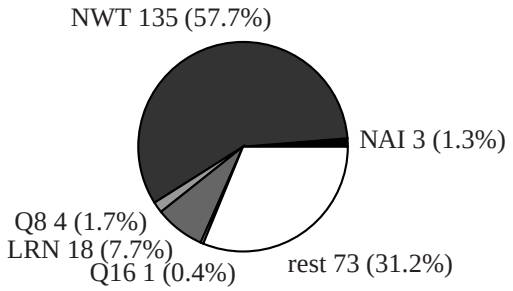

Figure 8: Left: The number of successful interceptions by each approach. Right: The number of times each method was better than *all* others. In 73 scenarios at least two methods were equally good.

Both the number of successes and the number of scenarios won indicate, that Newton's method (NWT) is slightly superior to the other approaches to ball interception. This is supported by calculating the means of the interception times, which are shown in Figure 9. The lowest average duration belongs to NWT. But all three figures show, that the learned method (LRN) is only slightly worse than NWT.

The naïve method NAI5 clearly is the worst approach to ball interception. It shows the lowest number of total successes (231), does not win a single scenario and on average takes the longest times to intercept a ball (31.5 simulation steps). Surprisingly the second naïve method (NAI) does quite well. On average it performs better than both variations of the qualitative interception method, although Q8 wins more scenarios than NAI.

In addition to the measurements described above, we took the average ratios $\overline{R}$ between interception times. The average ratio $\overline{R_{k,l}}$ of the interception times of methods k and l is given by

$$\overline{R_{k,l}} = \frac{1}{n}\sum_{i=1}^{n}\frac{d_i^k}{d_i^l}$$

where n denotes the total number of scenarios and d_i^j is the interception time of method j in scenario i. The average ratio of NWT and LRN is, for example, calculated by

$$\overline{R_{\mathrm{NWT,LRN}}} = \frac{1}{234}\sum_{i=1}^{234}\frac{d_i^{\mathrm{NWT}}}{d_i^{\mathrm{LRN}}} = 1.005.$$

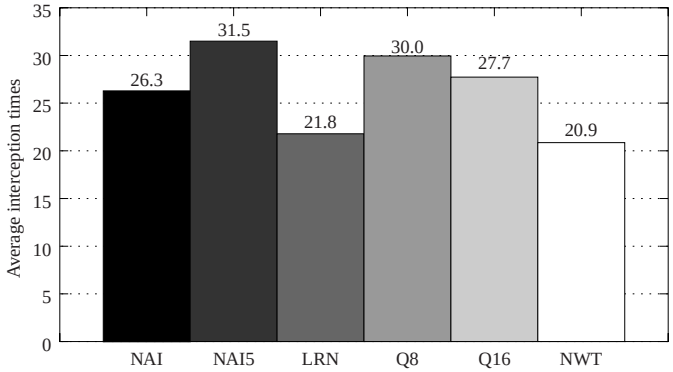

Figure 9: Average interception times of the different approaches.

Figure 10 shows the average ratios between all pairs of methods. This table once again supports the results from the measurements above, namely a slight superiority of NWT, but which is closely followed by LRN, although LRN seems to do better according to the ratio $\overline{R_{\text{NWT,LRN}}} = 1.005$. A direct comparison between the interception times of LRN and NWT shows, that NWT is faster than LRN 144 times, LRN beats NWT 21 times, and 69 times both methods produce equal interception times. This is also shown on the left in Figure 11. But if LRN is faster than NWT in a scenario, it is usually *much* faster (up to 15 simulation steps), which has a great impact on calculating the mean of the quotients.

	NAI	NAI5	LRN	Q8	Q16	NWT
NAI	—	0.832	1.436	0.981	1.014	1.413
NAI5	1.261	—	1.835	1.229	1.269	1.807
LRN	0.820	0.696	—	0.759	0.802	1.035
Q8	1.190	0.972	1.624	—	1.105	1.626
Q16	1.077	0.881	1.486	0.961	—	1.472
NWT	0.790	0.670	1.005	0.743	0.777	—

Figure 10: Average ratios between the methods

Let us now take a closer look at the methods, that we evaluated in two variations, namely NAI and NAI5 as well as Q8 and Q16. In both methods one variation proved to be clearly superior to the other.

4.2.1 NAI versus NAI5

The method NAI beats NAI5 by far. It is easy to see, that this behavior comes from the different values of the threshold angles. As this angle is much smaller in NAI5, the player has to turn often, because the ball moves out of the sector defined by the

threshold angle. For a player using NAI for interception this sector is much wider and thus the need for turning arises less frequently. If the threshold angle is not too wide, the additional way the agent has to run, does not matter very much, as we have shown in Section 3.2. Thus a player using NAI for interception can get to the ball much faster than a player using NAI5.

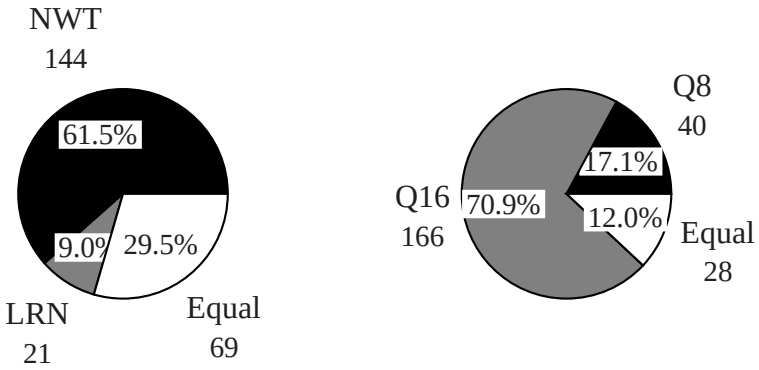

Figure 11: Direct comparisons between different interception methods. Left: LRN vs. NWT. Right: Q8 vs. Q16

4.2.2 Q8 versus Q16

The results of the evaluation show, that from the two variations Q8 and Q16 of our qualitative interception method Q16 is the faster one. Although both approaches are successful in the same scenarios, the average interception time is less for Q16. The direct comparison, shown on the right in Figure 11, shows that Q16 is faster than Q8 in 166 different scenarios, which roughly corresponds to 71% of all scenarios.

This result shows that the number of sectors, i.e. qualitative directions, has a significant influence on the performance of the qualitative ball interception. In Q16 the sectors are smaller than in Q8 and thus give the player a more finely grained control over its turns than Q8.

One phenomenon, that can repeatedly be observed, is an oscillation at the beginning of an episode. The player just keeps turning back and forth for some time before it starts to run. The agent decides to turn based on the qualitative position and velocity of the ball, but then it turns just too far. So, in the next step the player decides, that it has to turn back and now turns too far into the other direction. Clearly this effect can be reduced, if the player divides its surroundings in more sectors and thus has more directions to turn to at its disposal.

This oscillation effect is probably also the reason, why both qualitative methods come off worse than the naïve method NAI, as the agent wastes several simulation steps by turning forth and back before it takes any sensible action towards intercepting the ball.

5 Conclusions

In this paper we presented a variety of methods to intercept the ball in the virtual environment of the RoboCup simulation league. We introduced a qualitative approach to ball interception that makes use not only of qualitative directions, but of qualitative descriptions of velocity as well.

We conducted a number of experiments in order to compare two variations of this approach to several other methods, including a numerical method and a learned behavior. These experiments have shown the numerical method and the learned behavior to be the best. The differences between those two approaches are very small but show a slight advantage of the numerical method.

5.1 How Useful is Qualitative Velocity?

The qualitative interception methods did not do very well in the evaluation and turned out to be similar in performance to a naïve approach to ball interception which discards all information about the movement of the ball. But the evaluation also showed that the performance of both the naïve and the qualitative method is highly dependent on the right choice of parameter values.

One clear advantage the qualitative interception method has over the numerical and the learned methods is its robustness and portability. For the learned method to work it is necessary that the environment does not change after the learning period. Otherwise the whole behavior has to be trained again. The numerical method even depends on the complete knowledge of the physical model that describes movements in the environment.

In the simulated environment of the Soccer Server with its simple model of mechanics and the quantitative information about objects received by the clients the abstraction from quantitative data onto a qualitative level seems somehow artificial, as the agent has to abstract from quantitative information provided by the simulator, reason on the qualitative data and finally map qualities back to concrete numbers when it sends a command.

But consider a domain which is too complex to be described quantitatively or in the level of detail needed by the quantitative methods. In other (real world) domains it may be very expensive or slow to generate quantitative data from sensor inputs but relatively easy to achieve a qualitative representation of the world. As the qualitative interception method makes only very few basic assumptions about the mechanics involved—e.g. the ball motion does not change its direction without external influences—it is not hampered by complexity of the underlying physics, e.g. several kinds of friction and noise in the sensor data.

Furthermore, the representation of the ball position and movement as well as the actions taken by the player are more cognitively adequate. The representations are easy to understand and calculate. Real-time requirements are easily met because of the simplicity of the calculations.

5.2 Future Work

Future work includes further analysis of the data collected in the evaluation sessions. As no method was always fastest, but rather every method was better than the others in one scenario or another, we hope to gain more information about the strengths and weaknesses of the individual interception methods by further examining and classifying the data and the different scenarios.

Another piece of work, that will be tackled soon, is the optimization of the parameters of our qualitative interception method. As we have seen, the number of different directions the agent knows, have a significant influence on the performance of the method. In parallel to finding the optimal number of sectors for the proposed method, we will examine the influence of adding methods for reducing the oscillation effect on the overall performance of the method.

Last but not least, we plan to apply the proposed method to other domains in order to test its scalability to other applications and its feasibility outside the world of the Soccer Server. This might help us to understand better where a qualitative approach for spatial reasoning with velocity is appropriate—not only for the cognitively adequate description of physical behavior, but also for controlling agents.

Acknowledgments

We would like to thank Marion Levelink, Artur Merke, Reinhard Moratz, Martin Riedmiller and Manfred Stolzenburg for helpful comments, hints to relevant work, and helping us with the evaluation.

References

[1] Eliseo Clementini, Paolino Di Felice, and Daniel Hernández. Qualitative representation of positional information. *Artificial Intelligence*, 95(2):317–356, 1997.

[2] Johan de Kleer and Daniel G. Bobrow. Qualitative reasoning with higher-order derivatives. In *Proceedings of the American National Conference on Artificial Intelligence (AAAI-84)*, pages 86–91, 1984. Reprint in Daniel S. Weld and Johan D. Kleer (eds.), *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann, San Francisco, 1990.

[3] Ehsan Foroughi, Frederik Heintz, Spiros Kapetanakis, Kostas Kostiadis, Johann Kummeneje, Itsuki Noda, Oliver Obst, Pat Riley, Timo Steffens, and USTC9811 Group. *RoboCup Soccer Server User Manual (for Soccer Server Version 7.06 and later)*, 2001.

[4] Yumi Iwasaki. Real-world applications of qualitative reasoning. *IEEE Expert*, 12(3):16–21, 1997.

[5] Jan Murray, Oliver Obst, and Frieder Stolzenburg. Towards a logical approach for soccer agents engineering. In Peter Stone, Tucker Balch, and Gerhard Kraetzschmar, editors, *RoboCup 2000: Robot Soccer World Cup IV*, LNAI 2019, pages 199–208. Springer, Berlin, Heidelberg, New York, 2001.

[6] Alexandra Musto, Klaus Stein, Andreas Eisenkolb, Thomas Röfer, Wilfried Brauer, and Kerstin Schill. From motion observation to qualitative motion representation. In Christian Freksa, Wilfried Brauer, Christopher Habel, and Karl F. Wender, editors, *Spatial Cognition II*, LNCS 1849, pages 115–126. Springer, Berlin, Heidelberg, New York, 2000.

[7] Itsuki Noda. Soccer server: a simulator for RoboCup. In *JSAI AI-Symposium*, 1995.

[8] Martin Riedmiller. Concepts and facilities of a neural reinforcement learning control architecture for technical process control. *Journal of Neural Computing and Application*, 8:323–338, 2000.

[9] Martin Riedmiller, Artur Merke, D. Meier, A. Hoffmann, A. Sinner, O. Thate, C. Kill, and R. Ehrmann. Karlsruhe Brainstormers – a reinforcement learning way to robotic soccer. In Peter Stone, Tucker Balch, and Gerhard Kraetzschmar, editors, *RoboCup 2000: Robot Soccer World Cup IV*, LNAI 2019, pages 367–372. Springer, Berlin, Heidelberg, New York, 2001.

[10] Frieder Stolzenburg, Oliver Obst, Jan Murray, and Björn Bremer. Spatial agents implemented in a logical expressible language. In Manuela Veloso, Enrico Pagello, and Hiroaki Kitano, editors, *RoboCup-99: Robot Soccer WorldCup III*, LNAI 1856, pages 481–494. Springer, Berlin, Heidelberg, New York, 2000.

[11] Peter Stone et al. Robocup-2000: The fourth robotic soccer world championships. *AI magazine*, 22(1):11–38, 2001.

[12] Peter Stone and David McAllester. An architecture for action selection in robotic soccer. In *Fifth International Conference on Autonomous Agents*, 2001.

[13] Kai Zimmermann and Christian Freksa. Qualitative spatial reasoning using orientation, distance, and path knowledge. *Applied Intelligence*, 6:49–58, 1996.

Available Research Reports (since 1998):

2002

4/2002 *Frieder Stolzenburg, Oliver Obst, Jan Murray.* Qualitative Velocity and Ball Interception.

3/2002 *Peter Baumgartner.* A First-Order Logic Davis-Putnam-Logemann-Loveland Procedure.

2/2002 *Peter Baumgartner, Ulrich Furbach.* Automated Deduction Techniques for the Management of Personalized Documents.

1/2002 *Jürgen Ebert, Bernt Kullbach, Franz Lehner.* 3. Workshop Software Reengineering (Bad Honnef, 10./11. Mai 2001).

2001

13/2001 *Annette Pook.* Schlussbericht "FUN - Funkunterrichtsnetzwerk".

12/2001 *Toshiaki Arai, Frieder Stolzenburg.* Multiagent Systems Specification by UML Statecharts Aiming at Intelligent Manufacturing.

11/2001 *Kurt Lautenbach.* Reproducibility of the Empty Marking.

10/2001 *Jan Murray.* Specifying Agents with UML in Robotic Soccer.

9/2001 *Andreas Winter.* Exchanging Graphs with GXL.

8/2001 *Marianne Valerius, Anna Simon.* Slicing Book Technology — eine neue Technik für eine neue Lehre?.

7/2001 *Bernt Kullbach, Volker Riediger.* Folding: An Approach to Enable Program Understanding of Preprocessed Languages.

6/2001 *Frieder Stolzenburg.* From the Specification of Multiagent Systems by Statecharts to their Formal Analysis by Model Checking.

5/2001 *Oliver Obst.* Specifying Rational Agents with Statecharts and Utility Functions.

4/2001 *Torsten Gipp, Jürgen Ebert.* Conceptual Modelling and Web Site Generation using Graph Technology.

3/2001 *Carlos I. Chesñevar, Jürgen Dix, Frieder Stolzenburg, Guillermo R. Simari.* Relating Defeasible and Normal Logic Programming through Transformation Properties.

2/2001 *Carola Lange, Harry M. Sneed, Andreas Winter.* Applying GUPRO to GEOS – A Case Study.

1/2001 *Pascal von Hutten, Stephan Philippi.* Modelling a concurrent ray-tracing algorithm using object-oriented Petri-Nets.

2000

8/2000 *Jürgen Ebert, Bernt Kullbach, Franz Lehner (Hrsg.).* 2. Workshop Software Reengineering (Bad Honnef, 11./12. Mai 2000).

7/2000 *Stephan Philippi.* AWPN 2000 - 7. Workshop Algorithmen und Werkzeuge für Petrinetze, Koblenz, 02.-03. Oktober 2000 .

6/2000 *Jan Murray, Oliver Obst, Frieder Stolzenburg.* Towards a Logical Approach for Soccer Agents Engineering.

5/2000 *Peter Baumgartner, Hantao Zhang (Eds.).* FTP 2000 – Third International Workshop on First-Order Theorem Proving, St Andrews, Scotland, July 2000.

4/2000 *Frieder Stolzenburg, Alejandro J. García, Carlos I. Chesñevar, Guillermo R. Simari.* Introducing Generalized Specificity in Logic Programming.

3/2000 *Ingar Uhe, Manfred Rosendahl.* Specification of Symbols and Implementation of Their Constraints in JKogge.

2/2000 *Peter Baumgartner, Fabio Massacci.* The Taming of the (X)OR.

1/2000 *Richard C. Holt, Andreas Winter, Andy Schürr.* GXL: Towards a Standard Exchange Format.

1999

10/99 *Jürgen Ebert, Luuk Groenewegen, Roger Süttenbach.* A Formalization of SOCCA.

9/99 *Hassan Diab, Ulrich Furbach, Hassan Tabbara.* On the Use of Fuzzy Techniques in Cache Memory Managament.

8/99 *Jens Woch, Friedbert Widmann.* Implementation of a Schema-TAG-Parser.

7/99 *Jürgen Ebert, and Bernt Kullbach, Franz Lehner (Hrsg.).* Workshop Software-Reengineering (Bad Honnef, 27./28. Mai 1999).

6/99 *Peter Baumgartner, Michael Kühn.* Abductive Coreference by Model Construction.

5/99 *Jürgen Ebert, Bernt Kullbach, Andreas Winter.* GraX – An Interchange Format for Reengineering Tools.

4/99 *Frieder Stolzenburg, Oliver Obst, Jan Murray, Björn Bremer.* Spatial Agents Implemented in a Logical Expressible Language.

3/99 *Kurt Lautenbach, Carlo Simon.* Erweiterte Zeitstempelnetze zur Modellierung hybrider Systeme.

2/99 *Frieder Stolzenburg.* Loop-Detection in Hyper-Tableaux by Powerful Model Generation.

1/99 *Peter Baumgartner, J.D. Horton, Bruce Spencer.* Merge Path Improvements for Minimal Model Hyper Tableaux.

1998

24/98 *Jürgen Ebert, Roger Süttenbach, Ingar Uhe.* Meta-CASE Worldwide.

23/98 *Peter Baumgartner, Norbert Eisinger, Ulrich Furbach.* A Confluent Connection Calculus.

22/98 *Bernt Kullbach, Andreas Winter.* Querying as an Enabling Technology in Software Reengineering.

21/98 *Jürgen Dix, V.S. Subrahmanian, George Pick.* Meta-Agent Programs.

20/98 *Jürgen Dix, Ulrich Furbach, Ilkka Niemelä .* Nonmonotonic Reasoning: Towards Efficient Calculi and Implementations.

19/98 *Jürgen Dix, Steffen Hölldobler.* Inference Mechanisms in Knowledge-Based Systems: Theory and Applications (Proceedings of WS at KI '98).

18/98 *Jose Arrazola, Jürgen Dix, Mauricio Osorio, Claudia Zepeda.* Well-behaved semantics for Logic Programming.

17/98 *Stefan Brass, Jürgen Dix, Teodor C. Przymusinski.* Super Logic Programs.

16/98 *Jürgen Dix.* The Logic Programming Paradigm.

15/98 *Stefan Brass, Jürgen Dix, Burkhard Freitag, Ulrich Zukowski.* Transformation-Based Bottom-Up Computation of the Well-Founded Model.

14/98 *Manfred Kamp.* GReQL – Eine Anfragesprache für das GUPRO–Repository – Sprachbeschreibung (Version 1.2).

12/98 *Peter Dahm, Jürgen Ebert, Angelika Franzke, Manfred Kamp, Andreas Winter.* TGraphen und EER-Schemata – formale Grundlagen.

11/98 *Peter Dahm, Friedbert Widmann.* Das Graphenlabor.

10/98 *Jörg Jooss, Thomas Marx.* Workflow Modeling according to WfMC.

9/98 *Dieter Zöbel.* Schedulability criteria for age constraint processes in hard real-time systems.

8/98 *Wenjin Lu, Ulrich Furbach.* Disjunctive logic program = Horn Program + Control program.

7/98 *Andreas Schmid.* Solution for the counting to infinity problem of distance vector routing.

6/98 *Ulrich Furbach, Michael Kühn, Frieder Stolzenburg.* Model-Guided Proof Debugging.

5/98 *Peter Baumgartner, Dorothea Schäfer.* Model Elimination with Simplification and its Application to Software Verification.

4/98 *Bernt Kullbach, Andreas Winter, Peter Dahm, Jürgen Ebert.* Program Comprehension in Multi-Language Systems.

3/98 *Jürgen Dix, Jorge Lobo.* Logic Programming and Nonmonotonic Reasoning.

2/98 *Hans-Michael Hanisch, Kurt Lautenbach, Carlo Simon, Jan Thieme.* Zeitstempelnetze in technischen Anwendungen.

1/98 *Manfred Kamp.* Managing a Multi-File, Multi-Language Software Repository for Program Comprehension Tools — A Generic Approach.