The 6th Workshop on Humanoid Soccer Robots @ Humanoids Conference, Bled (Slovenia), October 2011.

Accurate Ball Tracking with Extended Kalman Filters as a Prerequisite for a High-level Behavior with Reinforcement Learning

Andreas Seekircher, Saminda Abeyruwan, Ubbo Visser
Department of Computer Science
University of Miami
1365 Memorial Drive, Coral Gables, FL, 33146 USA
{*aseek,saminda,visser*}*@cs.miami.edu*

***Abstract*—Controlling autonomous, humanoid robots in a dynamic, continuous, and real-time environment is a complex task. We have used an Extended Kalman Filter to track the position and velocity of the soccer ball in the RoboCup 3D soccer simulation scenario. The influence of reducing the error in the ball estimate on a high-level behavior is then demonstrated using the *keep-away* behavior. We have applied Sarsa(λ) with tile-coding as a linear function approximator. The keep-away task has been successfully learned in the 2D Soccer Simulation League a few years ago; in this paper, we apply similar ideas on a humanoid robot and describe new problems that arise with both the different robot model and the environment. The learned behavior depends highly on the underlying skills, which is shown using different ball localizations. The results are promising, yet reveal a new level of complexity.**

***Keywords*-Extended Kalman Filter, Reinforcement Learning, Sarsa(λ)**

I. Introduction

Controlling an autonomous, humanoid robot in a dynamic, continuous environment is a difficult task. Manually programming complex behaviors can be very time consuming and tedious, since the decisions made by the agents depend on many features and constraints imposed by the environment. Reinforcement learning has been successfully applied to learn complex multi-agent behaviors. The keep-away soccer behavior is an example where reinforcement learning outperforms hand-coded algorithms in the RoboCup 2D Simulation League [1]. In the RoboCup 3D Simulation League[1], problems arise when learning similar behaviors; for instance, agents must control humanoid robots that are constrained by physics.

In 3D simulation, behavior learning is tightly coupled with the state of the soccer ball, i.e., position and velocity, and the positioning of the players. In this paper, we first investigate the potential of ball tracking with an Extended Kalman Filter (EKF) [2]. We then show that the more ball tracking improves, the better the keep-away subtask (which is learned with the combination of Sarsa(λ) with CMAC [3]) will perform.

[1]http://simspark.sourceforge.net/

The subsequent sections of the paper are structured as follows. We first discuss the related work in section I-A. In section II we briefly discuss the RoboCup 3D Simulation League. In section III we present the soccer ball tracking with EKF. In section IV we summarize the learning algorithm and present the keep-away soccer scenario. Finally, we conclude the paper in section V and give suggestions for future work.

A. Related work

Extended Kalman Filters have been extensively used in many applications where non-linear dynamics are prevalent. There are many instances where EKFs have been used in different RoboCup leagues, e.g., robot self-localization as well as for ball tracking [4]–[6]. In our research, we are focusing on the effectiveness of better ball tracking in the 3D simulation league to improve the extended subtasks such as keep-away scenarios. Similar to EKFs, parametric function approximators such as tile coding have been successfully applied in real world control problems [7]–[11], especially when state and action spaces are continuous variables [3], [12]. Reinforcement Learning (RL) has also been applied to more complex multi-agent behaviors in the RoboCup domain. Stone et. al. [1] and Kalyanakrishnan & Stone [13] have used an episodic SMDP Sarsa(λ) with linear tile coding function approximation to learn a keep-away soccer behavior. That means a group of robots, the "keepers", try to keep the control of the ball despite the efforts of the other group, the "takers". They used RL to choose one of the lower level skills, such as passing to a teammate or holding the ball. This behavior was learned in the RoboCup 2D Simulation League. Skills in the 2D environment, such as passing the ball to another robot, are relatively reliable, especially when compared with the 3D league.

An important subtask of the keep-away behavior is the positioning of the keepers that are not in possession of the ball. In [13], this positioning is optimized using the cross-entropy method [14]. Learning this subtask simultaneously with the keep-away behavior can yield tightly-coupled multi-agent behaviors. So far, learning complex behaviors similar to keep-away has been done mostly in the 2D soccer simulation league. Applying the same methods in an environment using a humanoid robot model constrained by physics results in new and different problems.

II. 3D Simulation League

The RoboCup 3D Simulation League is based on the general purpose multi-agent simulator SimSpark[2]. The robot agents in the simulation are modeled based on the Aldebaran Nao[3] robots. Each robot has 22 degrees of freedom. The agents communicate with the server through message passing and each agent is equipped with noise free joint perceptors and effectors. In addition to this, each agent has a noisy restricted vision cone of 120^o. Every simulation cycle is limited to 20 ms, where agents perceive noise free angular measurements of each joint and the agents stimulate the necessary joints by sending torques to the simulation server. The vision information from the server is available every third cycle (60 ms), which provides the spherical coordinates of the perceived objects. The agents also have the option of communicating with each other every other simulation cycle (40 ms) by broadcasting a 20 $bytes$ message. Currently the simulation league competitions are conducted with 9 robots on each side (18 total).

III. Ball tracking with EKF

We have used the following modeling criterion to capture the non-linear dynamics of the ball: first, we have conducted two rotations and a translation to move the perceived vision information to a fixed coordinate frame relative to the robot's torso; second, we have developed EKF models for x and y axis separately; finally, we have used odometry information to capture the relative translation and rotations of this fixed coordinate frame. The ball state is given by:

$$\begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix} = \begin{bmatrix} x_{t-1} + \dot{x}_{t-1}\Delta t \\ \dot{x}_{t-1} \end{bmatrix} + \epsilon_t,$$

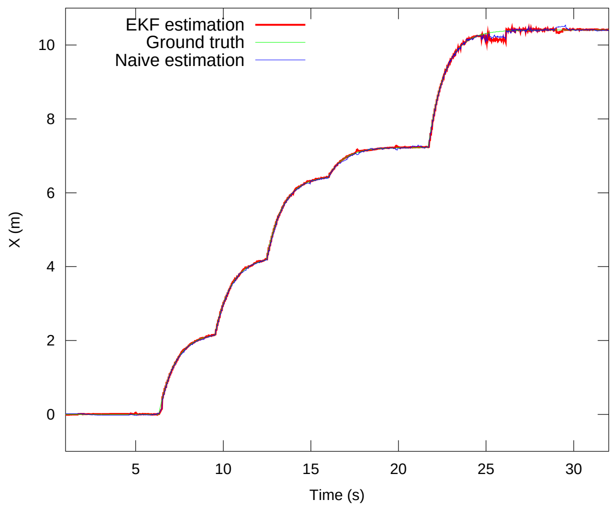
where t is the index of the sampling interval, $x_.$ is the position, $\dot{x}_.$ is the velocity, Δt is the time step size and $\epsilon_.$ is the process noise which is normally distributed with $\mathcal{N}(0, R_t)$. The measurement model is given by:

$$z_t = \begin{bmatrix} 1 & 0 \end{bmatrix} x_t + \delta_t,$$

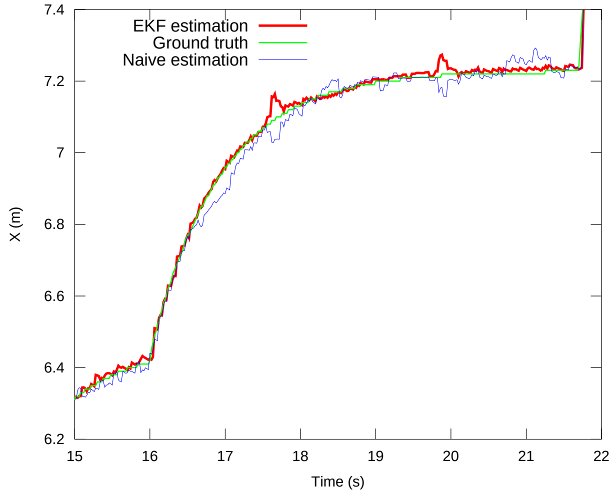
where $z_.$ is the measurement and $\delta_.$ is the measurement noise which is normally distributed with $\mathcal{N}(0, Q_t)$. We have used the following prediction model:

$$\begin{bmatrix} x_t \\ y_t \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} cos(-\Delta\theta_t) & -sin(-\Delta\theta_t) & 0 & -\Delta x_t \\ sin(-\Delta\theta_t) & cos(-\Delta\theta_t) & 0 & -\Delta y_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ 0 \\ 0 \end{bmatrix},$$

[2]http://simspark.sourceforge.net/
[3]http://www.aldebaran-robotics.com/eng/



(a) Displacement of the soccer ball along the x axis.

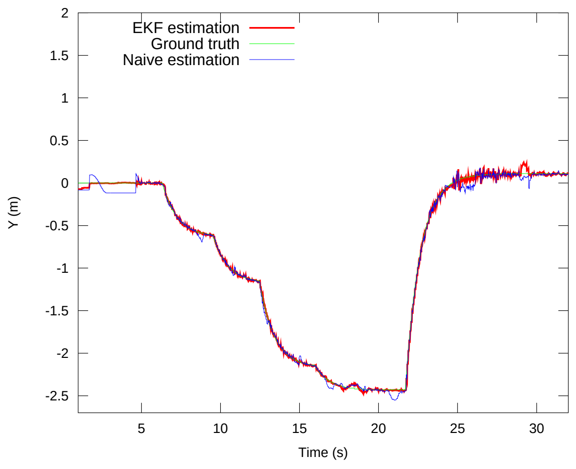


(b) Zoomed part of a) graph

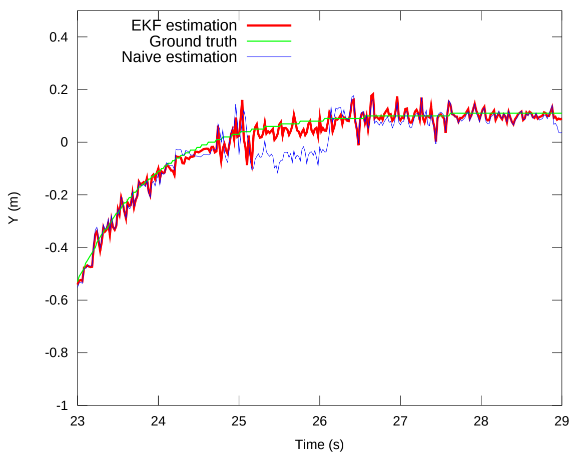
Figure 1: (a) Absolute displacement of the soccer ball in the global x axis until a goal is scored. (b) Zoomed part of the graph from $15s$-$22s$.

where $\Delta x_t, \Delta y_t$ is the odometry translation and $\Delta\theta_t$ is the odometry rotation. Since we are interested in state variables in the plane surface, the height is ignored from the prediction equations.

The robot receives vision information every third cycle. Therefore, the update cycle of EKF is performed approximately every third cycle. There are situations where the robot would not see the soccer ball for a considerable amount of time. We have used the communication network to broadcast the current ball state of each robot. If a robot does not perceive the ball percept for more that 3000 ms, we use the ball state in the communication message as the new percept. We have used the following initial error covariance matrices, P_0^x, P_0^y, process noise covariance matrices, R_0^x, R_0^y



(a) Displacement of soccer ball along the y axis



(b) Zoomed part of the a) graph

Figure 2: (a) Absolute displacement of the soccer ball in the global y axis until a goal is scored. (b) Zoomed part of the graph from $23s$-$29s$.

and measurement noise covariance matrices, Q_0^x, Q_0^y.

$$P_0^x = P_0^y = \begin{bmatrix} 0.0001 & 0.0001 \\ 0.0001 & 0.0001 \end{bmatrix}, Q_0^x = Q_0^y = 0.001,$$

$$R_0^x = R_0^y = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}$$

We have conducted the experiments based on the default soccer behavior encoded into the agent. Figure 1 shows the absolute displacement of the soccer ball in the global x axis until a goal is scored, and Figure 2 shows the absolute displacement in the y axis with respect to ground truth. We have also implemented a naive tracker, which uses the most recent percept as the estimation. Table I shows the total EKF error relative to ground truth, total naive error relative to ground truth and the EKF improvement over the naive method in percentage for each axes separately for 15 test runs. From these observations, our conclusion is that the EKF provides a better ball localization for the RoboCup 3D Simulation League.

IV. Learning a high-level behavior

In this section we describe the learning of a high-level behavior and show the influence of the improved ball localization on the experimental results.

Table I: The total EKF error relative to ground truth, total naive error relative to ground truth and the EKF improvement over the naive method in percentage for each axes separately for 15 test runs.

Axis	Naive error (m)	EKF error (m)	Improvement/Naive (%)
x	0.038	0.029	23.6
y	0.081	0.062	23.4

A. *Reinforcement Learning*

The EKF ball localization described in III has a direct influence on the robots soccer skills. Thus, high-level behaviors can also be improved. We demonstrate this by learning a keep-away soccer behavior with the naive and the EKF localization of the ball.

We have done the experiments with the Sarsa(λ) algorithm. We use Sarsa(λ) because of its linear time computation efficiency and it is an on-policy learning method where agents act using the current policy and update this policy simultaneously by updating the values $Q(s, a)$. These Q-values are the expected rewards for executing an action a in state s. We have used an ϵ-greedy action selection and replacing eligibility traces [15].

If the states and the actions of the problem are finite, the representation of the value function or the state-value function is classically formulated in tabular form [16]. When working with features in continuous spaces the representation of these features is usually covered by function approximation. There are many function approximators available in literature with different characteristics (cf. [15], [16] for more information). In our work, we have used a linear parametric function approximator based on tile coding. We have considered tile coding because of its efficiency in real-time operations and linear run-time behavior.

B. *Keep-away*

In the keep-away scenario some robots (the keepers) have to pass the ball to each other in a way that the opponents (the takers) can not get control over the ball. The keepers lose if a taker is too close to the ball or the ball leaves a given area.

The keep-away behavior has already been learned successfully by Stone et. al. [1] in the RoboCup 2D Simulation

League. In that environment, the robots have relatively fast and reliable skills, such as passing to a teammate. The ball can be manipulated when it is inside a kickable area around the robot. That means there is not much time needed for an exact positioning to be able to kick the ball. In contrast to that environment, the simulated humanoid robot in the 3D Soccer Simulation League has limited walking and kicking capabilities. It is bound by physics and has to get into the right position before a kick is possible. This makes the learning more complex.

The keep-away behavior is based on the following skills (we have used the same skills as proposed by [1] for better comparison): `GOTO-BALL()` (walk straight to the ball), `HOLD-BALL()` (stay close to the ball and try to be between ball and a taker), `PASS-BALL(`k`)` (pass ball to keeper k) and `GET-OPEN()` (positioning using SPAR [17]). These skills are hand-coded behaviors. The pass, for example, has a high degree of uncertainty. There can be an error in the direction or distance of a pass if the ball is not hit correctly. Although the skills are not optimal, the experiments will show that it is possible to learn a behavior tuned to these skills, which is better than a simple hand-coded behavior.

1) Keeper Behavior: In every time step one of the keepers is in possession of the ball or has to go to the ball. When a keeper is in possession of the ball, he has to choose one of the available actions, such as holding the ball or passing to another keeper.

This selection is done by reinforcement learning. Since the duration of the actions can vary, the problem has to be represented as a semi-Markov decision process (SMDP). The action `HOLD-BALL()` is always executed for a constant time. However, the duration of `PASS-BALL(`k`)` is the time the robot needs to kick the ball. The reward for a state transition is given by the time since the last action was selected. An episode ends when the keepers lose the possession of the ball. The reward for a sequence of actions is the sum of the time for each action. This means the expected reward is always the estimated remaining time of the episode. Thus, the keepers are maximizing the time the ball stays in their possession.

Each keeper learns the keep-away behavior individually. However, at every timestep only one keeper is learning. The other keepers are using the `GET-OPEN()` skill to get into a good position for receiving a pass.

2) Taker Behavior: The taker tries to intercept the ball by always heading directly to the ball. A second taker could be used to block passes. However, we used only one taker in the experiments, since more or better takers would likely require enhanced low-level skills.

3) State Representation: In the following, the keeper in possession of the ball is the first keeper K_1. The other keepers are sorted using the distance to K_1. The action `PASS-BALL(2)` is, for instance, a pass to the closest teammate.

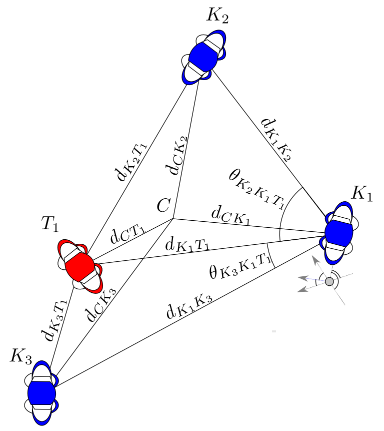


Figure 3: The keep-away state variables that have been used in [1]. The drawings near the ball on the left side of K_1 illustrate some additional state variables. These are shown in detail in Figure 4.

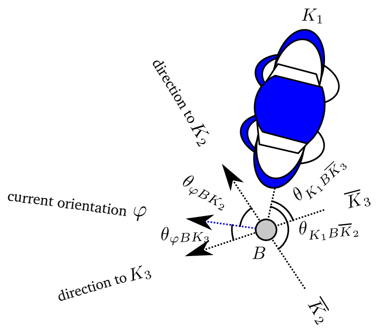


Figure 4: Additional state variables that are important for the positioning of K_1.

In the case of 3 vs. 1, each state is represented by 15 continuous state variables. In the following, d_{AB} is the distance between the points A and B and θ_{ABC} is the angle between $\overrightarrow{BA}$ and $\overrightarrow{BC}$. The first 11 state variables are the same variables as used in [1] (see Figure 3):

- Distance of each robot to the center of the area: d_{CK_1},d_{CK_2},d_{CK_3},d_{CT_1}
- Distance of each robot to K_1: $d_{K_1K_2}$,$d_{K_1K_3}$,$d_{K_1T_1}$
- Minimum distance of K_2 and K_3 to a taker, here always T_1: $d_{T_1K_2}$,$d_{T_1K_3}$
- Minimum angle between pass and the direction to a taker: $\theta_{K_2K_1T1}$, $\theta_{K_3K_1T1}$

Since the humanoid robot has to position itself correctly for the kick, we added two state variables for each possible pass target. These two variables are the angle errors in the position and orientation of K_1 for passing to another keeper (Figure 4). For 3 vs. 1 there are four additional state variables, that are using the ball position B and the current orientation φ of K_1:

- Angle K_1 has to walk around the ball to be able to pass:$\theta_{K_1B\overline{K}_2}$, $\theta_{K_1B\overline{K}_3}$

- Angle K_1 has to turn before a pass is possible: $\theta_{\varphi BK_2}$, $\theta_{\varphi BK_3}$

The position of K_1 relative to the ball can make a large difference in the time that is needed to finish the `PASS-BALL(`k`)` action. If K_1 is close to the ball, it can still take several seconds to position the robot correctly. On the other hand, with the right position it can pass almost immediately. Without these angles as state variables, the different situations are mapped to the same state or are at least very close in the state space such that the RL might not converge properly.

We chose the parameters of the keep-away problem appropriate to the capabilities of the robots, such as the walking speed and the maximum distance for passes. All experiments were done on a $6m$ by $6m$ area in the center of the field. Instead of playing 3 vs. 2 we used only one taker, because compared to the walking speed the pass speed and distance are relatively weak.

The time step of the learning algorithm depends on the actions, since it is a SMDP. The action `HOLD-BALL()` is set to 200 ms. The `PASS-BALL(`k`)` action is finished when K_1 has kicked. For the Sarsa(λ) we used the learning rate $\alpha = 0.125$ and the probability for random exploration $\epsilon = 0.01$. λ is set to 0.

The Q values for the RL are stored using tile coding. Each variable is encoded independently from each other on a set of 32 tiles, so a state is mapped on a large binary feature vector $\mathcal{F}_a$ with $15 * 32 = 480$ ones and a large amount of zeros. The value of $Q(s, a)$ is then calculated by $\mathcal{F}_a * \theta_a$.

For an evaluation of the results of the RL we implemented the following hand-coded behaviors:

- **random** selects a random action in each time step.
- **always-hold** only executes `HOLD-BALL()`, never passes.
- **hand-coded** always chooses the pass with the larger angle to T_1.

Table II shows the average episode lengths produced by the hand-coded behaviors. These values already show the difference in the quality of the soccer skills caused by the different ball localizations.

The fact that always executing the action `HOLD-BALL()` yields quite good rewards compared to the hand-coded passing behavior shows that the pass skill requires some improvements. The uncertainties in the pass skill will slow down the learning of the keepers, since the decisions learned by the RL have a smaller influence on the results when the action `PASS-BALL(`k`)` is unreliable.

Figure 5a shows the RL results for the naive ball localization. It starts on a similar level as the random behavior and increases slowly until it yields longer average episode durations as the simple hand-coded behavior. A behavior learned using Sarsa(λ) can easily outperform the simple hand-coded passing behavior. However the learned behavior is not better than only using the `HOLD-BALL()` action, since the pass skill is not reliable and fast enough.

Table II: The average episode lengths (in seconds) using different hand-coded behaviors and different ball localizations.

Behavior	Naive localization	EKF	improvement
random	9.26	11.66	25.9%
always-hold	18.06	26.31	45.7%
hand-coded	9.34	12.25	31.1%

As shown before in table II, the improved ball localization using an EKF has a high influence on high-level behaviors. Figure 5b shows the results of the same learning scenario using the EKF ball localization. Again, the learned behavior produces longer episodes than the hand-coded passing. Nevertheless, it still can not reach the results of the always-hold behavior. The RL can learn that the `HOLD-BALL()` action creates the highest expected rewards, but the random exploration sometimes chooses another action. Thus, the robot takes a higher risk by passing instead of only trying to hold the ball.

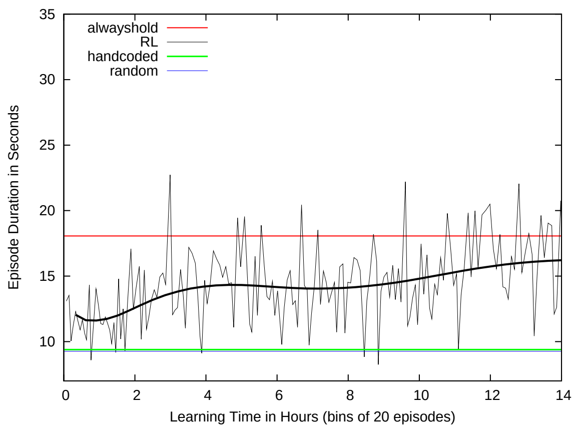
The more important fact is that all behaviors benefit from the improved ball localization. The passing skills used here are still not sufficient for a keep-away behavior that could be used in a soccer game in this environment. However, the experiments show a clear improvement in the high-level behaviors by reducing some errors in the world model. The same improvements can be done, for instance, for the localization of the opponent, which will bring us again closer to a usable keep-away behavior.

The experiments also show that further improvements of the pass skill are necessary, since the best strategy is still using only the `HOLD-BALL()` action. The risk of losing the ball when trying to pass is too high. In addition to comparing the error values in the localization, a high-level behavior such as keep-away can be used to evaluate the actual benefit from changes in the perception, modeling or low-level skills.

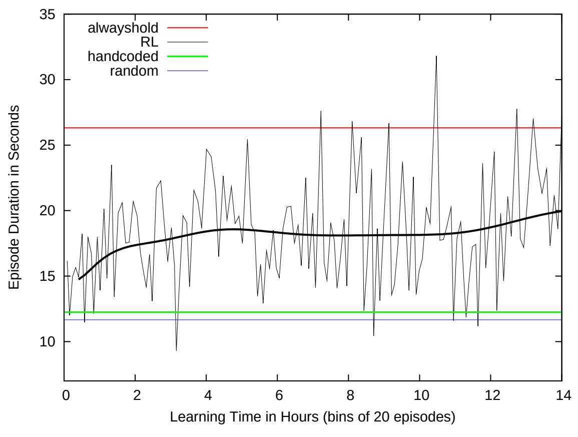
V. Conclusions and Future Work

Our experiments show that reinforcement learning can be used to provide a superior method of creating keep-away behaviors for agents in the RoboCup 3D Simulation League. However, the physical constraints imposed by the 3D environment and less robust robot skills adds additional complexity to a learning-based approach.

Robust modeling and reliable and fast skills are prerequisites for the keep-away behavior. The results show clearly that our agent needs some improvement, before a good keep-away behavior can be learned. However, we believe our results indicate strong potential for learned behaviors when the required skills are present. We presented ball localization using an EKF, that reduced the error in the estimated ball position by over 23%. This improvement in the modeling directly produced 20% to 30% longer episode lengths for the



(a)



(b)

Figure 5: Results of the RL keep-away compared to some constant behaviors. (a) Results using the naive ball localization. (b) These are the results for the ball localization using the Extended Kalman Filter. In both cases the RL keep-away learns a better policy than the simple hand-coded behavior. All behaviors are improved by the better ball localization.

passing behaviors (hand-coded and RL) and even an increase of more than 40% for holding the ball.

The results so far are promising since more improvements on low-level modules can be done on skills such as opponent localization or positioning for passes.

Eventually, our goal is to use keep-away during a soccer game by detecting the learned situations (e.g. 3 vs. 1 in the 6m by 6m area) and moving the region forward towards the opponent goal. In future, we are planning to compare the efficiency of Sarsa(λ) with other RL methods, e.g. *Greedy-GQ* [18].

References

[1] P. Stone, R. S. Sutton, and G. Kuhlmann, "Reinforcement learning for RoboCup-soccer keepaway," *Adaptive Behavior*, vol. 13, no. 3, pp. 165–188, 2005.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[3] R. S. Sutton, "Generalization in reinforcement learning: Successful examples using sparse coarse coding," in *Advances in Neural Information Processing Systems 8*. MIT Press, 1996, pp. 1038–1044.

[4] X. Li and A. Zell, "RoboCup 2006: Robot Soccer World Cup x," G. Lakemeyer, E. Sklar, D. G. Sorrenti, and T. Takahashi, Eds. Berlin, Heidelberg: Springer-Verlag, 2007, ch. Filtering for a Mobile Robot Tracking a Free Rolling Ball, pp. 296–303.

[5] K. Han and M. Veloso, "Physical model based multi-objects tracking and prediction in robosoccer," in *Working notes of the AAAI 1997 Fall Symposium on Model-directed Autonomous Systems*, MIT, Boston, November 1997.

[6] R. A. Lastra, P. A. Vallejos, and J. R. del Solar, "Integrated self-localization and ball tracking in the four-legged robot soccer league," in *1st IEEE Latin American Robotics Symposium*. IEEE, 2004, pp. 54–59.

[7] C. C. Ward and K. Iagnemma, "A dynamic-model-based wheel slip detector for mobile robots on outdoor terrain." *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 821–831, 2008.

[8] S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended kalman filter based slam." *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 1036–1049, 2007.

[9] J. Kim, Y. T. Kim, and S. Kim, "An accurate localization for mobile robot using extended kalman filter and sensor fusion." in *IJCNN*. IEEE, 2008, pp. 2928–2933.

[10] R. Jassemi-Zargani and D. S. Necsulescu, "Extended kalman filter-based sensor fusion for operational space control of a robot arm." *IEEE T. Instrumentation and Measurement*, vol. 51, no. 6, pp. 1279–1282, 2002.

[11] R. Huang, S. C. Patwardhan, and L. T. Biegler, "Robust extended kalman filter based nonlinear model predictive control formulation." in *CDC*. IEEE, 2009, pp. 8046–8051.

[12] C. Gaskett, D. Wettergreen, and A. Zelinsky, "Q-learning in continuous state and action spaces," in *Australian Joint Conference on AI*. Springer-Verlag, 1999, pp. 417–428.

[13] S. Kalyanakrishnan and P. Stone, "Learning complementary multiagent behaviors: A case study," in *Proceedings of the RoboCup International Symposium 2009*. Springer Verlag, 2009.

[14] P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method." *Annals OR*, vol. 134, no. 1, pp. 19–67, 2005.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[16] L. Buşoniu, R. Babuška, B. De Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*. Boca Raton, Florida: CRC Press, 2010.

[17] M. Veloso, P. Stone, and M. Bowling, "Anticipation as a key for collaboration in a team of agents: A case study in robotic soccer," in *In Proceedings of SPIE Sensor Fusion and Decentralized Control in Robotic Systems II*, vol. 3839, 1999.

[18] H. R. Maei, C. Szepesvári, S. Bhatnagar, and R. S. Sutton, "Toward off-policy learning control with function approximation," in *ICML*, 2010, pp. 719–726.