Learning Reliable and Efficient Navigation with a Humanoid

Stefan Oßwald Armin Hornung Maren Bennewitz

***Abstract*— Reliable and efficient navigation with a humanoid robot is a difficult task. First, the motion commands are executed rather inaccurately due to backlash in the joints or foot slippage. Second, the observations are typically highly affected by noise due to the shaking behavior of the robot. Thus, the localization performance is typically reduced while the robot moves and the uncertainty about its pose increases. As a result, the reliable and efficient execution of a navigation task cannot be ensured anymore since the robot's pose estimate might not correspond to the true location. In this paper, we present a reinforcement learning approach to select appropriate navigation actions for a humanoid robot equipped with a camera for localization. The robot learns to reach the destination reliably and as fast as possible, thereby choosing actions to account for motion drift and trading off velocity in terms of fast walking movements against accuracy in localization. We present extensive simulated and practical experiments with a humanoid robot and demonstrate that our learned policy significantly outperforms a hand-optimized navigation strategy.**

I. Introduction

Completing navigation tasks reliably and efficiently is one of the most essential objectives for an autonomous robot. While this problem is mainly solved for wheeled robots, it is still a challenging problem for humanoid robots. The reasons are, first, that the motion commands are executed rather unreliably due to backlash in the joints and foot slippage on the ground. Second, there is considerable noise in the observations according to the shaking movements of the humanoid. As a result, the estimate about the pose of the robot typically gets highly uncertain while navigating in the environment. However, as a precondition for finding the way to a designated navigation goal, the robot needs accurate knowledge about its pose.

In the case of small humanoid robots, which have only a limited payload, compact and lightweight cameras are often used as sensor for localization. The problem which arises when applying visual localization is that the movements of the humanoid robot typically introduce motion blur in the acquired images. As an example, Fig. 1 depicts an image acquired with a small, walking humanoid robot equipped with a camera observing the floor in front of the robot. As can be seen, the image is highly affected by motion blur, typically resulting in a decreased localization accuracy because of non-detection of features and matching failures. Thus, the robot may be prevented from executing the navigation task successfully due to a wrong pose estimate.

In this paper, we present an approach for reliable and efficient vision-based navigation with a humanoid. We use reinforcement learning (RL) to learn which navigation actions to chose in order to reach the destination as fast as possible and with a sufficient accuracy. In particular, we apply Sarsa(λ) RL [2] to determine the optimal action for each state, i.e., the humanoid decides in each step whether it should stop to integrate new observations extracted from sharp images, walk towards the goal, or rotate on the spot to re-align with the goal or to observe more features. We use an unscented Kalman filter (UKF) to track the pose of the robot and include the uncertainty of the filter as one component in the state representation of the Markov decision process (MDP) modeling the navigation task.

Fig. 1. The *Nao* robot [1] walking in our corridor environment (left). The robot observes floor patches from which it extracts features for localization. While walking, significant motion blur is introduced in the captured image (right), resulting in non-detection of features.

According to the learned policy, the robot automatically chooses actions so as to account for motion drift and for trading off velocity, i.e., fast walking movements against accuracy in localization. In this way, an efficient and reliable navigation strategy is generated. We provide experiments evaluating the learned policy in both simulated and real-world experiments with a humanoid. The experimental results show that our learned navigation behavior significantly outperforms a hand-optimized navigation strategy.

Several techniques for humanoid robots have been developed that aim at optimizing properties such as speed [3], [4], [5] or torso stability [6], [7] during walking. The corresponding authors were mainly interested in the resulting gait and did not consider the effects on the execution of motion commands, on the quality of the pose estimate of the robot, and on the resulting time to finish a navigation task. In contrast to all these techniques, our goal is to choose navigation actions so that the robot reaches the destination fast and reliably.

In this work, we extend the learning framework presented by Hornung *et al.* [8]. This includes, e.g., defining appropriate navigation actions for the humanoid. Our system directly learns which actions to choose so as to reach the goal reliably

All authors are with the Dept. of Computer Science, University of Freiburg, Germany. This work has partly been supported by the German Research Foundation (DFG) under contract number SFB/TR-8.

and fast. We do not require an additional controller steering the robot towards the goal, which was applied to the wheeled robot used in [8].

The remainder of this paper is structured as follows. We first discuss related work in the next section. Sec. III shortly describes the humanoid with which we performed our experiments. In Sec. IV, we present our vision-based localization system. The navigation task and our learning approach are detailed in Sec. V. Finally, experimental results are provided and discussed in Sec. VI.

II. Related Work

In the past, various frameworks were presented which employ active methods in the context of localization and navigation with wheeled robots or flying vehicles equipped with laser range scanners. Kollar and Roy [9] use reinforcement learning to optimize the robot's trajectory, i.e., they learn the translation and rotation behavior which minimizes the uncertainty in laser-based SLAM (simultaneous localization and mapping). Similarly, Huynh and Roy [10] generate navigation control laws to minimize the uncertainty by combining global planning and local feedback control. A different approach to minimize the pose uncertainty is planning a path which takes the information gain into account [11], or to plan paths such as to maximize the quality of the resulting SLAM estimate [12]. Strasdat *et al.* [13] developed a reinforcement learning approach for the problem of landmark selection in visual SLAM. They learn a policy on which new landmark to integrate into the environment representation in order to improve the navigation capabilities of a lightweight robot with memory constraints. Kwok and Fox [14] apply reinforcement learning to increase the performance of soccer-playing robots by active sensing. The robots learn where to point their camera at in order to localize relevant objects.

Regarding humanoid robots equipped with cameras for localization, few approaches exist that consider the effects of movements on the quality of the observations and on the pose estimate of the robot. Bennewitz *et al.* [15] developed a localization method based on visual features and mention the impact of motion blur on the feature extraction. To obtain clear observations, the robot interrupts its movement at fixed intervals. Seara *et al.* [16] developed an approach for intelligent gaze control for self-localization and obstacle avoidance of a humanoid robot. They avoid motion blur with an active view stabilization. Ido *et al.* [17] explicitly consider the shaking movements of the head and acquire images only during stable phases of the gait. Such an approach is not applicable with our humanoid robot, since there is some unsystematic delay in getting an image after a request.

Pretto *et al.* [18] propose an additional image processing step prior to feature extraction, in particular for humanoid robots. The authors estimate the direction of the motion blur for image patches and developed a feature detection and tracking scheme which is superior to standard techniques. While their approach increases the matching performance, motion blur cannot be completely removed by filtering. However, such a pre-processing technique could be easily combined with our learning approach in order to further improve the performance of the robot.

III. The Humanoid Robot Nao

The humanoid robot *Nao* (see Fig. 1) is 58 cm tall and has 25 degrees of freedom [1]. To observe its surroundings, two monocular cameras of webcam quality are located in its head. One of them points to the ground in front of the robot and is used for vision-based localization in our work. Nao is also equipped with two ultrasound sensors, which can be used for obstacle avoidance during navigation.

Nao has pre-configured walking behaviors for walking forward, sideways, on circles, and for turning. Internally, each walking behavior creates footstep placement patterns. From these, the zero-moment point (ZMP, [19]) is obtained through sampling, which is finally transformed into a trajectory of the center of mass. The joint angle trajectories which result from the executed walking pattern are hand-optimized and generate a smooth and stable walking movement. In our work, we use these default walking behaviors to walk forward or turn on the spot (API version 1.2). The default step length when walking is 5 cm, the default angle per step when turning is 11.5°.

IV. Localization

A. The Unscented Kalman Filter

In this work, we apply the unscented Kalman filter (UKF) to estimate the pose of the robot in a given map of the environment. The UKF is a recursive Bayes filter to estimate the state $\mathbf{x}_t$ of a dynamic system [20]. This state is represented as a multivariate Gaussian distribution $N(\mu, \Sigma)$. The estimate is updated using nonlinear controls and observations $\mathbf{u}_t$ and $\mathbf{z}_t$. The key idea of the UKF is to apply a deterministic sampling technique that is known as the unscented transform to select a small set of so-called sigma points around the mean. Then, the sigma points are transformed through the nonlinear functions and the Gaussian distributions are recovered from them thereafter. Compared to the extended Kalman filter, the UKF can better deal with the nonlinearities inherent to the motion of a humanoid, and leads to more robust estimates.

B. Vision-based Pose Estimation

To estimate the robot's pose in the 2D plane as state $\mathbf{x}_t = (x_t, y_t, \theta_t)$ with an UKF, controls $\mathbf{u}_t$ and observations $\mathbf{z}_t$ need to be available. Wheeled robots usually have rather accurate odometry information available from their wheel encoders, yielding a good estimate of the relative movement. However, this information is not directly available on humanoid robots. Nevertheless, the executed motion command can be used as control $\mathbf{u}_t$ together with an appropriate high covariance according to the considerable noise in the executed motion resulting from foot slippage.

To integrate observations $\mathbf{z}_t$ in the UKF, we extract the state of the art Speeded-Up Robust Features (SURF, [21]) from camera images as visual landmarks (see Fig. 2). Extracted descriptors of these features are matched to landmarks

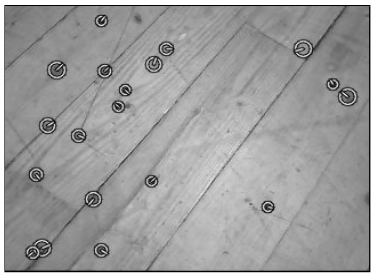

Fig. 2. A floor patch observed by the camera while the robot was standing still. The detected SURF features are marked with circles.

in a map which was constructed beforehand. This map contains the global 2D positions and descriptors of the landmarks on the floor. Whenever the robot matches a perceived feature to a landmark in the map, it integrates the relative 2D position of the landmark as observation $\mathbf{z}_t = (r_t, \varphi_t)$ in the UKF.

To improve performance and robustness, we only regard features within a local area around the current pose estimate for feature matching. The size of this area depends on the uncertainty given by the covariance of the UKF localization. An additional matching robustness is achieved by removing outliers using RANSAC [22] on the 2D positions of the matched landmarks and by limiting the maximum number of features to be integrated per frame.

V. Learning a Reliable and Efficient Navigation Strategy

A. Navigation Task

The task of the robot is to walk to a designated target location with the objective of reaching it reliably and as fast as possible. We assume that a path planner such as A^* plans a path towards the target location yielding waypoints the robots has to traverse. During learning, we consider the scenario of the humanoid walking from its current pose to such an intermediate goal location, which is in viewing distance. The task is completed as soon as the distance between the robot's true pose and the goal location is below a certain threshold. After learning, the policy can then be applied to a scenario containing several waypoints on a longer path.

As mentioned in Section III, we use the default walking patterns of the Nao robot, which are optimized for smoothness and stability. While walking, two problems arise. First, there is typically a high noise in the executed motion commands due to backlash in the joints and foot slippage. The robot's inertial sensors do not provide a motion estimate which could be used to correct the drift by considering it directly in the walking pattern generator. Second, the movement influences the perception of landmarks since motion blur is introduced in the images. Thus, the robot may be prevented from robustly detecting and matching visual features. An accurate localization, which is crucial for reaching the destination reliably and without delays caused by localization errors, is often not possible.

By adjusting its movement accordingly, e.g., by stopping from time to time to acquire sharp images from which features can be extracted and matched reliably, or by turning around to observe more features or to re-align with the goal, the robot learns to trade off an accurate localization versus movement velocity and to compensate for motion drift. This trade-off is hard to assess manually, but poses a typical problem that can be solved with reinforcement learning.

B. Reinforcement Learning

In reinforcement learning, an agent seeks to maximize its reward by interacting with the environment. Formally, this is defined as a *Markov decision process* (MDP) using the state space $\mathcal{S}$, the actions $\mathcal{A}$, and the rewards $\mathcal{R}$. By executing an action $a_t \in \mathcal{A}$ in state $s_t \in \mathcal{S}$, the agent experiences a state transition $s_t \rightarrow s_{t+1}$ and gets a reward $r_{t+1} \in \mathcal{R}$. The overall goal of the agent is to maximize its return

$$R_t = \sum_{i=t+1}^{T} r_i \ , \tag{1}$$

where T is the time when the final state is reached. One finite sequence of states $s_0, \ldots, s_T$ is called an *episode*.

The *action-value function*, also called *Q-function*, for the policy π is defined as

$$Q^\pi(s,a) = E_\pi\{R_t | s_t = s, a_t = a\} \ , \tag{2}$$

which denotes the expected return of taking action a in state s and following policy π afterwards. The optimal policy maximizes the expected return, which corresponds to choosing the action with the maximum Q-value in each state.

We apply Sarsa RL to update the current estimate of the *Q-function* based on its old value, the new reward, and the new value:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\big(r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\big) \tag{3}$$

The parameters α and γ are the step size and the discounting factor, respectively. This form of Sarsa uses only the immediate reward r_{t+1} for the update. By looking further into the future, a more accurate estimate of R_t can be obtained. We use Sarsa(λ), an extension of Sarsa which averages over a number of future rewards [2]. The parameter $\lambda \in [0,1]$ determines the decay of the impact of future rewards.

Throughout the learning phase, we use an ε-greedy action selection. This selection method chooses the action with the highest Q-value with probability ε and all non-greedy actions with equal probability in each step.

C. Augmented Markov Decision Process

We use an *augmented MDP* [23] to model our learning task. The difference is that in a standard MDP, it is assumed that the state of the agent is known. This is not the case in our application, where the belief about the state is represented by a unimodal distribution. In an augmented MDP, the belief about the state is represented by its most-likely estimate and the task is modeled as a MDP. As measure of the uncertainty of the belief distribution, the entropy is included in the state representation.

The probability distribution over the robot pose given all previous odometry information and visual observations is estimated by an UKF. For the MDP, we define the state space $\mathcal{S}$, the set of actions $\mathcal{A}$, and the rewards $\mathcal{R}$ as follows.

D. State Space $\mathcal{S}$

We define a set of features which characterizes the state sufficiently detailed and general as needed for learning. Based on the current, most-likely pose estimate $\mathbf{x}_t = (x_t, y_t, \theta_t)$ and the goal location, we compute the following features:

- The Euclidean distance to the goal location $(g_x, g_y)^T$

$$d = \sqrt{(g_x - x_t)^2 + (g_y - y_t)^2} \; . \tag{4}$$

- The relative angle to the goal location

$$\varphi = \operatorname{atan2}(g_y - y_t, g_x - x_t) - \theta_t \; . \tag{5}$$

  In combination with d, this completely characterizes the relative position of the destination.
- The uncertainty of the localization, computed as entropy over the pose covariance Σ of the UKF:

$$h = \frac{1}{2} \ln \left((2\pi e)^3 \cdot |\det(\Sigma)| \right) \; . \tag{6}$$

We represent the state-action space by a radial basis function (RBF) network, which is a linear function approximator [24]. The continuous features of the state are hereby approximated by a discrete, uniform grid with linear interpolation in between.

E. Action Set $\mathcal{A}$

As explained in Section III, we use the hand-optimized walking behaviors of the robot for navigation. The behavior of the humanoid robot can be influenced by changing the navigation action. We define the following set of actions for the reinforcement learning task:

- Walk forward: The robot walks 10 cm in forward direction.
- Rotate left / rotate right: The robot rotates 23° on the spot in the given direction.
- Stand still: The robot interrupts its movement and waits for 0.7 seconds to acquire a good quality image for its localization (after the robot has stopped, it needs some time until its body stabilizes from shaking).

We chose these actions since they proved to yield the most reliable and predictable behavior.

The actions all influence the quality of the state estimation differently. For example, walking forward generally increases the uncertainty due to the imprecise motion and a reduced probability of obtaining good landmark observations. Stopping to acquire a sharp image or turning around to possibly observe more features may result in a more accurate pose estimate. The rotate action is furthermore needed to re-align the robot with the goal when the robot lost its way.

If the robot is in danger of colliding with a wall due to temporary delocalization, a collision detection mechanism stops the robot. In this case, the robot is only allowed to rotate on the spot.

Note that in contrast to the approach by Hornung *et al.* [8] which only determines the maximum velocity for a separate navigation controller, we do not need such an underlying controller that steers the robot towards the goal. Instead, we directly learn the actions which are executed in each state.

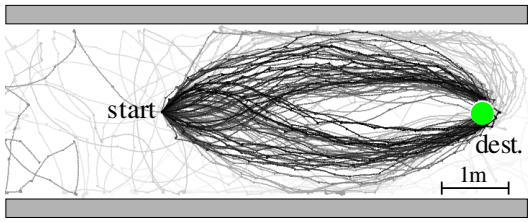

Fig. 3. Evolution of the robot's behavior throughout the 1500 learning episodes (each 10th episode is drawn). At the beginning, random exploratory actions are chosen and the robot does not reach the goal within a maximum of 700 seconds (light gray trajectories). Towards the end, the robot navigates successfully and efficiently towards the destination (black trajectories). Note that there is a high noise in the executed motion commands.

F. Rewards $\mathcal{R}$

We define the immediate reward at time t as

$$r_t = \begin{cases} 200 & \text{if } t = T \\ -\Delta_t & \text{otherwise} \; , \end{cases} \tag{7}$$

where T is the final time step and Δ_t is the time interval in between the update steps. The final state is reached when the robot's true pose is sufficiently close to the destination. This has the effect that the agent is driven to reach the destination as fast as possible, thereby avoiding delays caused by delocalization. Each episode has a maximum duration of 700 seconds. If the robot has not reached the goal within that time, the episode is aborted, resulting in a total reward of -700.

VI. Experiments

The practical experiments were carried out with a Nao humanoid robot, as described in Sec. III. The experimental environment is a hallway with parquet floor and a distance of approximately 5 meters between the starting pose of the robot and the destination. Since the robot has no knowledge about its true pose, we use a special marker to allow the robot to identify when it has reached the goal location. This artificial landmark can be reliably detected even while the robot is walking.

A. Learning the Navigation Policy in Simulation

The policy was learned in simulations. This allowed us to evaluate different parameter settings for the learning algorithms and to run a large number of learning and testing episodes without putting too much strain on the real robot. The simulated robot and its environment are modeled as close to reality as possible. We use a map of artificial landmarks whose positions are randomly distributed with an average density matching the real map (40 landmarks/m^2). To avoid an adaption of the robot's behavior to a specific environment, landmark positions are randomized in each new learning and evaluation episode.

We model motion blur in the simulated experiments as an effect on the probability of an observation, i.e., of a successful feature match, given the currently executed action (walk forward, rotate left/right, or stand still). The probability of observing a landmark for each action was experimentally determined with the humanoid in the real environment.

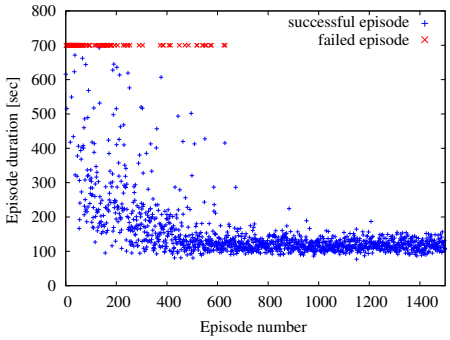

Fig. 4. The duration of the learning episodes shows the convergence of the policy after about 700 episodes.

The motion noise parameters in the simulations are also set to the parameters learned with the real robot. The only difference is the systematic drift to one side which can be observed on our humanoid. We randomly alternate the direction of the drift in each learning episode to avoid an adaption of the learned policy. While a policy adapted to the drift direction would demonstrate an even better performance in our test episodes, it would no longer be general enough to be used on a different robot with the same hardware.

As parameters for Sarsa(λ), we used $\alpha = 0.2$, $\gamma = 0.95$, $\lambda = 0.85$, and $\varepsilon = 0.1$ in 1500 learning episodes. Fig. 3 shows the evolution of the robot's behavior throughout the learning process. As can be seen, in the beginning the robot chose random exploratory actions. It did not reach the goal so that the episodes were aborted after a maximum of 700 seconds (the resulting trajectories are colored light gray). After a certain number of learning trials, however, the robot successfully navigated towards the destination (dark gray / black trajectories). The trajectories were getting more and more efficient towards the end of the learning process. Note that the robot had a systematic error in the executed motion command in each of the episodes. Fig. 4 shows the time to reach the goal throughout the learning episodes. After about 800 learning cycles, the average time to destination converges.

B. Comparison with a Hand-optimized Controller

We now compare our learned policy with a hand-optimized navigation controller. Using this policy, the robot walks forward while the estimated orientation towards the goal is smaller than some threshold $\hat{\varphi}$. Whenever the estimated angular distance to the goal is larger than $\hat{\varphi}$, the robot stops its forward motion and turns towards to goal. We chose the threshold of $\hat{\varphi} = 35°$ since smaller values lead to an oscillating behavior of the robot near the destination, whereas larger values lead to frequent collisions with the walls bounding the corridor. Similar to the dual-mode controllers introduced by Cassandra *et al.* [25], the robot stops in order to obtain a good quality image whenever the uncertainty about its pose exceeds a threshold. This threshold was empirically determined to minimize the time to reach the destination while still achieving a success rate of 100%. Thus, we optimized the parameters of this controller so as to perform best in our test environment.

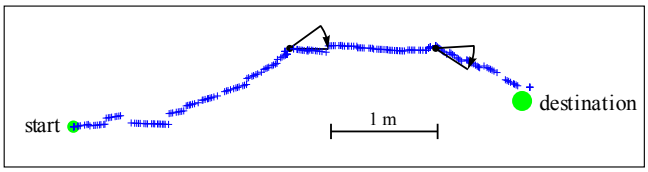

Fig. 5. Estimated trajectory of the Nao executing the learned navigation policy in our hallway. The robot walks forward with an error resulting from a drift to the left. Whenever it seems appropriate according to its belief, the robot executes a rotation action to re-align with the goal. The robot stops as soon as it recognizes the goal landmark.

Throughout the evaluation phase, the learned policy remained fixed and in each time step the action with the highest reward was chosen. We performed 100 evaluation episodes for each of the two navigation strategies in simulation and measured the average time it took to reach the destination from the starting pose. It took 117.18 s $\pm$ 8.41 s to reach the destination with the hand-optimized controller, and only 106.66 s $\pm$ 10.05 s using our learned policy. A t-test with 95% confidence reveals that the learned policy performs significantly better.

C. Evaluation on the Real Robot

We then performed experiments with our real Nao robot navigating in our hallway environment to evaluate whether the results from simulations can be transferred to the real system. We conducted test runs both with the hand-optimized controller and with the policy learned in simulation. The robot needs 93.16 s $\pm$ 10.14 s using the hand-optimized controller compared to 86.53 s $\pm$ 8.60 s using the learned policy, so the learned policy outperforms the hand-optimized controller by 7.1%. Again, a t-test with 95% confidence shows that the learned policy performs significantly better.

Figure 5 depicts a typical trajectory of the Nao robot while executing the learned navigation strategy (the drawn poses were estimated by the localization system). As can be seen, the robot rotates from time to time to compensate for its motion drift and to re-align with the goal. As mentioned before, the learned policy is not adapted to the specific drift direction.

Note that in all experiments presented so far, we used slow walking patterns as the Nao's stability is highly reduced when walking faster. Accordingly, the aquired images are only moderately blurred. The robot can still match an average of 3.25 features per frame while moving, and actions to reduce the uncertainty are rarely executed. Thus, the efficiency gain of the learned controller compared to the hand-optimized controller results mainly from choosing the navigation actions more foresightedly, which leads to shorter paths.

D. Analysis with Respect to Motion Blur

In future implementations, we will optimize the humanoid's gait, so faster walking patterns will be used. Faster movements increase the amount of motion blur, thus seriously reducing the average number of successfully matched features while walking or rotating on the spot. To evaluate the impact of motion blur, we learned policies for a different set

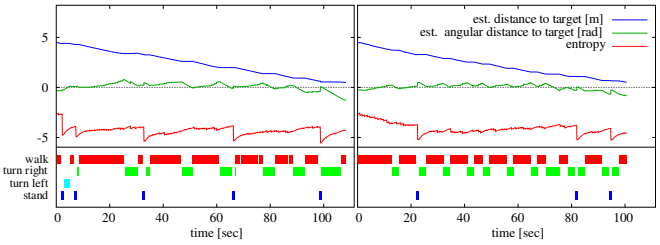

Fig. 6. The state features over time during typical runs with a hand-optimized controller (left) and the learned navigation policy (right) in simulation. The hand-optimized controller stops the robot in regular intervals to decrease the uncertainty, whereas the learned policy adapts the observation frequency according to the current state.

of estimated observation probabilities in the simulator, i.e., we decreased the probability of a successful feature match by 80% during walking and rotating.

Again, we compared the learned policy to a hand-optimized dual-mode controller that stops the robot whenever the entropy exceeds a fixed threshold. For each of the different observation probabilities, we selected the hand-optimized controller leading to the smallest average time to destination while still achieving a success rate of 100 %. The results show that the learned policy is significantly faster (9% gain) than the hand-optimized policy.

Figure 6 shows two typical runs with the hand-optimized controller (left image) and the learned policy (right image). The hand-optimized controller stops the robot in regular intervals to obtain good observations. In contrast to that, the learned policy accepts higher uncertainties as long as the distance to the destination is high, whereas it increases the robot's stand frequency when approaching the destination.

VII. Conclusions

We presented an approach to learn a reliable and efficient navigation strategy for a humanoid robot. Typically, the quality of the pose estimate seriously decreases during walking due to inaccurately executed motion commands and high noise in the observations. A wrong pose estimate, in turn, may prevent the robot from accomplishing its navigation task efficiently and successfully.

We formulate the task of navigating to a target location reliably and, at the same time, as fast as possible as a reinforcement learning problem. In our approach, the robot learns which navigation actions to choose in order to reach its goal efficiently. Increasing the speed of the gait typically leads to a higher amount of blurred images during walking and an uncertain localization. Our learned strategy then forces the robot to stop from time to time, so that it obtains good observations and avoids delays or failures caused by localization errors. In simulated and real-world experiments with a humanoid robot equipped with a camera for localization, we showed that our learned navigation policy significantly outperforms a hand-optimized navigation strategy.

References

[1] Aldebaran Robotics, "The Nao humanoid robot," http://www.aldebaran-robotics.com/eng/Nao.php, 2009.

[2] M. Wiering and J. Schmidhuber, "Fast online Q(λ)," *Machine Learning*, vol. 33, no. 1, pp. 105–115, October 1998.

[3] F. Faber and S. Behnke, "Stochastic optimization of bipedal walking using gyro feedback and phase resetting," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2007.

[4] C. Niehaus, T. Röfer, and T. Laue, "Gait-optimization on a humanoid robot using particle swarm optimization," in *2nd Workshop on Humanoid Soccer Robots at the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2007.

[5] T. Hemker, M. Stelzer, O. von Stryk, and H. Sakamoto, "Efficient walking speed optimization of a humanoid robot," *Int. Journal of Robotics Research*, vol. 28, no. 2, pp. 303–314, 2009.

[6] Q. Huang, Z. Yu, W. Zhang, X. Duan, Y. Huang, and K. Li, "Generation of humanoid walking pattern based on human walking measurement," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2008.

[7] R. Chalodhorn, D. Grimes, K. Grochow, and R. Rao, "Learning to walk through imitation," in *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, 2007.

[8] A. Hornung, H. Strasdat, M. Bennewitz, and W. Burgard, "Learning efficient policies for vision-based navigation," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009.

[9] T. Kollar and N. Roy, "Using reinforcement learning to improve exploration trajectories for error minimization," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2006.

[10] V. A. Huynh and N. Roy, "icLQG: Combining local and global optimization for control in information space," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[11] R. He, S. Prentice, and N. Roy, "Planning in information space for a quadrotor helicopter in a GPS-denied environments," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2008.

[12] M. Bryson and S. Sukkarieh, "Active airborne localisation and exploration in unknown environments using inertial SLAM," *IEEE Aerospace Conference*, 2006.

[13] H. Strasdat, C. Stachniss, and W. Burgard, "Which landmark is useful? Learning selection policies for navigation in unknown environments," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2009.

[14] C. Kwok and D. Fox, "Reinforcement learning for sensing strategies," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[15] M. Bennewitz, C. Stachniss, W. Burgard, and S. Behnke, "Metric localization with scale-invariant visual features using a single perspective camera," in *European Robotics Symposium 2006*, ser. STAR Springer tracts in advanced robotics, H. Christiensen, Ed., vol. 22, 2006.

[16] J. Seara and G. Schmidt, "Intelligent gaze control for vision-guided humanoid walking: methodological aspects," *Robotics and Autonomous Systems*, vol. 48, no. 4, pp. 231–248, 2004.

[17] J. Ido, Y. Shimizu, Y. Matsumoto, and T. Ogasawara, "Indoor Navigation for a Humanoid Robot Using a View Sequence," *The International Journal of Robotics Research*, vol. 28, no. 2, pp. 315–325, 2009.

[18] A. Pretto, E. Menegatti, M. Bennewitz, W. Burgard, and E. Pagello, "A visual odometry framework robust to motion blur," in *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2009.

[19] M. Vukobratovic and B. Borovac, "Zero-moment point – thirty five years of its life," *Int. Journal of Humanoid Robots*, vol. 1, 2004.

[20] S. J. Julier and J. K. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, 1997, pp. 182–193.

[21] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded-up robust features," *Proc. of the ninth European Conf. on Computer Vision*, 2006.

[22] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, 1981.

[23] N. Roy and S. Thrun, "Coastal navigation with mobile robots," in *Advances in Neural Processing Systems 12 (NIPS)*, vol. 12, 1999.

[24] K. Doya, "Reinforcement learning in continuous time and space," *Neural Computation*, vol. 12, no. 1, pp. 219–245, 2000.

[25] A. R. Cassandra, L. P. Kaelbling, and J. A. Kurien, "Acting under uncertainty: Discrete bayesian models for mobile-robot navigation," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 1996.