Using a Reinforcement Learning Controller to Overcome Simulator/Environment Discrepancies

Nancy Owens and Todd Peterson
Machine Intelligence, Learning, and Decisions Laboratory
Brigham Young University, Provo, UT, 84602

1 Introduction

As many researchers have noted [2, 3, 8, 9], robotic controllers often fail to perform well when transferred from a simulator to a situated domain. Several explanations for this phenomenon have been proposed [7, 10]. The most prevalent and, we believe, significant of these is the fact that no simulator can currently match the complexity of the real world. Discrepancies between a simulator and the real-world system which it is intended to model are therefore unavoidable. Depending on the sensitivity of the controller, the nature of the desired task, and the number and severity of the discrepancies, observable effects of the simulator-to-real-world transfer may range from negligible to dramatically detrimental.

Factors such as time, expense, and potential risk to robotic hardware make situated design and testing infeasible for a large percentage of applications. Simulations are therefore indispensable in many areas of research and development, and effective methods of dealing with simulator/environment discrepancies must be developed.

Traditional approaches to this problem attempt to reduce the number and severity of simulator/environment discrepancies in order to obtain better results when the controller is transferred to a situated domain. These approaches, although often effective, are limited in that they require the designer to successfully anticipate the types of discrepancies the controller is likely to encounter.

In this paper we present a different approach which focuses on overcoming discrepancies by designing a controller which is robust to unexpected changes in its environment. This approach is not intended as a replacement for previously developed techniques, but rather as a supplement to them. This combination of discrepancy reduction techniques and discrepancy-robust controllers is shown to be effective at overcoming artificially introduced discrepancies in several simulator-to-simulator transfers, as well as in an actual transfer from a Nomad simulator to a Nomad Scout robot.

2 Reducing Discrepancies

A common approach to simulator/ environment discrepancies is to alter simulator designs in order to create a model from which policies are more easily transferable to the real world. This could be accomplished through the use of high-precision simulations intended to model the real world more accurately, through minimized simulations intended to emphasize only those aspects of the real world which are useful to the agent, or through noisy simulations intended to force the agent to be robust to variation in its environment. Techniques such as calibration, sensor feedback, and stipulations on the environment itself have also been used to address this problem.

2.1 Specialized Simulations

Specialized simulations are simulations which attempt to create a simulator model as consistent with the real world as possible. This naturally increases the complexity of the simulator, but is often rewarded by increased accuracy in simulations.

One way to improve simulation accuracy is to use sensor input from a robot to build its own simulator model. Lee et. Al. [8] successfully used sensor data from a Nomad 200 autonomous mobile robot to train a neural network which modeled the robot's interactions with its environment. Lund and Miglino [9] demonstrate da similarly successful process with a Khepera miniature robot.

Specialized simulations are a valid approach for projects which require repeated trials using the same robot design over a long period of time. However, this type of increased-accuracy simulation assumes consistent functioning of the robot's sensors and actuators over time once the controller has been transferred to the real world, an assumption which does not apply in many situations.

Additionally, the amount of time required to build such a simulator is extensive, and the simulator, once built, is customized to the specific robot design (and perhaps even the individual robot) whose sensor data was used to create the model. If the robot design itself is being changed as part of the development process, this approach is entirely infeasible.

2.2 Minimal Simulations

An alternate approach is to simplify the simulation rather than make it more specific. Minimal simulationists assert that the majority of real-world data is irrelevant to the controller's ability to make appropriate decisions, and that it is therefore unnecessary to model that data accurately.

Nick Jakobi's *Radical Envelope of Noise Hypothesis* [7] is an excellent example of this approach. Jakobi defines a group of input features, called the *base set aspects* of a simulation, which can be safely relied upon by the controller to be an accurate model of reality. All other features of the simulation are referred to as *implementation aspects*, and are modeled incompletely and noisily to prevent the controller's reliance on irrelevant data. Random variation is also introduced into the base set aspects of the environment to encourage robustness in the controller.

Jakobi's methodology was successfully used by Perkins and Hayes [12] in the creation of a simulator for a genetic algorithm learning a visual tracking task.

Minimal simulations are an encouraging area for further exploration in reducing simulator/environment discrepancies, but they also have drawbacks. Perhaps the most significant of these is that the designer is performing half of the controller's work for it by creating a simulator which provides nothing but relevant data. This is acceptable for some applications, but may not be appropriate for situations where the designer himself does not know which input parameters are most relevant to the desired task. In addition, the controller's robustness is limited only to those discrepancies which the designer was able to foresee and incorporate into the simulation.

This approach is also not applicable to situations in which the agent is required to perform multiple tasks, each of which relies more heavily on different inputs. Whereas specialized simulations are specific to the robot design used, minimal simulations are specific to the task which is to be accomplished. Ideally, a simulation should be applicable across several robot implementations and task specifications.

2.3 Calibration and Controlled Environments

Two other methods which have been used frequently in the manufacturing world are calibration and controlled environments [6]. These methods differ from the techniques discussed above in that they attempt to constrain the real world to more closely approximate the simulator, rather than the other way around.

Calibration is a modification to the agent which attempts to reduce simulator/environment discrepancies by scaling the agent's inputs and outputs so that the real world appears more like the simulator to the agent. This approach has obvious benefits, but the agent's capacity to adapt to its environment is again limited by the foresight of the designer. In addition, calibration requires either the assistance of a human, specialized machinery, or a reliable baseline measurement in order to establish correct scaling values. These factors may not always be available. Also, non-static environments would require constant re-calibration, which would be time-consuming detrimental the agent's performance in some tasks.

In a controlled environment the robot's surroundings are constrained to match certain predefined standards so that the controller will never be faced with certain types of discrepancies. A robot soccer tournament, for example, constrains the environment of the playing field to guarantee specific conditions: the size and shape of the field, occasionally its color, and the fact that the only objects on the field are the robots and the balls. Obviously, this approach can be applied only to certain applications. It is neither possible nor practical to attempt to control complex interactive environments such as office building hallways.

We wish to emphasize that each of the methods discussed in this section is sufficient and effective for certain applications. But no single approach is sufficient for all applications, and for certain applications, none of them is sufficient. All of the methods of dealing with simulator/environment discrepancies discussed so far require the designer to possesses specific information about the quantity and type of discrepancies an agent will be subject to in a situated domain. Such information is often, but not always, available.

It is in the areas where the approaches described in this section fail that an additional aid to overcoming simulator/environment discrepancies would be helpful.

3 Overcoming Discrepancies

The concept of a robust controller is not new. In addition to creating simulations with enough random variation to ensure robustness, many designers have focused on designing inherently robust controllers. A notable example is FeatureBoost, a meta algorithm designed by O'Sullivan et. al. [11]. FeatureBoost deliberately exploits redundancy of the input features in order to create hypotheses which are robust to the failure, occlusion, or corruption of one or more inputs. FeatureBoost was successful in several test situations, and seems promising as an approach to designing more robust agents. However, the success of the algorithm relies on the discovery of an optimal schedule for biasing feature use, which is dependent on the specific algorithm and target function. This makes general application of FeatureBoost more difficult than some other algorithms.

We have chosen to explore a different approach. We opted to use a reinforcement learning controller which uses temporal differences to learn a control policy adapted to its environment. Such an agent is capable of adapting its behavior to respond appropriately to environmental changes.

Specifically, we use a Q-learning agent with a CMAC [1] function approximator as our controller. Q-learning [14] is a type of reinforcement learning in which the agent maintains a utility value, or Q-value, for each state-action pair. When the agent selects an action *a* from state *s*, it receives a reward R, which is used to update the Q-value for that action in that state according to the update function:

$$\Delta Q(s_t, a_t) = \alpha\,(R(s_t, a_t) + \gamma\, max_a Q(s_{t+1}, a))$$

where α is the learning rate, γ is the discount factor, and R(*s*, *a*) is the reward obtained by executing action *a* in state *s*.

Because the input spaces of real-world environments are often continuous, function approximators are used to group similar input combinations into collective states, thus allowing limited generalization to infrequently visited areas of the input space. Unfortunately, even with function approximation, standard Q-learning methods require a significant amount of time to adapt to changes in the environment or reward structure of the agent. If simulator/ environment transfers are to be effective and useful, the agent must be able to overcome discrepancies in a reasonable amount of time.

Simulator/environment discrepancies can be viewed as a jump from a source task (the simulator) to a similar but distinct target task (the situated domain). This means that task transfer algorithms may be used to reduce the amount of learning time required by the controller as it adapts to inconsistencies in its environment.

Task transfer and knowledge transfer algorithms have previously been studied in relation to reinforcement learning [4, 5, 13]. We have chosen three algorithms to focus on for the scope of this research. Each is discussed in turn.

3.1 Memory-Guided Exploration

The problem with directly transferring Q-values from a source task to a target task are twofold: First, portions of the source policy which are inapplicable to the target task may require more time to unlearn than it would have taken to learn those policies from scratch. This effect was noted by Bowling and Veloso in [2]. Second, during the unlearning phase, correct knowledge of a portion of the policy may be lost due to the back-propagation of information before states have been fully explored. (For a more detailed analysis of these problems see [4])

Memory-guided exploration [13] attempts to overcome these difficulties by initializing the Q-values for the target task to default initialization values, then using information from the source task's Q-value set to guide the agent's exploration towards those areas of the state space which were known to be most fruitful in the past. This has the effect of reducing unlearning time and preserving previous knowledge even as the Q-values are updated.

The exploration function is

$$a_t = max_a[W \times Q^{old}(s_t, a) + (1-W)Q^{new}(s_t, a)]$$

where W is the time-decayed weighting on the old Q-values.

3.2 Soft Transfer of Q-values

One drawback of the memory-guided exploration algorithm is that if the value for W and its decay rate are not set optimally, irrelevant portions of the previous task continue to have an effect on the agent's behavior even after the Q-values reflect a more efficient policy. The related Soft Transfer algorithm [4] avoids this problem by using a weighted average of the source task's Q-values and the default initialization values to initialize the Q-values for the target task. The initialization function is:

$$Q_0^{new}(s, a) = (1-W)I + W(Q_F^{old}(s,a))$$

where Q_0 is the initial Q-value, Q_F is the final Q-value, and *I* is the standard *tabula rasa* initialization value.

This solves the problem of over-reliance on past experience, but reintroduces the knowledge-loss problem discussed in the previous sub-section.

Memory-guided Exploration and Soft Transfer are both highly sensitive to the relationship between W, W's decay rate, and the mean and variance of the optimal Q-values of the source and target tasks. In order to maintain consistency in our data, we have chosen to use the same initial weighting W = 0.3 and a linear decay rate of 0.0006 every 100 iterations for all experimental runs in this paper.

3.3 KNN-based Q-value Prediction

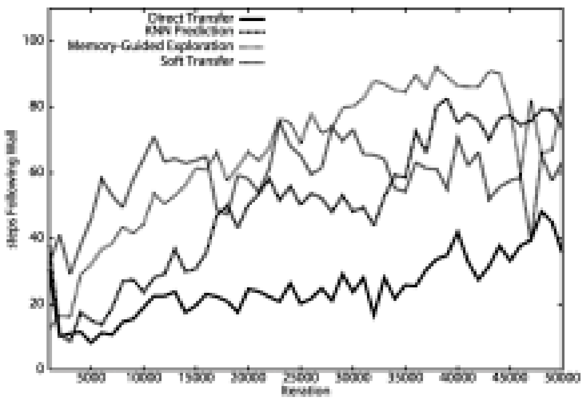
Memory-guided Exploration and Soft Transfer provide an adequate means for transferring information across similar states in related tasks, but simulator/environment discrepancies also frequently introduce completely new states which the agent has never visited before. We use a K-Nearest Neighbor approach to generalize to these previously unvisited states. In the interest of calculation time our algorithm is highly simplified, averaging only Q-values with a distance of 1 away from the unvisited state in the state space.

4 Experimental Results

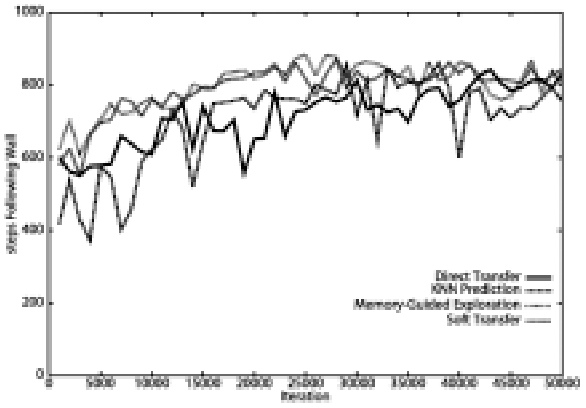
4.1 Simulator-to-Simulator Transfer

We used a Nomad Simulator to compare the effectiveness of these algorithms in several artificial cases of simulator/environment discrepancies. The Q-learning controller was initially trained on a simple wall-following task in a rectangular room and was then required to perform the same task in the face of various environment discrepancies which were artificially introduced into the simulation. We compare the performance of our baseline algorithm, Direct Transfer of Q-values, with the Memory-guided Exploration, Soft Transfer, and KNN Prediction algorithms discussed in the previous section.

The x-axis of the graphs indicates the number of time steps. The y-axis indicates the percentage of time which was spent within a pre-specified distance from a wall (i.e. the number of steps spent fulfilling task criteria.)



[Figure 1: "Malfunctioning sensor" scenario: The front-right sonar of the robot was hard-coded to return the maximum possible distance reading to the robot, regardless of the actual distance reading. Average of three runs.]



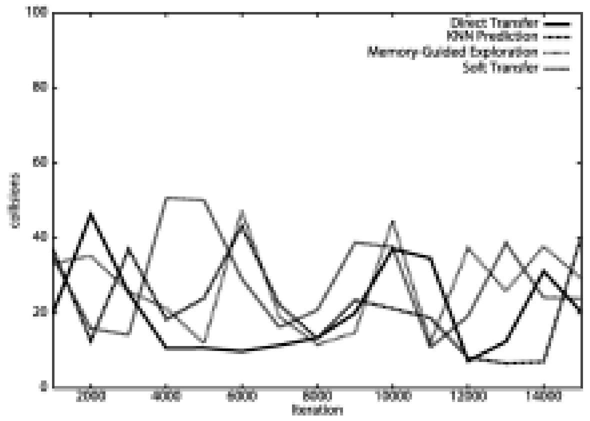
[Figure 2: "New wall" scenario: An extra wall was added down the length of the rectangular room. Average of three runs.]

4.2 Simulator-to-Real-World Transfer

We then applied these methods to a case of true simulator/environment transfer in which a simulator-trained controller was transferred to an actual Nomad Scout robot. The stipulations of using an actual real-world robot required some minor changes to the actuator implementation of the controller, as well as to the task specifications.

The controller was initially trained to perform a wall-avoidance task in a rectangular hallway, with a reward structure which encouraged forward motion while heavily discouraging collisions. The controller was then transferred to a real hallway with dimensions equivalent to those of the simulated map, and was required to perform the same wall-avoidance task.

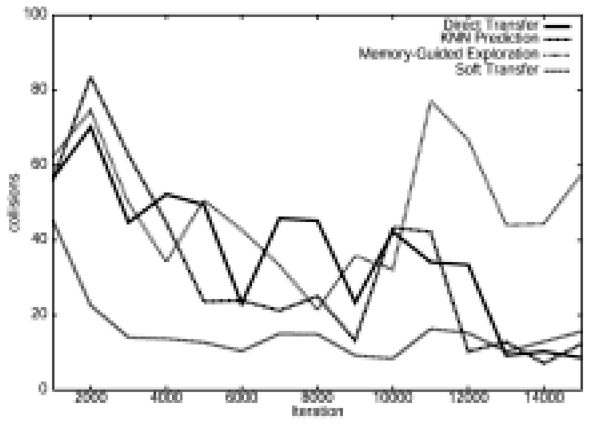
Again, the x-axis of the graphs indicates the number of time steps. The y-axis indicates the percentage of actions which resulted in collisions. Results are shown below.



[Figure 3: Pure simulator to environment transfer: No artificially introduced discrepancies. A single experimental run.]

In this case, all algorithms appear to be functioning with approximately the same efficiency. This is probably because the discrepancies between the simulator and the real world were relatively small. In cases of small discrepancies, direct transfer is likely to adapt to the new environment as rapidly as the other knowledge transfer mechanisms.

In order to test this theory, we made the transfer more difficult by introducing an artificial sensor malfunction identical to that which produced Figure 1. The results reveal that, when the severity of the discrepancy was increased in this manner, Soft Transfer adapted more quickly than the other algorithms.



[Figure 4: Simulator to environment transfer using the “Malfunctioning sensor” scenario. The front right sensor always returns maximum distance possible. A single experimental run.]

5 Discussion

As one can discern by analyzing the graphs, Memory-Guided Exploration and Soft Transfer tend to out-perform the other methods. This is not surprising, since in both the “malfunctioning sensor” and the “new wall” scenarios, the optimal action in a given state has changed in some areas of the state space.

More surprising is the behavior of the KNN-based Q-value Prediction algorithm. It performed well in the “malfunctioning sensor” scenario, which is to be expected because approximately 50% of the newly perceived states in this scenario are a distance of exactly 1 unit away from a state with an appropriate Q-value set for the unvisited state. The KNN Prediction algorithm’s behavior in the “new wall” task is a little more surprising. We hypothesize that the algorithm failed to improve the agent’s performance because each newly perceived state was exactly one unit away from two previously experienced states which required opposite behaviors. It is possible that these opposing states cancelled each other out during the KNN averaging process, leaving the agent with no useful generalization information. Further investigation into this matter is warranted.

The knowledge transfer methods studied in this paper appear to perform best when discrepancies are fairly large. When discrepancies are small, the transfer algorithms result in performance comparable or even inferior to that of the Direct Transfer baseline. One may therefore conclude that these knowledge transfer algorithms are extremely useful in overcoming significant discrepancies not anticipated by the designer, but are less effective in overcoming small discrepancies which cannot feasibly be removed through discrepancy-reduction techniques like those discussed in section 2.

It could be argued that function approximation and reinforcement learning alone are capable of overcoming small discrepancies, and that the use of knowledge transfer algorithms is therefore unnecessary. This may be true, but as the severity of a discrepancy increases, the time required to overcome it will also increase. Once the severity of the discrepancy increases beyond a certain point, knowledge transfer techniques will be required to make the reinforcement learning approach feasible.

6 Conclusions and Future Work

In this paper have demonstrated that reinforcement learning agents applying task transfer algorithms can successfully overcome some types of simulator/environment discrepancies in a reasonable amount of time. The task transfer algorithms appear to be most successful in cases where the discrepancies are more pronounced. This implies that such algorithms could be used to make the controller robust to any discrepancies which were not anticipated by the designer and minimized during the simulation phase.

We believe that further refinements to these algorithms may enable them to successfully overcome less severe discrepancies as well dramatic ones, making them even more effective as a supplement to traditional approaches.

One possible refinement would be to expand the Memory-Guided Exploration and Soft Transfer algorithms to automatically select an appropriate W and decay rate based on the statistical distribution of the Q-values of the source and target tasks. This could potentially expand the range of situations to which these knowledge transfer algorithms can successfully be applied.

A second potential refinement to the knowledge transfer algorithms would be to expand the KNN-based Q-value Prediction algorithm to use a double-weighted KNN calculation with arbitrarily large K, using both distance in the state space and a confidence factor (based on the number of updates received by a state) as distance metrics. This would enable more accurate prediction of the Q-values of unvisited states, as well as allowing the algorithm to predict Q-values for large, isolated areas of the state space which have never been visited.

The Q-value Prediction algorithm could also be used as a generalization technique in the initial learning of a task. In this case, the first time a state is visited by the agent, its Q-value would be initialized using the Q-value Prediction algorithm. This might dramatically increase learning rates for some types of tasks.

References

[1] J.S. Albus. A new approach to manipulator control: The cerebellar model articulator controller (cmac). *Trans. ASME, J. Dynamic Sys. Meas., Contr.,* 97:220-227, 1975.

[2] M. Bowling and M. Veloso. Reusing learned policies between similar problems. *In Proceedings of the AI*IA-98 Workshop on New Trends in Robotics*, Padua, Italy, 1998.

[3] B. Brunner, K. Arbter, and H. Hirzinger. Task directed programming of sensor based robots. *In IEEE/RSJ Int. Conf. On Intelligent Robots and Systems,* Munich, Germany, 1994.

[4] J.L. Carroll, T. Peterson, and N. Owens. Memory-guided exploration in reinforcement learning. *In IJCNN2001,* Korea, 2001. In Press.

[5] K. Dixon, R. Malak, and P. Khosla. Incorperating prior knowledge and previously learned information into reinforcement learning agents. *Technical Report,* Institute for Complex Engineered Systems, Carnegie Mellon, 2000.

[6] J. Gowdy and Z. Butler. An integrated interface tool for the architechture for agile assembly. *In IEEE International Conf. On Robotics and Automation,* 1999.

[7] N. Jakobi. Evolutionary robotics and the radical envelope of noise hypothesis. *Journal of Adaptive Behaviour*, 6(2):325—368, 2000.

[8] T. Lee, U. Nehmzow, and R. Hubbold. Computer Simulation of Learning Experiments with Autonomous Mobile Robots. *Proc. TIMR 99, "Towards Intelligent Mobile Robots",* Bristol, 1999.

[9] H. Lund, and O. Miglino. From simulated to real robots. *In Proceedings of IEEE 3rd International Conference on Evolutionary Computation.* 1996.

[10] M. Mataric. Reward functions for accelerated learning. *In Proceedings of the International Machine Learning Conference,* pages 181-189, 1994.

[11] J. O'Sullivan, J. Langford, R. Caruna, and A. Blum. FeatureBoost: A meta-learning algorithm that improves model robustness. *ICML '00*, pages 703—710. 2000.

[12] S. Perkins and G. Hayes. (1998) Evolving complex visual behaviors using genetic programming and shaping. *Presented at the 7th European Workshop on Learning Robots,* Edinburgh.

[13] T. Peterson, N. Owens, and J.L. Carroll. Automated shaping as applied to robot navigation. *In ICRA2001*, San Diego, CA, 2001. In Press.

[14] C. J. C. H. Watkins. Learning from delayed rewards. *PhD thesis,* University of Cambridge, 1989.