Action Selection in Robotic Motion Learning

Abstract

Robot motion planning and control is essential for robots to navigate in the real world. Autonomous robots are expected to take actions on their own in order to actively explore an environment, rather than being fed by hand-tuned knowledge by a human. Stronger and Stone designed an algorithm, called ASAMI, which allows robots to learn their environment efficiently. However, it assumes access to some knowledge of the effect of the actions even though we may not have access to such information in certain domains. In this paper, I propose an extension on ASAMI which overcomes these drawbacks, and allows the robot to learn the actions as well as the environment simultaneously.

1 Introduction

There are two critical models for robot motion—the action model and the sensor model. The action model predicts the effect of an action, or formally, $A \to \Delta S$, where A is the action space and S is the location space. The sensor model predicts the location given an observation, or formally, $O \to S$, where

Figure 1: Robots play soccer. [2]

O is the observation space. Traditionally, these two models are fed to the robot by a human. The robot only needs to accomplish a higher-level goal, by planning and learning (Figure 1).

However, robots can learn the action model and the sensor model simultaneously without any prior knowledge. One example of such learning is the ASAMI algorithm [10]. In the algorithm, there are two estimates of the current state of the robot — W_s, the state estimated by the sensor model, and W_a, the state estimated by the action model. They are both assumed to be polynomial functions, and the robot needs to estimate the parameters of each estimate. In a one-dimensional scenario, the robot is required to walk back and forth (which clearly requires some knowledge of the actions). In one iteration, the action model is used to update the sensor model, and vice verse. They will finally converge in a bootstrapping way. This idea is illustrated in Figure 2.

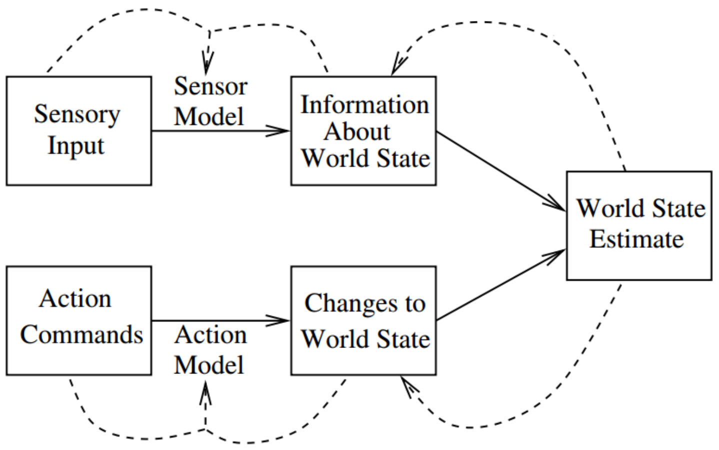

Figure 2: the robot can use redundant information to learn its action and sensor models, revised from Figure 2 in [10].

2 Analysis of ASAMI

In ASAMI, actions are selected randomly in the training. However, this can be biased according to the current belief. In this sense, the agent should be able to determine which action would lead to the most uncertain results and thus need more samples. It only knows the consistency of them, and has no knowledge about the correctness of the models. For example, states with a larger difference in the action model and the sensor model ($|W_a - W_s|$) should be considered inconsistent. For unobserved state, action pairs can be also assumed to be inconsistent.

The first author of [10], Dan Stronger, commented that "one possible reason the W_a is wrong after a certain action is that the action is just very noisy ... This is a good reason to gather more data for that action ... But another possible reason an action could be causing problems is because the action model function being fit, with the degrees of freedom that it has, just fits to a function that's not especially accurate for that action". This would be a problem in degree selection in the polynomial regression [12]. In this paper, I'll give our action model proper degrees of freedom. So, if the action model is inconsistent, the reason should be that either there are too few data gathered to make consistency happen, or the data gathered are too noisy.

Action selection is even necessary in some environment. The experiments in [10] assume that *we* know the effects of the actions (going forward, going backward, etc.). So to explore the environment, we make the robot walk forward and backward alternatively to explore the state space. However, the actions' effects may be completely unknown. If the agent still chooses the actions uniformly random, then exploration would be inefficient.

For example, we consider one-dimensional walk. Let the world be in the range of $[0, n]$. The agent starts at 0, and wants to reach n. Let the range of velocity be $[-2a, 2a]$ (positive velocities mean going forward, and negative ones mean going backward), and in every step the agent chooses an action uniformly random. Under these conditions, the expected absolute velocity of the walk is a. Assume the agent would "bump into" the boundary of 0 and stays there if it tries to go backward from position 0. The expected *moving forward* steps needed to take to reach position n from 0 is $\frac{n}{a}$. This forms a *random walk* [6] problem. The expected steps needed to take to reach distance n is $O(n^2)$. This implies that if an agent takes random actions in an unknown domain, it is likely that the agent would restrict itself within a small sub-domain.

3 Literature Review

In the perspective of model-adaptive agents [5], *action selection* is a critical problem because it accelerates the progress towards its goal—consistency between the action model and the transition model. This also solves the problem of *learning from experience*, as the inconsistency is an essential piece of information that can be used for further decision-making.

There is a two-dimensional version of ASAMI discussed in [11]. As the action space is larger in higher dimensions, biasing on actions instead of random selection on actions becomes more essential.

Compared with this proposed method, some similar ideas are discussed in the developmental robotics literature. They call the inconsistency described above as *error* [7], *surprise* [8] or *curiosity* [9]. This serves as a motivation to change the model. A metric can be used to describe the global inconsistency [7]. Then, a reinforcement learning framework can be used to plan to get to the states with the least global inconsistency. The authors in [8] used logic rules as the model, and let the robot figure out why there could be a surprise. In this paper, I assume that the surprise is caused by insufficient or noisy data. So the surprise, or inconsistency, of an action, is simply caused by the data gathered at that action.

Additionally, there is an assumption in [10] that there is a mapping from action to the difference of state, i.e., $A \to \Delta S$, where A is the action space and S is the state space. This is true in robot motion. So it's possible to learn the difference of the states, when an action is given. This is exactly how the task is designed in [4]. If this is not true, so that $S \times A \to S$ is the best we can estimate, this would become more challenging but still possible to be dealt with. We might know that a certain (s, a) is inconsistent and we want to gather more data on it. But our action model might not be well-learned, so the reward should be a compromise between the inconsistency of that (s, a) and the cost to reach it.

There are also other related works on learning the action model, but they make use of very different approaches. They assume that one model, usually the sensor model, is well calibrated. The data are represented as either statistics [3] or instances [1]. These methods would be further analyzed and compared with our approach in Section 6.

4 Proposed Algorithm

In this section, I propose an algorithm, named Strong ASAMI, which makes the agent choose actions in a computationally cheap way. It keeps the essentials of ASAMI, however, Strong ASAMI can be applied when there is no prior domain knowledge about the action model. The idea is that we divide the learning process into two phases. The agent learns the actions first and learns the state space later. In the first phase, it chooses the action with least confidence to learn. In the second phase, it uses the actions it learned to explore the environment.

Some essential functions are presented in Algorithm 1. In initialization, a sample list is initialized for each action (Line 5). A_0 is the initial action model (a roughly correct action model we have), and its prediction for each action is pushed to the corresponding sample list. This is a useful initialization, as the variance of the sample list would represent the error of our initial model,

Algorithm 1 Strong ASAMI

```
1: function INITIALIZE($rangeOfActions$)
2:     $trials \leftarrow 0$
3:     $actions \leftarrow$ discretized $rangeOfActions$
4:     for $action$ in $actions$ do
5:         $samples[action] \leftarrow \{\}$        ▷ Initialize a sample set for each action
6:         $s \leftarrow A_0(action)$
7:         push $s$ to $samples[action]$        ▷ add the prediction by the initial action model
8:     end for
9: end function
10:
11: function UPDATE($action, \Delta state$)
12:     add $\Delta state$ to $samples[action]$
13:     increase $trials$
14: end function
15:
16: function GETACTION
17:     if $trials \leq$ ACTION_LEARNING_TRIALS then
18:         $action \leftarrow$ the action such that $samples[action]$ has the largest variance
19:     else
20:         $action \leftarrow$ appropriate action for state exploration
21:     end if
22: end function
```

when the first observed sample is added.

In each iteration, `update` (Line 11) is called to update the knowledge of the action effects by adding new observations. When the robot needs to decide what action to take next, `getAction` (Line 16) is called. The action selection strategy in phase 2 (Line 20) always pick the action that has the largest variance in the observed samples. Then it keeps choosing actions to explore the environment. This is up to the agent to decide how to plan to traverse the state space. Consider a one-dimensional environment, it might want to keep trying actions with positive $\Delta state$. When it observes no state changes after executing such action, then it tries actions with negative $\Delta state$. With this, the robot should have the performance of walking forward and backward in the domain.

5 Experiments

The setup used in my experiment is the same setup used to evaluate ASAMI [10]. The plan for the experimental evaluation is detailed below. Firstly, the robot can walk back and forth in a one-dimensional path. There is a beacon placed at one end, and the robot is always facing towards the beacon. The only observation is the height of the beacon. The state is given by the distance to the beacon (Figure 3). The environment could be noisy - this is inherently true when doing experiment on real robots.

First, I test ASAMI in a toy domain. I only naively implemented the action model and the sensor model. Then, I made ASAMI and Strong ASAMI work on Aldebaran Nao robot. Action model and sensor model are both

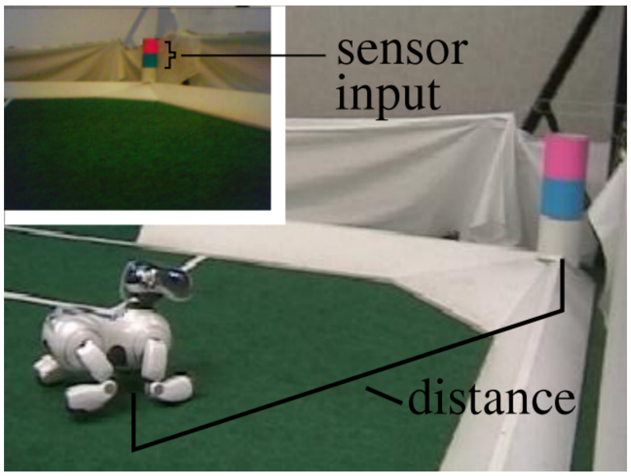

Figure 3: The observation is the height of the beacon in the experiment. The state is the distance to the beacon. Nao is used instead of Aibo shown in this figure. [10]

assumed to be cubic functions, and we have four parameters to estimate for each model. The initial action model A_0 is defined to be a constant function, $A_0(x) = c$, where c is a small random number. This is used to train the state estimated by the action model w_a in the first 100 iterations.

The goal is to have W_s and W_a both converge and consistent with the true position, or state. It is possible that they are not numerically equal to the state, but they should have the same shape, that is, they could be an offset or scale of the true function.

5.1 Test in Simulator

The actions in the simulator are simply walking backward or froward. The robot can choose from velocity of [-1, 1]. The change of state is affected by

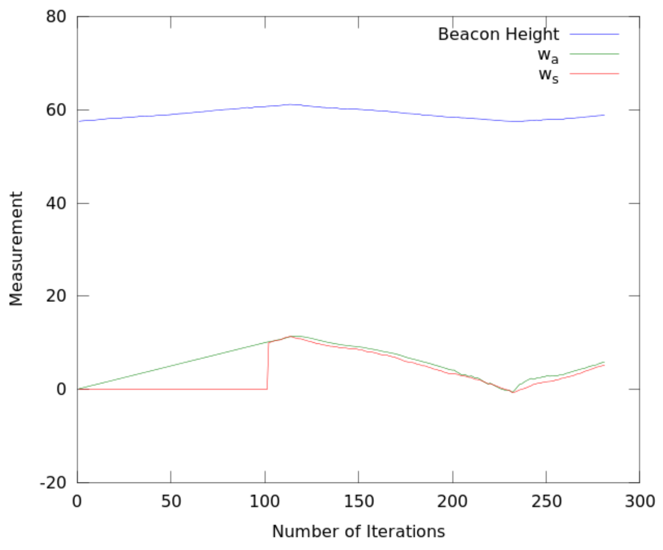

Figure 4: The learning performance in the simulator.

the velocity. The sensor model computes the height of the beacon according to the current state. The result is in Figure 4. There is no noise in the simulator so the results show that the learning on both models converge quickly. Estimates from both models have the same shape as the true states.

5.2 Test on Nao: First Attempt

The challenges of making ASAMI walk on Nao are that beacon height is too noisy, the height returned by the beacon detector could be noisy because the observed image can blur, and the real walking velocity can be different from what is set. The result is in Figure 5.

The observation is retrieved and ASAMI is updated in each iteration. The beacon height shows that the robot walks forward and backward two times, and then kidnapped[1] from a further point to a closer point to the beacon in around 3,500 iteration.

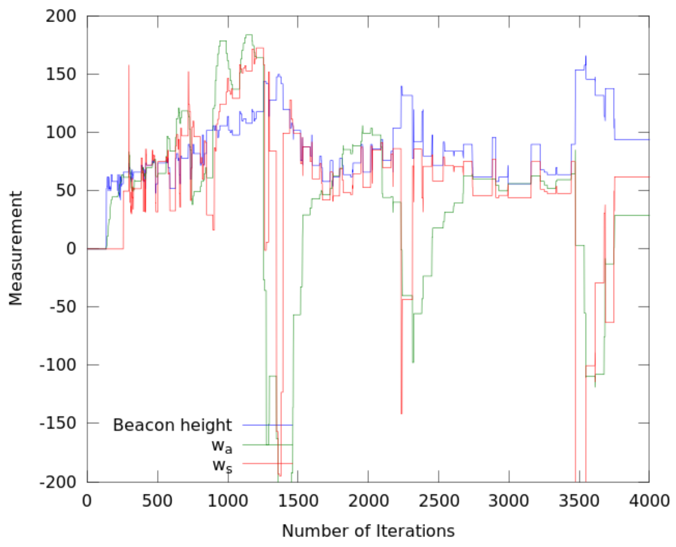

Figure 5: Testing of ASAMI on Nao during the class demo. The observation comes continuously, on average of 30 frames (iterations) per second. It is kidnapped at around 3,500th iteration.

5.3 Test on Nao: Second Attempt

I realized that making Nao observe while walking is not necessary, so I made the Nao observe while stationary. During the test, Nao will take an action, walk, and then stand still to observe the beacon. It keeps repeating this sequence. The result is in Figure 6. The beacon height in the figure is the

[1]Kidnapping means that while the robot is walking, you quickly put it to a different position. The robot would find its environment changes unpredictably, then it realizes its location changed.

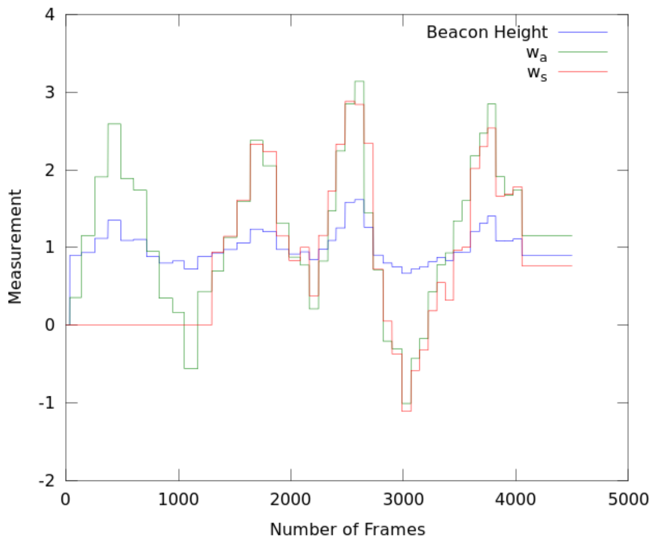

Figure 6: Testing of ASAMI on Nao. The observation comes only when it's in standing phase. So there is less update in the observation, and less noise compared to Figure 5. One iteration of ASAMI corresponds to one update in the figure, which corresponding to many frames on Nao (I re-scaled the height of beacon, so that they can be plotted in a same magnitude).

average of the observed beacon heights in the standing phase. Note that the x label in Figure 6 is different from that in Figure 5. In Figure 5, one iteration of ASAMI is exactly one frame on Nao. Here, one iteration of ASAMI takes many frames on Nao (one walking and one standing phase). This data is clearly more accurate because there is less noise in the observation.

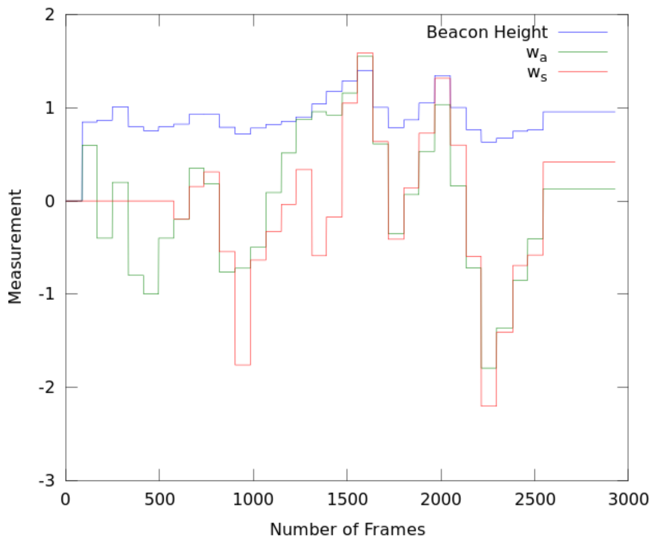

Figure 7: Testing of Strong ASAMI on Nao. The observation comes only when it's in standing phase - same as Figure 6. The robot tries to learn the action model first in the first 1,500 iterations, then continues to explore the state space (I re-scaled the height of beacon, so that they can be plotted in a same magnitude).

5.4 Test on Nao: using Strong ASAMI

Strong ASAMI is applied in this experiment. The assumption is different from the previous ones - Nao doesn't have the knowledge of the actions. The results are in Figure 7. The beacon height shows that the robot tries different actions first (within 1,500 iterations, or frames) and then moves forward and backward to explore the state space.

Another intended goal is to show that convergence in Figure 7 should be faster than that in previous experiments. Actually, the difference is not significant, at least for one-dimensional environment.

6 Discussion and Conclusion

The idea of action trials, which are the essential part in Strong ASAMI, is very similar to the idea of the multi-armed bandit problem [13]. In the multi-armed bandit problem, we choose one arm at each time and obtain a reward, with the objective to maximize the overall rewards. However, the goal of our problem should be to minimize the variance of all the actions. To solve this problem, I used an approach of choosing an action with the largest variance at each step. After such action, the variance of that action should decrease. The experiments have shown that this approach have worked, even though there is room for improvement. For example, as the actions are not mutually independent - they are picked up from a continuous space, getting one sample of an action can help us know its neighborhood.

We assume the action space as a continuous space in the experiments. However, it is possible that learning each discrete action independently would have a better performance (same metric as [1]). The action model and the sensor model are both assumed to be polynomial function in the experiments, but it has several constraints. First, degree selection should be determined, possibly using methods in [12]. But the model may not be a small-dimensional polynomial, so there might be too many parameters to estimate. Second, the inherent limitation of function approximation method is that local updates could have global effects. This is especially harmful in a noisy environment.

In this paper, I show that Strong ASAMI has its power in the environment where action learning is necessary. This is just for one-dimensional environment. So future work can be extending to testing in multi-dimensional environments.

7 Acknowledgment

This paper is revised from the report of the final project in Autonomous Robots class, instructed by Prof. Peter Stone, Department of Computer Science. He gave me valuable advices on this research. The first author of [10], Dan Stronger, also gave me comments when I proposed this project.

References

[1] Mazda Ahmadi and Peter Stone. Instance-based action models for fast action planning. In Ubbo Visser, Fernando Ribeiro, Takeshi Ohashi, and Frank Dellaert, editors, *RoboCup-2007: Robot Soccer World Cup XI*, volume 5001 of *Lecture Notes in Artificial Intelligence*, pages 1–16. Springer Verlag, Berlin, 2008.

[2] Samuel Barrett, Katie Genter, Yuchen He, Todd Hester, Piyush Khandelwal, Jacob Menashe, and Peter Stone. Ut austin villa 2012: Standard platform league world champions. In Xiaoping Chen, Peter Stone, Luis Enrique Sucar, and Tijn Van der Zant, editors, *RoboCup-2012: Robot Soccer World Cup XVI*, Lecture Notes in Artificial Intelligence. Springer Verlag, Berlin, 2013.

[3] Sonia Chernova, , and Abstract Legged Robots. Learning and using models of kicking motions for legged robots.

[4] Todd Hester and Peter Stone. Real time targeted exploration in large domains. In *The Ninth International Conference on Development and Learning (ICDL)*, August 2010.

[5] Pattie Maes. Modeling adaptive autonomous agents. *Artificial life*, 1(1_2):135–162, 1993.

[6] Rajeev Motwani. *Randomized algorithms*. Cambridge university press, 1995.

[7] Pierre-Yves Oudeyer and Frédéric Kaplan. Discovering communication. *Connection Science*, 18(2):189–206, 2006.

[8] Nadeesha Ranasinghe and Wei-Min Shen. Surprise-based learning for developmental robotics. In *Learning and Adaptive Behaviors for Robotic Systems, 2008. LAB-RS'08. ECSIS Symposium on*, pages 65–70. IEEE, 2008.

[9] Jürgen Schmidhuber. Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. *Connection Science*, 18(2):173–187, 2006.

[10] Daniel Stronger and Peter Stone. Towards autonomous sensor and actuator model induction on a mobile robot. *Connection Science*, 18(2):97–119, 2006. Special Issue on Developmental Robotics.

[11] Daniel Stronger and Peter Stone. Maximum likelihood estimation of sensor and action model functions on a mobile robot. In *IEEE International Conference on Robotics and Automation*, May 2008.

[12] Daniel Stronger and Peter Stone. Polynomial regression with automated degree: A function approximator for autonomous agents. *International Journal on Artificial Intelligence Tools*, 17(1):159–174, February 2008.

[13] Joannes Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In *Machine Learning: ECML 2005*, pages 437–448. Springer, 2005.