Action Awareness – Enabling Agents to Optimize, Transform, and Coordinate Plans

Freek Stulp
stulp@in.tum.de

Michael Beetz
beetz@in.tum.de

Intelligent Autonomous Systems Group, Department of Computer Science IX
Technische Universität München, Boltzmannstr. 3, D-85748, Garching bei Munich, Germany

ABSTRACT

As agent systems are solving more and more complex tasks in increasingly challenging domains, the systems themselves are becoming more complex too, often compromising their adaptivity and robustness. A promising approach to solve this problem is to provide agents with reflective capabilities. Agents that can reflect on the effects and expected performance of their actions, are more aware and knowledgeable of their capabilities and shortcomings.

In this paper, we introduce a computational model for what we call *action awareness*. To achieve this awareness, agents learn predictive action models from observed experience. This knowledge is then used to optimize, transform and coordinate plans. We apply this computational model to a number of typical scenarios from robotic soccer. Various experiments on real robots demonstrate that action awareness enables the robots to improve the performance of their plans substantially.

Categories and Subject Descriptors

I.2.9 [**Artificial Intelligence**]: Robotics—*autonomous vehicles*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*plan execution, formation, and generation*

General Terms

Computational Model

Keywords

learning action models, plan optimization

1. INTRODUCTION

Improvements in agent technology allows agent systems to solve more and more complex tasks. As the tasks and environments become more complex, so do the agent systems themselves. This often compromises the robustness and adaptivity of the system, leading to a shorter operational life-time. A promising approach to solve this problem is to provide the agents with reflective capabilities [2]. When the agents are aware of what they are doing, and why, they become more robust, and can adapt to new situations. We call this *action awareness*.

We investigate the issues of action awareness and its applications in the context of multi-robot control. Robotic soccer is a good example of a domain in which the agent's interactions with the environment are often very complex. Therefore, hand-coding the controller is intractable and failure-prone. However, by reasoning about their actions, robots are able to choose the appropriate action, and optimize its parameterization on-line themselves.

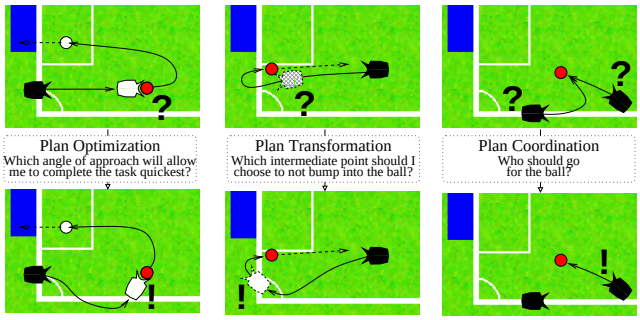

Figure 1: Three typical robotic soccer scenarios including control tasks requiring action awareness.

In this paper, we will propose a novel **computational model** for the acquisition and application of action awareness. It shows how action awareness can be acquired by **learning predictive action models** from observed experience (Section 2). We then explain how action awareness can be used to **optimize** (Section 3.1), **transform** (Section 3.2), and **coordinate** (Section 3.3) underspecified plans with highly parameterizable actions in the context of robotic soccer. While the individual contributions are not new, we do not know of a system implementing this as a comprehensive computational model, and applying it to autonomous robot control.

The computational model is depicted in Figure 2. To the left are the more or less 'traditional' action selection modules, and to the right are the novel modules that generate and exploit action awareness. When the robot has idle time, it learns prediction models for the actions in the ac-

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.

tion library. During operation time, action chains are generated. Three modules then take these chains, and optimize, transform, and coordinate them using the pre- and post-conditions, as well as the prediction models. The modified chains then simply replace the original ones, and are executed.

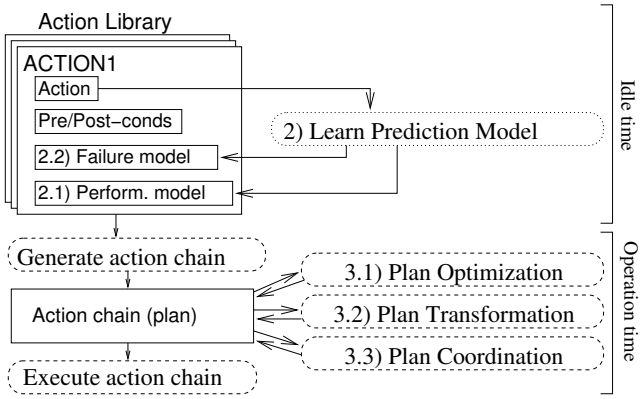

Figure 2: Computational Model

2. BECOMING ACTION AWARE: LEARNING PREDICTION MODELS

The robots achieve action awareness by learning action prediction models from observed experience. Given a specific scenario, these models should predict whether an action is likely to succeed, and what its expected performance will be. In this section, we will explain how temporal and failure prediction models were learned. The experiments in this paper were conducted with three customized Pioneer I robots of our RoboCup mid-size team the Agilo RoboCuppers. The robots use differential drive for locomotion, and a single forward facing CCD camera for state estimation.

2.1 Action Performance Prediction

Being aware of an action's expected performance is valuable knowledge. In many robotic domains, execution time is an important performance measure. Therefore we learn temporal prediction models. For our actions `goToPose` and `dribbleBall`, the model should predict the expected time of navigating from a start to a goal pose with this action. This model is acquired by training a model tree with data acquired on real robots. The training examples for the model tree were acquired by randomly choosing goal destination on the field, executing the `goToPose` or `dribbleBall` action, and measuring the time it took for the approach. In [6] we determined that 40 minutes operation time on a real robot yields enough examples to acquire an accurate temporal prediction model. The mean absolute error between the actual an predicted approach time for the `goToPose` and `dribbleBall` actions are 0.18s and 0.22s respectively.

2.2 Action Failure Prediction

The `goToPose` action can often be used well to approach the ball. However, in some situations it will bump into the ball before achieving the desired orientation, as can be seen in Figure 1. The second performance model we have learned predicts whether executing `goToPose` will lead to a collision with the ball or not. This model was learned by training a decision tree on data acquired in simulation. The training examples for this tree were acquired by having the robot executed `goToPose` a thousand times, with random initial and destination poses, the ball always positioned at the destination pose. For each run, we record if the robot had a `Collision` with the ball before reaching its desired position and orientation. If not, the run was a `Success`. On a test set, the learned decision tree predicts collisions correctly in almost 90%.

3. APPLYING PREDICTION MODELS TO AGENT CONTROL

In this section we will show how prediction models can be used to optimize, transform and coordinate plans.

3.1 Plan Optimization

Actions and action chains are often underspecified in plans. Take the example in Figure 3. The robot's plan is to approach the ball, and dribble it to the specified point. If it approached the ball as fast as possible, it would end up in the position on the top field in the image. This is an unfortunate position from which to start dribbling towards the goal. The problem is that, according to the planner, being at the ball is considered sufficient for dribbling the ball, and the angle of approach is considered to be irrelevant for the consecutive dribbling action. So, the intermediate state is under-specified, and we are allowed to choose any state from the set of intermediate states that the conditions deem valid.

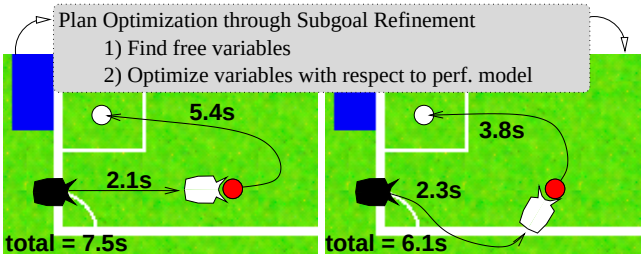

Figure 3: Overview of plan optimization.

What we would like the robot to do is to go to the ball *in order* to dribble it towards the goal afterwards. The robot should, as depicted in the right field in Figure 3, perform the first action sub-optimally in order to achieve a much better position for executing the second plan step. Our action-aware plan execution system can determine optimal intermediate states on the fly, using predictive models. We call this *subgoal refinement*.

First of all, the agent has to determine which variables can be used for optimization. That is, which variables are not bound by the plan. In this example, the only free variable is the angle of approach. For an explanation of how free variables are determined automatically, we refer to [4]. To optimize the action chain, we will have to find those values for the free variables for which the performance of the action chain is the highest. This overall performance is estimated by summing over the temporal prediction models of all actions that constitute the action chain. In Figure 3,

the predicted execution time of the first action in the left situation is 2.1s. However, the total time is 7.5s, because the second action takes 5.4s for this angle. Although the first action is executed very fast, this is not the optimum *overall* performance. By performing the first action suboptimally, an minimum overall predicted performance of 6.1s can be achieved, for an angle of 59°. This particular example yields a performance increase of 38%. The mean increase over a 1000 examples with random robot, ball and final goal positions was 12%.

Without subgoal refinement, the transitions between actions were very abrupt. In general, these motion patterns are so characteristic for robots that people trying to imitate robotic behavior will do so by making abrupt movements between actions. In contrast, one of the impressive capabilities of animals and humans is their capability to perform chains of actions in optimal ways and with seamless transitions between subsequent actions. It is interesting to see that requiring optimal performance can implicitly yield smooth transitions in robotic and natural domains, even though smoothness in itself is not an explicit goal in either domain.

3.2 Plan Transformation

In planning systems, the pre-conditions determine a set of states in which the action can be executed. Frequently, actions can be reused to perform other tasks than those they were originally designed for. For instance, the `goToPose` action can also be used to approach the ball. However, since `goToPose` does not consider the ball, it will sometimes bump into it before achieving the goal pose. In Section 2.2 we saw that a decision tree could be learned to accurately predict when this happens. In a sense, this decision tree can be seen as a learned pre-condition for a new `approachBall` action. This pre-condition can be used to determine when a plan to approach the ball will fail, and transform it so that a collision is avoided. Exactly how this is done is discussed in more detail in [5].

3.3 Plan Coordination

Many complex application tasks require two or more agents to cooperate in order to solve the task. A key aspect of these systems is that multiple agents share the same workspace, and can therefore not abstract away from the actions of other robots. Therefore, agents must not only be aware of their own actions, but also of those of others. Humans are very good at performing joint actions in shared workspaces, because they are very aware of the actions of others. Humans achieve this by inferring the intentions of others. Once the beliefs and desires of the cooperating party are known, we imagine what we would do in that situation. In contrast, coordination in multi-agent systems is usually achieved by extensive communication of intentions or utilities [3].

We believe that there are many domains in which implicit coordination is essential. Rescue robotics and autonomous vehicles operating in traffic are examples of domains in which robust communication is not guaranteed, but where correct coordination and action anticipation is a matter of life and death. We also consider implicit coordination to be essential for natural interaction between robots and humans. It cannot be expected of humans to continuously communicate their intentions. Instead, the robot must be able to anticipate a human's intentions, based on predictive models of human behavior.

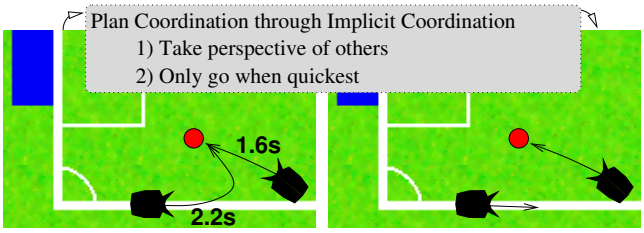

Figure 4: Overview of plan coordination.

A typical coordination task from the robotic soccer domain is to regain ball possession. Acquiring ball possession is a goal for the team as a whole, but it can be achieved by only one of the field players. Of course, the robots must agree upon which robot will approach the ball. The intuitive underlying rule is that only the robot who is quickest to the ball should approach it. The key to achieving implicit coordination is *perspective taking*. If the robots have estimates of each other's states, they can predict ball approach time for the others just as well as for themselves. In a sense, the awareness of their own action also provides them with awareness of actions of others. Once each robot has computed each robot's predicted ball approach time, each robot can determine if it is the fastest or not.

Our experiments have shown that the robots reach almost perfect agreement on this (99%), without communication [6]. The robot on which agreement is reached is indeed the quickest in 96% of the experiments.

Acknowledgments

The work described in this paper is partially funded by the German Research Foundation (DFG) under the contract RA 359/7-2, as part of the Priority Research Program SPP1125 *Cooperative Teams of Mobile Robots in Dynamic Environments*.

4. REFERENCES

[1] T. Belker, M. Hammel, and J. Hertzberg. Learning to optimize mobile robot navigation based on HTN plans. In *Proceedings of the International Conference on Robotics and Automation (ICRA03)*, 2003.

[2] R. Brachman. Systems that know what they're doing. *IEEE Intelligent Systems*, 2002.

[3] B. P. Gerkey and M. J. Matarić. Multi-robot task allocation: Analyzing the complexity and optimality of key architectures. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2003.

[4] F. Stulp and M. Beetz. Optimized execution of action chains using learned performance models of abstract actions. In *Proc. of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.

[5] F. Stulp and M. Beetz. Tailoring action parameterizations to their task contexts, 2005. IJCAI Workshop "Agents in Real-Time and Dynamic Environments".

[6] F. Stulp, M. Isik, and M. Beetz. Implicit coordination in robotic teams using learned prediction models. Accepted for the IEEE International Conference on Robotics and Automation (ICRA), 2006.