

---

# Situation Dependent Spatial Abstraction in Reinforcement Learning Based on Structural Knowledge

---

Lutz Frommberger

LUTZ@SFBTR8.UNI-BREMEN.DE

SFB/TR 8 Spatial Cognition, University of Bremen, Enrique-Schmidt-Str. 5, 28359 Bremen, Germany

## Abstract

State space abstraction reduces the size of a representation by factoring out details that are not relevant for solving a task at hand. But even in abstract representations not every detail is relevant in any situation. In cases where the structure of the environment only allows for one particular action selection, all information that does not relate to the structure can be omitted. We present a method to identify such cases in a reinforcement learning setting and abstract from non-structural details when appropriate to shrink the state space and allow for knowledge reuse. A significant performance improvement of this approach is demonstrated in a goal-directed robot navigation task.

## 1. Introduction

In human cognition, abstraction is an important concept. It enables humans to conceptualize situations by building categories that enable them to cope with overly detailed circumstances. It is widely accepted that abstraction is also a key technique for applying reinforcement learning (RL) to more complex tasks.

Abstraction of the state space has been used frequently with RL. Instead of confronting the learning system with all the details the sensors offer, an abstract representation is used to describe the state space. Not all pieces of information are required for solving the task. A central question of abstraction is which information to retain and which to drop.

Moreover, not everything is important under any circumstances. Depending on the situation, different levels of abstraction can be appropriate. While a detail

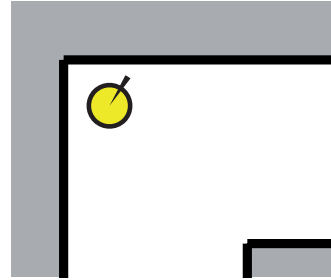


Figure 1. A robot in a corner of a corridor: A right turn is the only reasonable action here to prevent a collision.

may be critical for action selection in one situation, it can be completely irrelevant in another. Especially, it can be the case that the structure of the state space alone enforces a particular action. For example, when a robot navigates too closely into a corner such that any action besides a right turn would lead to a collision with a wall, this right turn is the only reasonable action (Figure 1). This action is enforced by the world’s structure, which does not allow for a change of decisions before the robot reaches a more open space. Before that *decision point* the knowledge of the concrete whereabouts of the turn within the world does not matter at all for action selection—actions are *only* induced by the world’s structure at this point in time.

In this paper we present SITSA, a method to abstract from irrelevant details of the state representation depending on the structure of the current situation based on knowledge gained in previous tasks. SITSA is applicable to a special class of abstract representations, so-called *structure space aspectualizable* state spaces that explicate the structure of the environment within the feature vector.

In the next section we introduce structure space aspectualizable state spaces. SITSA is described in Section 3 and evaluated in Section 4 within a simulated robot navigation scenario. Section 5 discusses the presented method before an overview on related work is given in Section 6. The paper ends with a conclusion.

## 2. Structure Space Aspectualizable State Spaces

We suppose the problem to solve to be represented as a Markov Decision Process (MDP)  $\langle \mathcal{S}, \mathcal{A}, T, R \rangle$  with a state space  $\mathcal{S}$ , a set of available actions  $\mathcal{A}$ , a transition function  $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$  denoting a probability distribution that the invocation of action  $a$  at state  $s$  will result in state  $s'$ , and a reward function  $R : \mathcal{S} \rightarrow \mathbb{R}$  assigning a reinforcement signal to any  $s \in \mathcal{S}$ . In this work, we assume both  $\mathcal{S}$  and  $\mathcal{A}$  to be discrete.

Under state space abstraction the resulting system usually is not Markov anymore and forms a Partially Observable Markov Decision Process (POMDP). However, approaches that solve a POMDP with reinforcement learning as if it was an MDP are often successful if the abstraction closely represents the dynamics of the underlying state space. We also follow this approach in this work.

For being able to adapt reinforcement learning to more complex tasks it is necessary to discover the structure of the world to abstract from its details (Thrun & Schwartz, 1995). The structure of a state space is built by recurring properties in the world that are significant for the task at hand. In indoor robot navigation, for example, this structure is induced by the position of walls which induce sensible paths inside the world which the moving agent is supposed to follow.

State space design for RL is usually driven by the designer’s domain knowledge about the task to solve. Within this process, we argue for explicitly modeling the structure of an environment as set of features within the state space representation. We call the space generated by these features *structure space* ( $\mathcal{S}_S$ ). If these features are a subset of the overall state space representation  $s = (s_1, s_2, \dots, s_n)$ , we call this a *structure space aspectualizable* state space, and a function  $\text{ssd} : \mathcal{S} \rightarrow \mathcal{S}_S$  returns the features of structure space. The remaining features generate *task space* ( $\mathcal{S}_T$ ) and are delivered by a function  $\text{tsd} : \mathcal{S} \rightarrow \mathcal{S}_T$ . W.l.o.g., we assume that the state space  $\mathcal{S}$  is represented as a Cartesian product of task and structure space representation  $\mathcal{S}_T \times \mathcal{S}_S$  such that we have a state representation  $s = (\text{tsd}(s), \text{ssd}(s))$ .

The distinction between task and structure space relates to the framework of problem space and agent space (Konidaris & Barto, 2006). An agent space retains the same semantics over different related tasks and thus can be used for knowledge transfer between them. In particular, structure space is an agent space for tasks sharing the same structure like, for example, two different office environments.

Structure space aspectualizable abstractions can be achieved by applying abstraction to the percepts of the world (Frommberger & Wolter, 2008). They allow for easy access to the represented structural knowledge. In previous work it has been shown that this allows for generalization within the same and across tasks and leads to a significant improvement in learning performance (Frommberger, 2008). In the following, we present a method to further abstract from irrelevant details depending on the current structural circumstances that operates on structure space aspectualizable state spaces.

## 3. Situation Dependent Spatial Abstraction

In many tasks we encounter situations where the choice of actions only depends on structural information. In other words: When structure space strictly enforces a certain action selection, task space information becomes irrelevant for the control process. We say that these *non-decision states* have a *non-decision structure*.

### 3.1. Identifying Non-Decision Structures

In autonomous learning we usually cannot identify non-decision structures in advance. To detect them, we resort to knowledge gained in a previous task (which we refer to as the *source task*) that shares the same structure space. Let us say that we used Q-learning (Watkins, 1989) to learn a Q-function  $Q(s, a)$  in the source task with a structure space aspectualizable state space  $\mathcal{S}$ .

In any non-decision state, the agent is certain which action to take, as there is only one reasonable choice that leads to a high reward expectation. That means that we encounter the Q-value for one action being significantly higher than those for any other action. In the example of the robot in the corner given above, the Q-values for anything but turning right would be rather low because the action would result in a collision, but the Q-value for turning right would be quite high because this action can enable the robot to reach its goal in the end.

To identify such non-decision structures, we sum up all Q-values for states with an identical structure space representation. Because higher Q-values can be encountered at states near a goal state they are normalized with regard to the absolute value of the maximum Q-value at each state such that they are all in  $[-1, 1]$ . We average over all states where we have knowledge of, that is, where the Q-value is not 0 for all  $a \in \mathcal{A}$ . This

leads to a new function  $Q_S : \mathcal{S}_S \rightarrow \mathbb{R}$  with  $\omega \in \mathcal{S}_S$ :

$$Q_S(\omega, a) = \sum_{y \in \mathcal{S}_T} \frac{Q((y, \omega), a)}{\max_{b \in \mathcal{A}} |Q((y, \omega), b)|} \cdot \frac{1}{|\{x \in \mathcal{S}_T | Q((x, \omega), a) \neq 0\}|}$$

We now again iterate over structure space and look at the normalized averaged values of  $Q_S$  to determine the difference between the highest and second highest values of  $Q_S$  for different actions, that is, to check whether one action leads to outstandingly high Q-values in the original Q-function. This defines a *decision certainty function*  $\text{cert} : \mathcal{S}_S \rightarrow \mathbb{R}$ :

$$\text{cert}(\omega) = \frac{1}{|\mathcal{A}|} (\max_{b \in \mathcal{A}} Q_S(\omega, b) - \text{smax}_{b \in \mathcal{A}} Q_S(\omega, b))$$

with  $\text{smax}$  denoting a function for “second highest value” to ease readability.

An algorithmic description of this procedure is given in Algorithm 1. It loops over all sensations perceived in the source task, so it is necessary to keep track of all encountered observations there. The runtime of the algorithm is linear to the number of observations.

A very high value of  $\text{cert}(\omega)$  denotes that one specific action was definitely preferred at states with a certain structural representation  $\omega$  and thus is an indicator for  $\omega$  being a non-decision structure.

### 3.2. SITSA: Structure Induced Task Space Aspectualization

Using Algorithm 1 we first identify a set  $\mathcal{S}_{\text{NDesc}} \subset \mathcal{S}_S$  of obvious non-decision structures. It contains all structures where  $\text{cert}(\text{ssd}(s)) > c_{\text{min}}$ , with  $c_{\text{min}} \in \mathbb{R}$  being a pre-defined constant that defines the minimal needed decision certainty that qualifies for a non-decision structure.

We abstract from task space information at non-decision states, because it is not needed there. This is achieved by enforcing the same unique task space representation at any state with a non-decision structure such that  $\text{tsd}(s) = \perp$  for all  $s \in \mathcal{S}_{\text{NDesc}}$  (with  $\perp$  denoting a pre-defined “empty” representation). All other states remain unmodified. Formally, we obtain an abstraction function  $\Theta : \mathcal{S}_T \times \mathcal{S}_S \rightarrow \mathcal{S}_T \cup \{\perp\} \times \mathcal{S}_S$ :

$$\Theta(s) = \begin{cases} (\perp, \text{ssd}(s)) & \text{if } \text{ssd}(s) \in \mathcal{S}_{\text{NDesc}} \\ s & \text{else} \end{cases}$$

$\Theta$  abstracts from task space features depending on the situation that is given by knowledge about the structure of the environment and thus shrinks state space

---

#### Algorithm 1 Determining Decision Certainties

---

**Require:**  $\mathcal{S}_v$  contains all perceived observations

**Require:**  $T$  is an empty associative table

$M_S \leftarrow \emptyset$  ▷ set to store detected structures

**for all**  $s \in \mathcal{S}_v$  **do**

$i \leftarrow 0$

**for all**  $a \in \mathcal{A}$  **do** ▷ Q-values for each action

$v[i] \leftarrow Q(s, a)$

$i \leftarrow i + 1$

**end for**

$m \leftarrow \max_j |v[j]|$  ▷ max. for weighting factor

**for all**  $a \in \mathcal{A}$  **do**

$Q_S(\text{ssd}(s), a) \leftarrow Q_S(\text{ssd}(s), a) + \frac{1}{m} Q(s, a)$

**end for**

$M_S \leftarrow M_S \cup \text{ssd}(s)$  ▷ store processed structure

$T[\text{ssd}(s)] \leftarrow T[\text{ssd}(s)] + 1$  ▷ count appearance

**end for**

**for all**  $\omega \in M_S$  **do** ▷ average Q-values

$Q_S(\omega) \leftarrow Q_S(\omega) / T[\omega]$

**end for**

**for all**  $\omega \in M_S$  **do** ▷ determine certainties

$m_1 = \max_{b \in \mathcal{A}} Q_S(\omega, b)$  ▷ highest

$m_2 = \text{smax}_{b \in \mathcal{A}} Q_S(\omega, b)$  ▷ second highest

$\text{cert}(\omega) \leftarrow \frac{1}{|\mathcal{A}|} (m_1 - m_2)$

**end for**

---

size. At any non-decision state now only structural information is represented and anything else omitted. Thus, we encounter a generalization effect, because structurally identical locations (that is: states  $s$  and  $s'$  with  $\text{ssd}(s) = \text{ssd}(s')$ ) now share the same abstract state  $(\perp, \text{ssd}(s))$  and thus learned knowledge applies to various places within  $\mathcal{S}$ . We call this method *structure induced task space aspectualization* (SITSA).

## 4. Experimental Results

### 4.1. A Goal Directed Robot Navigation Task

Our evaluation scenario for SITSA is a simulated indoor navigation task where an autonomous robot learns to find a specified location in an unknown environment, the goal area.

The robot is able to perceive and uniquely identify walls around it within a certain maximum range.<sup>1</sup> It is capable of performing three different actions: moving forward and turning a few degrees both to the left and

---

<sup>1</sup>The property of uniquely identifying walls is a strong assumption. We note that the presented approach also works well with point-based landmarks distributed over the environment. However, we chose the walls as landmarks because in this case the number of landmark scales homogeneously with the complexity of the environment.

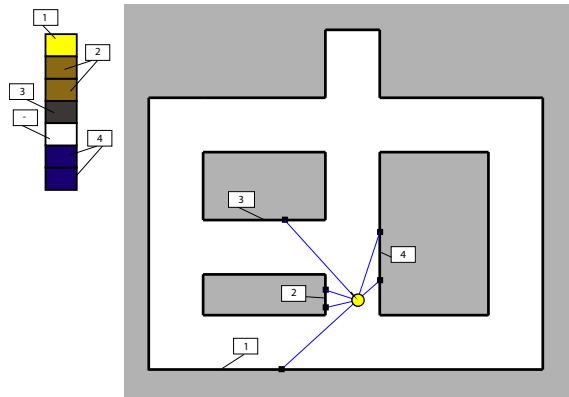


Figure 2. The navigation task: a robot in a simplified simulated office environment with uniquely distinguishable walls. The lines departing from the robot visualize landmark scans. Detected landmarks are depicted in the upper left.

to the right. Both turns include a small forward movement, and some noise is added to all actions. There is no built-in collision avoidance or any other navigational intelligence provided. See Figure 2 for a look on the simulation testbed.

## 4.2. State Space Representation

In this section, we shortly introduce the abstract state space representation we use to learn a policy in the described scenario. *Landmark-enriched RLPR* (Frommberger, 2008) is a structure space aspectualizable representation tailored for indoor robot navigation tasks.

### 4.2.1. REPRESENTING TASK SPACE

To represent task space it is sufficient to encode an approximate qualitative position of the agent within the world. We do this by representing a circular order of detected landmarks. We regard a sequence of detected walls  $c_i$  at 7 discrete angles around the robot and encode each wall by a number starting with 1:  $\text{tsd}(s) = (c_1, \dots, c_7)$ .

### 4.2.2. REPRESENTING STRUCTURE SPACE

To encode structure space, we use the *relative line position representation* (RLPR). It benefits from the domain knowledge that the structuring elements for navigation inside of buildings are walls. RLPR encodes the position of line segments perceived by the agent’s sensory system relative to its moving direction. The space around the agent is partitioned into bounded and unbounded regions  $R_i$  (see Figure 3). Two functions  $\bar{\tau} : \mathbb{N} \rightarrow \{0, 1\}$  and  $\bar{\tau}' : \mathbb{N} \rightarrow \{0, 1\}$  are defined:  $\bar{\tau}(i)$  denotes whether there is a line segment detected

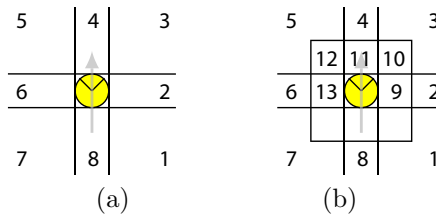


Figure 3. RLPR: Neighboring regions around the robot in relation to its moving direction. Note that the regions in the immediate surroundings (b) are proper subsets of  $R_1, \dots, R_8$  (a).

within a sector  $R_i$  and  $\bar{\tau}'(i)$  denotes whether a line spans from a neighboring sector  $R_{i+1}$  to  $R_i$ .

$\bar{\tau}_i$  is used for bounded sectors in the immediate vicinity of the agent ( $R_9$  to  $R_{13}$  in Figure 3(b)). Objects that appear there have to be avoided in the first place. The position of detected line segments in  $R_1$  to  $R_8$  (Figure 3(a)) is interesting information to be used for general orientation and mid-term planning, so  $\bar{\tau}'$  is used for  $R_1$  to  $R_8$ . Summed up, the structure space representation  $\text{ssd}(s)$  is defined as  $(\bar{\tau}'(R_1), \dots, \bar{\tau}'(R_6), \bar{\tau}(R_9), \dots, \bar{\tau}(R_{13}))$ .

Landmark-enriched RLPR is the concatenation of task and structure space representation:

$$s = (c_1, \dots, c_7, \bar{\tau}'(R_1), \dots, \bar{\tau}'(R_6), \bar{\tau}(R_9), \dots, \bar{\tau}(R_{13}))$$

This representation has shown to outperform approaches operating on the underlying MDP regarding learning speed and robustness (Frommberger, 2008).

## 4.3. Experimental Setup and Results

For this experiment we used Watkins’  $Q(\lambda)$  algorithm (Watkins, 1989). During training, the agent uses an  $\epsilon$ -greedy policy with  $\epsilon = 0.2$  for exploration. It started from 1900 starting positions randomly distributed over the corridors. Every 500 episodes, the success was evaluated on a cross-validation set of 100 starting positions disjoint from the ones used for training. A step size of  $\alpha = 0.05$ , a discount factor of  $\gamma = 0.99$ , and  $\lambda = 0.9$  was used in all the trials. A positive reward is given when a goal area is reached and a negative one if the agent collides with a wall. A learning episode ends when the agent reaches the goal state, collides with a wall, or after a certain number of actions. In this setting, we apply SITSA to abstract from landmark information in le-RLPR at non-decision points.

For determining non-decision structures for SITSA, a successful policy for a goal finding task in the world depicted in Figure 2 was analyzed using Algorithm 1 and a set  $\mathcal{S}_{\text{NDesc}}$  of 110 non-decision structures has been de-

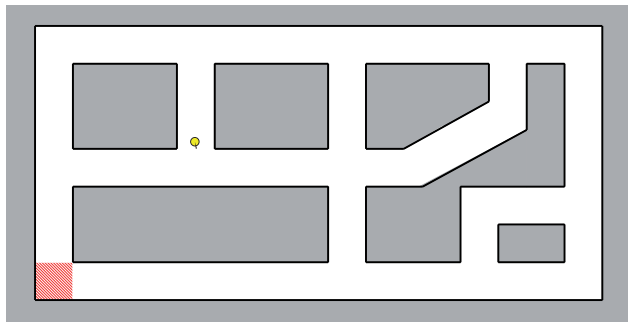


Figure 4. Screenshot: The simulated environment used for evaluation. The goal area is marked in the lower left corner.

rived from it. The evaluated learning task then took place in the larger environment depicted in Figure 4.

Figure 5 shows the goal finding success with and without the use of SITSA averaged over 15 different trials. When using the situation dependent abstraction SITSA offers, the learning speed is improved especially in the early learning phase. With SITSA, a stable 95% success is reached after 6,500 episodes while it takes 15,000 episodes without.

## 5. Discussion

The experiment described in the previous section indicates a significant improvement in learning performance. One prerequisite for this is the availability of sufficient starting positions within the environment. When using very few starting positions (as, for example, 20 instead of 1,900), we occasionally experienced to run into the exploration-exploitation dilemma: Because the agent generalizes very fast it acquires a sensible behavior early. In the following, the agent mainly follows this behavior while learning which leads to an insufficient exploration. A more sophisticated exploration method than  $\epsilon$ -greedy policies might be needed here.

Another prerequisite for identifying non-decision structures as described in Section 3.1 is that the consequences of the actions  $a \in \mathcal{A}$  that are used in the source task have to be considerably different. This has been the case in the presented scenario with only 3 actions. But if we had finer grained actions as, for example, several differently strong turns, non-decision structures might not have been identified successfully. Thus, it is reasonable to restrict to few, but different actions in the source task. In the task where SITSA is applied, a finer granularity of actions can then be made available, and different actions can be favored for non-decision states than learned in the source task.

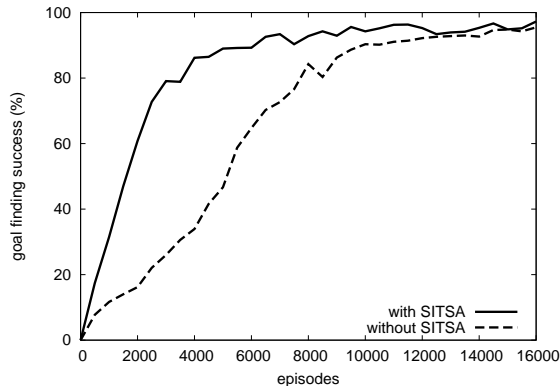


Figure 5. Comparison: Goal finding success with SITSA (straight line) and without (dotted line). With SITSA, learning is considerably faster.

Finally, SITSA, as presented in this paper, needs a discrete state space, because it iterates over different state space representations. Its main purpose is to add to an existing abstract discrete representation. But SITSA can be adapted to work with continuous state spaces as well if a value function approximation is used that allows to assign a representative with the same internal representation to any observed state. For example, this is the case for value function approximation with CMACs (Sutton, 1996).

## 6. Related Work

One of the earliest methods for state space abstraction is the Parti-Game algorithm (Moore & Atkeson, 1995) that automatically creates an abstract state space by using adaptive partitions. This idea has been followed by various other work (Dean & Givan, 1997; Munos & Moore, 1999; Reynolds, 2000, for example). Usually, these approaches require a model of the state space or a successful near-optimal policy working on an initial state space partition.

Domain knowledge has frequently been incorporated for knowledge reuse within the task. Glaubius et al. use a value function approximation based on manifolds to reuse experience across similar parts of the state space (Glaubius et al., 2005). Relational policies for spatial environments and conditions under which they can be approximately relocated within the learning task have shown to speed up learning (Lane & Wilson, 2005). Porta and Celaya introduced *partial views* and *partial rules* operating on them (Porta & Celaya, 2005). As in SITSA, only a reduced fraction of features is regarded. All these approaches need the a-priori formulation of rules or maps to be applied.

Experience from previous tasks has been used to achieve temporal abstraction, for example, in the SKILLS algorithm (Thrun & Schwartz, 1995) or to create portable options that have been learned in agent space (Konidaris & Barto, 2007). The distinction of problem and agent space has also been used to learn shaping rewards to speed up learning in a new task (Konidaris & Barto, 2006).

## 7. Conclusion

In this paper we introduced SITSA, a method to abstract from irrelevant details in the state space representations depending on the structure of the actual situation. In non-decision states where the structure of the environment only allows for one successful action we can ignore all information that does not describe the state's structure. We have shown that this is possible in structure space aspectualizable state spaces, where this can be achieved by factoring out the corresponding features. Thus, SITSA creates a single abstract state for all states with the same non-decision structure which shrinks the state space and allows for knowledge reuse within the task.

We presented an algorithm to identify non-decision structures by analyzing the Q-function of a previous learning task in a state space sharing the same structure. It averages over the corresponding Q-values to identify structures where one action is significantly preferred over the others. We could show that the use of SITSA enables a significant increase in learning performance in a simulated goal-directed robot navigation task.

## Acknowledgments

This work was supported by the DFG Transregional Collaborative Research Center SFB/TR8 "Spatial Cognition". Funding by the German Research Foundation (DFG) is gratefully acknowledged.

## References

- Dean, T., & Givan, R. (1997). Model minimization in markov decision processes. *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI)* (pp. 106–111).
- Frommberger, L. (2008). Learning to behave in space: A qualitative spatial representation for robot navigation with reinforcement learning. *International Journal on Artificial Intelligence Tools*, 17, 465–482.
- Frommberger, L., & Wolter, D. (2008). Spatial abstraction: Aspectualization, coarsening, and conceptual classification. In *Spatial cognition VI: International conference spatial cognition*, 311–327.
- Glaubius, R., Namihira, M., & Smart, W. D. (2005). Speeding up reinforcement learning using manifold representations: Preliminary results. *Proceedings of the IJCAI Workshop "Reasoning with Uncertainty in Robotics"*.
- Konidaris, G. D., & Barto, A. G. (2006). Autonomous shaping: Knowledge transfer in reinforcement learning. *Proceedings of the International Conference on Machine Learning (ICML 2006)* (pp. 489–49).
- Konidaris, G. D., & Barto, A. G. (2007). Building portable options: Skill transfer in reinforcement learning. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.
- Lane, T., & Wilson, A. (2005). Toward a topological theory of relational reinforcement learning for navigation tasks. *Proceedings of the International Florida Artificial Intelligence Research Society Conference (FLAIRS-2005)*.
- Moore, A. W., & Atkeson, C. G. (1995). The partigame algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning*, 21, 199–233.
- Munos, R., & Moore, A. (1999). Variable resolution discretizations for high-accuracy solutions of optimal control problems. *Proceedings of the International Conference on Artificial Intelligence (IJCAI)* (pp. 1348–1355).
- Porta, J. M., & Celaya, E. (2005). Reinforcement learning for agents with many sensors and actuators acting in categorizable environments. *Journal of Artificial Intelligence Research*, 23, 79–122.
- Reynolds, S. I. (2000). Adaptive resolution model-free reinforcement learning: Decision boundary partitioning. *Proceedings of the International Conference on Machine Learning (ICML)*.
- Sutton, R. S. (1996). Generalization in reinforcement learning: Successful examples using sparse tile coding. In *Advances in neural information processing systems: Proceedings of the 1995 conference*.
- Thrun, S., & Schwartz, A. (1995). Finding structure in reinforcement learning. In *Advances in neural information processing systems: Proceedings of the 1994 conference*.
- Watkins, C. (1989). *Learning from delayed rewards*. Doctoral dissertation, Cambridge University.