Spatial Abstraction: Aspectualization, Coarsening, and Conceptual Classification

Lutz Frommberger and Diedrich Wolter

SFB/TR 8 Spatial Cognition
Universität Bremen
Enrique-Schmidt-Str. 5, 28359 Bremen, Germany
{lutz,dwolter}@sfbtr8.uni-bremen.de

Abstract. Spatial abstraction empowers complex agent control processes. We propose a formal definition of spatial abstraction and classify it by its three facets, namely aspectualization, coarsening, and conceptual classification. Their characteristics are essentially shaped by the representation on which abstraction is performed. We argue for the use of so-called aspectualizable representations which enable knowledge transfer in agent control tasks. In a case study we demonstrate that aspectualizable spatial knowledge learned in a simplified simulation empowers strategy transfer to a real robotics platform.

Keywords: abstraction, knowledge representation, knowledge transfer.

1 Introduction

Abstraction is one of the key capabilities of human cognition. It enables us to conceptualize the surrounding world, build categories, and derive reactions from them to cope with a certain situation. Complex and overly detailed circumstances can be reduced to much simpler concepts and not until then it becomes feasible to deliberate about conclusions to draw and actions to take.

Certainly, we want to see such abstraction capabilities in intelligent artificial agents too. This requires us to implement abstraction principles in the knowledge representation used by the artificial agent. First of all, abstraction is a process transforming a knowledge representation. But how can this process be characterized? We can distinguish three different facets of abstraction. For example it is possible to regard a subset of the available information only, or the level of detail of every bit of information can be reduced, or the available information can be used to construct new, more abstract entities. Intuitively, these types of abstraction are different and lead to different results as well. Various terms have been coined for abstraction principles, distributed over several scientific fields like cognitive science, artificial intelligence, architecture, linguistics, geography, and many more. Among others we find the terms *granularity* [1,2], *generalization* [3], *schematization* [4,5], *idealization* [5], *selection* [5,6], *amalgamation* [6], or *aspectualization* [7]. Unfortunately, some of these terms define overlapping concepts, different ones sometimes have the same meaning, or a single term is used for different concepts. Also, these terms are often not distinguished in an exact manner or only defined by giving examples.

In this work we take a formal view from a computer scientist's perspective. We study abstraction as part of knowledge representation. Our primary concern is representation of spatial knowledge, yet we aim at maintaining a perspective as general as possible, allowing adaption to other domains. Spatial information is rich and can be conceptualized in a multitude of ways, making its analysis challenging as well as relevant to applications. Handling of spatial knowledge is essential to all agents acting in the real world.

One contribution of this article is a formal definition of abstraction processes: *aspectualization*, *coarsening*, and *conceptual classification*. We characterize their properties and investigate into the consequences that arise when using abstraction in agent control processes. Applying the formal framework to a real application in robot navigation exemplifies its utility. Appropriate use of abstraction allows knowledge learned in a simplified computer simulation to be transferred to a control task with a real autonomous robot. Aspectualizable knowledge representations which we introduce and promote in this paper play a key role. The exemplified robot application shows how abstraction principles empower intelligent agents to transfer decision processes, thereby being able to cope with unfamiliar situations. Put differently, aspectualizable knowledge representations enable knowledge transfer.

This paper is organized as follows: In Section 2 we give our definition of the spatial abstraction paradigms and discuss the role of abstraction in knowledge representation and its utility in agent control tasks. Section 3 covers the case study of learning navigational behavior in simulation and transferring it to a real robot. The paper ends with a discussion of formal approaches to spatial abstraction and their utility (Section 4) and a conclusion.

2 A Formal View on Facets of Abstraction

The term *abstraction* is etymologically derived from the Latin words "abs" and "trahere", so the literal meaning is "drawing away". However, if we talk about abstraction in the context of information processing and cognitive science, abstraction covers more than just taking away something, because it is not intended merely to reduce the amount of data. Rather, abstraction is employed to put the focus on the relevant information. Additionally, the result is supposed to generalize and to be useful for a specific task at hand. We define abstraction as follows:

Definition 1. *Abstraction is the process or the result of reducing the information of a given observation in order to achieve a classification that omits all information that is irrelevant for a particular purpose.*

We first concentrate on information reduction. Let us say that all potential values of a knowledge representation are elements of a set $\mathcal{S}$ which can be regarded as a Cartesian product of features from different domains: $\mathcal{S} = \mathcal{D}_1 \times \mathcal{D}_2 \times \ldots \times \mathcal{D}_n$. We call $s = (s_1, \ldots, s_n) \in \mathcal{S}$ a *feature vector*, and every s_i is a *feature*. $\mathcal{S}$ is also called *state space* and its elements are *states*.

Abstraction is a non-injective function $\kappa : \mathcal{S} \to \mathcal{T}$ mapping the source space $\mathcal{S}$ to a target set $\mathcal{T}$. Non-injectiveness is important as otherwise no reduction (n:1-mapping) is possible. In the case of $\mathcal{S}$ being finite it holds that $|\mathcal{S}| > |\text{Image}(\kappa)|$. Without loss of generality we will assume in the following, simply to ease readability, that all domains $\mathcal{D}$ are of the same kind: $\mathcal{S} = D^n$.

In the following we will formally classify abstraction into three different categories: *aspectualization*, *coarsening*, and *conceptual classification*.

2.1 Aspectualization

Aspects are semantic concepts. They are pieces of information that represent certain properties. For example, if we record the trajectory of a moving robot, we have a spatio-temporal data set denoting at what time the robot visited which place. Time and place are two different aspects of this data set. Aspectualization singles out such aspects.

Definition 2. *Aspectualization is the process or result of explicating certain aspects of an observation purely by eliminating the others. Formally, it is defined as a function $\kappa : \mathcal{D}^n \to \mathcal{D}^m (n, m \in \mathbb{N}, n > m)$:*

$$\kappa(s_1, s_2, \ldots, s_n) = (s_{i_1}, s_{i_2}, \ldots, s_{i_m}),\ i_k \in [1, n],\ i_k < i_{k+1}\, \forall k\,.$$

Thus, aspectualization projects $\mathcal{D}^n$ to $\mathcal{D}^m$.

Example 1. An oriented line segment s in the plane is represented as a point $(x, y) \in \mathbb{R}^2$, a direction $\theta \in [0, 2\pi]$, and a length $l \in \mathbb{R}$: $s = (x, y, \theta, l)$. The reduction of this line segment to an oriented point is an aspectualization with $\kappa_a(x, y, \theta, l) = (x, y, \theta)$.

Aspects may span over several features s_i. However, to be able to single out an aspect from a feature vector by aspectualization, it must be guaranteed that no feature refers to more than one aspect. We call this property *aspectualizability*:

Definition 3. *If an aspect is exclusively represented by one or more components of a feature vector $s \in \mathcal{S}$ (that is: no s_i refers to more than one aspect), then we call $\mathcal{S}$ aspectualizable regarding this aspect.*

Example 2. The oriented line segment representation in Example 1 can be bijectively mapped from point, angle, and length to two points (x_1, y_1, x_2, y_2). Then aspectualization as defined in Def. 2 cannot single out the length of the line segment, because length is not represented explicitly. $\mathcal{S}$ is not aspectualizable regarding length.

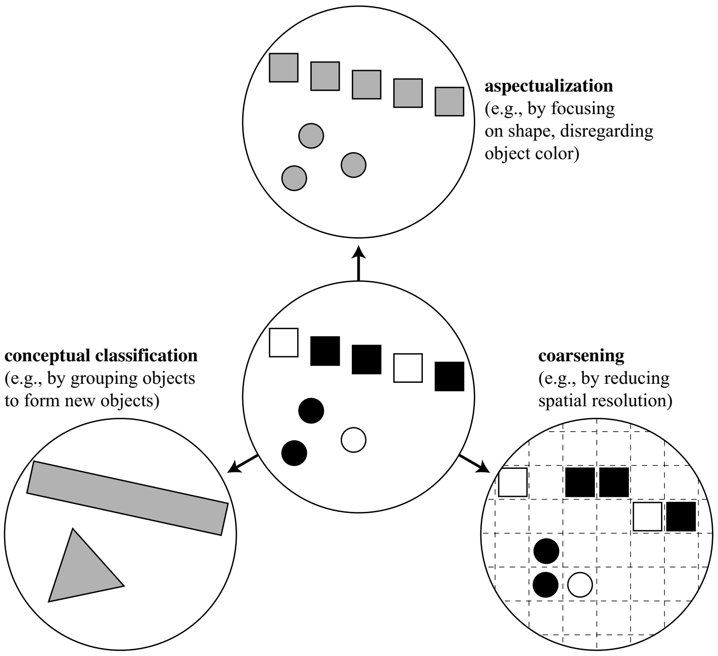

aspectualization
(e.g., by focusing on shape, disregarding object color)
conceptual classification
(e.g., by grouping objects to form new objects)
coarsening
(e.g., by reducing spatial resolution)

Fig. 1. Iconographic illustration of the three abstraction principles aspectualization, coarsening, and conceptual classification applied to the same representation

2.2 Coarsening

When the set of values a feature can take is reduced, we speak of a coarsening:

Definition 4. *Coarsening is the process or result of reducing the details of information of an observation by lowering the granularity of the input space. Formally, it is defined as a function* $\kappa : \mathcal{D}^n \to \mathcal{D}^n (n \in \mathbb{N})$,

$$\kappa(s) = (\kappa_1(s_1), \kappa_2(s_2), \ldots, \kappa_n(s_n))$$

with $\kappa_i : \mathcal{D} \to \mathcal{D}$ *and at least one* κ_i *being not injective.*

The existence of a non-injective κ_i ensures that we have an abstraction.

Example 3. An important representation in the area of robot navigation is the occupancy grid [8], a partition of 2-D or 3-D space into a set of discrete grid cells. A function $\kappa : \mathbb{R}^2 \to \mathbb{R}^2, \kappa(x, y) = (\lfloor x \rfloor, \lfloor y \rfloor)$ is a coarsening that maps any coordinate to a grid cell of an occupancy grid.

2.3 Conceptual Classification

Conceptual classification is the most general of the three proposed abstraction facets. It can utilize all components of the input to build new entities:

Definition 5. *Conceptual classification abstracts information by grouping semantically related features to form new abstract entities. Formally, it is defined as a non-injective function* $\kappa : \mathcal{D}^n \to \mathcal{D}^m (m, n \in \mathbb{N})$,

$$\kappa(s_1, s_2, \ldots, s_n) = (\kappa_1(s_{1,1}, s_{1,2}, \ldots, s_{1,h_1}), \kappa_2(s_{2,1}, s_{2,2}, \ldots, s_{2,h_2}), \ldots, \\ \kappa_m(s_{m,1}, s_{m,2}, \ldots, s_{m,h_m}))$$

with $\kappa_i : D^{h_i} \to D$ *and* $h_i \in \{1, \ldots, n\}$*, whereby* $i \in \{1, \ldots, m\}$.

Conceptual classification subsumes the other two abstraction concepts: If all κ_i have the form $\kappa_i : \mathcal{D} \to \mathcal{D}$ and $m = n$, a conceptual classification is a coarsening; and if all κ_i have the form $\kappa_i(s_j) = s_j$, $i \leq j$, $m < n$, and $\kappa_i = \kappa_j \Rightarrow i = j$, then a conceptual classification becomes an aspectualization.

Example 4. Data gathered from a laser range finder comes as a vector of distance values and angles to obstacles in the local surrounding, which can be represented as 2-D points in a relative coordinate frame around the sensor. Abstraction of these points to line segments by the means of a line detection algorithm (as, for example, described in [9]) is a conceptual classification.

To sum up, Fig. 1 illustrates aspectualization, coarsening, and conceptual classification in an iconographic way.

2.4 Abstraction and Representation

Intuitively, aspectualization and coarsening describe two very different processes: The first one reduces the number of features of the input, the latter one the variety of instances for every single feature. While aspectualization necessarily reduces the dimensionality of a representation, coarsening preserves dimensionality.

Depending on the representation of the feature vector, coarsening can produce a result that is equivalent to an aspectualization though: Let one or more mappings κ_i in a coarsening be defined as mappings to a single constant value: $\kappa_i = c_i$, $c_i \in \mathcal{D}$. Assume all other mappings κ_i to be the identity function. Now, consider an aspectualization that retains exactly the components not mapped to single constant values c_i by the coarsening. Obviously, this aspectualization has a canonical embedding in the result of the coarsening. We illustrate this by an example:

Example 5. As in Example 1, an oriented line segment in the plane is represented as a point $(x, y) \in \mathbb{R}^2$, a direction $\theta \in [0, 2\pi]$, and a length $l \in \mathbb{R}$: $s = (x, y, \theta, l)$. κ_a is defined as in Example 1. The function $\kappa_c(x, y, \theta, l) = (x, y, 1, l)$ is a coarsening, and it trivially holds:

$$\{\kappa_c(x, y, \theta, l)\} = \{(x, y, 1, l)\} \cong \{(x, y, l)\} = \{\kappa_a(x, y, \theta, l)\}$$

It is also possible to transform a knowledge representation such that a coarsening can be expressed by an aspectualization. For example, this is the case when abstraction operates on a group:

Theorem 1. *If κ_c is a coarsening on a group, for example $(\mathcal{S}, +)$, then there exists an isomorphism φ and an aspectualization κ_a such that the following diagram commutes:*

$$\begin{tikzcd}
\mathcal{S} \arrow[rr, "\varphi"] \arrow[dr, "\kappa_c"'] & & \mathcal{S}' \arrow[dl, "\kappa_a"] \\
& \mathcal{T} &
\end{tikzcd}$$

Proof. Choose $\varphi(s) = (s + \kappa_c(s), \kappa_c(s))$, $\varphi^{-1}(t_1, t_2) = t_1 + (-t_2)$ and $\kappa_a(t_1, t_2) = t_2$, and define $(\mathcal{S}', \oplus)$ with $\mathcal{S}' = \text{Image}(\varphi)$ and $t \oplus u = \varphi\left(\varphi^{-1}(t) + \varphi^{-1}(u)\right)$ for each $t, u \in \mathcal{S}'$. Checking that $(\mathcal{S}', \oplus)$ is a group and φ a homomorphism is straightforward. $\square$

We illustrate this theorem by the following example:

Example 6. Coordinates $(x, y) \in \mathbb{R}^2$ can be bijectively mapped to a representation $(\lfloor x \rfloor, x - \lfloor x \rfloor, \lfloor y \rfloor, y - \lfloor y \rfloor)$ which features decimal places separately. The function $\kappa(x, x', y, y') = (x, y)$ is an aspectualization with the same result as the coarsening in Example 3.

Note that Theorem 1 does not introduce additional redundancy into the representation. If we would allow for introducing redundancy we could bijectively create new representations by concatenating s and an arbitrary abstraction $\kappa(s)$ with the effect that any abstraction, including conceptional classification, can always be achieved by an aspectualization from this representation. Therefore, we do not regard this kind of redundancy here.

Not every representation allows for coarsening, as the following example shows:

Example 7. Commercial rounding is defined by a function $f : \mathbb{R}_0^+ \to \mathbb{R}_0^+$, $f(x) = \lfloor x + 0.5 \rfloor$. f is a coarsening. If, similar to Example 6, $x \in \mathbb{R}$ is represented as $(\lfloor x \rfloor, x - \lfloor x \rfloor)$, then commercial rounding can neither be expressed by aspectualization (because the representation is not aspectualizable regarding this rounding) nor by coarsening (because the abstraction function operates on both components x and $x - \lfloor x \rfloor$ of the feature vector, which contradicts Def. 4). So even if commercial rounding reduces the number of instances in half of the components, the example above cannot be expressed as a coarsening under this representation following Def. 4. It then must be seen as a conceptual classification, which is the most general of the three facets of abstraction presented here.

Different abstraction paradigms, even if describing distinct processes, can thus lead to the same result: Applicability of a specific abstraction principle relies heavily on the given representation, and usually different types of abstraction can be utilized to achieve the same result. Thus, the choice of an abstraction paradigm is tightly coupled with the choice of the state space representation. In the following we will argue for an action-centered view for choosing appropriate representations.

2.5 Abstraction in Agent Control Processes

Abstraction, as we define it, is not a blind reduction of information, but comes with a particular purpose. It is applied to ease solving a specific problem, and the concrete choice of abstraction is implied by the approach to master the task.

If we want to utilize spatial abstraction in the context of agent control tasks, we try to reach three goals:

1. Significantly reducing the size of the state space the agent is operating in
2. Eliminating unnecessary details in the state representation
3. Subsuming similar states to unique concepts

The first goal, reducing state space size, is a mandatory consequence of the latter two, which must be seen in the context of action selection: The question whether a detail is "unnecessary" or whether two states are "similar" depends on the task of the agent:

– A detail is considered *unnecessary* if its existence does not affect the action selection of the agent.
– Two states are considered to be *similar* if the agent should select the same action in any of the states.

This action centered view expands classical definitions of similarity, as it is for example given by Fred Roberts [10]: Two states s and s' are indistinguishable (written $s \sim s'$) if there is a mapping $f : \mathcal{S} \rightarrow \mathbb{R}$ and an $\epsilon \in \mathbb{R}^+$ with $s \sim s' \Leftrightarrow |f(s) - f(s')| < \epsilon$. Roberts' concept is data driven whereas ours is action driven in order to account for the task at hand. States may be very near concerning a certain measure, but nevertheless require different actions to take in certain contexts. Grid based approaches, achieved by coarsening, easily bear the danger of not being able to provide an appropriate state separation due to missing resolution in critical areas of the state space. Furthermore, a "nearness" concept as presented by Roberts is again a matter of representation and may only be appropriate in homogeneous environments.

Abstraction shall ease the agent's action selection process. If those details are eliminated that are irrelevant for the choice of an action, difficulty and processing time of action selection is reduced, and action selection strategies may be applicable to a broader range of scenarios.

When choosing an abstraction paradigm for a given data set, the result must be regarded in the context of *accessibility* of information. The goal of abstraction must be to enable easy access to the *relevant* information. Which piece of information is relevant, of course, depends on the task at hand: A computer-driven navigation control may require different concepts than a system interacting with a human being. Abstraction retains information that is relevant for a certain purpose. Therefore, it can never be regarded as purely data driven, but requires a solid a-priori concept of the problem to solve and, consequently, the actions to take.

As we have seen in Section 2.4, we can use different abstraction paradigms to achieve the same effect, given an appropriate state space representation. From the view point of accessibility we now argue for preferring the use of aspectualizable representations, as relevant aspects are clearly separated and easy to access and aspectualization itself is a computationally simple process. Accessibility eases knowledge extraction: Section 3.3 will show an example of an algorithm that makes use of the aspectualizability of a representation. Once again, aspectualizability can be achieved by abstraction. In particular, conceptual classification is a powerful means. So abstraction helps to create representations that allow for distinguishing different aspects by using aspectualization.

3 Knowledge Transfer of Simulation Strategies to a Real Robot

In this section we show how abstraction supports knowledge transfer. We regard the problem of transferring navigation skills learned with reinforcement learning (RL) [11] in a simple simulator to a real robot and demonstrate that on the one hand abstraction allows us to cope with real world concepts in the same way as with simulated ones and on the other hand that the transfer of knowledge benefits from aspectualizability of the state space representation. The key benefit of this approach is that learning is much more efficient in simple simulations than in real environments or complex simulations thereof. In particular, we show that the use of an aspectualizable representation empowers the derivation of aspectualizable behavior that is key to successful knowledge transfer.

3.1 The Task

The task considered here is the following: A simulated robot shall learn to find find a specified location $s^* \in \mathcal{S}$ within an unknown environment (see Fig. 2 left for a look on the simulation testbed).

This scenario is formalized as a Markov Decision Process (MDP) $\langle \mathcal{S}, \mathcal{A}, T, R \rangle$ with a continuous state space $\mathcal{S} = \{(x, y, \theta)— \ x, y \in \mathbb{R}, \ \theta \in [0, 2\pi)\}$ where each system state is given by the robot's position (x, y) and an orientation θ, an action space $\mathcal{A}$ of navigational actions the agent can perform to move to another state, a transition function $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denoting a probability distribution that performing an action a at state s will result in state s', and a reward function $R : \mathcal{S} \to \mathbb{R}$, where a positive reward will be given when a goal state $s^* \in \mathcal{S}$ is reached and a negative one if the agent collides with an obstacle. A solution to this MDP is a policy $\pi(s)$ that assigns an action to take to any state s. RL is a frequently used method to compute such a strategy. After successful learning, following π in a greedy way will now bring the robot from every position in the world to the goal location s^*.

In general, this strategy π is bound to the goal state the agent learned to reach. For mastering another task only differing in s^*, the whole strategy would need to be re-learned from scratch, including low-level skills as turning actions and collision avoidance—the knowledge gained in the first learning task would not be applicable. The challenge of avoiding this and re-using parts of the gained

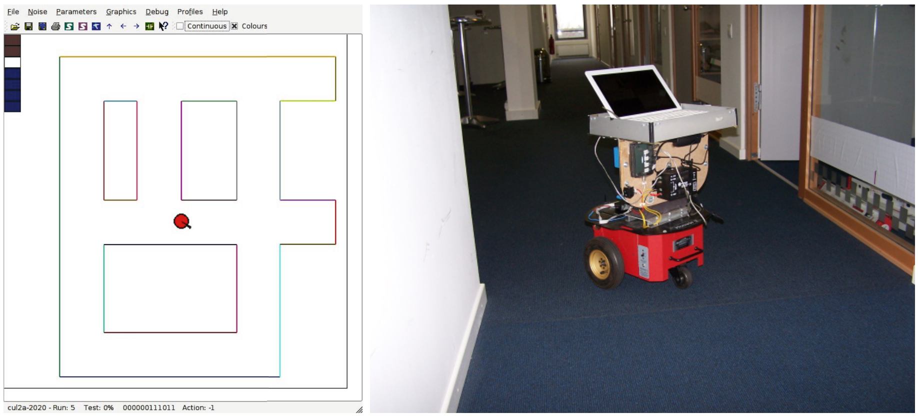

File Noise Parameters Graphics Debug Profiles Help
Continuous
Colours
cul2a-2020 - Run: 5 Test: 0% 000000111011 Action: -1

Fig. 2. Left: a screenshot of the robot navigation scenario in the simulator, where the strategy is learned. Right: a Pioneer 2 in an office building, where the strategy shall be applied. The real office environment offers structural elements not present in the simulator: open space, uneven walls, tables, and other obstacles.

knowledge of a learning task and transferring it to another one has recently been labeled *transfer learning*, and several approaches have been proposed to tackle this problem [12,13,14, e.g.]. We will describe how such transfer capabilities can be achieved by *spatial state space abstraction* and we will point out how abstraction mechanisms allow for knowledge transfer in a more general sense: Learned navigation knowledge is not only transferable to a similar task with another goal location, but abstraction allows us to operate on the same abstract entities in quite different tasks. We will show that the spatial state space abstraction approach even allows for bridging the gap between results gained in a simple simulator and real robotics just by the use of spatial abstraction.

3.2 Learning a Policy in Simulation

In our simulation scenario, the robot is able to perceive walls around it as line segments within a certain maximum range. This perception is disturbed by noise such that every line segment is detected as several smaller ones. The agent can also identify the walls. In our simulator, this is modeled in a way that every wall has a unique color and the agent perceives the color of the wall. The robot is capable of performing three actions: moving forward and turning a few degrees either to the left or to the right. Turning includes 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.

For learning we use the reinforcement learning paradigm of Q-learning [15]. The result is a Q-function that assigns an expected overall reward to any state-action pair (s, a) and a policy $\pi(s) = \operatorname{argmax}_a Q(s, a)$ that delivers the action with the highest expected reward for every state s.

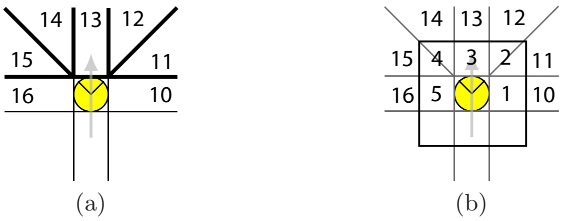


Fig. 3. Neighboring regions around the robot in relation to its moving direction. Note that the regions $R_1, \ldots, R_5$ in the immediate surroundings (b) overlap $R_{10}, \ldots, R_{16}$ (a). The size of the grid defining the immediate surroundings is given a-priori. It is a property of the agent and depends on its size and system dynamics (for example, the robot's maximal speed). In this work, only the thick drawn boundaries in (a) are regarded for building the representation.

This learning task is a complex one, because the underlying state space is large and continuous, and reinforcement learning processes are known to suffer from performance problems under these conditions. Thrun and Schwartz stated that for being able to adapt RL to complex tasks it is necessary to discover the structure of the world and to abstract from its details [16]. In any case, a sensible reduction of the state space will be beneficial for any RL application.

To achieve that structural abstraction, we make use of the observation that navigation in space can be divided into two different aspects: *Goal-directed behavior* towards a task-specific target location, and *generally sensible behavior* that is task-independent and the same in any environment [17]. According to [12], we refer to the first as *problem space* and to the latter as *agent space*. It is especially agent space that encodes structural information about the world that persists in any learning task and therefore this knowledge is worth transferring to different scenarios.

The structure of office environments as depicted in Fig. 2 is usually characterized by walls, which can be abstracted as line segments in the plane. Even more it is relative position information of line segments with respect to the robot's moving direction that defines structural paths in the world and leads to sensible action sequences for a moving agent. Thus, for encoding agent space, we use the qualitative representation RLPR (Relative Line Position Representation) [17]. Inspired by the "direction relation matrix" [18], the space around the agent is partitioned into bounded and unbounded regions R_i (see Fig. 3). Two functions $\overline{\tau} : \mathbb{N} \rightarrow \{0, 1\}$ and $\overline{\tau}' : \mathbb{N} \rightarrow \{0, 1\}$ are defined: $\overline{\tau}(i)$ denotes whether there is a line segment detected within a region R_i and $\overline{\tau}'(i)$ denotes whether a line spans from a neighboring region R_{i+1} to R_i. $\overline{\tau}_i$ is used for bounded sectors in the immediate vicinity of the agent (R_1 to R_5 in Fig. 3(b)). Objects that appear there have to be avoided in any case. The position of detected line segments in R_{10} to R_{16} (Fig. 3(a)) is helpful information to be used for general orientation and mid-term planning, so $\overline{\tau}'$ is used for R_{10} to R_{16}. This abstraction from line segments in the simulator to a vector of RLPR values is a conceptual classification.

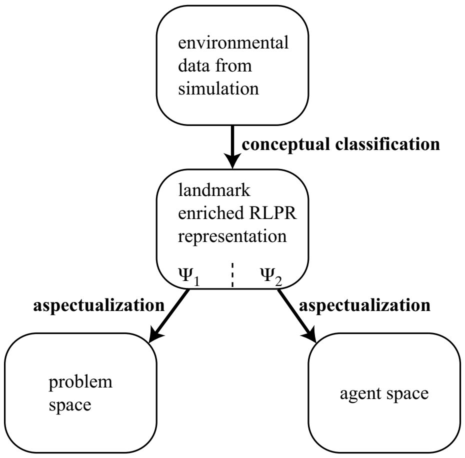

environmental data from simulation
conceptual classification
landmark enriched RLPR representation
Ψ1
Ψ2
aspectualization
aspectualization
problem space
agent space

Fig. 4. Abstraction principles used to build an aspectualizable state space representation

For representing problem space it is sufficient to encode the qualitative position of the agent within the world. We do this by representing a circular order of detected landmarks, as for example proposed in [19]. Therefore we regard a sequence of detected wall colors c_i at seven discrete angles around the robot: $\psi_l(s) = (c_1, \ldots, c_7)$.

As suggested in Section 2.5, we now use these two conceptual classifications to create an aspectualizable state space representation by concatenating ψ_l and ψ_r. The result $\psi(s)$ is the *landmark-enriched RLPR representation*:

$$\psi(s) = (\psi_l(s), \psi_r(s)) = (c_1, \ldots, c_7, \overline{\tau}(R_1), \ldots, \overline{\tau}(R_5)), \overline{\tau}'(R_{10}), \ldots, \overline{\tau}'(R_{16})$$

We call the new emerging state space $\mathcal{O} = \text{Image}(\psi)$ the *observation space*. It is a comparably small and discrete state space, fulfilling the three goals of abstraction we defined in Section 2.5. The RLPR based approach has been shown to outperform metrical representations that rely on distances or absolute coordinates with regard to learning speed and robustness [17]. For an example of deriving RLPR values refer to Fig. 5.

So conceptual classification is employed twice for both problem and agent space to create a compact state space representation. $\psi(s)$ is aspectualizable regarding the two aspects of navigation (see Fig.4). Let us now investigate how to take advantage of that to transfer general navigation knowledge to a new task.

3.3 Extracting General Navigation Behavior

The learning process results in a policy $\pi : \mathcal{O} \to \mathcal{A}$ that maps any $o \in \mathcal{O}$ to an action to take when the agent observes o. The corresponding Q-values are stored in a lookup table. We now want to apply this policy to a totally different domain, the real world, where we cannot recognize the landmarks encountered during learning. So the policy must provide sensible actions to take in the absence of known landmarks. This is the behavior that refers to the aspect of general navigation behavior or agent space. It has to be singled out from π.

By design, $\psi(s)$ is aspectualizable with regard to agent space, and the desired information is easily accessible. An aspectualization $\kappa(o) = \kappa(\psi_l(s), \psi_r(s)) = \psi_r(s))$ provides structural world information for any observation. That is, structurally identical situations share an identical RLPR representation.

A new Q-function $Q_{\pi'}$ for a general, aspectualized policy π' for arbitrary states with the same aspect $\psi_r(s)$ can be constructed by Q-value averaging over states with identical $\psi_r(s)$, which are easily accessible because of the aspectualizability of $\mathcal{O}$. Given a learned policy π with a value function $Q_\pi(o, a)$ ($o \in \mathcal{O}$, $a \in \mathcal{A}$), we construct a new policy π' with $Q_{\pi'}(o', a)$ ($o' \in \mathcal{O}'$, $a \in \mathcal{A}$) in a new observation space $\mathcal{O}' = \text{Image}(\psi_r)$, with the following function [20]:

$$Q_{\pi'}(o', a) = \frac{\sum_{c \in \{\psi_l(s)\}} (\max_{b \in \mathcal{A}}(|Q_\pi((c, o'), b)|))^{-1} \; Q((c, o'), a)}{|\{((c, o'), a) | Q_\pi((c, o'), a) \neq 0\}|}$$

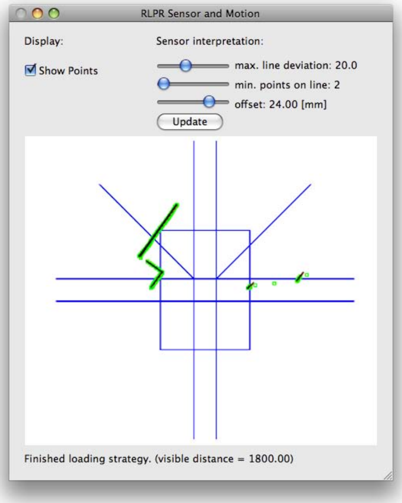
This is a weighted sum over all possible landmark observations (in reality, of course, only the visited states have to be considered, because $Q(o, a) = 0$ for the others, so the computational effort is very low). It is averaged over all state-action pairs where the information is available, that is, the Q-value is not zero. A weighting factor scales all values according to the maximum reward over all actions.

This procedure has been applied to a policy learned in the simulated environment depicted in Fig. 2 for 40,000 learning episodes. For the exact experimental conditions of learning the policy in simulation refer to [20]. The resulting policy $Q_{\pi'}$ can then be used to control a real robot as shown in the following section.

3.4 Using the Aspectualized Strategy on a Mobile Robot

Controlling a mobile robot with a strategy learned in simulation requires sensory input to be mapped to the same domain as used in the simulation. This can be accomplished in a straightforward manner given that an abstract intermediate representation is constructed on the robot from raw sensor data. We now detail this approach using a Pioneer-2 type robot equipped with a laser range finder.

Laser range finders detect obstacles around the robot (the field of view of the sensor used on our robot is 180°). By measuring laser beams reflected by obstacles one obtains a sequence of (in our case) 361 points in local coordinates. We use the well-known iterative split-and-merge algorithm that is commonly used in robotics to fit lines to scan data (see [9])—another conceptual classification, as we have seen in Example 4. With respect to parameters of the procedure we point out that a precise line fitting is not required [20]. Rather, we want to make sure that all obstacles detected by the laser range scanners get represented by lines, even if this is a crude approximation. The detected line configuration is then mapped every 0.25 seconds to the RLPR representation and fed into the learned strategy to obtain the action primitive to perform. See Fig. 5 for a view



[✓ Show Points]
max. line deviation: 20.0
min. points on line: 2
offset: 24.00 [mm]
Finished loading strategy. (visible distance = 1800.00)

$\overline{\tau}(R_1) = 1$
$\overline{\tau}(R_2) = 0$
$\overline{\tau}(R_3) = 0$
$\overline{\tau}(R_4) = 1$
$\overline{\tau}(R_5) = 0$

$\overline{\tau}'(R_{10}) = 1$
$\overline{\tau}'(R_{11}) = 0$
$\overline{\tau}'(R_{12}) = 0$
$\overline{\tau}'(R_{13}) = 0$
$\overline{\tau}'(R_{14}) = 1$
$\overline{\tau}'(R_{16}) = 1$

Fig. 5. Screenshot: Abstraction to RLPR in the robot controller. Depicted are the qualitative regions (see Fig. 3) and the interpreted sensor data which has been acquired from the robot position shown in Fig. 2 right. The overall representation for this configuration is $\psi_r(s) = \{1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1\}$.

on the line detection of the laser range finder data and the corresponding RLPR representation. Fig. 6 gives an overview on the development of representations in both simulator and robot application.

In the simulation three action primitives (straight on, turn left, turn right) have been used that always move the robot some fixed distance. Rather then implementing this step-wise motion on the real robot, we mapped the action to commands controlling the wheel speeds in order to obtain continuous motion. Additionally, movement is smoothed by averaging the most recent wheel speed commands to avoid strong acceleration/deceleration which the robot drive cannot handle well. We applied the averaging to the last 8 actions which (given the 0.25 second interval of wheel commands) yields a time of 2 seconds before reaching the wheel speed associated with the action primitive. In accordance to the robot's size and motion dynamics the inner regions of the RLPR grid (Fig.3(b)) have been set to 60 cm in front and both 30 cm to the left and the right of the robot.

We analyzed the behavior of the Pioneer 2 robot with the learned policy in our office environment. In contrast to the simple simulation environment the office environment presents uneven walls, open spaces of several meters, plants, and furniture like a sofa or bistro tables. The robot shows a reasonable navigation behavior, following corridors straightly and turning smoothly around curves. It also showed ability to cope with structural elements not present in the simulated environment, such as open space or tiny obstacles. In other words, general navigation skills learned in simulation have been transferred to the real-world environment. The robot only got stuck when reaching areas with a huge amount of clutter

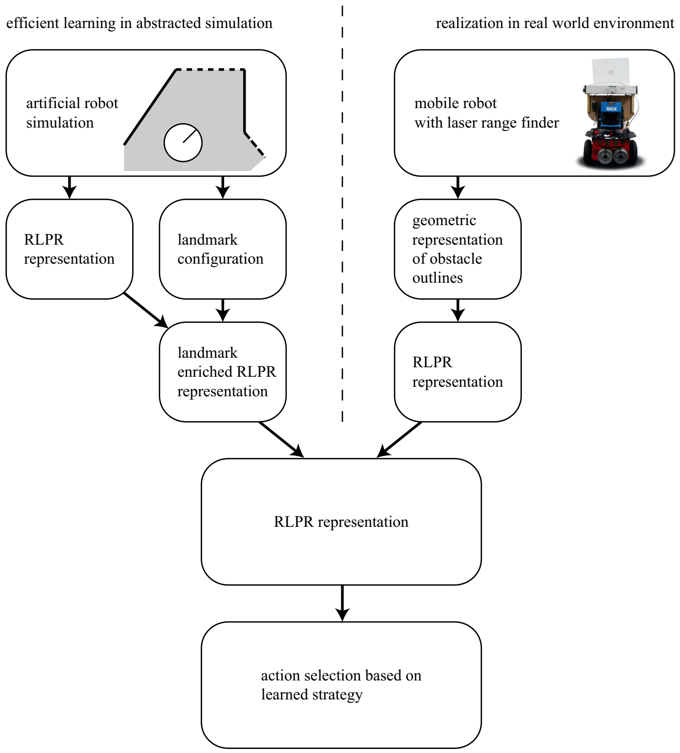

efficient learning in abstracted simulation
realization in real world environment
artificial robot simulation
mobile robot with laser range finder
RLPR representation
landmark configuration
geometric representation of obstacle outlines
landmark enriched RLPR representation
RLPR representation
RLPR representation
action selection based on learned strategy

Fig. 6. Evolution of spatial representations in both simulation and real robot application. Abstraction techniques enable both scenarios to operate on the RLPR representation to achieve a reasonable action selection.

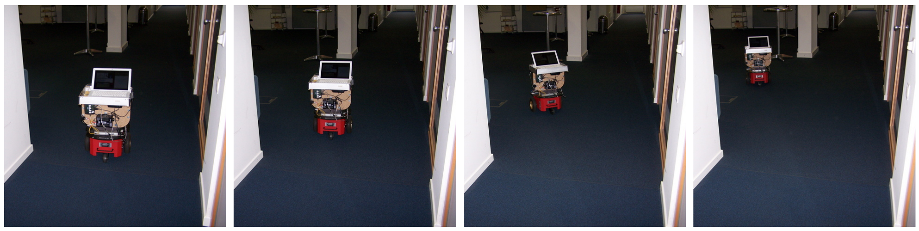


Fig. 7. Pioneer 2 entering an open space, using the aspectualized policy learned in the simulator. It shows a reasonable navigation behavior in a real office environment, driving smoothly forward and safely around obstacles.

(such as hanging leaves of plants) and in dead ends where the available motion primitives do not allow for collision-free movement anymore. Because the original task was goal-oriented (searching a specific place), the robot also showed a strong tendency of moving forward and thus actively exploring the environment instead of just avoiding obstacles. This generally sensible navigation behavior could now, for example, be used as a basis for learning new goal-oriented tasks on the robotics platform. Fig. 7 gives an impression of the robot experiment.

4 Discussion

Performing abstraction is a fundamental ability of intelligent agents and different facets of abstraction have thus been issued in previous work, addressing various scientific fields and considering a rich diversity of tasks. First, we comment on a critical remark by Klippel et al.: In their thorough study on schematization, they state that “there is no consistent approach to model schematization” [4]. We believe that by our formal definitions of abstraction principles the manifold terms used to describe abstraction can very well be classified and related.

The insight that abstraction can be divided into different categories has been mentioned before. Stell and Worboys present a distinction of what they call “selection” and “amalgamation” and formalize these concepts for graph structures [6]. Our definition of aspectualization and coarsening corresponds to selection and amalgamation, which Stell and Worboys describe as being “conceptually distinct” types of generalization. Regarding this, we pointed out that this conceptual distinctness does only apply to the *process* of abstraction and not the *result*, as we could show that the effect of different abstraction paradigms critically depends on the choice of the initial state space representation.

Bertel et al. also differentiate between different facets of abstraction (“aspectualization versus specificity”, “aspectualization versus concreteness”, and “aspectualization versus integration”), but without giving an exact definition [7]. “Aspectualization versus specificity” corresponds to our definition of aspectualization, and “aspectualization versus concreteness” to coarsening. However, our definition of aspectualization is tighter than the one given by Bertel et al.: According to them, aspectualization is “the reduction of problem complexity through the reduction of the number of feature dimensions”. In our definition, it is also required that all the other components remain unchanged.

The notion of *schematization*, which Leonard Talmy describes as “a process that involves the systematic selection of certain aspects of a referent scene to represent the whole disregarding the remaining aspects” [21] is tightly connected to our definition of aspectualization. If we assume the referent scene to be aspectualizable according to Def. 3, then the process mentioned by Talmy is aspectualization as defined here.

Annette Herskovits defines the term schematization in the context of linguistics as consisting of three different processes, namely abstraction, idealization, and selection [5]. According to our definition, abstraction and selection would both be an aspectualization, and idealization refers to coarsening.

The action-centered view on abstraction we introduced in Section 2.5 is also shared by the definition of *categorizability* given by Porta and Celaya [22]. The authors call an environment categorizable, if “a reduced fraction of the available inputs and actuators have to be considered at a time”. In other words: In a categorizable environment, an abstraction can be achieved that subsumes identical action selection to identical representations.

5 Conclusion

In this article we classify abstraction by three distinct principles: aspectualization, coarsening, and conceptual classification. We give a formal definition of these principles for classifying and clarifying the manifold concept names for abstraction found in the literature. This enables us to show that knowledge representation is of critical importance and thus must be addressed in any discussion of abstraction. Identical information may be represented differently, and, by choosing a specific representation, different types of abstraction processes may be applicable and lead to an identical result. Also, as abstraction is triggered by the need to perform a certain task, abstraction can never be regarded as purely data driven, but it requires a solid a-priori concept of the problem to solve and, consequently, the actions to take.

We introduce the notion of aspectualizability in knowledge representations. Aspectualizable knowledge representations are key to enabling knowledge transfer. By designing an aspectualizable representation, it is possible to transfer navigation knowledge learned in a simplified simulation to a real-world robot setting.

Acknowledgments. This work was supported by the DFG Transregional Collaborative Research Center SFB/TR 8 "Spatial Cognition" (project R3-[Q-Shape]). Funding by the German Research Foundation (DFG) is gratefully acknowledged. The authors would like to thank Jan Oliver Wallgrün, Frank Dylla, and Jae Hae Lee for inspiring discussions. We also thank the anonymous reviewers for pointing us to further literature from different research communities.

References

1. Hobbs, J.R.: Granularity. In: Proceedings of the Ninth International Joint Conference on Artificial Intelligence (IJCAI), pp. 432–435 (1985)
2. Bittner, T., Smith, B.: A taxonomy of granular partitions. In: Montello, D. (ed.) Spatial Information Theory: Cognitive and Computational Foundations of Geographic Information Science (COSIT), pp. 28–43. Springer, Berlin (2001)
3. Mackaness, W.A., Chaudhry, O.: Generalization and symbolization. In: Shekhar, S., Xiong, H. (eds.) Encyclopedia of GIS (2008)
4. Klippel, A., Richter, K.F., Barkowsky, T., Freksa, C.: The cognitive reality of schematic maps. In: Meng, L., Zipf, A., Reichenbacher, T. (eds.) Map-based Mobile Services – Theories, Methods and Implementations, pp. 57–74. Springer, Berlin (2005)
5. Herskovits, A.: Schematization. In: Olivier, P., Gapp, K.P. (eds.) Representation and Processing of Spatial Expressions, pp. 149–162. Lawrence Erlbaum Associates, Mahwah (1998)
6. Stell, J.G., Worboys, M.F.: Generalizing graphs using amalgamation and selection. In: Güting, R.H., Papadias, D., Lochovsky, F. (eds.) SSD 1999. LNCS, vol. 1651, pp. 19–32. Springer, Heidelberg (1999)

7. Bertel, S., Vrachliotis, G., Freksa, C.: Aspect-oriented building design: Toward computer-aided approaches to solving spatial contraint problems in architecture. In: Allen, G.L. (ed.) Applied Spatial Cognition: From Research to Cognitive Technology, pp. 75–102. Lawrence Erlbaum Associates, Mahwah (2007)
8. Moravec, H.P., Elfes, A.E.: High resolution maps from wide angle sonar. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), St. Louis, MO (1985)
9. Gutmann, J.S., Weigel, T., Nebel, B.: A fast, accurate and robust method for self-localization in polygonal environments using laser range finders. Advanced Robotics 14(8), 651–667 (2001)
10. Roberts, F.S.: Tolerance geometry. Notre Dame Journal of Formal Logic 14(1), 68–76 (1973)
11. Sutton, R.S., Barto, A.G.: Reinforcement learning: an introduction. In: Adaptive Computation and Machine Learning. MIT Press, Cambridge (1998)
12. Konidaris, G.D., Barto, A.G.: Building portable options: Skill transfer in reinforcement learning. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI) (2007)
13. Taylor, M.E., Stone, P.: Cross-domain transfer for reinforcement learning. In: Proceedings of the Twenty Fourth International Conference on Machine Learning (ICML 2007), Corvallis, Oregon (2007)
14. Torrey, L., Shavlik, J., Walker, T., Maclin, R.: Skill acquisition via transfer learning and advice taking. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 425–436. Springer, Heidelberg (2006)
15. Watkins, C., Dayan, P.: Q-learning. Machine Learning 8, 279–292 (1992)
16. Thrun, S., Schwartz, A.: Finding structure in reinforcement learning. In: Tesauro, G., Touretzky, D., Leen, T. (eds.) Advances in Neural Information Processing Systems: Proceedings of the 1994 Conference, vol. 7. MIT Press, Cambridge (1995)
17. Frommberger, L.: A generalizing spatial representation for robot navigation with reinforcement learning. In: Proceedings of the Twentieth International Florida Artificial Intelligence Research Society Conference (FLAIRS 2007), Key West, FL, USA, pp. 586–591. AAAI Press, Menlo Park (2007)
18. Goyal, R.K., Egenhofer, M.J.: Consistent queries over cardinal directions across different levels of detail. In: Tjoa, A.M., Wagner, R., Al-Zobaidie, A. (eds.) Proceedings of the 11th International Workshop on Database and Expert System Applications, Greenwich, UK, pp. 867–880 (2000)
19. Schlieder, C.: Representing visible locations for qualitative navigation. In: Carrete, N.P., Singh, M.G. (eds.) Qualitative Reasoning and Decision Technologies, Barcelona, Spain, pp. 523–532 (1993)
20. Frommberger, L.: Generalization and transfer learning in noise-affected robot navigation tasks. In: Neves, J., Santos, M.F., Machado, J.M. (eds.) EPIA 2007. LNCS (LNAI), vol. 4874, pp. 508–519. Springer, Heidelberg (2007)
21. Talmy, L.: How language structures space. In: Pick Jr., H.L., Acredolo, L.P. (eds.) Spatial Orientation: Theory, Research, and Application. Plenum, New York (1983)
22. Porta, J.M., Celaya, E.: Reinforcement learning for agents with many sensors and actuators acting in categorizable environments. Journal of Artificial Intelligence Research 23, 79–122 (2005)