TOWARDS CONCEPT FORMATION GROUNDED ON PERCEPTION AND ACTION OF A MOBILE ROBOT

Volker Klingspor and Katharina Morik*

**University of Dortmund, Computer Science Dept. LS VIII, D-44221 Dortmund, Germany*
{volker,morik}@ls8.informatik.uni-dortmund.de

Abstract. The recognition of objects and, hence, their descriptions must be grounded in the environment in terms of sensor data. We argue, why the concepts, used to classify perceived objects and used to perform actions on these objects, should integrate action-oriented perceptual features and perception-oriented action features. We present a grounded symbolic representation for these concepts. Moreover, the concepts should be learned. We show a logic-oriented approach to learning grounded concepts.

Key Words. Operational concepts, inductive logic programming, combining sensing and action, symbol grounding

1. INTRODUCTION

Up to now, application programming of a robot is a time consuming task. The programming is carried out at a very low level. The goal position of the individual commands must be given by exact real world coordinates. This has three undesired consequences. First, even if the new application is very similar to the one before, the control program must be rewritten completely. Second, these commands require a fixed environment and complete knowledge of the environment. If either an unforeseen event occurs or the environment is (partially) unknown, this approach to preparing an application fails. Third, communication between a knowledgeable user and a robot is hindered by the low-level representation of the application program. The user cannot change the robot's behavior directly according to small changes in the environment, nor can he help the robot out of a failure situation.

One way to overcome the problems described above is to develop robots that learn. From its experience in an environment the robot could adapt its behavior to the environment. Moreover, the robot could transfer what it has learned in one environment to other, similar environments. A second way is to use less specific commands. If, for example, the user wants a mobile robot to leave the room, turn to the right until it reaches a desk, and then stop, it should be possible for the user to state just this. This requires the robot to be capable of recognizing doors and – in the absence of specific goal coordinates – be capable of searching for them. The robot must assign its perceptions to classes of objects, i.e., to concepts (e.g., doors). Our aim is to combine both of the described approaches to build up more applicable and more flexible robots in future.

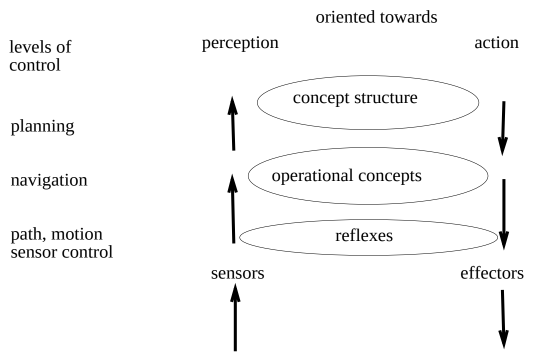

Fig. 1 Levels of control

Machine learning can be applied to robot technology in several ways. Consider a hierarchical control structure with different levels of behaviors as it is proposed by (Brooks, 1986) (Fig. 1). At all levels, perception-oriented and action-oriented processing is related. Machine learning can enhance the performance at each level.

Biologically inspired work in robotics has developed artificial beings that adapt to their environment (Brooks, 1991; Steels, 1993). This type of learning is restricted to reflex-like behavior. Similarly, the system described by (Millán and Torras, 1992) operates on the reflexes level. It learns how to avoid obstacles and reacts very fast to sensory input, but higher levels of cognition such as reasoning and concept formation are excluded. The system is unable to find a path in a maze-like environment in an appropriate time. Nevertheless, learning at this lower level enables the system to react immediately to unexpected situations and thus decreases effort at higher levels.

In order to enhance a robot's high-level processing, planning can be tailored to particular robot tasks by learning techniques (Segre, 1988).

Learning from failures of robot actions can be used to enhance the planning capabilities of a robot (Zercher, 1992; Bennett, 1989). Learning can also be used to link the environment model with the perceptions when executing a plan (Jong and Bennett, 1993; Gil, 1994), thus adapting a general plan to the specific environment.

Our approach to learning robots links low-level features that integrate sensing and action with high-level concepts. These concepts, which we call *operational concepts* combine sensing and action and are grounded on basic features. The use of these concepts sets up the following learning tasks:

- learning to recognize perceived objects as instances of a concept,
- learning to enhance a map of the environment by adding instances of perceived concepts,
- and learning which sensing and moving actions are appropriate for easing an object's recognition.

The learned concepts can then be used to formulate high-level commands and to control their execution (Wallner et al., 1994).

In this paper, we focus on two main properties of operational concepts: how they can be grounded in the environment (Sect. 2.1) and how to integrate perception and action (Sect. 2.2). In Section 3, we describe the data used for the experiments. Section 4 focuses on the representation of the data and the applied learning techniques. Finally, we discuss open problems and restrictions of our approach (Sect. 5).

2. REQUIREMENTS OF OPERATIONAL CONCEPTS

Two requirements were to be regarded during the development of representing operational concepts. First, the concepts should be usable by real robots in a real world. Therefore, the features used to describe the concepts must be grounded in the environment (Harnad, 1990; Wrobel, 1991; Cottrell et al., 1990). Second, inspired by psychological results, we argue why the representation should integrate perception and action.

2.1 Symbol Grounding

From the AI point of view, concept characterization means the definition of a concept based on features, so that examples for this concept will be covered by the characterization. If a formula representing this characterization, is true for an object, this object is an instance of the concept, otherwise it is not an instance.

Our topic is to learn concept descriptions for mobile robots. The input data used for learning are sensor measurements. In contrast to most problems examined in symbolic learning, no explicitly given features exist, only real values are given. But this data is too specific to be applied to more than a single object. Thus, more abstract features must be constructed from the sensory data. The choice of these features and their calculation is a very important task, because it influences the expressiveness of the symbol system. Concepts that cannot be described by these features can neither be described by newly built intermediate concepts, since intermediate concepts only simplify the representation, but do not enlarge the "closure of existing symbols" (Wrobel, 1991).

Until now, feature construction was programmed by an application developer. After this programming step, he had to test the quality of the constructed features with the used system which might be a machine learning system, and then he had to change the way features are constructed if the performance of the system were not sufficient. This process is very time consuming, and if the environment or the robot changes, feature construction must be adapted to this change.

Feature construction cannot be seen as an isolated task. If the concept learning step is not taken into account while developing the feature construction, the feature construction step cannot get any external evaluation about the quality of the constructed features. So learning how to construct features must be combined with concept learning.

Additionally, the features must be delivered online, because the robot must be able to react immediately to its perceptions. Thus, the algorithm that calculates these features must be incremental, analyzing each new input as soon as it is given.

For learning the basic features, a set of functions and their possible parameters are given. The functions calculate basic features on the basis of sensor measurements. The system learns the appropriate degree of granularity by searching for an appropriate function and a useful set of parameters. The search space is partially ordered by the granularity of the functions and parameters. The granularity is measured in terms of how many feature statements are delivered for a series of measurements. If the measurements are described by few feature statements, it is a rough, highly compressing abstraction. If many feature statements are delivered for the same series of measurements, it is a finely grained, less compressing abstraction. The appropriateness of the granularity is measured in terms of the quality of learning results using the features. If no good concept learning was possible using a rough abstraction, a finer granularity is tried, and so forth, until a good learning result has been achieved or no further function and parameter set is available.

2.2 Integration of Perception and Action

The second main issue of operational concepts is the integration of perception and action, where perceptual features must be action-oriented and action features must be perception-oriented. Humans represent events by relating the objects involved and the action, that is performed at these objects. Abelson called *predicates* what comprises an action and its corresponding object (Abelson, 1963), e.g., drinking from a cup, throwing a ball, moving through a doorway. Nelson states, that "..., young children must represent their own roles and the roles of other and be able to reciprocate actions of the other with actions of their own, ..." (Nelson, 1983, p. 135). Objects should be represented as relations between actions and reactions, where an object implies a specific action, like a "ball implies throwing". Nelson regards three kinds of relations between objects and actors, where the two higher levels are the most appropriate ones in our scenario. In the medium level, "different objects may occur in the same position of an event", like a specific tennis ball or a red rubber ball in the event "ball implies throwing". In our representation, the different objects that can occur in an event are represented by their perceptual features, and the action these objects imply are integrated in the definition of operational concepts as action features. Additionally, "an object or category of objects may recur in a number of different events, in similar or different relationships". The category of doors may occur in different events like moving along them or moving through them, so we have to relate different actions with the perceptions of the concept door.

This integration of perceptual features and action features also supports the classification of an object as an instance of a concept. In a conventional representation, a cup, for example, is defined by having a flat bottom and a handle and being concave. But it is easy to find arbitrarily many objects with these properties that are not cups because it is impossible to drink from them, e.g., if the handle bridges over the opening (De Jong and Mooney, 1986). Finding a complete description of a cup excluding all exceptions is impossible because of the infinite number of these exceptions. This is the qualification problem (McCarthy and Hayes, 1969). So, how to define a cup suitably? The main issue of a cup is that you can drink from it. If a cup is defined as a receptacle from which drinking must be possible, the object classification can be verified by performing this action. If it is possible to drink from the perceived object, it can be classified as a cup. In this way, actions are integrated into concept descriptions and their recognition functions.

Kedar-Cabelli has gone a first step in this direction (Kedar-Cabelli, 1988), using properties that describe action applicability, e.g., that a cup must be liftable. But the applicability is described by perceptual features such as material, weight, handle, etc., – not by action features proper. Giordana and Saitta have proposed to use *executable features* in concepts descriptions (Giordana and Saitta, 1990). These features are true for an object, if a particular handling of this object is successful. For instance, the feature "movable" for a concept can be verified by moving that object. We want to develop their approach further and propose that features at all levels should integrate action and perception. Perceptual features require the integration of the action that is performed while perceiving an object. Suppose that you are looking at a cup from above. This position does not allow to determine whether the bottom is flat or not. The perception is restricted by the action during which an object is perceived. Action features, in turn, require the integration of perception. Particular sensor patterns express the applicability conditions, the conditions for successful performance of the action, and the conditions for ending the action. In this way, an action is expressed by perceptions.

3. DATA FROM THE NAVIGATION SCENARIO

Before describing the representation hierarchy and the learning algorithms, we present the scenario and the robot we used for getting data. PRIAMOS, developed by the University of Karlsruhe (Dillmann et al., 1993), is a mobile robot with three degrees of freedom for motion, i.e., it can move into every direction and rotate simultaneously. It has 24 sonar sensors measuring the distance to the nearest object within the emission cone. Errors can occur, if the sonar beam is multiply reflected or the angle between the beam and the object is inconvenient. At every side of the robot three sensor are installed, emitting in parallel. Three sensors are mounted at every corner emitting with a difference of 15°.

The aim of our first learning trials was to learn descriptions for the concepts move_along_door and move_through_door. We used data from 28 traces in a simple room, most of them being paths

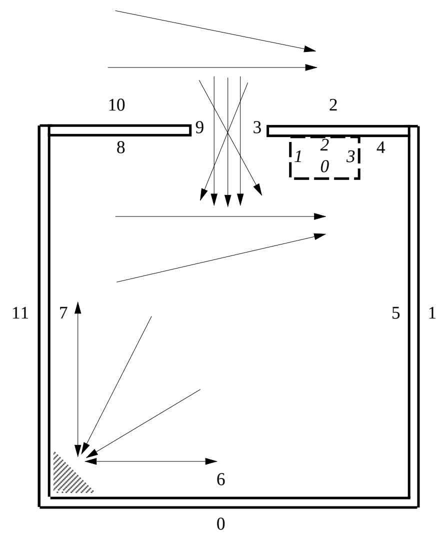

Fig. 2 Room with traces and edge numbers

along or through a doorway in different directions (Fig. 2). The cupboard, displayed as the hatched object, appeared only in some of the traces. For every trace, we got between 25 and 35 measurements of the 24 sensors, summing up to 17472 in total. Each measurement consists of the following data:

- trace number and point of time; for identification of the measurement,
- robot position in world coordinates,
- sensor number, its position and orientation; for identification of the sensor,
- measured distance,
- position of the sensed point in world coordinates,
- object and edge number of what is sensed.

4. THE REPRESENTATION AND LEARNING APPROACHES

Learning can take place at several levels within a concept hierarchy representing concepts at different degrees of abstraction. In this section, we first present our approach to overcome the symbol grounding problem, namely the learning of basic features (Section 4.1). Then, we give an overview of the representation hierarchy, based on the basic features. Finally, we show how learning can be applied to higher levels of the hierarchy. The used learning algorithm will not be described in detail (but see (Klingspor, 1994)). The reader may think of any learning algorithm for (restricted) first-order logic because, in principle, any first-order learning algorithm can solve our learning tasks. Note, that first-order learning is necessary in order to handle time intervals and their relations. The classical learning algorithms for attribute-value representations cannot solve our learning tasks.

4.1 Learning Basic Features

In Section 2.1 we have argued that the appropriate construction of basic features is crucial for further concept learning. Moreover, we have explained the necessary interaction of feature construction and concept learning. Now we present the learning algorithm for feature construction, starting with the specification of the learning task:

> **Learning basic features:**
> **Given** a set of functions and parameter values of these functions and an ordering on these different ways to calculate basic features –
> **Try** to find the function and parameters that lead to good results of concept learning.

Our learning task is to select appropriate values from a given set. This is simpler than the construction of different functions for calculating basic features. The goal of learning is to find the parameter setting, whose application results in basic features with the appropriate level of detail. In this context "appropriate" means, that, on the one hand, the basic features contain only those details, which are necessary to recognize and distinguish operational concepts. On the other hand it means, that they do not contain superfluous, redundant details, which would impair the recognition process by overloading it. The set of parameter settings can be considered as a *version space* (Mitchell, 1982) for learning basic features. The learning algorithm can be described by the following steps:

1. Start with a predefined set of parameters.
2. Calculate the basic features based on the function determined by the actual parameters.
3. Learn concept descriptions.
4. **If** the evaluation of concept learning is not acceptable and other parameter values are available, take the next, less abstract parameter setting and iterate at step 2.
   Otherwise: stop.

The learned (i.e. selected) functions and parameters are used to calculate basic features from sensory input. The calculation is incremental so that it can be performed while the robot is moving.

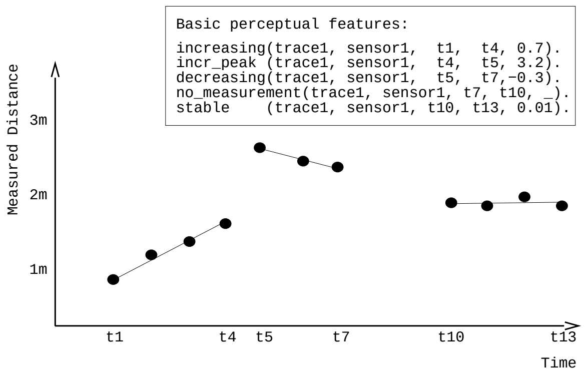
```
Basic perceptual features:
increasing(trace1, sensor1,  t1,  t4, 0.7).
incr_peak (trace1, sensor1,  t4,  t5, 3.2).
decreasing(trace1, sensor1,  t5,  t7,-0.3).
no_measurement(trace1, sensor1, t7, t10, _).
stable    (trace1, sensor1, t10, t13, 0.01).
```

Fig. 3 Measurements and basic features.

Basic features are calculated for each sensor using the following algorithm:

1. Start with the first measurement.
2. Get the next measurement and calculate the gradient between the previous and the current one, i.e., the quotient of the difference of the measurements and the moved distance. T_1 and T_2 becomes an interval.
3. **If** the gradient is close enough to the average gradient of the interval:

   then enlarge the interval by the new time point and adapt the average gradient depending on the way defined by a parameter.

   else close the previous interval at the previous time point and start a new one at that time point with the new gradient as average gradient.

   "Close enough" is defined by a parameter, too.
4. A closed interval will be named by a symbol, depending on the average gradient attached to that interval. The set of symbols and the gradients they belong to are determined by a further parameter.
5. Go to step 2.

In Figure 3, an example for a short sequence of measurements is displayed together with the corresponding basic features. First, the gradient between two measurements is more or less 0.7. The measurements are increasing linearly. That is represented by the basic feature increasing(trace1, sensor1, t1, t4, 0.7). Then, from time point t4 to t5, the gradient is greater than 1. Since the difference between two measurements cannot be greater than the moved distance, if the sensor sensed the same edge, two different edges must have been sensed. This situation is expressed by the basic feature incr_peak. Sometimes a sensor gets no echo, e.g., if there is no object within the

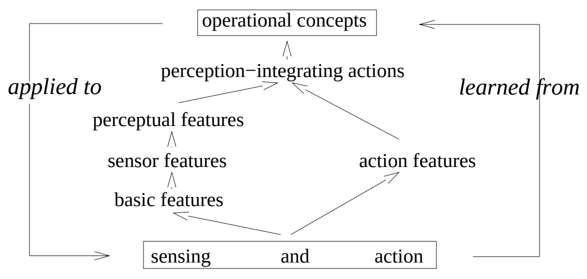

Fig. 4 Representation Hierarchy

maximal sensing distance. This situation, in the example the interval from t7 to t10, is labeled no_measurement. Other basic features are stable (describing more or less equal measurements), decreasing, decreasing_peak, no_movement (we cannot compute the gradient, if the robot does not move), and straight_to and straight_away (if the gradient is about 1 or -1, respectively).

Note, that the basic features compress a sequence of measurements to a statement about a time interval. The finely grained basic features summarize only a short sequence of measurements where the rough features summarize a long sequence of measurements. In that way, the basic features determine the time intervals used by concept learning. The time intervals, in turn, determine what can be achieved by concept learning. It is, for instance, impossible to learn good concepts descriptions, if the time intervals determined by the basic features do not match the time intervals used in the examples to be classified.

4.2 The Representation Hierarchy

Several levels of abstraction link the level of raw sensor and motion data with the level of operational concepts. The hierarchy of more and more abstract representations comprises perceptual features at several levels which are all related with actions as well as action feature which become integrated with perceptual features at a higher level. Figure 4 shows the representation hierarchy.

In the first step, *basic features* are constructed based on the measurements and the robot's position. At the next higher level, the perceptions while moving are represented by *single sensor features.*

s_line: a single edge.

s_jump: two parallel edges, sensed in sequence.

s_convex: two orthogonal edges, sensed from the convex side of this corner.

s_concave: two orthogonal edges, sensed from the concave side.

Each sensor feature is characterized by sequences of basic features. The sequences are represented by rules. An example rule for s_jump is:

stable(Trace, Or, Sensor, Time_1, Time_2, Grad_1) &
incr_peak(Trace, Or, Sensor, Time_2, Time_3, Grad_2) &
stable(Trace, Or, Sensor, Time_3, Time_4, Grad_3)
$\rightarrow$ s_jump(Trace, Sensor, Time_1, Time_4, parallel).

At the next level, situations are represented where different sensors of a class sensed the same constellation of edges while moving. Single sensor features are grouped together in order to increase the evidence for an edge constellation. This helps to cope with the noise of single sensor measurements. It can be sufficient for a *sensor group feature* to be derived, that two of three sensors of a sensor class sensed the same constellation. Additionally, we abstract from the actual sensor numbers in this step. Instead, the orientation of the perception relative to the orientation of the motion is used.

s_jump(Trace, Sensor_1, Start_1, End_1, parallel) &
sclass(Trace, Sensor_1, right_side) &
s_jump(Trace, Sensor_2, Start_2, End_2, parallel) &
sclass(Trace, Sensor_2, right_side) &
succ(Start_1, Start_2) & succ(End_1, End_2)
$\rightarrow$ sg_jump(Trace, right_side, Start_1, End_2, parallel).

At the next level of abstraction, the level of *action-oriented perceptual features*, perceptions at different sides of the robot will be combined.

sg_jump(Trace, right_side, T_1, T_2, parallel) &
sg_jump(Trace, left_side, T_1, T_2, parallel)
$\rightarrow$ through_door(Trace, T_1, T_2, both_sides, parallel).

At the action side of the representation hierarchy, only *basic actions* occur. They represent intervals of uniform motions. From the absolute orientation and speed of the motion features are calculated that relate the orientation to the previous direction of motion. The basic action features used are move, rotate, and stand, with the arguments *trace, time interval, speed,* and *direction*.

Action features and perceptual features are then combined to *perception integrating features* representing single actions and the corresponding perceptions. An example of such an action is a steady movement while measuring through_door in a specific direction PDirect.

through_door(Trace, PDirect, T_1, T_2, parallel) &
move(Trace, T_1, T_2, Speed, Dir)
$\rightarrow$ move_parallel(Trace, T_1, T_2, Speed, Dir,
through_door, PDirect).

Finally, *operational concepts* use perception integrating features for characterizing preconditions, verification conditions, and end-of-action conditions for an abstract action and its corresponding object.

standing(Trace, T1, T2, *in_front_of_door*,
PDirect, small_side, *PrevP*) &
parallel_moving(Trace, T2, T3, MSpeed,
PDirect, *through_door*, right_and_left) &
standing(Trace, T3, T4, *in_front_of_door*, rear,
small_side, *through_door*)
$\rightarrow$ move_through_door(Trace, T1, T4).

The rule describes the operational concept of moving through a doorway. It combines the recognition of the doorway with the action of moving through it on a parallel track. Moving diagonally through a doorway is a different operational concept. Action and perception are linked by the perceptual features occurring as arguments in the action predicates. In our example rule, the perceptual features are written in italics.

The first premise of the rule states that the robot is standing in a time interval from T1 to T2 of a particular (Trace) and senses the perceptual feature in_front_of_door from the sensors of a small side after having perceived the perceptual feature PrevP prior to T1. This premise denotes the precondition for moving through a doorway on a parallel track. The precondition can also be viewed as a trigger for planning: whenever recognizing that it stands in front of a door, the robot may execute the action of parallel moving through the doorway.

The second premise states that in a following time interval the action parallel_move is executed, measuring by the sensors at the right and the left side of the robot the perceptual feature through_door. Note, that the particular values for the time intervals do not matter for the operational concept but are instantiated by the specifics of a particular path. Only the time relation is important. This makes the operational concept independent of the particular depth of a doorway and the robot's particular rate of advance.

The third premise describes the end of the movement through the doorway: the robot is standing in front of the door, now sensing with its rear sensors mounted at its small side the feature in_front_of_door after having perceived the feature through_door. PDirect, the orientation of perception (e.g., left, right, front, rear) is not fixed by the rule. It is only relevant that the orientation should not change during the parallel move and that with respect to this orientation the doorway is sensed in the rear after the movement. That means, whatever the orientation was at the beginning of the action, it is defined as being front so that after completing the action the doorway is sensed by the opposite, the rear sensors.

4.3 Learning Perceptual Features

Learning of higher-level features and, finally, of operational concepts is performed in the paradigm of *inductive logic programming* (ILP, see, e.g., (Muggleton, 1992)). The learning task is the following:

> **Learning higher-level features:**
> **Given** the target feature for various situations and background knowledge –
> **learn** rules that characterize the target feature in terms of the background knowledge.

The background knowledge is a set of facts from the level below the target feature and general information about the robot (e.g., sensor classes). This learning task is performed several times: learning sensor features from basic features, learning sensor group features from sensor features, learning action-oriented perceptual features from sensor group features, and learning operational concepts from perception-integrating action features.

We applied learning algorithms that search a restricted hypothesis space completely. The hypothesis space is restricted syntactically by rule schemata that express the form of learnable rules. Rules that are not instantiations of a rule schema cannot be learned. This prohibits learning rules that cannot be put to use by the robot. The rule schemata are ordered according to their generality. Starting with the most general rule schema, all its instantiations (i.e. rules) are tested whether they cover positive but not negative examples of the target feature. Until no negative (or only few) negative examples are covered, the next special rule schema is instantiated and tested. The most applicable learning algorithms were RDT (Kietz and Wrobel, 1992) and a modification of it, GRDT (Klingspor, 1994).

In a nutshell, the results are as follows. Given 1004 examples for the four sensor features, we learned 129 rules, covering 87% of the given examples. For learning sensor group features, we had given 956 examples, from which GRDT learned 136 rules. Using the learned rules for sensor features and sensor group features, we got a coverage of 64%. For the operational concept through_door, we had given ten examples. We learned three rules, two of them covering exactly the ten examples. The third rule was a poor one that could be dropped. The quality of the learned rules increases as the representation hierarchy is climbed. This shows that the hierarchy makes learning robust against the noise which is produced by the weak ultrasonic sensors. The levels can be regarded as a filter – they only pass the more reliable information to the next higher level.

5. CONCLUSION

In this paper, we introduced operational concepts that integrate action-oriented perceptual features and perceptual-oriented action features. Both psychological and practical arguments indicate the necessity of this integration. We sketched a representation for operational concepts that is grounded in the perception and action of a mobile robot. We proposed two learning tasks: one that finds appropriate basic features in order to summarize sequences of sensor data, and one that characterizes higher-level features (or operational concepts) in terms of features from the level below. Experiments have shown that learning becomes more accurate and the coverage of learned rules increases when progressing to higher levels of the representation hierarchy. This means that even noisy input to learning leads to results that summarize the data such that at the next higher level the noise is reduced. These results are promising, but further learning experiments in other environments need be conducted.

In order to achieve first results, we restricted sensing and action of the mobile robot. For representing the sensory data, we reduced the degrees of freedom of the robot allowing only one-dimensional motions at once. That will lead to problems, if we incorporate a reactive module executing the elementary operations, if this module generates traces where the robot deviates from straight courses. Then, the calculation of basic features is probably not able to counterbalance the divergence of the sensor measurements, because in the moment, we only use linear functions to calculate the features.

Another restriction concerns the sensors used. Sonar sensors are not able to get precise measurements, they cannot be used to discriminate fine structured objects. E.g., it is not possible to detect closed doors, so we used only simple structured objects in the moment. An interesting future investigation might be the integration of another sensor system like a vision system to get further information in specific situations.

The next challenge will be to validate the contribution of learning to easing robot applications. Until now we have a small planning component that computes basic actions of a robot from an operational concept. This component uses learned rules in a backward-chaining manner. The basic perceptual features are incrementally calculated from sensor measurements as shown in this paper (Section 4.1). Using the learned rules in a forward-chaining manner, higher-level features are derived. Using these techniques, learned rules are made available for the robot's processing. First, experiments with the robot are to be made where the robot uses learned rules in the environment in which learning took place. Then, the same rules should be used by the robot in similar but different environments from the one in which learning took place. Finally, the efforts of preparing these robot applications need be compared.

ACKNOWLEDGEMENTS

This work is partially funded by the European Community under the project B-Learn II (P7274) and the Ministry for Sciences and Research of the German federal state Nordrhein-Westfalen.

We would like to thank Anke Rieger for her contributions to representing and learning perceptual features, Stefan Sklorz for developing the representation for action features and programming a meta-interpreter for planning, and Stephanie Wessel for her work on learning basic features. Thanks to Joachim Hertzberg and Stephanie Wessel for reading drafts of this paper.

REFERENCES

Abelson, R. P. (1963). Hot cognition. *Computer Simulation of Personality.*

Bennett, S. W. (1989). Learning uncertainty tolerant plans through approximation in complex domains. Tech. Rep. UILU-ENG-89-2204, University of Illinois at Urbana-Champaign.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2.

Brooks, R. A. (1991). The role of learning in autonomous robots. In Valiant and Warmuth, editors, *COLT'91, Proc. of the fourth Annual Workshop.* Morgan Kaufmann.

Cottrell, G. W., Bartell, B., and Haupt, C. (1990). Grounding meaning in perception. In Marburger, editor, *GWAI-90, 14th German Workshop on AI.* Springer.

De Jong, G. and Mooney, R. (1986). Explanation-based-learning: An alternative view. *Machine Learning*, 2(1):145–176.

Dillmann, R., Kreuziger, J., and Wallner, F. (1993). PRIAMOS – an experimental platform for reflexive navigation. In Groen, Hirose, and Thorpe, editors, *IAS-3: Intelligent Autonomous Systems.* IOS Press.

Gil, Y. (1994). Learning by experimentation – incremental refinement of incomplete planning. In Cohen, W. and Hirsh, H., editors, *Procs. 11th Int. Machine Learning Conference.*

Giordana, A. and Saitta, L. (1990). Abstraction – a general framework for learning. In *Procs. AAAI – Workshop on Automatic Generation of Approximations and Abstractions.*

Harnad, S. (1990). The symbol grounding problem. *Physica D*, 42:335–346.

Jong, G. D. and Bennett, S. (1993). Permissive planning – a machine learning approach to linking internal and external worlds. In *AAAI-93.*

Kedar-Cabelli, S. (1988). Toward a computational model of purpose-directed analogy. In Prieditis, A., editor, *Analogica.* Morgan Kaufmann.

Kietz, J.-U. and Wrobel, S. (1992). Controlling the complexity of learning in logic through syntactic and task-oriented models. In Muggleton, editor, *Inductive Logic Programming.* Academic Press.

Klingspor, V. (1994). GRDT: Enhancing model-based learning for its application in robot navigation. In Wrobel, S., editor, *Proc. of the Fourth International Workshop on Inductive Logic Programming*, GMD-Studien Nr. 237, St. Augustin, Germany.

McCarthy, J. and Hayes, P. J. (1969). Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 5:463–502.

Millán, J. and Torras, C. (1992). A reinforcement connectionist approach to robot path finding in non-maze-like environments. *Machine Learning*, 8.

Mitchell, T. M. (1982). Generalization as search. *Artificial Intelligence*, 18(2):203–226.

Muggleton, S. (1992). *Inductive Logic Programming.* Academic Press, London.

Nelson, K. (1983). The derivation of concepts and categories from event representations. In Scholnick, editor, *New Trends in Conceptual Representation: Challenges to Piaget's Theory?* Lawrence Erlbaum.

Segre, A. (1988). *Machine Learning of Robot Assembly Plans.* Kluwer, Boston.

Steels, L. (1993). Building agents out of autonomous behavior systems. In Steels and Brooks, editors, *The 'artificial life' route to 'artificial intelligence' – Building situated embodied agents.* Lawrence Erlbaum.

Wallner, F., Kaiser, M., Friedrich, H., and Dillmann, R. (1994). Integration of topological and geometrical planning in a learning mobile robot. In *Proc. of IROS-94.*

Wrobel, S. (1991). Towards a model of grounded concept formation. In *Proc. 12th Int. Joint Conf. on AI.* Morgan Kaufman.

Zercher, K. (1992). *Wissensintensives Lernen für zeitkritische technische Diagnoseaufgaben.* infix, Sankt Augustin.