HTN Robot Planning in Partially Observable Dynamic Environments*

Martin Weser, Dominik Off and Jianwei Zhang

TAMS, Department of Informatics, Hamburg University

Vogt-Kölln-Strasse 30, 22527 Hamburg, Germany

{weser,off,zhang}@informatik.uni-hamburg.de

Abstract— Experiments showed that *Hierarchical Task Network (HTN)* planners are suitable to find solutions for non-trivial tasks in complex scenarios. Mobile service robots are able to execute actions which may constitute the basic building blocks to achieve high-level goals. However, only few experiments demonstrate the application of a general purpose deliberative planner in the domain of mobile service robots. One challenging problem arises from the fact that adaptive AI-based planners presume the *closed-world assumption (CWA)* and are therefore unable to deal with incomplete information. Unknown objects which are not represented in the planning domain, for example, cannot be integrated into the planning process. Since mobile service robots act in a real dynamic environment and construct or adapt their world model autonomously based on sensory data, they are inevitably confronted with uncertain and incomplete information about the world. This conflict between simplified assumptions for planning on the one hand and the complexity of the real world on the other constitutes a major problem of modern robotics. This paper describes two approaches to dealing with incomplete world knowledge in the context of HTN robot planning. Several experiments demonstrate that the approaches can successfully be applied in a dynamic and unstructured environment.

I. INTRODUCTION

One of the striking conclusions drawn from literature is that an autonomous service robot has to feature both close sensor-control loops and abstract planning that foresees the effects of the intended course of action. The application of plans to mobile service robots has many advantages: The use of planning mechanisms enables robots to flexibly interleave complex and interacting tasks, exploit opportunities and plan their future courses of action [1]. Attempts to combine planning and control in one integrated homogeneous system are still of a theoretical nature [2], focus on specific applications, or clearly do not reach the level of abstraction that is needed for applied service robots [3]. This paper describes an architecture that integrates a state-of-the-art configurable planner and a robot control system. The architecture has been successfully implemented on the mobile service robot TASER, which is shown in Figure 1.

A major difficulty in integrating AI-based planning into robot control architectures is the need to overcome the limitations of the *closed-world assumption (CWA)*. The CWA assumes that every fact in the modeled part of reality is known. This leads to the assumption that if an atomic fact p cannot be deduced from a given knowledge base KB,
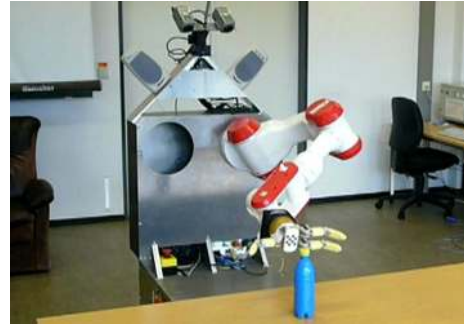
Fig. 1. The mobile robot TASER executing a manipulation task.

then $\neg p$ can be assumed [4]. Based on this implication, a knowledge base KB is extended to a KB^+ in the following way: $KB^+ = KB \cup \{\neg p \,|\, p$ is atomic and $KB \nvDash p\}$.

Partial observability and probabilistic information has been shown to be a hard problem in symbolic planning, both theoretically and experimentally [1], [5]. If applied to realistically large domains, such planners tend to be computationally intractable and are therefore of no practical relevance for intelligent service robots. In the context of mobile service robots, all applicable general purpose planners presuppose the CWA and are therefore unable to directly generate plans that depend on objects or facts which are unknown at planning time. In contrast to artificial theoretic planning domains, real environments are perceived by robot sensors and thus inevitably contain incomplete and unreliable information about the world. Thus we have to find applicable solutions to overcome the limitations of the CWA in applied robotics.

The main contributions of this paper are 1) to describe a control architecture that integrates a state-of-the-art symbolic planner with a library of robot control programs that act as an interface to physical perception and execution of basic actions, 2) to propose practical solutions to overcome limitations of the CWA if applied to the robotic domain, and 3) to demonstrate the applicability of our approach in a set of long-term experiments with the mobile manipulator TASER.

The rest of this paper is organized as follows. The next section reviews the state-of-the-art in plan-based robot control and in planning with partial observability. In Section III the plan-based robot architecture is introduced. Two patterns that enable HTN planners to deal with incomplete knowledge are introduced in Section IV. The feasibility of the patterns is evaluated in Section V. We conclude with a summary and a discussion of future work in Section VI.

II. State-of-the-art in plan-based robot control

This section describes the state-of-the-art in plan-based robot control platforms and planning with partial observability.

A. Plan-based robot control

Plan-based robot control can be considered as the state of the art in autonomous robot systems [6]. Deliberative symbolic planning systems are integrated with reactive robot control [7] to benefit form the strength of both paradigms. In recent years, a number of experiments showed the potential of this research direction. The selection of the presented systems is based on the similarity to the approach proposed in this paper and on the degree to which they integrate planning into robot control.

The interactive tour guide robot MINERVA [8] interacted successfully with thousands of people and traversed more than 44 km during its two weeks of operation. Plan revision policies, the deletion of completed plans and tour scheduling has been used by MINERVA to achieve its tasks. A high-level planner was implemented on top of reactive controllers to reason about the long-term goals of the robot.

The ARMAR-III [9] robot was developed at the University of Karlsruhe in order to act in a household environment. The three-layered software architecture is composed of a task planning layer, a synchronization and coordination layer, and a sensor-actor layer. The planning layer decomposes abstract tasks into sets of subtasks and is responsible for the scheduling of tasks and management of resources and skills. The execution layer works in terms of the control theory and executes dedicated sensory-motor control programs. The middle layer integrates the planning and the sensor-actor layer. It invokes the subtasks from the planning layer sequentially or in parallel on the execution layer.

The robot control system of RHINO [10] consists of a special purpose planning system and several low-level control modules. Each of the control modules is designed to monitor or control a dedicated aspect of the physical robot system. A high-level interface module integrates the different control modules by composing low-level operations to abstract tasks. It allows the planning system to start and terminate these tasks and thereby mediates between reactive robot control and symbolic planning.

All above-mentioned architectures have the use of specifically implemented planning systems in common. Requirements for the particular area of application of the systems are explicitly modeled in the planning domain. To the best of our knowledge, no robot system utilizes a general purpose planner which has been established in the AI community and verifiably provides a good performance. For the proposed architecture in this paper, the HTN planning system SHOP2 will be used as a deliberative component.

B. Planning with partial observability

The closed world assumption is made as part of a scientific modelling process. Whether this assumption is reasonable or not depends on the modelled slice of reality. The assumption simplifies the representation of the actual world state since only positive facts are stated explicitly. Furthermore, the CWA simplifies the planning process considerably and permits efficient planning. Currently all practically applicable planners are based on this assumption.

In the context of a mobile service robot, the CWA is inadequate since it assumes that the world model is a complete representation of the environment. Apparently, it is impossible to endow a robot that acts in a dynamically changing environment with complete knowledge. A planning system for real-world service robots therefore has to deal with partial observability.

Planning based on Markov decision processes or model checking, as well as planning as satisfiability are approaches that are able to deal with partial observability [5].

A comparably new approach tries to enable forward-chaining planners like SHOP2 to natively support partially observable environments. The adaptation of SHOP2 is called *CondShop2* and seems to outperform existing planners that are able to deal with partial observability as a result of initial experiments [11]. However, this approach is in its infancy and further investigations are necessary. Furthermore, an open implementation of CondShop2 is not available to date.

In addition to planning under incomplete knowledge, sophisticated reasoning and knowledge representation capabilities as well as the need for an extendable high-performance implementation are major demands for plan-based robot control. Planning approaches that are able to directly deal with partial observability still perform poorly even in simple scenarios [1], [11], [5]. Thus, these approaches do not meet necessary requirements and are not adequate for a mobile service robot. Summing up, current deliberative planners that natively support partial observability are not applicable to mobile service robots. The approach proposed in this paper permits the generation of plans in the presence of incomplete information while maintaining the performance advantages of a HTN planner.

III. Integrated planning and control on the service robot TASER

The control architecture for the service robot TASER features a symbolic planner as deliberative component. We purposefully did not implement a dedicated specific purpose planner; instead the java implementation of the SHOP2 planning system [12] is used. This approach has the following advantages:

- maintenance and further development is provided by a large open source community
- state-of-the-art methods from theoretical AI are implemented; provably good performance [13]
- the well-defined interface to reactive parts of the architecture permits future replacement of the planning component

A unique feature of the JSHOP2 planner is a two-fold compilation and planning process [14]. While other planners interpret domain descriptions directly, JSHOP2 compiles domain descriptions to programs specifically generated for

that planning domain. This process permits implementation-level optimizations that are otherwise not possible and have not been explored in previous research on AI planning [15].

The system is able to autonomously generate the planning domain and the planning problem based on the current circumstances for each planning process.

The domain file describes primitive robot operators which are directly dependent on the available robot devices and their hierarchical abstractions in the form of HTN methods. It describes generally possible capabilities of the robot platform and does not reflect changing hardware setups of the robot. A change in the hardware setup (e.g. due to malfunction of several sensors) is represented in the problem description. However, the required hardware components are stated in the preconditions of every primitive operator. If, for example, the robot arm is not available, manipulation operators cannot be used in the planning process. Using this approach, the domain description has to be created and compiled only once when the robot system is started and repetitive adaption and compilation are avoided.

In contrast to the planning domain, the planning problem changes whenever the situation changes and whenever the system is confronted with new instructions. Thus it has to be generated prior to every planning process. The compilation process is autonomously triggered by the robot's meta controller every time the robot receives new tasks or the circumstances are changed. The robot circumstances—as already hinted—include the external environment and the availability of the robot devices. This approach makes it possible to find plans even in the case of unavailable robot modalities.

Figure 2 shows the idealized control flow of the overall robot architecture. TASER remains immobile while waiting for new commands. The only active internal processes in such an idle mode are passive perception routines, such as localization. Every time the robot receives a new instruction, it is stored in a task queue which is sequentially executed. As long as the task queue is not empty, the robot generates a problem description based on the current state of the world, the available robot devices, and the given instruction from the task queue. Based on this, the planner generates a plan which is executed sequentially by invoking the control programs of the execution layer according to the listed operators. The process marked with (A) generates the planning problem as described above. Process (B) denotes the activation to the planning process, which results in a solution of the problem. This plan is then sequentially executed (C).

Obviously, the control flow in Figure 2 is the ideal case. Several problems may occur. The following paragraphs categorize possible exceptions and describe how the system behaves in such situations.

1) **Invalid internal world representation**: The symbolic representation of the world may be inconsistent with the external world, for instance, because the external world has changed in the meantime. Examples are doors that have been closed recently or objects that have been taken away.

exception handling: The control program being executed when the exception occurred will change the world representation autonomously in both symbolic and continuous representations. Replanning will be initiated based on the improved world state.

2) **Unavailability of robot devices**: Malfunction of the robot hardware prevents the execution of an action.
exception handling: The executed control program will add a symbol to the world state that marks the particular device as unusable, and causes replanning. The planner will then try to find an alternative plan in which the corrupted device will not be used.

3) **Planning Exception:** The Planner cannot find a solution to a problem.
exception handling: For many situations, a change of the circumstances of the robot is sufficient to enable the planner to find a solution. In such cases the planner may provide a preliminary plan that first changes the robot's position and then tries to find a solution for the problem again. However, this approach has to be specified in the planning domain.

4) **Execution exception**: An action may not succeed due to insufficiencies of the control or the hardware of the robot. This exception may have several reasons, for example the lack of satisfactory perception routines, an upcoming collision which cannot be avoided autonomously at the execution layer, or a singularity in the robot arm trajectory.
exception handling: In such cases, the action will mark its own symbolic representation as unpromising and abort to cause replanning. In this way, the planner tries to find an alternative plan. In order to remove the mark after execution of the task, an operator will be added to the task queue that has no other effect than ridding the symbolic world description of bookkeeping information.

The execution exception states the fallback position for all kinds of exceptions that cannot be attributed to any of the first three cases. As an additional safety precaution, the meta controller checks every two successive problem definitions for similarity to prevent loops in the planning process.

IV. PLANNING UNDER PARTIAL OBSERVABILITY WITH A (STATE-OF-THE-ART) HTN PLANNER

This section describes practical approaches to generate reasonable robot behavior under incomplete knowledge.

A. A pattern for perception routines: deliberative replanning

Goal-directed sequences of robot actions often include perception routines that provide information at execution time which would have been needed at planning time. For example, instructions for the robot can depend on external entities that are not known to the robot at planning time. A robot may be instructed to grasp an unknown object from a table, or it may bring all cups to an unknown table in the kitchen. Since the unknown objects are not part of the
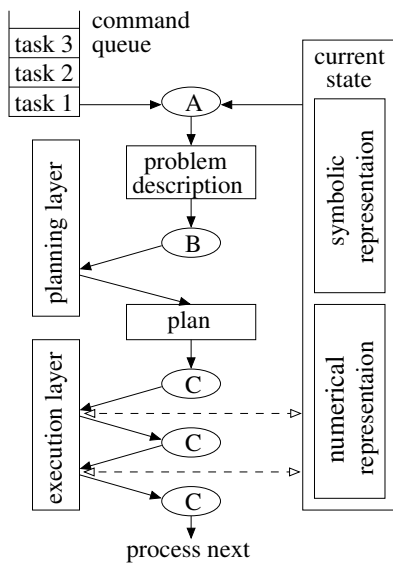
Fig. 2.   Control Flow of the overall robot architecture

```
(:method (pick_from ?table)
 ((use arm) (use ptu) (use camera)
  (use mobile) (table ?table)
  (cup object?) (on object? ?table))
 ((approach ?table)
  (!localize camera ?cup)
  (!reach_for ?cup) (!grab ?cup)
 )
 ; without knowledge of object
 ((use arm) (use ptu) (use camera)
  (use mobile) (table ?table)
  (not (nosuccess find_cup)))
 ((approach ?table) (!find_cup)(!replan))
)
```

Fig. 3.   Listing for (pick_from ?table) task.

symbolic world state, an unmodified HTN planner cannot provide solutions to such tasks.

The proposed approach to deal with such indefinite tasks is to first generate a (possibly complex) plan to acquire the unknown information physically via perception, and than reinvoke the planner to generate a solution to the initial task based on the improved world state. This behavior is achieved in four steps: planning for perception, execution of perception, planning for initial task based on additional knowledge, execution of plan. In order to invoke the replanning process, a virtual primitive operator is defined that causes the robot to replan deliberatively. This procedure is implemented in the symbolic HTN description of abstract tasks which may depend on unknown objects. The (pick_from ?table) task with unknown object to grasp is defined as shown in Listing 3. Two possible decompositions for this task are defined[1]: first, if an object on the table is represented in

[1]The syntax is similar to PDDL and does not require further explanation.

the world state, the abstract task will be substituted by an ordinary plan to approach the table, localize the object and pick it up. Second, if the object is not known, the task decomposition will let the robot approach the table, search for an object and generate a plan based on the acquired knowledge. The (find_cup) operator changes the state of the world according to the results of its own execution. If the object of interest is added to the symbolic state of the world, the planner will find a valid plan in the second attempt. A bookkeeping literal (nosuccess find_cup) will be added if no object could be found and prevents infinite looping of planning and execution of the perception.

In this simple example, the pattern of perception and replanning appears trivial. However, more complex plans may be generated to acquire the needed knowledge. The pattern is applied in tasks that depend on unknown objects at planning level. It provides a solution to the problem of the planner's inability to handle unknown objects and turns around the usual dependency of the planning and execution layer. Instead of being triggered by the deliberative layer, the execution layer invokes deliberative planning actively.

B. A pattern for predictable perception routines: using exceptions

As already mentioned, applicable state-of-the-art deliberative planners, e.g. JShop2, are unable to plan with incomplete information. Acquiring the needed information via perception is—as proposed in the previous section—suitable to generate adequate robot behavior under incomplete information. However, replanning is a resource and time consuming process. Furthermore, the interplay between perception and replanning requires possibly complex bookkeeping mechanisms.

The approach described in this section is able to prevent the above drawbacks in the case of a predictable outcome of perception routines. In such cases, the planning layer generates a solution for the task based on the predicted result of perception. This avoids repetitive planning if the perception does not come up with unexpected results. In other cases, the plan will fail and a replan exception will be handled as described in Section III.

For example, the assumption that most doors in the environment are usually open can be used to generate plans to cross a door without opening it actively. This plan can be generated directly without definite knowledge about the state of the particular door. Technically, the presumed information is artificially added to the world state. For each assumption a confirmation-operator is added to the generated plan.

Using this approach, replanning will be caused only in the unlikely case of a closed door as a result of the unsuccessful physical execution of the (!confirm...) operator.

Common applications for this approach to deal with incomplete world knowledge are human instructions that contain implicit information. For example, (grasp_from ?table) implicitly claims that an object exists on the particular table.
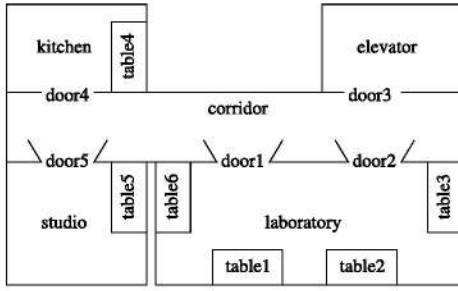
Fig. 4. Simplified Environment

(a) Taser grasps a bucket   (b) Taser opens a door

(c) Taser detects a cup using SIFT features   (d) Taser grasps a cup

Fig. 5. Several robot skills that where executed in different experiments

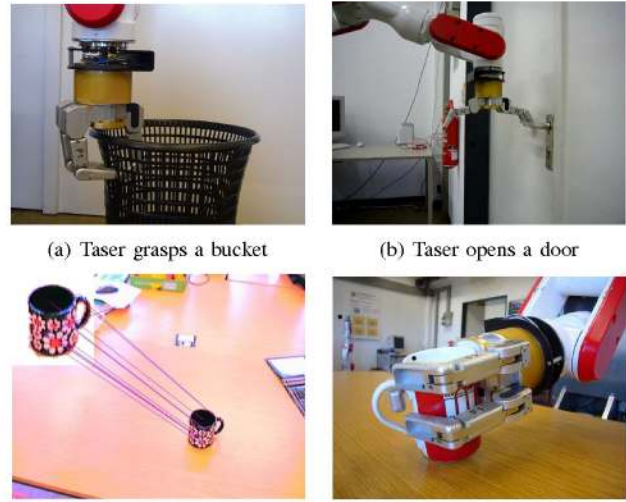## V. EXPERIMENTS AND RESULTS

We made several experiments in order to demonstrate the applicability of the proposed approaches to dealing with incomplete knowledge in the context of symbolic planning in the domain of mobile service robots. A multitude of robot skills were used for the purpose of fulling the given tasks. A few example robot skills are shown in Figure 5(a) - 5(d). Describing all the made experiments would go beyond the scope of this paper. In this section we therefore only describe one experiment that illustrates all important aspects of the proposed patterns.

All experiments took place on the floor of our robotic laboratory. Figure 4 shows a schematic view of our experimental environment. It consists of four rooms that can be accessed by the robot. All rooms are connected with the corridor, the laboratory is connected to the corridor via two doors. Six tables that can be accessed by the robot to put objects on or grasp objects from are distributed over these rooms . The robot is able to grasp and place buckets in every room and the corridor. Several different objects like cups and small bottles are distributed on the tables. All of them are symbolically labeled as cup since only the execution layer is affected by the type of an object. For all experiments, the robot initially started in the laboratory with an empty hand.

In the presented experiments, the robot had the task of bringing a bucket from the laboratory to the elevator. This experiment was carried out under three different conditions:
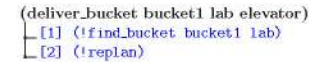
1) The artificially added assumption conforms to reality and the door to be passed is open.
2) The door that the robot had to pass was closed unexpectedly. Thus the robot was in the unlikely situation that the artificially added assumption about the state of the door turns out to be wrong. The initial plan could not be achieved and the robot had to replan based on the changed state.
3) The bucket is symbolically unknown at planning time. The deliberative replanning pattern (see section IV-A) will be applied in order to make planning possible.

`door1` is assumed to be open. This assumption is probably but not certainly correct. For the purpose of dealing with this incomplete information while avoiding unnecessary replanning, the second pattern that has been introduced in section IV-B is applied in all scenarios.
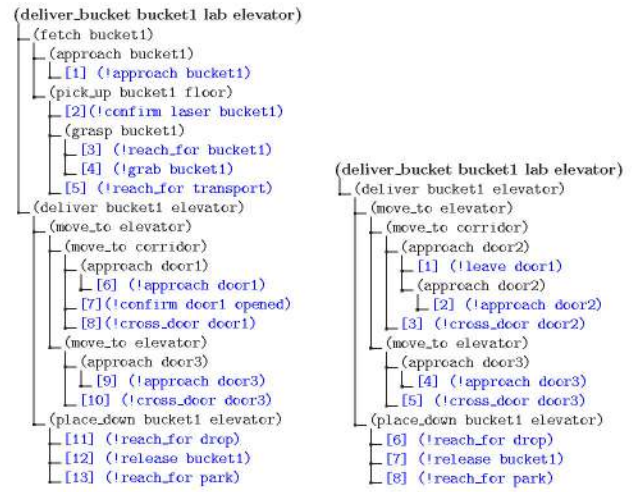
(a) Initial Plan in the case of correct assumptions   (b) Initial Plan in the case of incorrect assumptions

(c) Final Plan in the case of an unknown object

Fig. 6. Initial plans for the first and the third experiment, and the final plan for the second experiment

A. Planning with correct assumptions

In the first experiment, the artificially added assumption conformed to reality and no unexpected events were introduced. Therefore the plan could be executed successfully and additional replanning was not needed. The resulting plan is shown in figure 6(a).

B. Planning with incorrect assumptions

Under condition 2, the assumption that `door1` is open was not confirmed by the corresponding perception routine. Thus, the robot was in the improbable situation that the assumed

door state turned out to be wrong. The initial plan could not be achieved and the robot had to replan based on the changed state. The planner scheduled to pass the second door which also connects the laboratory with the corridor. This required fewer primitive actions to be executed than opening the door. The finally generated plan is shown in figure 6(b)

C. Planning with an unknown object

Under the third condition, the bucket does not exist in the planner's symbolic world model. The deliberative replanning pattern (see section IV-A) was applied in order to enable planning in the presence of incomplete information. A perception routine was planned and executed in order to find the unknown bucket in the laboratory. The bucket was found and added to the symbolic state representation. Subsequently, the planner was invoked to generate a solution to the initial task based on the improved world model. The initially generated plan is shown in Figure 6(c). Finally, replanning produced a plan that looks exactly like the plan from the first experiment, which is shown in Figure 6(a).

D. Results

The trouble-free execution of the task under the first condition took 1:40 minutes in total.

Recognition of the unexpectedly closed door, generation and compilation of a revised problem description and planning under the third condition took less then 10 seconds altogether and is thus negligible. The detour through the alternative door took 1:54 minutes in the experiment. This is more time than the trouble-free execution (condition 1) needed in total. However, it is still less than opening the door, which requires more than two minutes.

The time needed for the compilation of the improved world model and replanning under condition two and three was negligible. Perception of the unknown bucket under condition three cannot be avoided and took 47 seconds.

VI. DISCUSSION AND CONCLUSION

The plan-based robot control paradigm is the most promising approach to endow robots with autonomy and intelligence. Current symbolic planners are usually based on the closed world assumption which cannot hold in the domain of real-world robots that construct their world model autonomously. This makes it a challenging task to integrate such planners in robot control systems.

This paper described the integration of a state-of-the-art symbolic planner with the control architecture of the service robot TASER. Two approaches are presented to deal with the CWA in applied robotics. They are characterized by patterns of interleaved symbolic planning and physical execution. Default assumptions about the world are used in the planning process whenever available. Several experiments showed that the proposed approaches lead to reasonable robot behavior even under incomplete world knowledge.

Although every plan-based robot control architecture is inevitably confronted with difficulties that arise from incomplete world knowledge, this problem has not been addressed explicitly. Most of the existing systems make use of planners that are specifically developed for certain tasks. Our approach, in contrast, preserves all advantages of general purpose planning and utilizes sophisticated AI reasoning techniques. This greatly simplifies the design of the planning domain since elaborated tools for definition and consistency checks are available. We believe that this contributes to the future development of intelligent systems. The high performance of optimized symbolic planning allows for a comparably low abstraction level of the atomic symbols and thus makes autonomous sensor-based construction of the current world state possible.

Directions of future work include an extended HTN planning approach that natively supports an incomplete world model, the integration of planning and rich knowledge representations, and the autonomous integration of external knowledge sources into the planning process.

REFERENCES

[1] Michael Beetz. Towards comprehensive computational models for plan-based control of autonomous robots. In *Mechanizing Mathematical Reasoning*, pages 514–527, 2005.

[2] Michael Beetz. Structured reactive controllers. *Autonomous Agents and Multi-Agent Systems*, 4(1-2):25–55, March 2001.

[3] K. H. Low, W. K. Leow, and M. H. Ang, Jr. A hybrid mobile robot architecture with integrated planning and control. In *Proc. 1st International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS-02)*, pages 219–226, 2002.

[4] Ronald J. Brachman and Hector J. Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufman Publ Inc, 2004.

[5] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning Theory and Practice*. Elsevier Science, 2004.

[6] Malik Ghallab, Rachid Alami, Joachim Hertzberg, Maria Gini, Maria Fox, Brian Williams, Bernd Schattenberg, Daniel Borrajo, Patrick Doherty, Jos M. Morina, Araceli Sanchis, Patrick Fabiani, and Martha Pollack. A roadmap for research in robot planning, 2006.

[7] Rodney A. Brooks. A robust layered control system for a mobile robot. In *ICRA*, volume RA-2, pages 14–23, 1986.

[8] S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. MINERVA: A second generation mobile tour-guide robot. 1999.

[9] T. Asfour, K. Regenstein, P. Azad, J. Schrder, A. Bierbaum, N. Vahrenkamp, and R. Dillmann. ARMAR-III: An integrated humanoid platform for sensory-motor control. In *IEEE-RAS International Conference on Humanoid Robots*, 2006.

[10] Michael Beetz, Tom Arbuckle, Thorsten Belker, Armin B. Cremers, Dirk Schulz, Maren Bennewitz, Wolfram Burgard, Dirk Hähnel, Dieter Fox, and Henrik Grosskreutz. Integrated plan-based control of autonomous robots in human environments. *IEEE Intelligent Systems*, 16(5):56–65, 2001.

[11] Ugur Kuter, Dana Nau, Elnatan Reisner, and Robert Goldman. Conditionalization: Adapting forward-chaining planners to partially observable environments. In *Proceedings of the ICAPS-07 Workshop on Planning and Plan Execution for Real-World Systems*, 2007.

[12] Dana Nau, Tsz C. Au, Okhtay Ilghami, Ugur Kuter, William Murdock, Dan Wu, and Fusun Yaman. SHOP2: An HTN planning system. *Journal on Artificial Intelligence Research*, 20, 2003.

[13] Derek Long and Maria Fox. The 3rd international planning competition: Results and analysis. *Journal of Artificial Intelligence Research*, 20:1–59, 2003.

[14] Okhtay Ilghami and Dana S. Nau. A general approach to synthesize problem-specific planners. Technical report, 2003.

[15] Okhtay Ilghami. Documentation for JSHOP2. Technical Report CS-TR-4694, University of Maryland, Department of Computer Science, 2006.