A General-purpose Transportation Robot
An Outline of Work in Progress

Mattias Wahde and Jimmy Pettersson

***Abstract*— An outline of a current joint project between Chalmers University of Technology (in Sweden) and several Japanese universities (Waseda University, Future University, and the University of Tsukuba) is presented. The aim of the project is to build a general-purpose transportation robot for use in hospitals, industries, and similar facilities. The project will provide a thorough test of the recently developed utility function method for behavior selection, which will be used for generating the decision-making system in the transportation robot.**

In this paper, an outline of the proposed transportation robot is given, along with a brief description of some of the challenges arising from this project. Furthermore, the utility function method is presented. Finally, the results obtained thus far are briefly discussed, and some directions for further work are provided.

I. INTRODUCTION

The combination of reduced hardware prices and the development of behavior-based (and related) techniques [2] has led to a rapid development of autonomous robots during the last two decades. Some of the tasks carried out by such robots include vacuum cleaning [16], entertainment [1] or general assistance to people, either at their place of work [7] or in their home [13], [15].

Another task that could potentially be carried out by robots is internal transportation (or delivery), i.e. the task of reliably moving objects from an arbitrary point A to another arbitrary point B in some (indoor) environment, without human supervision. Robots equipped with the means of carrying out such a task would be useful for internal transportation of various objects in hospitals, offices, or factories.

The development of a transportation robot is the main goal of a current joint project involving researchers at Chalmers University of Technology, in Göteborg, Sweden, Waseda University in Tokyo, University of Tsukuba, and Future University in Hakodate. Similar robotic platforms are being developed within the framework of other projects as well, e.g. the TUG robot [19], the Xavier robot [17], and the MB385 mobile transportation system [10].

While the definition of the problem may appear to be quite simple, the problem poses several difficult challenges that will be described in Sect. II below. The challenges pertain to hardware and software alike. On the hardware side, the construction of the robot and, in particular, the choice of an adequate set of sensory modalities must be considered carefully. On the software side, the problem of behavior selection is the main challenge. In Sect. III, a brief description of the utility function method for behavior selection will be given, and in Sect. IV, a brief discussion of the results obtained so far will be presented, along with a brief outline of future work.

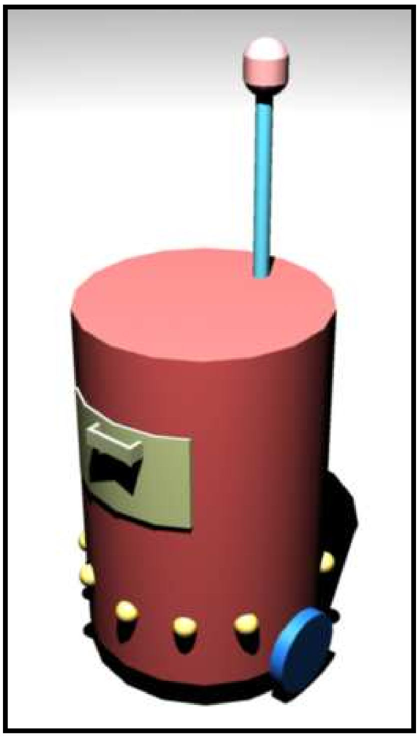

Fig. 1. A schematic illustration of the transportation robot. The laser range finder is located at the top of the pole.

II. PROJECT OUTLINE

As mentioned above, the main goal of this project is to generate an autonomous robot capable of reliable internal transportation in indoor settings. An additional goal, however, is to test (and further develop) the utility function method for behavior selection. The transportation robot will constitute one of the first stringent tests of this method outside a laboratory setting.

A schematic drawing of the (differentially steered) robot is shown in Fig. 1. It is assumed that the brain of the robot[1] has been equipped with a map of the stationary parts

---

Both authors are with the Dept. of Applied Mechanics, Chalmers Univ. of Technology, Goteborg, Sweden. Corresponding author: Mattias Wahde, `mattias.wahde@chalmers.se`

[1] The first prototype of the transportation robot will use a laptop computer. In later versions, a set of microcontrollers might be used.

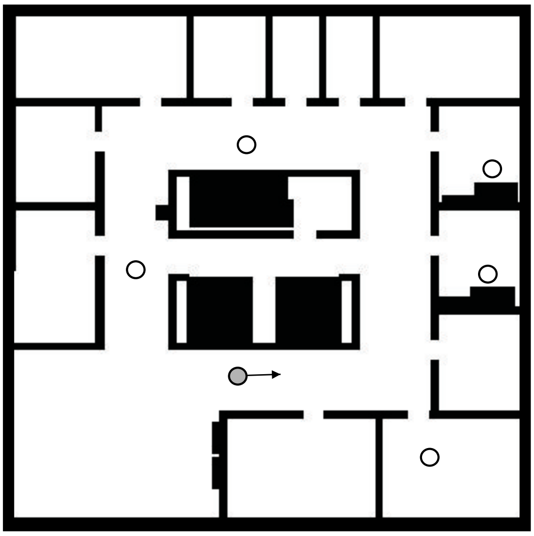

Fig. 2. An example of a typical environment for the transportation robot. The robot is shown as a filled circle, and the open circles indicate moving obstacles (e.g. people).

(e.g. the walls and doorways) of the arena. The sensory modalities involve (1) an array of IR sensors (or, possibly, a sonar array) for proximity detection, (2) a (2D) laser range finder, to be used for localization, in conjunction with digital optical encoders (one for each wheel), (3) a battery sensor, measuring the amount of energy available in the onboard battery, and (4) bumper sensors for detecting collisions. However, the robot will *not* be equipped with a GPS localization system[2]. Furthermore, it is assumed that the compartment of the robot used for transporting objects (hereafter: the transportation compartment) is equipped with scales, so that it can determine whether or not it is carrying an object[3].

An example of a typical arena for the transportation robot is shown in Fig. 2. The arena can represent, for example, a hospital ward, an office floor, or a factory. In this (schematic) figure, the robot is represented as a filled circle, and moving obstacles (e.g. people) as open circles. A brief description of a typical task for this robot will now be given.

A. Basic functionality

In a typical situation, the robot will start at (an arbitrary) point A, as indicated in the upper left panel of Fig. 3. A user will open the door to the transportation compartment, and place an object there. The robot will measure the weight of the object, giving a warning should the object be too massive. Next, the user will enter (via an, as yet unspecified, user interface) the intended navigation goal (point B) of the robot. The position of the navigation goal can possibly be given in the form of coordinates (x, y) or, more simply, chosen from a list of allowed positions, supplied to the robot in connection with the map. Possibly, for calibration, the robot may request information concerning its current position. Next, the robot will activate its *navigation* behavior (B1), generating a path towards its target location (point B), and begin moving. The path will be generated using the A* algorithm [6], which has been integrated with the UFLibrary software package, see Sect. III below. During the motion, the robot will constantly update its measured position through integration of the kinematic equations using the information supplied by the digital optical encoders. In addition, the robot will check continuously its immediate surroundings for obstacles. Should such an obstacle be detected in the direction of motion, the robot will suspend B1 and instead activate an *obstacle avoidance* behavior (B2). In B2, the robot will first stop moving in order to make sure that it does not collide e.g. with a person. Next, the robot will wait for a moment to see if the obstacle disappears. If it does not, the robot will then attempt to circumnavigate the obstacle, as indicated in the upper right panel of Fig. 3, again keeping track of its position, using the odometric readings. Once free of the (stationary or moving) obstacle, the robot will again activate B1, generating a new path towards point B, and resume its navigation.

Clearly, at some stage, the drift in the odometry will begin to pose problems. This is indicated in the lower left panel of Fig. 3, where the dashed circle indicates the position as perceived by the robot which, at this stage, differs from the actual position (indicated by the filled circle). Now, the robot should re-calibrate its odometric readings, and will thus activate an *odometry calibration* behavior (B3). The re-calibration will be carried out by matching the current readings of its laser range finder to the readings obtained at a given snapshot. Thus, a further assumption will be that a number of such laser range finder snapshots have been stored in advance, for example in connection with the storage of the map. The snapshots can either be in the form of a finite number of actual laser range finder readings, or in the form of estimates, for any point in the arena, based on the map. The former case is illustrated in the lower right panel of Fig. 3, where the snapshot points are indicated as small filled squares. Some of the rays of the laser finder are shown as well, as the robot attempts to match its current readings to those obtained at a nearby snapshot p.

Provided that the robot carries out the calibration with sufficient frequency (see Subsect. II-B below), it will only need to try to match its current location to the nearest snapshot. Once the (far from trivial) matching has been completed, the robot can again resume operation of its *navigation* behavior (B1).

Upon reaching the target location (point B), the robot will activate a *waiting* behavior (B4), in which it simply remains at standstill until a user removes the object it is carrying, and possibly gives the robot a new task.

Additionally, the robot will be equipped with an *emergency* behavior (B5) which can be activated if, despite its efforts to find point B, it finds itself stuck or lost.

---

[2] In general, the GPS signal is too weak to penetrate the walls of buildings. This problem can be solved, see e.g. [9], but here it will, nevertheless, be assumed that neither GPS nor any similar system for indoor applications is used.

[3] The maximum weight for objects transported by the robot will be around 20 kg.

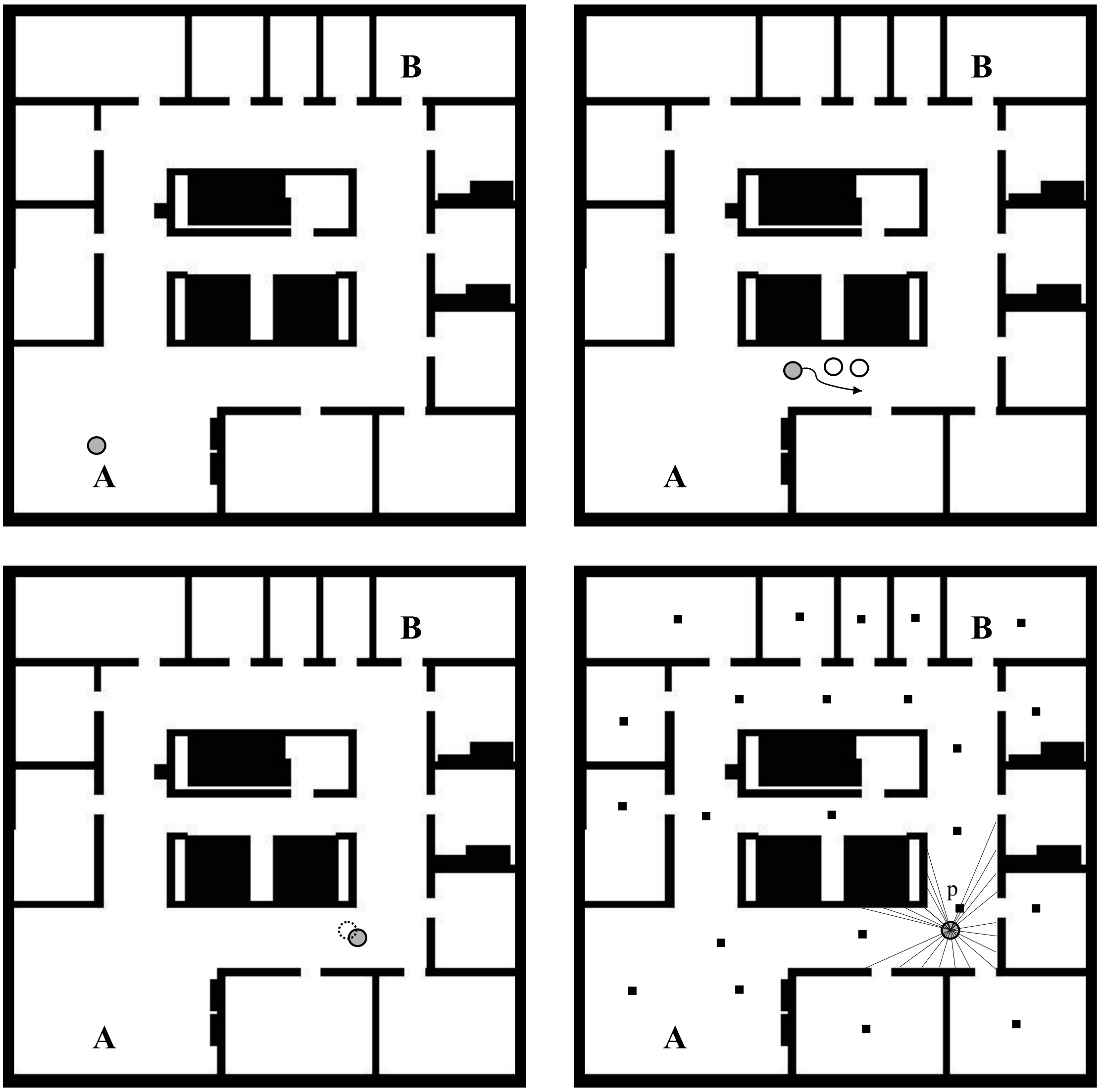

Fig. 3. A schematic illustration of a sequence of typical situations encountered by the transportation robot. See the main text for a more detailed description of the four panels in this figure.

The battery of the robot should be such as to allow continuous operation for several hours, and preferably for a full working day. This might be difficult to achieve and the robot should therefore be equipped with a *battery charging* behavior (B6), allowing it to locate a charging station and charge its batteries when needed. However, for simplicity, the first prototype of the robot will not be equipped with such a behavior. Thus, the problem of battery charging will not be considered further in this paper. Even in the absence of battery charging, the development of a robot capable of carrying out the task outlined above will be a challenging task. Here, a few details concerning some of the challenges will be given.

B. Challenges

1) Safety: First and foremost, the robot must operate in a safe way, i.e. it must never collide with people or obstacles. Note that, even though the robot is equipped with a map, it may still encounter stationary obstacles. An example is the case of a hospital environment, where many different stationary (but movable) objects may be present in certain situations, and absent in others.

In an encounter with a single person, avoiding collisions is not so difficult. However, in a congested environment, the problem becomes more difficult. For example, if the robot moves backwards quickly in order to avoid a collision, it may bump into a person behind the robot. The robot's first action, therefore, will always be to stop if an obstacle is detected in the direction of motion. This will have the additional benefit of making the robot's behavior predictable from the point-of-view of the people working in the same environment.

Another possibility will be for the robot to choose a different way, in case its current path (as obtained from the A* algorithm) is blocked. Here, however, the robot must be careful not to change its path too frequently, as this may result in a considerable delay in the delivery of the transported object.

2) Snapshot matching: As is well known, reliable self-localization is a common difficulty encountered in navigation problems involving autonomous robots [18], [5].

In order to recalibrate its odometry, the robot developed in this project must find and match its current location against stored snapshots. Clearly, other options exist for localization, such as e.g. the NorthStar system [12]. However, this project is aimed at achieving navigation without any adaptation of the environment, such as installation of beacons, transmitters, or other hardware [9]. In addition, the snapshot matching method has a biological equivalent in the procedure used by some species of ants [8], [3], [4], and is interesting in its own right, particularly in the light of the biologically inspired approach to behavior selection defined by the utility function method. The snapshot matching could, of course, also be based on vision using two video cameras, and the use of binocular vision is certainly retained as a possibility. However, the simulations carried out so far have indicated that the 2D laser range finder ought to be sufficient for the snapshot matching, provided that it is carried out frequently.

3) Sensory integration: In order for the robot to operate robustly, it should preferably be able to navigate even if some sensory modality fails. For example, if the IR proximity sensors suddenly break, the robot should be able to switch to alternative proximity detection methods, e.g. based on the laser range finder readings. This would not be optimal, since the range finder will be located at a different height than the proximity sensors, and may therefore miss certain obstacles that would have been detected by the IR sensors. A possible solution, in case of IR sensor failure, is to navigate more slowly, using a combination of the readings from the laser range finder and the bumper sensors. An alternative approach is to provide the robot with sensor redundancy, using e.g. two sets of IR proximity detectors, or a sonar. The problem of sensor failure can thus be solved either mainly as a software problem (dynamically switching from IR sensors to the laser range finder in case the former break down) or mainly as a hardware problem (providing the robot with redundant sensors).

4) Behavior selection: From a software point of view, one of the main challenges is behavior selection. The problem is made more difficult by the fact that the robot will operate in an unstructured, rapidly changing environment. Clearly, the robot must always avoid collisions with people (see Subsect. II-B.1 above) or stationary obstacles, but it will nevertheless operate under conditions that require a certain trade-off: If the robot is made *too* careful, it will most likely move too slowly to be useful. A similar problem will occur if, for example, the robot misjudges the amount of congestion along a certain path and unnecessarily selects a much longer path. Thus, finding the right balance between efficiency on the one hand, and safety and self-preservation on the other, is likely to be one of the main challenges encountered during the evolution of the behavior selection system.

Another, related, challenge is to evolve a behavior selection system that is sufficiently general, so that it can cope with any situation (within reasonable limits) that may occur. In view of the rather long time taken to evaluate robots in simulations, this problem will be far from trivial.

III. THE UTILITY FUNCTION METHOD

Behavior selection (also called behavioral organization or action selection), i.e. the problem of activating (in any situation) the correct behavior among the behaviors available in a robot's behavioral repertoire, is a challenging task that has been approached in many different ways (see e.g. [14] for a review).

The utility function (UF) method [22], [23], [21] is a method for behavior selection based on evolutionary optimization of utility functions. It is described in detail by Wahde [22] and therefore only a brief introduction will be given here.

A. Brief description

The UF method is an arbitration method, i.e. a behavior selection method in which a single behavior is active at any given time. The method deals with the *selection* of behaviors that are already present. Thus, in order to apply the method, one must first generate a set of basic behaviors (e.g. by hand, in simple cases, or using evolutionary optimization in more complex cases). Some examples of behaviors are described in the project outline above.

In the UF method, a set of state variables is defined. These can be of three kinds: (1) External variables (denoted $\mathbf{s}$) based e.g. on the readings of IR proximity sensors or a laser range finder, (2) internal physical variables (denoted $\mathbf{p}$) measuring e.g. the energy level in the robot's batteries and (3) internal abstract variables (denoted $\mathbf{x}$), whose variation may be either hand-coded or evolved, and which roughly correspond to (the action of) hormones in biological systems. For example, an internal abstract variable can be used to model fear. In that case, its value would rise e.g. in cases where a collision or battery depletion is imminent.

Each behavior B_i contained in the brain of the robot is associated with a utility function U_i that depends on (a subset of) the state variables, i.e.

$$U_i = U_i(\mathbf{s}, \mathbf{p}, \mathbf{x}), \;\; i = 1, \ldots, n, \tag{1}$$

where n is the number of behaviors available.

Once the utility functions have been generated, behavior selection is straightforward: At any given time, the robot simply activates the behavior corresponding to the largest utility value, i.e.

$$i_{\mathrm{active}} = \mathrm{argmax}\,(U_i)\,, \tag{2}$$

where i_{active} denotes the index of the currently active behavior. Thus, in this method, the utility values are used as a common currency [22], [11] allowing the robot to assess, on a dynamical basis, the relative merit of the different behaviors.

The problem, of course, is to generate the utility functions. In the UF method [22], the utility functions are optimized by means of an evolutionary algorithm. This procedure is carried

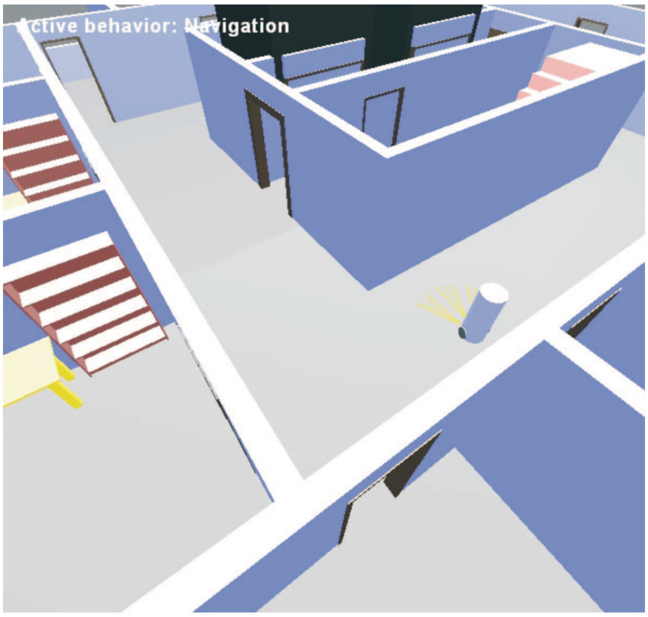

Fig. 4. A snapshot from a simulation based on the UFLib simulation package.

out in simulations, based on the UFLib software package, which will now be described briefly.

B. *Software*

The application of the UF method requires that many different behavior selection systems (i.e. sets of utility functions) should be evaluated. In order for this to be possible, simulations must normally be used. The UF method has been implemented in the UFLib software package [20]. Written in Delphi (object-oriented Pascal) the UFlib package contains software for 3D simulation of wheeled robots in arbitrary arenas, using the UF method for behavior selection. The package also implements an evolutionary algorithm allowing evolution of the utility functions that determine the behavior selection. UFLib supports multiple evaluations, so that each behavior selection system can be tested in a variety of situations. Furthermore, the software package supports the use of behavioral hierarchies, i.e. layers of sub-behaviors within each behavior. However, these concepts will not be described further in this paper. Note that the current version of UFLib can be downloaded for academic use [20].

IV. PRELIMINARY RESULTS

Until the present time (June 2006), the main work in the project has been the development of the necessary software. A significant amount of time has been spent on completing the UFLib [20], and testing it in various circumstances [21], [23]. Furthermore, all of the required behaviors (B1 - B5 as listed above), except B3, have been completed. In particular, the *navigation* behavior B1 has been finalized and thoroughly tested in simulation. A snapshot from such a test is shown in Fig. 4.

Recently, a specification of the hardware requirements has been made, and the initial design phase has been started.

A. *Future work*

The next step is to complete the hardware design, and then to begin hardware construction. Obviously, this will be an iterative process, involving both system identification aimed at making the simulator as accurate as possible, and repeated modification of both hardware and software.

The aim is to have a working prototype completed in the spring of 2007.

ACKNOWLEDGEMENTS

The authors would like thank Mr. Krister Wolff (Chalmers Univ. of Technology), Drs. Pitoyo Hartono (Future Univ., Hakodate) Kenji Suzuki (Tsukuba Univ.), Ryo Saegusa (Waseda Univ.), and Prof. Shuji Hashimoto (Waseda Univ.) for helpful discussions during the preparation of this paper. Also, the project participants wish to thank the Carl Trygger foundation for its financial support for this project.

REFERENCES

[1] AIBO, (Sony, www.sony.net/Products/aibo).

[2] R. Arkin, *Behavior-based robotics*. MIT Press, 1998.

[3] T. S. Collett and M. Collett, “Memory use in insect navigation,” *Nature Reviews Neuroscience*, vol. 3, pp. 542–552, 2002.

[4] M. O. Franz, B. Schöllkopf, H. A. Mallot, and H. H. Bülthoff, “Where did i take that snapshot? scene-based homing by image matching,” *Biological Cybernetics*, vol. 79, pp. 191–202, 1998.

[5] U. Frese, “A discussion of simultaneous localization and mapping,” *Autonomous Robots*, vol. 20, no. 1, pp. 25–42, 2006.

[6] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths in graphs,” *IEEE Transactions on Systems Science and Cybernetics*, vol. SSC-4, no. 2, pp. 100–107, July 1968.

[7] HRP 2, (Kawada Industries www.kawada.co.jp/global/ams).

[8] S. P. D. Judd and T. S. Collett, “Multiple stored views and landmark guidance in ants,” *Nature*, vol. 392, pp. 710–714, 1998.

[9] C. Kee *et al.*, “Development of indoor navigation system using asynchronous pseudolites,” in *Proceedings of ION GPS-2000*, 2000, pp. 1038–1045.

[10] MB385, (BlueBotics, www.bluebotics.com/automation/MB835/).

[11] D. McFarland and T. Bösser, *Intelligent behavior in animals and robots*. Cambridge, MA, USA: MIT Press, 1993.

[12] Northstar (Evolution robotics, www.evolution.com).

[13] Papero (NEC, www.incx.nec.co.jp/robot).

[14] P. Pirjanian, “Behavior coordination mechanisms – state-of-the-art,” Institute of Robotics and Intelligent Systems, USC, Los Angeles, Tech. Rep., 1999.

[15] M. E. Pollack *et al.*, “Pearl: A mobile robotic assistant for the elderly,” in *AAAI Workshop on Automation as Caregiver*, August 2002.

[16] Roomba (iRobot, www.irobot.com).

[17] R. Simmons, R. Goodwin, K. Z. Haigh, S. Koenig, and J. O’Sullivan, “A layered architecture for office delivery robots,” in *Proceedings of the first international conference on Autonomous agents*, 1997, pp. 245–252.

[18] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. The MIT Press, 2005.

[19] TUG (University of Maryland, www.umm.edu/news/releases/robot.html).

[20] www.me.chalmers.se/˜mwahde/robotics/UFMethod/UFLibrary.

[21] M. Wahde, J. Pettersson, H. Sandholt, and K. Wolff, “Behavioral selection using the utility function method: A case study involving a simple guard robot,” in *Proc. of the 3rd Int. Symp. on Autonomous Minirobots for Research and Edutainment (AMiRE 2005)*, 2005, pp. 261–266.

[22] M. Wahde, “A method for behavioural organization for autonomous robots based on evolutionary optimization of utility functions,” *Journal of Systems and Control Engineering*, vol. 217, pp. 249–258, 2003.

[23] M. Wahde and J. Pettersson, “Application of the utility function method for behavioral organization in a locomotion task,” *IEEE Trans. Evol. Comp.*, vol. 9, no. 5, pp. 506–521, 2005.