A general-purpose method for decision-making in autonomous robots

Mattias Wahde

Department of Applied Mechanics, Chalmers University of Technology,
412 96 Göteborg, Sweden
{mattias.wahde@chalmers.se}

Abstract. In this paper, it is argued that the standard taxonomy of behavior selection is incomplete. In order to overcome the limitations of standard behavior selection, a novel method for decision-making, the extended utility function (EUF) method, has been developed. Based on the concept of utility as a common currency for decision-making, the method handles decision-making involving both cognitive processes and (motor) behaviors, and is applicable as a general-purpose framework for decision-making in autonomous robots (as well as software agents). The EUF method is introduced and described, and it is then illustrated by means of an example. Preliminary tests indicate that the method performs well, allowing users rapidly to set up a decision-making system.

Key words: Intelligent systems, autonomous agents

1 Introduction

Traditionally, decision-making in behavior-based robotics has been referred to as *behavior selection* or *action selection* (see e.g. [1] for a recent review). The standard taxonomy of behavior selection systems [2] identifies two main approaches to the problem. In *arbitration methods*, normally a single behavior controls the robot at any given time, other behaviors being inactive. By contrast, in *command fusion methods*, the action taken by a robot represents a weighted average of the actions suggested by several different behaviors. However, this taxonomy is incomplete. First of all, it is strongly directed towards motor behaviors, i.e. behaviors that use one or several motors in order to move the robot or a part thereof. Indeed, the entire approach of behavior-based robotics (BBR) has been criticized for its inability to generate solutions to anything other than simple toy problems. In BBR, one commonly ties action directly to sensing; in other words, not much (cognitive) processing occurs. Second, even though cooperative methods (such as, for example, potential field navigation [3]) allow a robot to generate an action as a weighted average of the suggestions from several elementary behaviors, the taxonomy described above does not include methods that allow, for example, one or several cognitive processes ("thinking", ranging from very simple low-level skills to much more complex processes) to execute simultaneously with a motor behavior ("acting").

The purpose of this paper is to introduce a new approach to robotic decision-making, referred to as the *extended utility function* (EUF) *method*. The EUF method aims to overcome many of the limitations inherent in the methods defined in the framework of the taxonomy described above and to allow users of the method to define robots displaying sufficiently advanced overall behavior to be useful in industrial and other applications.

2 The EUF Method

The EUF method has been developed as an extension of the utility function (UF) method [4], which, being an arbitration method in the taxonomy described above, has mainly been used for handling behavior selection among motor behaviors. The field of behavior-based robotics includes a bewildering array of terms that are often used slightly differently by different authors. Thus, before describing the actual method, an attempt will be made to clarify the various terms used in connection with the EUF method.

2.1 Nomenclature

The term *robotic brain* will be used to describe the program (running in the processor(s) of the robot) responsible for making decisions and taking actions. The term *control system* is specifically *not* used, as it signifies the more limited systems defined in the field of classical control theory. This is not to say that classical control is irrelevant: On the contrary, in (motor) behaviors (see below), the basic actions taken are often formulated in the form of classical control systems, such as e.g. PID controllers. However, the decision-making of the robot (i.e. selecting behaviors for activation or de-activation) is clearly another matter, hence the use of the term *robotic brain*. It should also be noted that this term does *not* imply that neural networks are used by default. Certain behaviors *may* of course make use of neural networks, but the term robotic brain, as used in the EUF method, does not in itself indicate that such structures are employed. Thus, one may say that the term is more oriented towards the overall functionality of a brain rather than its detailed structure.

In the EUF method, the robotic brain consists of a decision-making system (described below) and a repertoire (i.e. a set) of *brain processes*. Three kinds of (brain) processes are defined in the EUF method: (i) *cognitive processes* that do not involve any motor action, (ii) *locomotive behaviors* that displace the entire robot, using its motors, and (iii) *movement behaviors* that carry out movements not related to locomotion, such as moving the arms or the head (if available on the robot), or any combination thereof. Locomotive and movement behaviors both constitute motor behaviors. Thus, what one normally thinks of as a behavior (in BBR) is a special case of a brain process in the EUF method. Typically, a robot defined in the EUF framework is equipped with several cognitive processes and several motor behaviors. As for the motor behaviors, most wheeled and legged robots are equipped with at least *some* locomotive behaviors. On the

other hand, movement behaviors are commonly used in, say, humanoid robots, whereas wheeled robots (of the kind considered in the example below) without arms or other manipulators normally do not use movement behaviors. A few specific examples of brain processes are given in connection with the example presented in Sect. 4 below.

In addition to the repertoire of brain processes, a *decision-making system* is defined, which is responsible for activation and de-activation of the various brain processes, in response to the different situations encountered by the robot.

2.2 Brain processes

First and foremost, the EUF method deals with decision-making, i.e. selecting brain processes for activation, rather than the problem of actually generating the brain processes in the first place. However, in order to apply the EUF method, one must first generate a set of brain processes that will be used in the robot. Thus, a set of basic brain processes, briefly described in Sect 4.1 below, have been defined for the purpose of testing the method. For the remainder of this section, it will be assumed that such a set of brain processes is available.

2.3 Description of the method

In the EUF method, the decision-making is based on the concept of *utility*, which can be seen as a common currency used for weighing different brain processes against each other in any given situation. The concept of utility, which was originally formalized by von Neumann and Morgenstern [5], has been applied in fields as diverse as economics, ethology and robotics [6]. However, before describing the use of utility in the decision-making process, it is necessary to introduce the concept of *state variables*.

State variables A robot normally obtains information via its sensors or other input devices such as keyboards or touch screens. Some sensors, such as infrared (IR) sensors, provide a scalar reading that (in the case of IR sensors) can be used, for example, in proximity detection. Other sensors provide vector-valued readings (e.g. laser range finders) or matrix-valued readings (e.g. digital cameras). In principle, all those readings could be used for determining the state of the robot. However, the data flow would be massive. Taking a cue from biology, one may note the ability of the (human) brain to filter out irrelevant information, in such a way that a person will only be consciously aware of information that is likely to be relevant for assessing the situation at hand. Thus, in the EUF method, a *sensory preprocessing system* (SPS) is introduced, which maps the raw data obtained through all the sensors (or a subset thereof) to a manageable number of *state variables*.

A variety of sensory preprocessing methods can be envisioned, and the EUF method does not introduce any restrictions on the type of mappings used in the SPS. As a simple example, a state variable z_1 that measures proximity to

obstacles may be defined as the average value of the distances measured over an angular range (say $[-0.25, 0.25]$ radians, relative to the direction of heading of the robot), using a laser range finder (LRF), assuming that the robot is equipped with such a sensor. An SPS normally contains many mappings, each of which produces a scalar output z_k. The state variables are then collected in a vector (denoted $\mathbf{z}$) containing all outputs from the SPS. This vector is used as input to the utility functions, which will be described next.

Utility functions In the EUF method, each brain process is equipped with a *utility function* that determines the relative merit of the brain process in the current situation. The utility u_i of brain process i is determined as

$$\tau_i \dot{u}_i + u_i = \sigma_i \left(\sum_{k=1}^{m} a_{ik} z_k + b_i + \Gamma_i \right), \quad i = 1, \ldots, n, \tag{1}$$

where n is the number of brain processes, τ_i is a time constant determining the reaction time of the robot (typically set to around 0.1 s), m is the number of state variables[1], a_{ik} and b_i are tunable parameters (the procedure of setting parameters is exemplified in Sect. 4.2 below), and $\sigma_i(x)$ is taken as $\tanh(c_i x)$, where c_i is a positive constant. Thus, the squashing functions σ_i serve to keep utility values in the range $[-1, 1]$ provided, of course, that the values are initialized in this range.

The parameter Γ_i, which is normally equal to zero, allows direct activation or de-activation of a brain process. Ideally, the state variables z_k $(k = 1, \ldots, m)$ should provide the robot with all the information needed to make an informed decision regarding which brain processes to keep active in any situation encountered. In practice, of course, it is very difficult to summarize, in a set of scalar state variables, all the many situations that may occur. As an alternative, a brain process j may set the parameter Γ_i of some brain process i either to a large positive value (in order to raise the utility u_i, so as to activate process i) or a large negative value (to achieve the opposite effect, i.e. de-activation of process i). However, once the intended result has been achieved, Γ_i should return to its default value of zero. This is achieved by letting Γ_i vary (at all times) as

$$\tau_i^{\Gamma} \dot{\Gamma}_i = -\Gamma_i \tag{2}$$

where τ_i^{Γ} is a time constant[2], determining the decay rate of Γ_i. Thus, in the normal situation Γ_i is equal to zero, whereas if some brain process abruptly sets Γ_i to a value different from zero, it subsequently falls off exponentially.

[1] In practice, Eq. (1) is discretized and is integrated with a time step (typically around 0.01 s) much smaller than the smallest time constant. Note that the latest available state variables are used as inputs to the utility functions. Some state variables may change very frequently, whereas others (e.g. those based on LRF readings) are updated more seldom (typically with a frequency of 10 Hz, in the case of an LRF).

[2] The superscript (which is not an exponent!) is introduced in order to distinguish this time constant from τ_i defined in Eq. (1).

```
repeat
 t ← t + dt
 Update sensor readings
 Determine state variables, using the SPS
 Obtain non-zero Γ_i parameters (if any) from the (active) brain processes
 Update the utility functions and the Γ_i parameters
 Activate and de-activate brain processes based on the utility values
 Execute active processes (for a time interval of length dt)
until (Terminated)
```

Fig. 1. Pseudo-code summarizing the activities taking place in a robotic brain defined in the framework of the EUF method. SPS = sensory preprocessing system. See also Eqs. (1) and (2). The integration time step dt is set to a value smaller than the smallest time constant in the system.

The differential form of Eq. (1), has the important effect of filtering out the inevitable noise present in the sensor readings and therefore transmitted to the state variables.

Decision-making The generation of a specific decision-making system in the EUF method consists of setting the parameters of the utility functions, as well as specifying both the mappings constituting the SPS and the use (if any) of the Γ_i parameters, so as to achieve the desired overall result. In many cases, this is in fact easier than it sounds. However, in complex cases, it may be difficult to provide appropriate parameter values by hand, as noted also in connection with the original UF method [4]. One may then resort to optimization of the parameters, using stochastic optimization algorithms such as, for example, genetic algorithms or particle swarm optimization.

In any case, assuming that one has been able to find appropriate values for the parameters (either by hand or using some optimization method), the forms of the state variables and the utility functions are thus determined. At this stage, decision-making is quite simple in the EUF method, and works as follows: Any cognitive process with utility larger than zero is active. Thus, it is possible for several cognitive processes to be active at the same time. By contrast, exactly *one* locomotive behavior is active at any given time, motivated by the fact that the robot cannot move in two different directions at the same time. In the EUF method, the locomotive behavior with highest utility is active, and all other locomotive behaviors are inactive. Of course, this does not imply that the robot must constantly be on the move: A locomotive behavior may also allow the robot to stand still. The activation of movement behaviors (if available) works in the same way: Exactly one such behavior is active at any given time, namely the one with highest current utility among the movement behaviors. Note, however, that movement behaviors may, of course, combine the use of several (non-locomotive) motors. For example, in a robot consisting of a wheeled base and a humanoid upper body, a movement behavior may be responsible for turning the head (and, possibly, the eyes) as well as moving the arms, to

carry out a hand-eye coordination task. At the same time, the active locomotive behavior may move the entire robot forward. It should also be noted that the utility functions of *all* brain processes are continuously updated, meaning that an inactive process can be activated, should its utility value become sufficiently high. The activities occurring in the brain of the robot are summarized in pseudo-code in Fig. 1.

3 Implementation

Even though the EUF method is defined irrespective of any particular implementation, in order to actually *use* the method one must, of course, provide an implementation, normally in the form of a computer program. As in any problem involving autonomous robots, one may implement the method either in a computer simulation or in the processor(s) of a real robot. Ultimately, it is the ability of the method to provide a decision-making system for *real* robots that is of importance. However, simulations are useful in the early stages of the development of an autonomous robot since, for example, the construction of the actual robot is normally a costly and time-consuming process. Using simulations, one may reduce the risk of making serious mistakes in the construction process such as, for example, fitting the robot with inappropriate sensors for the task at hand. Also, if the development of the robot (body or brain, or both) involves some form of optimization, the time required for carrying out such a procedure in a real robot is often prohibitively long. On the other hand one should be careful to remember that the results obtained in a simulator at best represent a very rough approximation of the performance of the corresponding real robot, and only if the simulator manages to capture (most of) the relevant aspects of the real world such as, for instance, noise in sensors and actuators.

Note that using simulations for evaluating the EUF method is well motivated, since the complexity of the actual decision-making process is no different in a simulation (with appropriate noise levels in sensors, actuators etc.) than in the real world. In addition, the EUF method may, in fact, be used for decision-making in software agents (in computer games, information systems etc.), further motivating the use of simulations.

The EUF method has been implemented in a simulation program. Written in Delphi (object-oriented Pascal), the simulator allows a user to set up a simulation involving a differentially steered robot equipped with a repertoire of brain processes as well as a decision-making system following the EUF method (with an SPS, utility functions etc., as described above). The simulator introduces appropriate noise levels in both actuators and sensors. The detailed description of the robotic brain (i.e. the brain process repertoire, the parameter values of the decision-making system, the SPS mappings etc.) must be provided, by the user, in the form of a text file. Similarly, the parameters of the robot (physical characteristics, motors, sensors etc.) must also be provided, in the same way. A third text file, describing the arena, should also be generated. As these files tend to become rather complex, a set of template files has been generated, so that

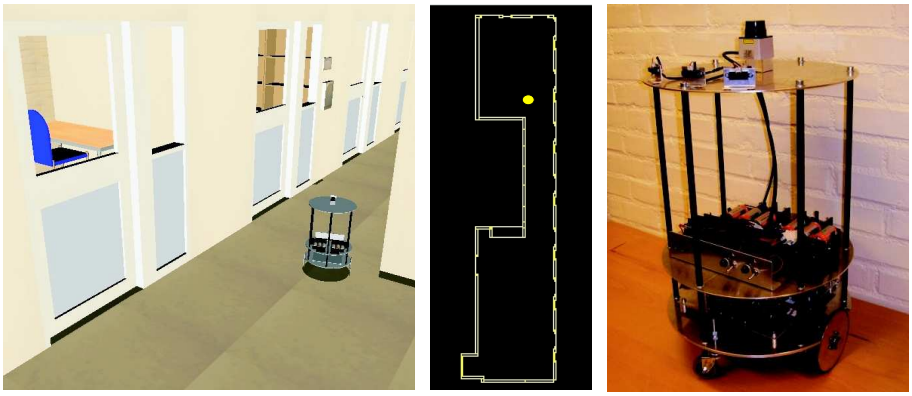

Fig. 2. Left panel: A screenshot from a simulation, showing a wheeled robot in a typical office environment. Middle panel: A schematic view (from above) of the office environment in which the simulations were carried out. Right panel: The physical counterpart (currently under construction) to the robot used in the simulations.

the user normally only needs to make minor modifications to those files. Once the setup files have been generated, the simulation can be executed. The program allows 3D visualization using OpenGL, so that the user easily can follow the performance of the robot, and also stop execution in order to modify, for example, the parameters determining the decision-making procedure.

An important aspect of the simulation is the possibility of emulating parallel processing, since the EUF method allows any number of cognitive processes to be active simultaneously, in parallel with one locomotive behavior (and, if available, one movement behavior). Thus, the simulator has been equipped with this feature, in such a way that all active brain processes are allowed to execute for dt s (the time step length) before an actual movement of the robot is carried out. Since the simulator typically runs much faster than real time, an illusion of parallel processing is easily obtained.

As for implementation in real robots, the simulation code is currently being translated to `C#`, which was deemed more suitable for this purpose. Here, the parallel execution of brain processes is handled using multi-threading, in which each brain processes runs as a separate thread (in Microsoft Windows).

4 Examples and results

The EUF method is currently being tested and evaluated. A brief summary of such a test will now be given, in which a robot is tasked with navigating between two arbitrary points in a typical office arena. In practical applications, such a robot could be used, for example, in delivery tasks. The physical robot, shown in the right panel of Fig. 2, is currently being constructed. It is equipped with two (DC) motors, one for each drive wheel, and is supported underneath

by two castor wheels. For sensing, the robot has been fitted (on its top plane) with a Hokuyo URG-04LX LRF and three infrared proximity detectors. It is also equipped with wheel encoders, providing the raw data for odometry (described below). The robotic brain will be implemented on a small laptop (not shown) placed on the robot. Here, only the results obtained from simulations will be described. These simulations (involving a simulated version of the robot just described) have been carried out mostly to test the EUF method, but also as a part of an iterative procedure, involving both simulations and hardware construction, aiding the process of designing the physical robot. A screenshot from the simulator is shown in the left panel of Fig. 2, and a schematic view of the arena is given in the middle panel of the same figure.

4.1 Simulation setup

The brain of the (simulated) robot consisted of a decision-making system in the EUF framework. The behavioral repertoire contained five brain processes, namely *Potential field navigation* (B1), *Side collision avoidance* (B2), *Frontal collision avoidance* (B3), *Odometry* (B4), and *Localization* (B5). B4 is a cognitive process, whereas the other four are locomotive processes. In B1, the robot follows a potential field [3], with (stationary) obstacles defining bumps in the field and the goal defining a gently sloping valley attracting the robot. Hence, just like B2 and B3, B1 also handles obstacle avoidance to some extent. However, potential field navigation is dependent on accurate positioning. Thus, B2 and B3 are needed in cases where the robot's positioning (through the odometry process described below) is inaccurate so that, for example, the robot may be headed for, say, a wall, even though the potential field (based on the inaccurate position) tells it that the path is clear. B2 and B3 can also be used for avoiding collisions with moving obstacles (e.g. people) that are not included in the potential field. B2 handles situations in which, for example, the robot moves along an extended obstacle (e.g. a wall) in a slightly incorrect direction, i.e. towards the obstacle rather than parallel to it. By contrast, B3 is used for avoiding head-on (or near head-on) collisions. B4 generates estimates of position, velocity and heading angle, based on the readings of wheel encoders on the motors. The localization process (B5) is used in order to counteract the inevitable drift in the odometry. When activated, this process matches the latest available wide-angle readings from the LRF to a map provided (along with the potential field) to the robot. This behavior is, in fact, quite complex, and has been tested thoroughly both in simulations and in physical robots [7], using the Hokuyo URG-04LX LRF.

Thus, in order for the robot to solve its navigation task successfully, it must handle several instances of decision-making. For example, while its task is to reach the current target position as fast as possible, it must do so without hitting any obstacles. Thus, the robot must occasionally activate B2 or B3 even though their activation delays the robot's arrival at the navigation target. Furthermore, from time to time, the robot must also activate B5 in order to recalibrate the odometry. However, this should not be done too often, since B5 (in its current form) requires the robot to reach a standstill, thus causing another delay.

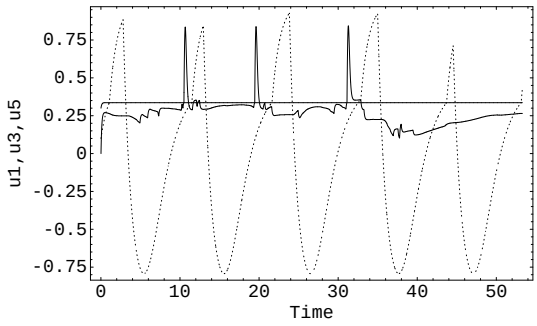

Fig. 3. The variation of utility values for brain processes B1 (solid, almost constant), B3 (solid, with spikes), and B5 (dotted), during a simulation.

4.2 Decision-making system

The state variables needed for the utility functions (see Eq. (1)), were obtained from an SPS that, in this case, contained three mappings, each formed as an average over LRF readings. For example, the state variable z_1 was defined as the average of the distances detected by the LRF in the angular interval $[-0.25, 0.25]$. Thus, a low value of this variable would signal an imminent frontal collision. Using the three state variables, the five utility functions (one for each behavior) were set up. In the preliminary test described here, the parameters (i.e. a_{ik}, b_i etc.) were set by hand, which was possible in this rather simple example.

First of all, τ_1, τ_2, τ_3, and τ_4 were all set to 0.1 s, representing the reaction time of the robot. For B4, all a_{4k} were set to zero, and b_4 was set to a small positive value. The parameter Γ_4 was not used by any brain process. Hence, following Eq. (1), u_4 was always positive, meaning that the odometry process was constantly active. For the locomotive behaviors, where only the process with highest utility is active, it implies no restriction to set the utility of *one* such process to a constant value. Thus, the utility of B1 was set as that of B4. For the other three processes, some experimentation was needed and, in the end, B2 and B3 were, in fact, the only processes with non-zero a_{ik} parameters. For B5, τ_5 was set to 1.0 s. The bias (b_5) was set to a positive value, and the Γ_5 parameter was used to make sure that localization, when activated, was normally allowed to run to completion. Thus, upon activation, B5 would itself set Γ_5 to a large positive value, raising (within the next second or so) its utility. The fall-off of Γ_5 was regulated by setting the τ_5^{Γ} parameter to an appropriate value (2 s, in this case). In cases where B5 managed to complete its task rather quickly, instead of waiting for u_5 to drop, B5 would then essentially de-activate itself by setting Γ_5 to a large negative value. After only a few iterations involving different parameter settings, appropriate settings were found, and the robot was then capable of navigating reliably between arbitrary points in the arena.

An illustration of the robot's decision-making is shown in Fig. 3. The figure shows the variation in utility for B1, B3, and B5 over a simulation. For clar-

ity, the utility functions for B2 and B4 have been omitted from the figure. In this particular run, B2 was never activated, whereas the cognitive process B4 (odometry) had positive utility throughout the run, and was therefore continuously active. The robot began by activating B4 and B1. B1 was then deactivated after around 1 s, when instead an initial localization (B5) was carried out. Next, the robot resumed its navigation until, after around 10 s, B1 was deactivated in favor of B3, in order to avoid a collision etc. Having covered the entire length of the arena, the robot reached its target after 53 s.

5 Discussion and conclusion

The EUF method for decision-making in autonomous robots has been introduced and described. The preliminary tests carried out thus far are promising, and indicate that the EUF method allows a user to set up the decision-making structure quite rapidly, even though optimization is likely to be needed in complex cases. The use of differential equations for determining the utility functions effectively filters out the noise in the state variables. This represents an improvement over the earlier versions of the method (see e.g. [4]), where utility values were determined directly, i.e. without the derivative term present in Eq. (1). Current work involves implementation of the EUF method in physical robots, as well as the formulation of more complex test cases, involving even larger behavioral repertoires (including, for example, processes for human-robot interaction).

References

1. Bryson, J.J.: Mechanisms of action selection: Introduction to the special issue. Adaptive Behavior **15**(1) (2007) 5–8
2. Pirjanian, P.: Behavior-coordination mechanisms – state-of-the-art. Technical report IRIS-99-375, Institute for Robotics and Intelligent Systems, University of Southern California (1999)
3. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. In: Proceedings of the IEEE International Conference on Robotics and Automation. (1985) 500–505
4. Wahde, M.: A method for behavioural organization for autonomous robots based on evolutionary optimization of utility functions. Journal of Systems and Control Engineering **217**(4) (2003) 249–258
5. von Neumann, J., Morgenstern, O.: Theory of Games and Economic Behavior. 3 edn. Princeton University Press, Princeton, N. J. (1953)
6. McFarland, D.: Animal Behaviour: Psychobiology, Ethology and Evolution. 3rd edn. Prentice Hall (December 1998)
7. Pettersson, J., Hartono, P., Wahde, M.: A behavior module for odometry recalibration in autonomous robots. In: Proceedings of the Fourth International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE2007). (2007) 11–17