Underwater Vehicle Control: Minimum Requirements for a Robust Decision Space

Michael R. Benjamin*
Naval Undersea Warfare Center (code 2211) - Division Newport, RI
Computer Science Department, Brown University
Building 1171-1 NUWC, Newport RI, 02841
phone (401) 832-4148 fax (401) 841-4749
(mrb@cs.brown.edu)

Abstract

Development of automated tactical control in underwater vehicles is motivated by the growing need for both decision aids in manned vehicles, as well as for controllers in unmanned autonomous vehicles. While these systems may vary widely in architectural design, ultimately the system interacts with the world via a sequence of decisions or control variable assignments. The choice of these variables, and their domains, ultimately affects both the complexity (i.e. quickness) of the decision process and the precision of the individual decisions or actions. Operating a vehicle situated in the real, dynamic world means there are both real-time requirements and precision requirements. In this paper we examine the precision required in both submarine tactical decision aids and autonomous underwater vehicles, to identify a minimum set of control/decision variables. We also argue that existing techniques for handling the resulting large decision space are inadequate and often lead to damaging oversimplifications. We discuss these common simplifications and their drawbacks, and suggest new techniques that may handle the large spaces, preserving the necessary level of precision in control.

1. Introduction: autonomous agent control and action selection

Deliberative and reactive decision-making are both essential components for rational agents seeking long-term goals while surviving unforeseen events. Control of manned or unmanned vehicles in an underwater environment involves consideration of other moving, and potentially hostile vehicles, as well as long-term navigation objectives. The ability to choose an action that optimizes a prioritized combination of objectives may dramatically affect the chances of surviving to carry out one's mission. In this section, behavior-based control, and multifusion-based action selection are introduced. This type of action selection typically comes with the need to manage a large decision space. We first argue here that the virtues of these techniques are worth the trouble, before discussing the particulars, in later sections, concerning the large decision space required for effective underwater vehicle control.

* This material is based upon work supported in part by NUWC-Division Newport's Independent Research program and the Office of Naval Research, Dr. Ralph Wachter.

1. **Behavior-based control**

Traditional AI planning techniques that reason from a starting state to a goal state have been criticized (e.g. Brooks [1986]; Maes [1990]; Payton et al. [1990]) for being too rigid and unable to cope in unpredictable environments, as well as for being dependent on a single world model. Focusing on environments that were not overly contrived or simplified, Brooks [1986] proposed a control architecture that that had independent components primarily designed to react to features of the environment of specific concern to each component. This architecture later became generalized to typically also contain deliberative or long-term planning components, and became commonly referred to as the *behavior-based* control architecture. This decentralized architecture (Fig. 1)

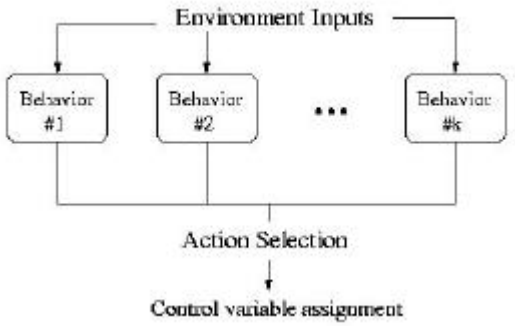

Figure 1: Behavior-based control architecture

not only contributes to a modular design process and robust performance, but also alleviates the need to form a centralized, comprehensive model of the entire world state.

As Figure 1 indicates, a single action is chosen by an *action selection method* (ASM) from the inputs received from each behavior. By limiting communication between behaviors to the ASM, context about what should be sensed, modeled and acted upon is determined by each behavior. Nowhere in the system is there an attempt to formulate and update a complete world-state, and no communication or model sharing is proposed between behaviors. Many of the recent implementations of agents operating in real-world environments with conflicting objectives and a need to both plan and react effectively (e.g. Humphrys [1997]; Mataric [1997]; Pirjanian [1998]; Pirjanian and Christensen [1997]; Pirjanian and Mataric [1999]; Riekki [1999]; Rosenblatt [1997, 1999]; Rosenblatt and Thorpe [1995]; Rosenblatt et al. [2000]; Saffiotti et al. [1999]; Tunstel [1995]; Williams et al. [2000]), have based their control architecture on the behavior-based architecture. The point of contention among these works is how to make the action selection method work properly and sufficiently fast.

1.2 **Using the right action selection method**

The earliest action selection methods (e.g. Brooks [1986]; Kosecka and Bajcsy [1994]) simply let the most dominant or highest priority behavior have complete control of the vehicle, switching in between control cycles. Later designs of ASM's sought to let the behaviors *cooperate*, by having each behavior influence the selected action. The earliest of these simply averaged the single action taken from each behavior (Khatib [1985]). The flaws of these two early approaches are well documented (e.g. Pirjanian [1998], Rosenblatt [1997], Saffiotti et al.

[1999]). In situations where behaviors are competing for different actions, cooperation is indeed a crucial aspect of effective action selection. To achieve this, each behavior must output more than its single best action. Preferably it outputs a rating of all possible alternative actions. We call such methods *multifusion* methods. To see the importance of using multifusion, consider the situation depicted in figure 2,

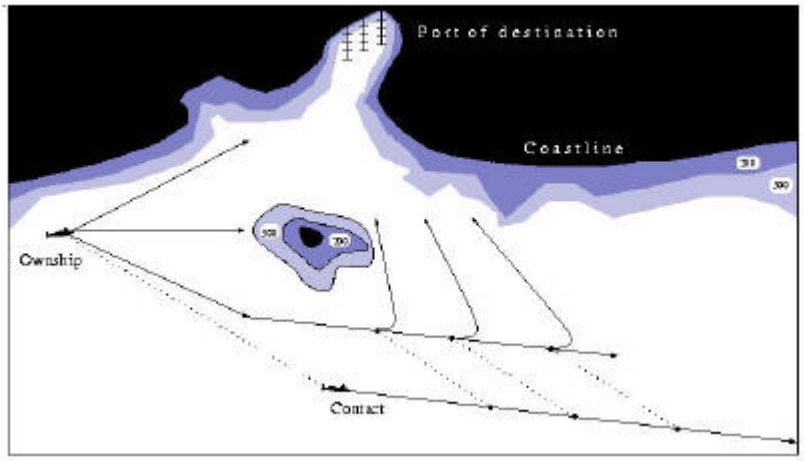

Figure 2: Dilemma: return to port as planned, or follow and monitor?

where ownship is depicted with a dilemma of either returning to port as planned, or following a contact to monitor its actions. Assume each objective corresponds to one behavior, and that returning to port is of higher priority. An ASM that does not allow cooperation would promptly send ownship to port. An ASM that cooperates by averaging the *single* best action from each behavior would send ownship directly toward the island, satisfying neither behavior. An ASM that cooperates via multifusion would weigh the alternative rated-actions from each behavior, in addition to its most-preferred action. This multifusion ASM would seek a compromise allowing ownship to follow the contact for a while, and then return to port via an alternative route. Virtually all recent agent implementations using behavior-based control use some form of multifusion.

1.3 **The decision space: size matters**

The price of using multifusion methods is that the size of the decision space may become difficult to deal with, since the typical practice is to evaluate each possible action explicitly. An action is typically denoted by a change of value to one or more control variables, and control variables are typically discrete variables with a finite, uniformly-spaced domain. The number of distinct actions is thus determined by the Cartesian product of each variable's domain. Since the multidimensional decision space grows exponentially with the number decision variables, explicit evaluation of each candidate decision jeopardizes the ability to meet the real-time requirements of the application.

The motivation for using the behavior-based control architecture and using multifusion-based action selection is clear, but currently there are only two options for dealing with the resulting large decision spaces, and they are both problematic. The first method treats the

decision space properly, as the Cartesian product of each variable's domain. But a reduction is made in the number of variables modeled, or in the size of their domains, to reach a point where explicit evaluation of each decision can fit under the real-time requirements. The second method separates the decision problem into a sequence of constrained decisions, one for each variable. A single decision over a high dimension decision space is flattened into several one dimension decisions, which results in sufficiently faster decisions. Both methods result in an unacceptable compromise of optimality.

The first aim of this paper is to look at the underwater vehicle maneuvering problem and to determine a minimal set of decision variables, their relationship to each other, and their domains. The second aim is to argue that the decision space should not be simplified using any of the methods typically used to meet real-time requirements. We argue instead for the development of a better method for handling large decision spaces.

2. **Decision variables for underwater vehicle control**

The aim of this section is to establish a minimal set of control or decision variables for effective maneuvering of both submarines and AUVs. An argument will be made not only for what variables should be included in this set, but also what is a fair domain for each variable.

2.1 **The core set of maneuver decision variables**

A submarine's maneuvering behavior is composed of a series of *maneuver legs* where each leg begins and ends with a *maneuver*. What constitutes a maneuver, in the decision maker's mind, is open to debate and is the focus of this subsection. The definition of a maneuver is determined by the choices made for the set of decision variables, where each variable corresponds to some aspect of controlling the ship. Once these variables are identified, a maneuver is marked by a change in value for any one (or more) of these variables. If, for example, course and speed are the only two decision variables, then a change of depth could be made *within* one maneuver. But if depth is also a decision variable, the depth change would mark a new maneuver leg.

Underwater vehicle control involves at least three control variables: *course*, *speed*, and *depth*. As Figure 3 shows,

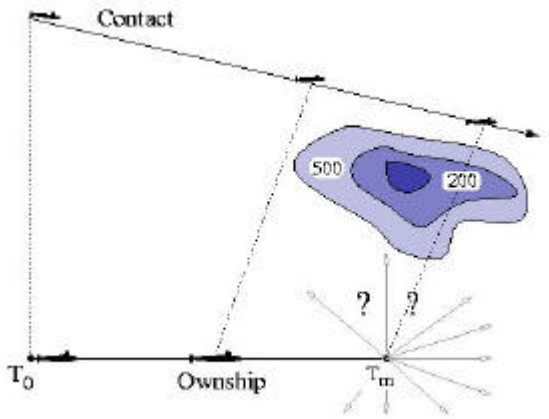

Figure 3: Underwater vehicle maneuvering

ownship control is viewed as a sequence of assignments of new values to these variables that enable safe navigation, and tactical superiority with respect to the environment and other vessels. The time between T_0 and T_m is the time between maneuvers and shapes the notion of what *real-time* means in this application. The *course* variable typically has the domain [0, 359] with 1-degree increments. The exact upper limits for *speed* and *depth* are platform dependent and classified. Unclassified numbers generally attributed to modern submarines give an upper value of 30 knots [Burcher and Rydill, 1998, pg. 189] and 450 meters [Burcher and Rydill, 1998, pg. 73] respectively. We assume 1-knot and 1-meter units for these domains.

2.2 **Rate of change in control variables**

Considerations for detection and counter-detection bring into question the adequacy of the three control variables, *course, speed* and *depth*. For each variable, the *rate-of-change* influences the ability of ownship to hear and be heard. For course, the *rudder angle* determines the period of time in which a towed array is unstable. For speed, the *acceleration* determines the amount of noise emanating from ownship. For *depth*, the *hydroplane angle* may affect the rate of hull compression, which may affect ownship noise. Furthermore, these rate-of-change variables have considerations in navigation (turn radius) and ownship safety (SOE, see Fig. 8). A strong candidate set of control variables for submarine maneuvering should not only include the variables *course*, *speed* and *depth*, but also the corresponding rate-of-change variables rudder-angle, *acceleration* and *hydro-plane angle*. The domain size for each of the three rate-of-change variables is debatable, but using a rough number of 10 for each variable will suffice for our purposes.

2.3 **The position, or time-on-leg, decision variable**

A chosen *course*, *speed*, and *depth*, along with the corresponding rate-of-change decision variables, constitute a complete maneuver (and a new position) only when the *duration* of that leg is specified. For this reason, *time-on-leg* is also added to the set of decision variables. For our purposes, we will consider this variable to be in minutes, and a domain of [1, 100]. Typical notions of action selection, prevalent in the works using behavior-based control, would not include *time-on-leg* as a decision variable. The common way to determine the duration of an action is to have a fixed control cycle, where in every cycle, the current control variable settings are re-evaluated. The idea of selecting an action duration is inconsistent with this, since no duration of action is ever committed to.

On this same idea, note the relationship between the rate-of-change variables, and their counterparts *course*, *speed*, and *depth*. Each of the latter three is actually a function of *time* and the corresponding rate of change. It can be argued that, *course* for example, is in fact a *plan* in the sense that a *rudder angle* was chosen, but it takes a commitment in time to actually achieve the intended *course*. Certainly, if an unexpected event were to be sensed during the course change, the chosen course may be over-ridden. A distinction can be made, however, between *committing* to time, and *reasoning* about time. If we include *time-on-leg* as a decision variable, this does not necessarily imply a commitment to that time. Reasoning about time is not inconsistent with the idea of a fixed control cycle where control/decision variables are subject to re-evaluation. In fact, one way to simplify decision-making is to only decide a sequence of values for rate-of-change variables. This is actually done in some works employing behavior-

based control [Rosenblatt, 1997]. This method of simplification, and others, along with their drawbacks, are discussed in the next section.

3. **Simplifying the decision space: methods and dangers**

In the previous section, the following seven decision variables were identified: *course*, *speed*, *depth*, *rudder-angle*, *hydroplane angle*, *acceleration*, and *time-on-leg*. Treating the decision space as the Cartesian product of each of the domains, we have a seven-dimensional space with a half trillion[1] distinct decisions. In section 1, the merits of behavior-based control and multifusion action selection were presented and defended. Recall that this approach requires the explicit rating of each possible decision by each behavior. Requiring each behavior module to rank a half trillion decisions, and then to require the action selection method to accept and combine this input, is not feasible in practice. To overcome this problem, rather than find methods to deal with such large spaces, the common approach is to go back and simplify (e.g. reduce the size of) the decision space. The following four subsections list four ways to do this and the sacrifices that accompany each method.

3.1 **Reducing the domain sizes**

The easiest way to simplify the decision space is to reduce the size of the variable domains. With some corner cutting, it may be argued that, for example, the *course* variable can be treated with 5-degree increments instead of 1-degree increments. This argument can be made for each variable on a case-by-case basis, to varying degrees. Furthermore, this idea can be combined with some of the ideas described in the next three subsections. There are two drawbacks to this however. The first is that, if we are building a decision aid for experts, there is little tolerance for recommendations that may have missed an optimal solution that fell through the cracks of an overly coarse decision space. The second is that, even with generous corner-cutting, the space is still too large in practice.

3.2 **Raising the decision perspective: fixing the lower variables**

The seven decision variables given above can be split into three different levels, where the higher levels are viewed as a function of *time* and the variables on the lower level. This is depicted in Figure 4 below. One method to simplify the decision space is to *fix* the variables below a certain level. In this case, this means assuming all turns are conducted with a certain rudder angle, all depth changes with a certain coordination of the hydroplanes and, and all speed changes with certain rates of acceleration or deceleration. This indeed simplifies the decision space, but comes at a certain cost. Consider for example the situation depicted in Figure 5.

[1] From: $360*30*450*10*10*10*100 = 486,000,000,000

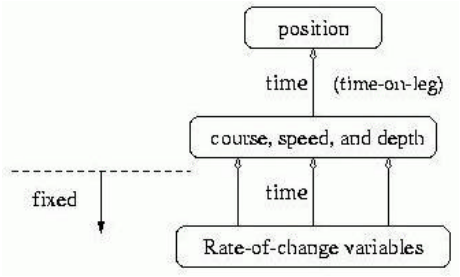

Figure 4:
Three decision perspectives

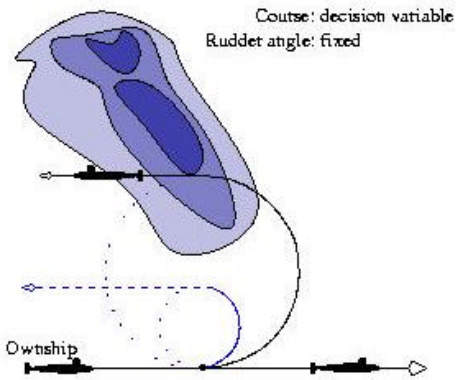

Figure5:
Deciding course with fixed rudder angle

A turn of 180 degrees to port is desired, but, by assuming a fixed rudder angle (resulting in the larger turn radius in Fig. 5), it appears the desired turn is restricted by navigation constraints. Only by considering a different rudder angle, with a smaller turn radius, does the 180 degree turn become feasible.

Similar problems can be revealed when fixing the acceleration and depth changing rates. A fair reaction to these shortcomings is the following: using the example again of *course* in Figure 5, first decide a new *course*, and then decide a *rudder angle* that will achieve this without violating any (e.g. navigation) constraints (assuming s*ome* angle works before committing to the turn). The general problem with this is that the values of the rate-of-change variables, in this case *rudder-angle*, affect other parts of the problem. For example, the tighter turn radius may entail a lower speed, which in turn may be restricted by the current depth (see Fig. 8), or may be restricted in shallow water by concerns about damaging a towed array. The tighter turn may also increase ownship noise during the turn. Any of these effects of turning at a tighter radius may subsequently affect the merits of making the originally desired 180 degree turn to port. The message is clear: a decision that changes *course*, *speed*, or *depth*, must simultaneously consider their rates of change.

3.3 **Lowering the decision perspective: marginalizing the higher variables**

Another way to simplify the decision space can be carried out in a manner opposite to the one just described in section 3.2. In this method, a decision focuses on values only to the rate-of-change variables (Fig. 6). A new *course* for example, consists of a sequence of two separate decisions for *rudder-angle*. Such an approach has been used by Rosenblatt [1997] for land vehicle control, and in Pirjanian [1998] for robot control. The advantage of this approach is that the decision space is indeed greatly reduced by the elimination of the higher level variables. Specifically, in this case the size of the decision space is reduce from a half trillion to one thousand.

To see the drawback in this approach, recall how decisions are selected by a multifusion action selection method. Each possible action is rated by each behavior based on how it fares with respect to the behavior's objectives. A fundamental property of the action selection method is that if two or more behaviors benefit from one action, that action *may* be selected over another

action that benefits a single behavior, even if the single behavior is of higher priority. When we lower the decision perspective in the manner suggested above, this property can easily be lost, resulting in significantly poorer decisions. Consider for example the situation in Figure 7.

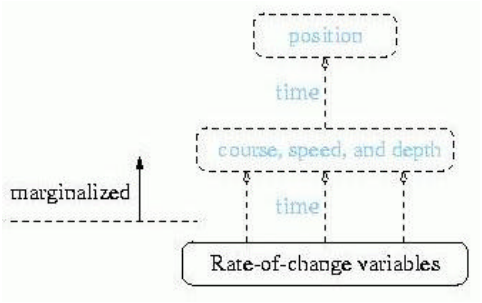

Figure 6:
Three decision perspectives

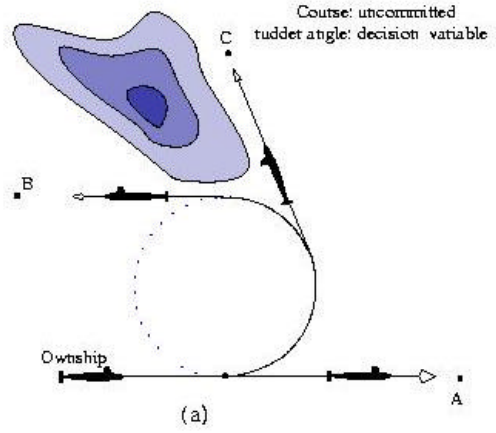

Figure 7:
Choosing rudder with no course commitment

Ownship has three mutually exclusive objectives of reaching positions **A**, **B**, and **C** indicated. Reaching position **A** would result in the highest payoff of the three. However, to reach points **B** and **C**, the same *rudder-angle* would suffice, differing only in the duration of time at that angle. It appears (incorrectly), from this lower perspective, that the two objectives, **A** and **B**, can be achieved with the same rudder angle. The result will be that only one objective will have been achieved, and it will not have been the highest priority one. Its interesting to note that, by this same argument for not marginalizing *course*, *speed*, and *depth*, the same argument can be made for not marginalizing *time-on-leg* (i.e. position).

3.4 **Flattening the decision space: advantages and dangers**

Recall that the decision space, when taken to be the Cartesian product of each variable's domain, grows exponentially with the number of decision variables. A common approach to dealing with this growth is to ``flatten" the decision space into a sequence of separate decisions, one for each variable, where initial decisions constrain the later ones. In our application, this turns a problem with one decision, with a half trillion choices, into a problem with seven decisions, each with no more than 450 choices. This method, despite its shortcomings, is actually quite common in practice [Rosenblatt, 1997; Saffiotti et al., 1999].

The drawback of flattening the decision space is that, while parts of the space are not removed (as in the other methods), parts of the decision space are *ignored*. This, in effect, leads to the same problem as reducing the decision space: optimal solutions will be missed. The method relies on a commitment to the order in which the variables will be assigned. By constraining later variable assignments based on the earlier ones, certain combinations of variable assignments will never be considered. It is interesting to contrast this procedure with constraint satisfaction techniques, where the same procedure is done, with backtracking invoked when a later decision is left with no feasible assignments. If the aim is to optimize, not just

satisfy, a full sequence of feasible assignments can be made, with no resulting idea, of how the chosen set of assignments stacks up against the global optima. Consider Figure 8,

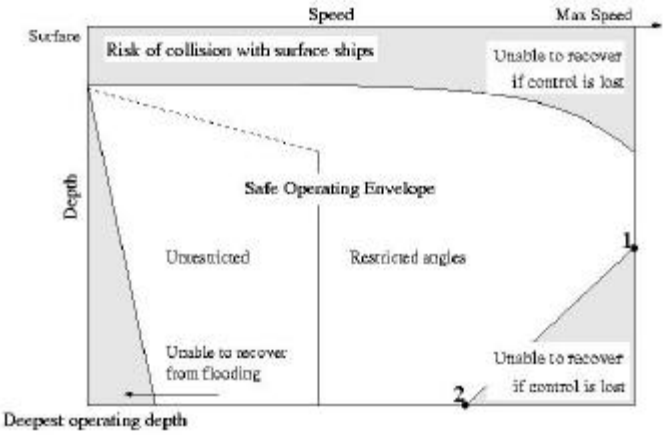

Figure 8: Safe Operating Envelope [Burcher and Rydill, 1998, pg. 190]

which depicts a typical safe operating envelope (SOE) relating safe submarine depths to certain speeds. Suppose there were two objectives, one wishing to maximize *speed*, and one wishing to maximize *depth*. If *speed* is optimized first, followed by *depth*, the point labeled *1* would be the resulting pair. If *speed* were instead optimized after depth, the resulting pair would be the point labeled *2*. It is likely that some point in between would have been better overall, especially if the objective to maximize *speed* were higher and the decision about *depth* were made first.

Its important to note that the penalty paid for flattening the decision space is highly dependent on the specifics of the application. For example, in the case above, if the depth of the ocean floor were less than the deepest operating depth of the platform, the effect of flattening the decision space would be harmless. Conversely, if the shape of the safe operating envelope were changed by sliding the point labeled *2* to the left or right, the effects of flattening could be much worse. The question then turns to the general problem of underwater vehicle control: can the proposed seven-dimensional decision space be safely flattened into a sequence of seven separate decisions? The answer is clearly *no*. The argument for this proceeds by providing examples of how pairs (or more) variables have a dependency on each other with respect to certain objectives or constraints. The dependency, for example, found between *speed* and *depth* in the SOE in Figure 8 provides a convincing argument for not separating these two variables.

3.5 **Conclusions about methods for simplifying the decision space**

In sections 3.1 through 3.4 we gave four methods for simplifying or reducing the seven dimensional decision space formed by the seven variables given in section 2. Recall that this large decision space was a result of adopting the multifusion method of action selection described in section 1, which claimed that (a) behavior-base control is an effective architecture for balancing reactive and deliberative decision making, and (b) that the most effective type of action selection takes a rating of each action, from each behavior. The situations where *non*-multifusion ASM's result in unacceptably poor decisions are easy to conjure and are well documented [Pirjanian, 1998; Rosenblatt, 1997; Saffiotti et al., 1999]. In virtually every work employing multifusion ASM's, the large decision space is simplified using one of the approaches

of this section. In the case of submarine tactical control, we argue two things, (a) there is less tolerance for sub-optimal actions/maneuvers resulting from these simplifications (it takes just one stupid decision to create a disaster or ruin the confidence in a decision aid), and (b) the size of the decision space in this application strongly points to the need for methods that can actually handle the large space, instead of simplifying the decision space to meet the inadequate methods.

4. **Decision making with large decision spaces**

The preceding sections claim that the decision space for submarine maneuver control is not only large, but that attempts to simplify or reduce its size come at too great a sacrifice. This problem of decision space size is tied to the method prevalent in all multifusion-based action selection methods: the simple exhaustive evaluation of each possible decision. In the behavior-based architecture, this is done not only by the action selection method, but also by each of the behaviors, as input to the ASM (recall Fig. 1). From section 3, it is clear that we do not want to eliminate, or ignore parts of the decision space, but perhaps instead we can *implicitly* evaluate many elements of the space. While the primary message of this paper is that such an approach is necessary, as opposed to how it might actually be done, we nevertheless present two methods here that work in such a manner. The latter of these two methods is the primary focus of this author's ongoing research.

4.1 **Traditional analog methods**

One approach to dealing with a large, high-dimensional decision space is to utilize an underlying function that may have motivated the behaviors to produce a set of action-value ratings. For example, in Figure 9,

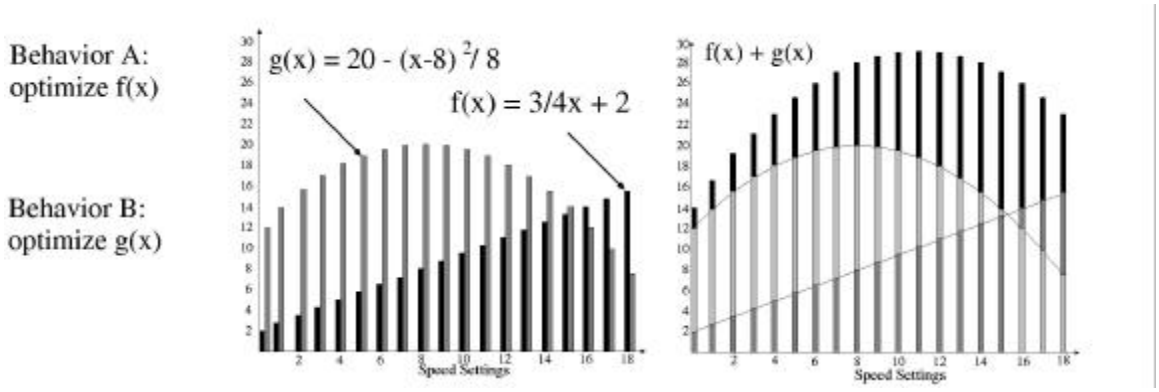

Figure 9:Combining two objective

two behaviors may have rated the eighteen speed settings on the left at the shown levels. By combining them via additive voting as in Rosenblatt [1997], the best combined speed can be identified as shown on the right. If, however, we have the knowledge that the two analog functions shown, f(x) and g(x), were the source of the action-ratings, then we can avoid an explicit evaluation of each action. The two functions, when added together, produce $-x^2/8 + 11/4x + 14$. The derivative of this function has a root at x=11, confirming the maximum found after explicit evaluations. While such nonlinear programming problems are intractable in general [van Hentenryck et al., 1997], there are many methods that work well in practice for

significant subclasses of the general problem. This is true in the case where *multiple* objective functions are present as well [Miettinen, 1999]. Unfortunately, each subclass comes with restrictions on allowable function form. In modeling the objectives of a decision maker controlling a vehicle operating in a real-world physical environment, with an open-ended set of potential circumstances to be faced, a particular kind or class of function form cannot guaranteed. An approach is needed that places little or no restrictions on function form, yet retains sufficient tractability.

4.2 **Interval programming**

Two methods have been discussed for representing and combining objective functions: the first method, explicit evaluation of each point in the decision space, is accurate, flexible, but unacceptably slow. The second method, the use of analog functions, is accurate and fast, but unacceptably inflexible. All three categories are crucial, and we seek a method that fairs well in all aspects. We call this method *interval programming*. Interval programming uses intervals of the decision space to represent objective functions in a piecewise defined manner. In Figure 10(a),

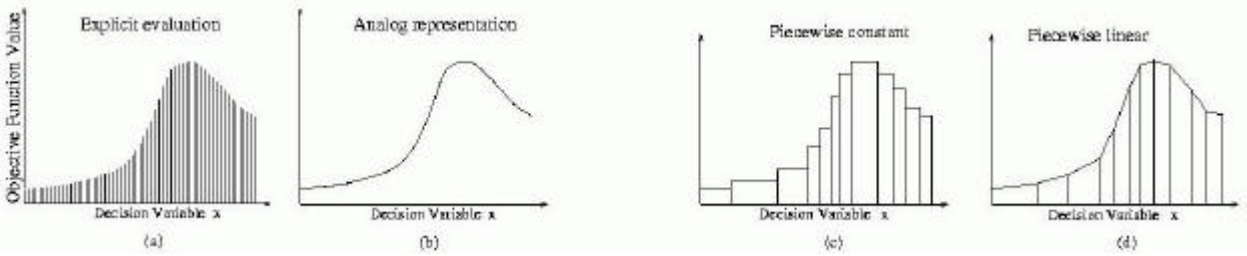

Figure 10: Four ways to represent the same objective function

an objective function over one decision variable is depicted that explicitly evaluates each element of the decision space. In Figure 10(b), the same objective function in analog form is depicted. In Figures 10(c) and (d), the same function is depicted using piecewise constant and piecewise linear functions. Solving an IP problem amounts to overlaying the objective functions generated by each behavior (Figure 11),

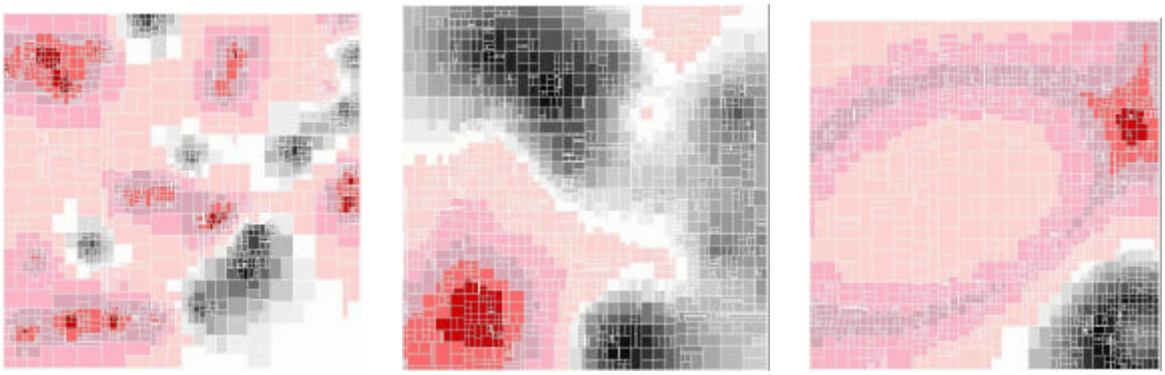

Figure 11: Three piecewise defined objective functions over two variables

then figuring out which pieces intersect from different objective functions, and finding the set of points where the best weighted sum is achieved. Unlike the analog methods, it is extremely

flexible since the solution algorithms make no assumptions about the underlying function form. The solution process deals only with the individual pieces and the simple functions within them.

Beyond flexibility, there is the issue of accuracy and tractability. In this regard there is a clear tradeoff revolving around the number of pieces used and the shape of each piece. More pieces mean greater accuracy but slower solution times. Furthermore, if the pieces have more flexible shapes (in 2D or above), less pieces can be used with better accuracy, but the algorithm must do more work to handle the more complicated shapes. Current techniques can solve problems with ten decision variables and five objective functions, each using 10,000 pieces, in less than a second. For more on interval programming techniques and results, see Benjamin [2000].

5. **Conclusions**

In this paper, we proposed a set of decision variables for underwater vehicle control, and variable domains, that were argued to be minimal for effective, robust decision making. We reviewed a technique, behavior-based control with multifusion-based action selection, that promises effective vehicle control, but has in practice required the explicit consideration of each element of a decision space that grows exponentially with the number of variables. We presented four methods, frequently used in multifusion-based applications to simplify the large decision spaces, and the corresponding penalty in optimality paid by each simplification. We argued that, while multifusion methods provide the action selection qualities necessary to effectively combine deliberative and reactive decision making, each type of penalty incurred by simplifying the decision space is unacceptable. We argued that instead of simplifying the decision space to fit the existing techniques, it is more appropriate to find techniques to meet the large decision space. We concluded by reviewing two such techniques.

6. References

[Arkin, 1998] Ronald C. Arkin. *Behavior-Based Robotics*. MIT Press, Cambridge, MA, 1998.

[Benjamin, 2000] Michael R. Benjamin. *Virtues and Limitations of Multifusion Based Action Selection*. Technical Report CS-00-01, Brown University, Department of Computer Science, 2000.

[Benjamin et al., 1993] Michael R. Benjamin, Thomas Viana, Karen Corbett, Ann Silva, *Satisfying Multiple Rated-Constraints in a Knowledge Based Decision Aid*. Proceedings IEEE Conference on Artificial Intelligence Applications, Orlando, FL, 1993.

[Brooks, 1999] Rodney A. Brooks, *Cambrian Intelligence: The Early History of the New AI*. MIT Press, Cambridge, MA, 1999.

[Burcher and Rydill, 1998] Roy Burcher, Louis Rydill, *Concepts in Submarine Design.* Cambridge University Press, 1998.

[Humphrys, 1997] Mark Humphrys. *Action Selection Methods Using Reiforcement Learning*. PhD thesis, University of Cambridge, 1997.

[Khatib, 1985] Oussama Khatib, *Real-time Obstacle Avoidance for Manipulators and Mobile Robots*. In Proceedings IEEE International Conference on Robotics and Automation, 1985.

[Kosecka and Bajcsy, 1993] Jana Kosecka, Ruzena Bajcsy. *Discrete Event Systems for Autonomous Mobile Agents*. In Proceedings Intelligent Robotic Systems 1993, pages 21-31, 1993.

[Mataric, 1997] Maja Mataric. *Behavior-Based Control: Examples from Navigation, Learning, and Group Behavior*. Journal of Experimental and Theoretical Artificial Intelligence, special issue on Software architectures for Physical Agents, 9(2):323-336, 1997.

[Miettinen, 1999] Kaisa M. Miettinen, *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, MA, 1999.

[Payton et al., 1990] David W. Payton, Julio K. Rosenblatt, David M. Keirsey. *Plan Guided Reaction*. IEEE Transactions on Systems, Man, and Cybernetics, 20(6):1370-1382, 1990.

[Pirjanian, 1998] Paolo Pirjanian, PhD Thesis: *Multiple Objective Action Selection & Behavior Fusion*, Aalborg University, 1998.

[Pirjanian and Christensen, 1999] Paolo Pirjanian, Henrik I. Christensen. *Behavior Coordination Using Multiple-objective Decision Making*. In SPIE Conference on Intelligent Systems and Advanced Manufacturing, Pittsburgh, PA, October 1997.

[Pirjanian and Mataric, 1999] Paolo Pirjanian, Maja J. Mataric. *Multiple Objective vs. Fuzzy Behavior Coordination*. In D. Drainkov and A. Saffiotti, editors, Lecture Notes in Computer Science on Fuzzy Logic Techniques for Autonomous Vehicle Navigation, 1999.

[Riekki and Roning, 1997] Jukka Riekki, J. Roning, *Reactive Task Execution by Combining Action Maps*. International Conference on Integrated Robots and Systems (IROS), Grenoble, France, pp 224-230, 1997.

[Riekki, 1999] Jukka Riekki, PhD Thesis: *Reactive Task Execution of a Mobile Robot*. Oulu University, 1999.

[Rosenblatt, 1997] Julio K. Rosenblatt, PhD Thesis: *DAMN: A Distributed Architecture for Mobile Navigation*. Carnegie Mellon University, Pittsburgh, PA, 1997.

[Rosenblatt, 1999] Julio K. Rosenblatt, *Maximizing Expected Utility for Behavior Arbitration*. In Proceedings of 12th Australian Joint Conference on Artificial Intelligenc, Sydney, NSW, Australia, December 1999.

[Rosenblatt and Thorpe, 1995] Julio K. Rosenblatt, Charles E. Thorpe, *Combining Multiple Goals in a Behavior Based Architecture*. In International Conference on Integrated Robots and Systems (IROS), 1995.

[Saffiotti et al., 1999] Alessandro Saffiotti, Enrique H. Ruspini, Kurt Konolige. *Using Fuzzy Logic for Mobile Robot Control*. In H.J. Zimmerman, editor, Practical Applications of Fuzzy Technologies, chapter 5, pages 185-206. Kluwer Academic, 1999.

[Tunstel, 1995] Edward Tunstel. *Coordination of Distributed Fuzzy Behaviors in Mobile Robot Control*. In IEEE International Conference on Systems, Man, and Cybernetics, pages 4009-4014, Vancouver, BC, Canada, October 1995.

[Williams et al., 2000] Stefan B. Williams, Paul Newman, Gamini Dissanayake, Julio K. Fosenblatt, Hugh Durrant-Whyte. *A Decoupled, Distributed AUV Control Architecture*. In Proceedings of 31st International Symposium on Robotics, Montreal, 2000.

[Yen and Pfluger, 1995] John Yen, Nathan Pfluger. *A Fuzzy Logic Based Extension to Payton and Rosenblatt's Command Fusion Method for Mobile Robot Navigation*. IEEE Transactions on Systems, Man and Cybernetics, 25(6):971-978, 1995.