Behavioral Mapless Navigation Using Rings

Randall P. Monroe, Samuel A. Miller, and Arthur T. Bradley

***Abstract*—This paper presents work on the development and implementation of a novel approach to robotic navigation. In this system, map-building and localization for obstacle avoidance are discarded in favor of moment-by-moment behavioral processing of the sonar sensor data. To accomplish this, we developed a network of behaviors that communicate through the passing of rings, data structures that are similar in form to the sonar data itself and express the decisions of each behavior. Through the use of these rings, behaviors can moderate each other, conflicting impulses can be mediated, and designers can easily connect modules to create complex emergent navigational techniques. We discuss the development of a number of these modules and their successful use as a navigation system in the Trinity omnidirectional robot.**

I. Introduction

A great deal of the work in ranging-based autonomous navigation has focused on the dual problems of map-building and localization. It seems clear that mapping and localization are essential to autonomous robotics. However, the assumption is often made that these internal maps should be the key to the robot's moment-by-moment navigational decision-making. This places enormous constraints on the character of the map, and often requires that they be extremely detailed. The majority of the work on obstacle avoidance in both static and dynamic situations either assumes the existence of a map [1], [3], [6] or assumes that the building of a map should be the first task [2], [4], [5]. These ideas are based on the assumption that for the purposes of obstacle avoidance, the robot should carry a highly detailed internal representation of the world. Elfes remarks in [4] that for successful autonomous operation,

> . . . it is necessary to develop systems . . . able to operate in unstructured environments with little *a priori* information. To achieve this degree of independence, the robot system must have an understanding of its surroundings, by acquiring and manipulating a rich model of its environment of operation.

Similarly, Chang and Song describe in [3] a sensor based real-time navigation system which uses global path-planning. They assume that the map – updated via sensors when possible – is the basis for navigation, and obstacle avoidance routines should be functions that take the map as their input. But as early as 1986, Dr. Brooks of the MIT CS/AI Lab was creating robots capable of this type of real-time navigation in a dynamic environment [7]. In [8], Brooks explains that his robot was successful because

> . . . it was using the world as its own model. It never referred to an internal description of the world that would quickly get out of date if anything in the real world moved.

High-level navigation requires an internal map of some kind. However, the use of a single-level high resolution spatial map in moment-by-moment obstacle avoidance is extremely inefficient. Attempts to model the world internally for these purposes typically involve gross oversimplifications regarding the shape of obstacles [1], [2], or run into serious difficulty involving computational complexity [5]. Frequently running path-planning algorithms on full high-resolution maps in real-time is not merely computationally difficult, it is unnecessary. In [8], Brooks argues that intelligent interaction with the world is generated by relatively simple rules running on top of complex perceptual processes. He describes the construction of a framework of simple systems for simple behaviors, with more complex behaviors emergent through the interaction of these systems with each other and with higher-level systems. By letting the simple systems run on a separate layer below the complex ones, the robot can respond intelligently and efficiently to its immediate surroundings while simultaneously pursuing whatever high-level goals it may have.

II. Our System

A. Trinity

The Trinity robot was designed as a prototype for a number of systems that gather and process information, and as such makes use of a number of high-level sensors. These include a stereoscopic camera, an omnidirectional camera, x-ray/visible spectrum fluoroscopes, and a thermal imaging camera. These sensors can be used to identify and gather data on science objectives, as well as serving as a basis for broad localization – that is, determining where the robot is.

For the purposes of obstacle avoidance and moment-to-moment navigation, the robot is equipped with a ring of fifteen ultrasonic ranging sensors, each of which measures the distance to the nearest obstacle within a 50-degree arc in the direction in which the sensor is pointed.

Manuscript received September 15, 2012. This work was supported by NASA Langley Research Center.

R. P. Monroe was with Christopher Newport University, Newport News VA. He is now with xkcd.com (e-mail: randall@xkcd.com).

S. A. Miller is with NASA Langley Research Center, Hampton, VA (e-mail: samuel.a.miller@nasa.gov).

A. T. Bradley is with NASA Langley Research Center, Hampton, VA (e-mail: arthur.t.bradley@nasa.gov).

B. Behavioral Navigation

Tasks such as the identification of dead ends, recognizing landmarks and rooms, and the construction of an absolute positional map are often combined with the task of low-level obstacle avoidance. These high-level tasks, however, are different in nature and often quite disconnected from the low-level ones – e.g. a person can be completely lost in a building and yet be no more likely to walk into a wall than a person who knows exactly the layout of the corridors and rooms.

Our approach was to split the navigation problem at the obstacle-avoidance level – that is, we approached the problems of obstacle avoidance and low-level motion as separate from that of high-level goal-based navigation. Instead of modeling the world internally at a high spatial resolution, we chose to implement a series of perception-based behaviors which could run independent of a high-level map or goal. These behaviors – computationally trivial when compared with path-planning and mapping algorithms such as A* and SLAM [5] – run in parallel, processing the sensor data in various ways and interacting with each other to decide what the robot's motion would be at any given moment.

Fig. 1. Trinity: an omnidirectional robot.

These lower-level behaviors proved capable of navigating the robot smoothly around and between obstacles. They allowed it to move quickly through open space, and slow down for careful navigation in cluttered space. They assumed nothing about the structure of the environment and required no warm-up or mapping period to become familiar with a region. Through moving these reflexive, sensor-based behaviors onto their own computational level, we have made the process of navigation more robust and efficient.

Since these behaviors can handle the moment-to-moment problem of obstacle avoidance, the higher-level navigation algorithms have far more freedom in the way they handle the still-important problems of localization and mapping. The behavioral system lays a foundation on which navigation systems such as [9], [10], and [11] can be built. These detail work on systems that use high-level vision systems that perform localization by recognizing and remembering regions. As a side project, undertaken in anticipation of the utility of these systems to the Trinity project (among others), the histogram-based topological localization algorithm described in [12] was successfully implemented and tested using web cameras in both indoor and outdoor environments. These and other techniques provide a robust and accurate method of localization and mapping that creates maps well-suited to established graph-theory navigational techniques. a

C. Rings

The low-level behaviors clearly needed to interact with and moderate one another, ultimately reconciling their varying impulses to arrive at a set of commands to the robot. The actual construction of behaviors themselves and a pattern of linkage between them would involve simulation and experiment, but a prerequisite of any work in this area was the development of a basic system for communication between the behaviors. The low-level behaviors, or 'reflexes,' were designed to be perceptual rather than cognitive – that is, we tried to avoid abstract reasoning about the environment. To this end, we considered carefully the type of data received from the sensors. The data consists of an array of n range values, one from each of the sensor directions, which can be thought of as a 'ring' of distances corresponding to the ring of directional sensors. We decided to implement behaviors that would use these rings of numbers as their means of transferring information. A ring consists of a list of values between 0 and 1, one for each sensor (that is, each direction). These rings proved effective in part because they could be easily combined, compared, and manipulated.

One of the other formats we considered for communication was a single vector -- the main output of each behavior would be a direction and a speed. This, however, did not encourage interaction. If one behavior reported that it wanted to go in one direction, and another behavior outputted a different direction, there was no obvious way to come to a consensus between them. The vector average (or any other method for producing a third vector distinct from the first two) could conceivably be strongly unacceptable to both behaviors. The rings, however, rated the acceptability of each direction. We found that they could be combined easily through element-by-element multiplication, effectively giving each behavior a veto for each direction. If either behavior gave a particular direction a low rating, the corresponding product for that direction would be low. The strengths of the preferences expressed by each behavior could also be easily moderated by outputs from other behaviors (through a process described later), allowing for the possibility of a great variety of interactions as the network of behaviors grows.

The rings also provided a conceptual framework that made the development of simple behaviors straightforward, as evidenced by the relative ease with which the basic navigational system was developed. The primary goals at this level of behavior were simple movement and collision avoidance. Regardless of whether the robot has a high-level goal, it should be capable of simple directional movement while avoiding collisions. The design of the initial set of behaviors reflects this goal and accomplished it quite effectively.

III. Sample Behaviors and Modules

We developed a number of behaviors and modules for the simple navigational framework. Several of these sample behaviors are detailed here, along with polar plots of example ring values. In addition to the examples discussed here, modules we developed included behaviors for light-seeking, wall-following, and finding open spaces.

A. Weak Avoid

The Weak Avoid behavior biases the robot toward open spaces. We wanted the value of each element in the ring (values of a sample ring are graphed on a polar plot in Fig. 1) to reflect the 'openness' of the space.

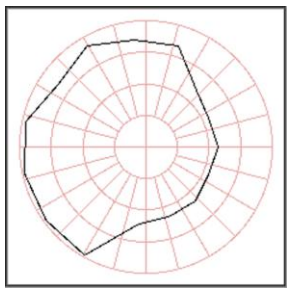

Fig. 2. Sample *Weak Avoid* ring plot.

The ring value was calculated from the sonar value by a simple logistic transform:

$$ringOut[i] = \frac{1}{e^{\left(2.197225\left(K_p - ringIn_n[i]\right)/K_d\right)}} \quad (1)$$

In equation (1), K_p and K_d are parameters determining the radius and 'strength' of the perimeter and 2.20 is a constant chosen to scale the parameters conveniently so that they represent 50% and 10% points on the logistic curve. Because of the wide field-of-view of the sonar sensors used in this implementation, this behavior can use the sensor data directly. If the sensors are more directional (i.e. laser rangefinders), this behavior should first transform the data to get a better measure of 'openness.'

B. Manual Direction

The Manual Direction behavior takes a direction as input and outputs a ring with preference for that direction. The ring has the value 1 in the direction, 0 in the opposite direction, and some gradient (often linear) in between.

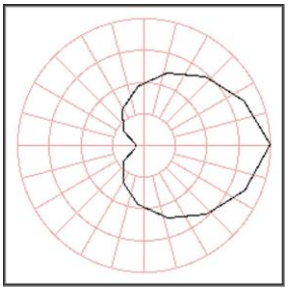

Fig. 4. Sample *Manual Direction* ring plot.

In the simplest case, this gradient value for an angle can be made proportional to the linear distance from the angle to the desired direction:

$$ringOut[i] = \frac{\min\left(\left(d - \theta_i\right)\bmod(2\pi), \left(\theta_{i2} - d\right)\bmod(2\pi)\right)}{\pi} \quad (2)$$

In equation (2), d is the desired direction, and θ_i is the angle to ith sensor. With this behavior – which is used as a component of several other behaviors – the robot can be easily biased to move toward or away from something. If the desired direction is 'vetoed' by another behavior, the directions nearest to the desired one become the robot's next choices.

C. Strong Avoid

The behavior outputs a near-zero value for headings that, if followed, would result in an immediate collision. It establishes a virtual perimeter around the robot with a range of roughly 20 centimeters. If an object is detected within that perimeter, it prohibits all directions less than 90 degrees from that object. It does this by replacing each value range[i] with the minimum value within 90 degrees angle of angle[i], and then taking the logistic transform of the data (as described in Equation (1) with constants F_p = 35 cm and F_d = 10 cm.

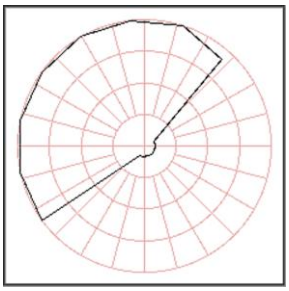

Fig. 3. Sample *Strong Avoid* ring plot.

This prevents the robot from driving straight toward the obstacle and from driving in a manner which would cause it to sideswipe the object. This behavior interacts constructively with the Manual Direction behavior – if a direction is selected that would lead to collision, the nearest direction parallel to the obstacle will have the largest value after the two are multiplied. That is, if a user tries to drive the robot into a wall, the robot will simply slide along the wall as if blocked by an invisible barrier.

D. Inhibitor Module

The inhibitor module takes as input a ring and a parameter X (with value on [0,1]) and it outputs a ring with uniformly weaker preferences (closer to 1). The inhibitor does this by scaling the input ring's values such that they fall between X and 1. If X is 0, the ring is output unchanged. If X is 1, the output ring is 1 everywhere – that is, the ring has been inhibited completely from having an effect.

The inhibitor transforms the input as follows:

$$ringOut[i] = \left(ringIn_n[i]\right)\left(1 - X\right) + X \quad (3)$$

In equation (3), X is the mitigation factor.

Each behavioral module can be paired with an inhibitor, and any module can output a value which can be used to inhibit another. For example, when the robot is closing in on a waypoint, it is moving slowly and there is less concern of hitting more distant walls, so the Weak Avoid behavior is more strongly inhibited the closer the robot gets to the waypoint (via a module that responds to a nearby waypoint

by outputting a higher *X* to the *Weak Avoid* inhibitor). This allows it to reach a waypoint that is near an obstacle even if the *Weak Avoid* behavior would normally prohibit this.

E. Mediation Module

The final outputs from the various behaviors are combined with a mediation module. This module creates a ring whose elements are given by the products of the corresponding elements in each of the n input rings:

$$ringOut[i] = \prod_n ringIn_n[i] \tag{4}$$

This simple behavior arbitrator easily aggregates many behaviors into one ring graph. The largest value of the ring corresponds to the travel direction compatible with all the behaviors. Unlike other behavioral aggregation techniques, were the effect of individual behaviors is frequently hard to determine, a cursor inspection of each behavior's rings, in the context of the aggregated ring, readily give human observers intuition into the interactions among all the behavior.

IV. Emergent Dynamic Navigation

As implemented, the robot is capable of efficiently navigating through dynamic environments. Its performance is not affected by quasi-static obstacles such as doors (as in [2]), and it is capable of maneuvering through a field of moving objects. This is notable, because this capability was in no way programmed explicitly into our system. Dynamic obstacle avoidance, often treated as a very different problem from static avoidance [1], was an emergent behavior of the static system.

V. Future Work

With the basic point-to-point navigation system described in this paper as a framework, we are free to develop higher-level navigational systems using topological methods making use of other sensors. In addition, we will be developing a version of the system using laser rangefinders as a replacement for the somewhat unreliable sonar sensors, and further exploring the way that these behaviors can be developed and connected to optimize navigation and intuitively avoid dead ends and other such pitfalls.

Often left unclear in Brooks' work [7] [8] is the methodology for choosing the way in which these reflexes are to be connected. In some cases, it is fairly obvious what connections will create the desired overall behaviors, but in others it is not. We have shown that even a few behaviors joined by primitive linkages chosen somewhat intuitively can create impressive functionality after only slight refining by experiment. There does not seem to be an easy way to determine, without simulation, what behaviors will emerge from a particular linked reflex system. However, these reflexes and their linkages can be quantified into a fairly restricted search space – a series of parameters representing inhibitions and individual behavior variables. If a metric is chosen that measure the robot's success at exhibiting a desired overall behavior – such as how quickly and/or frequently the robot reaches a randomly chosen waypoint – the parameters can be tuned using techniques borrowed from neural networks and mathematical optimization. These techniques, including simulated annealing [12], Tabu search, genetic algorithms, and reactive search [13], are extremely powerful and can be used to finely adjust parameters when a full mathematical description of the system is absent.

As a preliminary exercise toward demonstrating the feasibility of this parameter-tuning, we varied several parameters and measured the time the robot took to complete a simple obstacle course. The experiment was not intended for data gathering, but merely to demonstrate that completion times could serve as a useful measure of a particular choice of settings. Each set of parameters can be mapped as a point in an n-dimensional parameter space. Therefore, through use of a simulation or automated obstacle course, the parameters could be tuned by the mathematical optimization algorithms mentioned above. These algorithms can attempt to find the set of parameters that minimized the metric – in this case, the time taken to complete the course.

The techniques can be applied not only to the reflex parameters and ways in which the reflexes are connected, but to the very structure of the reflexes themselves. If the system can be simulated at a high enough rate, it is conceivable that entirely new reflex blocks could be developed and optimized solely through genetic algorithms. These self-developed reflexes could be phased in gradually, built first as small modifications of existing reflexes designed to augment a functioning system. As the techniques are refined, behavioral networks could be built entirely via these automated optimization routines. The reflex-ring behavior structure can serve as an excellent framework for the development of more sophisticated navigational algorithms while retaining the simplicity and characteristics that are the reason for their effectiveness.

References

[1] K. Fujimura and H. Samet, "A hierarchical strategy for path planning among moving obstacles," *IEEE Trans. on Robotics and Auto.*, Vol. 5, No. 1, 1989, pp. 61-69.

[2] W. D. Rencken, "Autonomous sonar navigation in indoor, unknown and unstructured environments," *Intelligent Robots and Systems*, Munich, Sept. 1994.

[3] C. C. Chang and Kai-Tai Song, "Sensor-based motion planning of a mobile robot in a dynamic environment," in *Proc. of IEEE Int. Conf. on Indust. Elec.*, Vol. 2, 5-10 Aug. 1996.

[4] A. Elfes, "Sonar-based real-world mapping and navigation," in *IEEE J. of Robotics and Auto.*, Vol. RA-3, No. 3, June 1987.

[5] J. E. Guivant and E. M. Nebot, "Optimization of the simultaneous localization and map-building algorithm for real-

time implementation," in *IEEETrans. on Robotics and Auto*., Vol. 17, No. 3, June 2001.
[6] J. O. Kim and P. K. Khosla, "Real-time obstacle avoidance using harmonic potential functions," in *IEEE Trans. on Robotics and Automation*, Vol. 8, No. 3, June 1992.
[7] R. A. Brooks, "A robust layered control system for a mobile robot," in *IEEE J. of Robotics and Auto.*, Vol. RA-2, No. 1, Mar. 1986.
[8] R. A. Brooks, *Flesh and Machines*, Pantheon Books, New York, 2002.
[9] J. Gaspar, N. Winters, and J. Santos-Victor, "Vision-based navigation and environmental representations with an omnidirectional camera," in *IEEE Trans. On Robotics and Auto.*, Vol. 16, No. 6, Dec. 2000.
[10] N. Aihara, H. Iwasa, N. Yokoya, "Memory-based self-localization using omnidirectional images," in *Proc. Fourteenth Int. Conf. on Pattern Recognition*, Vol. 2, 16-20, Aug. 1998.
[11] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *Proc. of IEEE Int. Conf. on Robotics and Auto.*, Vol. 2, 24-38, Apr. 2000.
[12] W. Jackson and M. McDowell, "Simulated annealing with dynamic perturbations: a methodology for optimization," in *Aerospace Applications Conference Digest*, pp. 181-191, Feb. 1990.
[13] R. Battiti and M. Protasi, "Reactive search, a history-sensitive heuristic for MAX-SAT," in *ACM J. of Experimental Algorithms*, 2, 1997.