UNDERGRADUATE EXPERIMENTS WITH MOBILE ROBOTS

R. Tanner, W. Mitchell, M.Z. Atashbar, and D.A. Miller

Department of Electrical and Computer Engineering
Western Michigan University, Kalamazoo, MI 49008

This paper details an independent undergraduate research project centered around using a Rug Warrior™ mobile robot for several types of experiments. The Rug Warrior™ is a mobile robot platform developed at the Massachusetts Institute of Technology by Joseph Jones, Anita Flynn, and Bruce Seiger and marketed by AK Peters Publishers. This paper includes a description of the robot, a discussion of programming techniques used for this robot, a description of a set of experiments conducted using the Rug Warrior™, and the Rug Warrior™'s advantages as a research tool.

The basis for the experiments is a series of "thought experiments" proposed in 1984 by Valentino Braitenberg. The experiments consisted of a few simple goals to establish basic operations such as movement, avoidance, and attraction. Later experiments consisted of various external stimuli such as light to judge responses to a given situation. This robot is a test bed for conceptual ideas that could then be scaled to other projects, including multiple cooperating robots.

Introduction

The increased power of microcontrollers and microprocessors in the past twenty years has augmented the ability of robots to perform independent missions with little or no human intervention. Robots can now use microcontrollers that exceed the power of older IBM AT systems of the early eighties. Organizations such as NASA and the Department of Defense are becoming increasingly dependent on the use of such robots to accomplish missions that would prove too costly and dangerous otherwise. An example is the Sojourner robot on Mars. The microcontroller allowed the robot to perform mission tasks that required little human intervention. This paper will discuss the basic principles of how to implement *behavior* programming techniques on a relatively low cost robotic platform.

The principle of fusion behavior programming is a different approach from the traditional method of programming robots. The traditional system relies on having an accurate model of the world the robot will function in to perform its task well. Many different problems arise when applying this type of programming approach to complicated and ever changing problems in a non-controlled environment. Fusion behavior programming offers a different method of programming based on layers and priorities. The most basic functions have the lower layer and could include simple activities such as motion control. The higher functions might include object avoidance or investigation of objects with certain attributes. An example of behavior programming is as follows:

A robot has a basic level of programming for motion. This program will execute indefinitely if left as is. In our layering schema, we put extra programs into the system. These programs could range from simple ones such as obstacle avoidance or low battery power, to higher ones such as search for heated objects that make sound. In either case, the additional program interrupts the normal run program to execute a function with higher priority. This type of programming can mimic some forms of behavior thus allowing for functions that are more versatile.

The robot used for the series of experiments must fulfill several goals. The robot platform needed to have a proven design with adequate processing power as well as flexibility at a moderate cost. The objective of the experiments was to study different types of behavior algorithms. Therefore we choose the Rug Warrior™ [1] robot kit.

Braitenberg[2] describes a series of thought experiments. These thought experiments used varying internal configurations and different stimuli to evoke behavior patterns similar to primitive organisms. Using the set of thought experiments provided in the book and fusion programming techniques, the Rug Warrior™ robot replicated the thought experiments proposed by Braitenberg.

Description of the Rug Warrior™

The Rug Warrior™ is a mobile robot in kit form, commercially available from A.K. Peters Publishers. The mobility of the robot consists of a pair of DC servo drive motors and a third wheel caster for balance. The robot also comes with a host of sensors ranging from limit switches to implement a "bump" sensor to infrared emitters/detectors to sound emitters/detectors. The builder can add additional sensors depending upon the use of the robot. Control of the robot and processing of the sensory data is accomplished through a Motorola MC68HC11 micro-controller.

The physical configuration of the Rug Warrior™ is a "garbage can" consisting of a platform approximately seven inches in diameter. The motors and batteries are mounted upon this platform. Then the circuitry associated with the micro-controller and the sensors are mounted above the motor platform. Finally, a protective Plexiglas cover, connected to the "bump" sensors, is placed over the entire arrangement.

Programming Techniques

At the center of all the robot control strategy is a programming technique called fusion behavior programming. The overall program is interrupt driven. This type of program control provides control over several subtasks. A typical program is broken into several tasks. Examples can include movement, escape, avoid and follow. Each task is programmed at first as if that was the only thing it needed to do. The next step is to assign a level of priority to each subtask. All subtasks are available for immediate execution. At any point in the process, a condition can occur that causes a subtask to seek control of the robot. It is at this point that a subroutine in the program allows the subtask with the highest priority to take control. Once the need of the higher priority subtask is completed, the subroutine passes control to a lower priority routine. This type of programming approach has several advantages.

The first advantage is a modular programming approach. This is a familiar technique for most programmers. A small task is defined and implemented by a program. The subtask is put into the main program and used as necessary. The modular programming enables simpler modifications as needed. Another advantage of using interrupts is using the priority subroutine to change the interrupt priority. As a byproduct, it is possible to combine priorities on the same level and give them the same priority. This represents a level of programming difficulty that increases in complexity rather quickly. This method of prioritizing is not for the novice programmer. In addition to programming complexity, programs that combine priorities consume large amounts of processor power.

Experiments with the Rug Warrior™

The first experiment was the task of movement. Many individuals do not understand how that can be difficult. The idea that simply applying an equal voltage to two motors will cause the robot to move in a straight line is false. The primary problem is no two motors are the same. Each motor will turn at a different rate even with the same voltage applied. The experiment called for the robot to go in a relatively straight line for some distance. The function that regulates power to the motor using pulse width modulation exists - the primary problem is what type of feedback is best suited for the experiment. The decision to use an open loop approach came about because of its simplicity, and that the robot did not need to go in straight lines for long distances. The algorithm was straightforward: apply a pulse width signal to the motors with an open loop bias value to restrict the amount of power available to either motor depending on which needs correction.

The second experiment was a logical progression from experiment 1. An autonomous mobile robot must not get stuck or haphazardly bump into objects in its path. Ideally, the robot should avoid obstacles it encounters. If the robot hits an object in its path, it needs to choose an alternate path to get around the object. This would also apply in situations when the robot gets into confined areas. The algorithm used for this experiment was as follows. The robot was put in motion in a direction. The microcontroller polled the infrared system constantly for any obstacles. The microcontroller also polled the bumper switches to detect any collisions. Should the infrared system or bumper switches activate an interrupt or flag notifies the prioritization function. Prioritization of the subtask avoids confusion of actions. The highest priority goes to the escape function. The escape function activates when the bumper switches hit an obstacle. The function that poles the infrared system takes second highest priority. If the escape function is not active and an obstacle is detected, the avoid function takes the control. The movement function has the lowest priority. If the first two conditions of escape and avoid are not active, the movement function is in control of the robot. The experiment for movement was to put the robot in a room or hallway with lots of obstacles and paths to see if it could perform the movement task without becoming stuck

The third experiment added the additional task of seeking light. A function in the program polled the two photoelectric cells for a detection of light. The stronger the light source, the larger the signal output from the photoelectric cells. The photoelectric cells are approximately at the ten o'clock and the two o'clock positions. The position helps to insure that each photoelectric cell receives a different amount of light from the same source. The program reads the value that represents the amount of light each photoelectric cell receives. The difference between the two values is calculated. The value returned determines which way he

robot will turn. The stronger the light is on one side, the more strongly it turns toward the light. Other functions such as avoid and escape still have priority over the seek function. The primary difference is the seeking light function now has a higher priority than movement. The experiment consisted of a single light on an opposite side of a classroom. The classroom had several workbenches and assorted obstacles. The goal was to see if the robot would eventually find the light. Additionally, later parts of the experiment had several lights with varying intensity and additional background light in the classroom.

The fourth experiment was a variation of the seeking experiment. The robot would seek light, but keep a predetermined distance from the light. In addition, the robot could go to another source if it were at a point that the second source could attract it. The algorithm for the orbit experiment is much like the seek algorithm with one alteration. A chosen value represents a certain intensity of light. When the robot is at a distance that represents that intensity, an additional interrupt occurs. This interrupt turns the robot away from the light source. As soon as the threshold value is low enough for the interrupt to stop, the robot resumes its seek mode for light. The experiment setup involved several parts. The first part used one light put in an open area first without obstacles and then with obstacles. The next part involved the use of two lights. The first set of lights where the same intensity. The second set of lights had a light with a different intensity. Both experiments ran with and without different obstacles.

The last series of experiments involved modifying existing functions and programs. A few of the programs such as seek light and follow objects allow a simple bias adjustment to affect the reaction of the robot. The intention of this series of experiments was to find out the limitations of the processor with the existing sensors. A brief explanation of the follow program is necessary. The follow program uses the infrared system to actively find an object that crosses it sensor path. When the object reflects infrared light back to the sensor, the robot follows where the side with most strength. The modification involved incrementally adding a bias or multiplier to the section of code that controlled the amount of power applied to the robot's motors. Eventually the bias level increased to a level that caused a processor error. After recording the number, the target moved at a slower pace to see if the robot would stabilize. Likewise, the target moved at a faster pace with a lower bias to find out if the robot could enter an unstable condition. The same process was used for the seek function to explore the limits of the robot.

Results of the Experiments

The first experiment resulted in a robot that moved in a generally straight line. The robot did tend bear to the right. As stated, the motors do not turn at the same rate when the same voltage feeds both motors. The solution exists in the standard library that comes with the interactive C program for the robot. A bias for the motor function can modify the output to the motors to correct for imbalances. If the robot bears to the right, use of positive correction applies. If the robot bears to the left, use of negative correction applies. The robot bears to the right without bias correction hence, the bias had a positive value. The objective was met for this experiment. The robot moved in a straight line for short distances. The program required to accomplish this is simple and robust.

In the second experiment, the robot encountered some minor problems that needed attention. As mentioned previously, the infrared system proved too sensitive on the left side of the robot. The first attempt of this series of experiments caused the robot to jerk around and

shake in one spot. The robot's infrared system showed that an obstacle was in the robots path at all times. The robot acted erratically, veering first one way then another. Eventually the processor overloaded and the robot simply spun in circles. To correct the problem, the eyeholes where cut until a window formed the eyehole.

The robot performed very well after the adjustment to the bumper shield. Objects used in the experiment, ranged in shape, color, material, and size. The robot avoided most objects before having to resort to the bumper switches to avoid an obstacle. The problem obstacles were objects that were a dark color. These objects did not reflect infrared very well and the sensors did not sense then until the robot was too close. The bumper system did very well to help the robot in situations that the infrared system did not work well. One situation for the robot did prove quite a challenge. In the lab the wall has a flat black trim at the point where the floor and wall join. The area that had this black trim also formed a corner or narrow corridor. The robot eventually went into this particular area. The infrared system did not pick up the wall very well, so the robot had to rely on the bumper switches to avoid the wall. Because the robot hit a corner about the same time the infrared system did finally sense the wall, the robot became confused for a brief period. This confusion sometimes caused the robot to take a few extra attempts to navigate out of the situation. To help alleviate the confusion, extra time was added to the escape function to allow it to have more time to back away from an object. Adding too much time could cause the robot to encounter another obstacle in conditions where numerous objects exist. Overall, the robot performed this experiment well and avoided being stuck. The only problem encountered was the avoidance of flat black objects. If several flat black objects were in the path of the robot, the robot could occasionally become stuck.

For the third experiment, the robot performed several light seeking and avoidance tasks. The first series of experiments had the robot find a single light source. The lab has several aisles formed from workbenches as well as numerous obstacles. In various locations, there were light sources. The robot then sought out one of the lights. Location of the light, the contrast between the source and the ambient light, and the narrowness of the beam determined how quickly the robot found the source. Location determined how much ground needed exploration before the robot found the light. If there were several obstacles or pathways, the robot needed to choose different paths to find the light. Several times the robot avoided an obstacle only to loose track of the light source and locate it using a different path. If the ambient light and the light source were approximately the same, the robot did not detect the source at further distances. Consequently, if there was a very bright ambient source, an open window with direct sunlight for example, the robot tended to ignore the light source. During some of the test runs, a flashlight with a narrow focus beam became the light source. When the area had low light conditions, the robot detected the light relatively easy. When the robot found the beam of light directly, the robot tended to stay on the beam regardless of surrounding light.

The second series of light seeking experiments involved several light sources. In the first set, several lights placed in close proximity were the source. These lights had approximately the same intensity. The robot invariably chose the light source closest to its sensors. When the robot encountered the light source, the escape function caused the robot to back away and turn from the light. Depending on the side bumped by the light, and the orientation of another light, the robot sought out the other light. The choosing of any individual light depended on the escape path of the robot and the position of lights. The next part of the experiment had light with

different intensities. The robot acted predictably if the robot encountered the lights in an open area with each light at about the same distance from the robot. The robot chose the brightest object. Arranging the lights in a line caused a slightly different reaction, with the approach of the robot determining its reaction. An approach where the robot arrived at the brightest one caused the robot to essentially lock onto the light and stay. If the robot approached from a side in line with the less bright light, it still locked on the light and stayed, with some exceptions. If an obstacle near the path caused the robot to turn at an angle slightly greater then ninety degrees, the robot pursued the brighter light. The only time the robot locked on the dimmer light happened when the orientation of the sensors was one hundred eighty degrees from the brighter light. The robot turned to the dimmer light rarely and only with several obstacles or distractions present. The robot followed lights and avoided objects well in most cases. There were some problems encountered when the ambient light and the light sources were approximately the same intensity.

During the first part of the fourth experiment, the robot orbited the lights in smooth motion. The second part of the experiment involved two lights. The robot would alternately orbit the lights if they were at a certain distance, dependent on the ambient light. If the distance between the lights was too great for the conditions, the robot orbited one light only. When the condition allowed the robot to orbit two lights alternately, the robot chose a figure eight path or an elliptical path. The path depended on the turning bias applied when the robot sensed a light source on either photo detector. A high turning bias caused a figure eight path; a lower turning bias caused an elliptical path.

The reason for this action is what happens when the robot turns toward or away from the light. A high turn bias causes the robot to turn quickly into the light when in orbit. When the turning action is completed the robot's photoelectric cell that is closest to light will tend to stay that way. The design of the program is to react to a difference in light, the robot will turn into the light it is already orbiting until the opposite photoelectric cell detects enough light to turn towards the other light. A lower turn bias causes a slower reaction to light. The slower reaction allows the robot's photoelectric cells more opportunity to compare the light sources, allowing the robot to move in a straight line long enough to detect the other light source and start an orbit around it.

The next portion of the experiment involved light sources with different intensities. If the light sources were close in intensity, the same results occurred as in the previous portion of this experiment. To make the robot orbit the lights in the same manner as the previous portion, the lights had to move closer. If one of the light source intensity is much greater, the robot reverts to the situation in experiment 2 where it locks on to a light and stays with that one exclusively.

If obstacles are in between the two light sources and relatively small, the robot generally performed the experiments the same as without obstacles. When the obstacle size approached the width of the robot path when it orbited in the non figure eight path is when different results occurred. The setting that determined the orbiting distance caused the robot to act differently. If the robot had a large orbiting path, the robot almost never orbited the lights in a figure eight path. The obstacle blocked the opposite source in a way that prevented the photoelectric cells from sensing the opposing light early enough to go into a figure eight path. The large orbiting path, combined with the obstacle avoidance, caused the robot to go around the obstacle. Consequently, if the robot had a small orbiting distance, the robot orbited around one light almost exclusively.

A small orbit path combined with the large obstacle tended to keep the robot focused on one light only. The robot performed the series of experiments well and to expectations.

For the fifth experiment, the robot remained stable until a bias multiplier for the turning approached eight. At that level, the processor would overloaded, causing erratic behavior, when a situation called for the robot to turn. For operations such as tracking moving objects and avoiding tight spaces, a bias of three to five, depending on the level of difficulty, work well without overloading the processor. When the bias was at a six, the robot overloaded in many situations except in the case where it tracked from a standing position. When the robot was in the follow program, an object must cross the path of the infrared system before it activates. A high bias caused to the robot to lock on the target very well. Unfortunately, if the object turns too quickly or there is several objects encountered while tracking the robot processor overloads. The experiment worked well only if the object turned slowly and there was no obstacles. Overall, the robot performed most tasks well at a bias level of three. The object of this series of experiments was finding the level of bias that would prove useful and not overload the processor.

Advantages of the Rug Warrior™ as a Research Tool.

The Rug Warrior™ is a reasonably priced (less than one thousand dollar) platform that provides mobility, control circuitry, and a basic set of sensors which will allow an investigator to explore issues associated with autonomous mobile robotics without the requirement to first develop the basics of a mobile platform itself. The MC68HC11 micro-controller can be programmed using an included interpretive C compiler, the 6800 assembly language, or a combination of the two. This provides the investigator the ability to work at the lowest level when a topic requires that level of interface while allowing the investigator capacity to formulate more complex problems at a conceptual level.

The Rug Warrior™ is also an expandable platform. The architecture of the system provides the ability to add various emitters and sensors to experiment with alternative control schemas. In addition to open locations on the existing printed circuit board, the platform has expansion ports providing the industrious investigator with the capability of providing parallel and serial communications for add-on boards of the investigator's own design.

The basic control programs that are provided with the Rug Warrior™ help move the investigator quickly beyond the basics of motion, thereby allowing pursuit of more advanced queries regarding autonomous mobile robotic platforms.

Suggestions For Further Research

A recharging station for the robot is almost a necessity. The robot could continually monitor the battery level on the circuit board. During operations, the battery pack will eventually drain to a point that requires the robot to recharge. The robot would then find its way back to the charging station and parks itself in the station for recharging. The implications for this function becomes obvious in an off world exploration project. The mother station could land on a planet and disperse several exploration robots. The mother station could comprise of a solar collector or more likely a nuclear generator. The exploration robots would come to the mother station when a recharge is necessary. To add greater versatility, the mother station could move when another area needs exploration.

Several Rug Warrior™s could be programmed to work together as a robot colony. A dedicated program would handle the communication among the robots. An experiment might comprise of the following scenario. The lead robot searches for an event that is of interest, e.g. a source of heat or noise. The robot signals a need for more specialized robots for investigation. Several variations of this experiment provide a wealth of possible experimentations.

Exploration of hostile or inaccessible needs accurate plotting of where the robot is and has been is of paramount importance. Therefore, an experiment where the Rug Warrior™ maps the surrounding area provides an interesting experiment. One possible scenario uses the Rugbat™ add on kit for the Rug Warrior™. The Rugbat™ uses sonar to locate objects; the same sonar can map objects as well. Consumption of memory resources requires additional memory modules several megabytes as a minimum. A memory management function similar to the battery function provides a means to manage large amounts of data. The primary difference is the robot could transmit the data to a central location for processing.

Bibliography

1 Joseph L. Jones, Anita M. Flynn and Bruce A. Seiger. *Mobile Robots: Inspiration to Implementation.* Second Edition, A K Peters, Ltd. Natick, MA, 1999.

2 Valentino Braitenberg. *Vehicles: Experiments in Synthetic Psychology.* MIT Press. Cambridge, MA, 1984.

Biographies

RALPH TANNER is an Associate Professor in the Electrical and Computer Engineering Department. His research interests include mobile robots, walking machines, and controls. He may be contacted at: ralph.tanner@wmich.edu.

WILLIAM MITCHELL is an undergraduate student majoring in Electrical Engineering.

MASSOOD ATASHBAR is an Assistant Professor in the Electrical and Computer Engineering Department. His research interests include mobile robots, sensors, and VLSI. He may be contacted at: massood.atashbar@wmich.edu.

DAMON MILLER is an Assistant Professor in the Electrical and Computer Engineering Department. His research interests include nonlinear circuits and systems, soft computing, and engineering education. He may be contacted at: damon.miller@wmich.edu.