Final Report: Project 1

Team 3:
Mark Branson
Amit Mathur
Matt Roman
Mike Taylor

February 16, 2003

Chapter 1

Hardware Description

1.1 Introduction

The key to the hardware design for this project is simplicity. Because of the fairly simple nature of the problem and the relatively brief time allotted, we chose an approach which minimized inaccuracy in robot movements. By minimizing error, we could avoid the costly (both in time and power) movements caused by error correcting motion. For this reason, we chose a design which was small and compact.

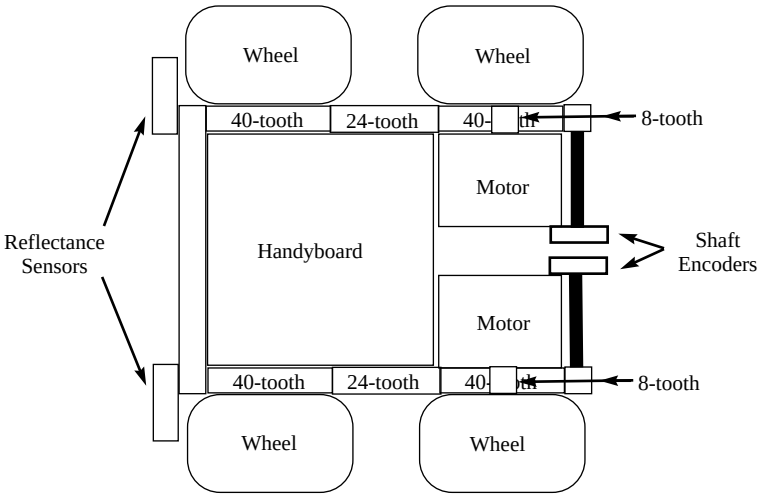

Figure 1.1: View of the robot from the top, with significant pieces labeled.

1.2 Sensors

Our robot design involved exactly four sensors: two break-beam sensors and two top hat reflectance sensors. The two break-beam sensors were built into shaft encoders in the rear of the robot. We built two shaft encoders to ensure that we were accurately measuring the

distance traveled by each side. The two shaft encoders facilitated calibration of the motors, which in turn contributed to the extremely straight paths followed by the robot between goal squares.

The top hat reflectance sensors were used for course corrections when leaving the goal squares. These two sensors were placed on the very front of the robot, approximately 1-2 millimeters from the test surface. The two sensors were placed approximately 13 cm apart (compare with an overall robot width of 18.5 cm). By placing the sensors close to the test surface, we obtained extremely different readings on the black tape and on the dingy grayish-white floor. Their extreme horizontal displacement enhanced their usefulness for our purpose by ensuring that, if both sensors were reading black, then the robot must be oriented perpendicularly to the tape to a very high degree of accuracy.

1.3 Motors and Gearing

Our primary concern in our motors and gearing was obtaining a system which incurred minimal slippage, and thus maximal accuracy in direction. Because of the nature of the project, a small error induced by slippage of one wheel could cause major inaccuracy by the time the next goal square was reached. Additionally, no method existed for determining whether the robot was on course while between goal squares. Operating under both these conditions, we made several design decisions.

First, we chose to have a very short wheelbase. The distance from the front axle to the rear axle was just 6.5 cm. This extremely short wheelbase (practically the minimum, since the wheels have 2.5 cm radius) gave the robot extremely tight control, especially during turns.

The second design decision lay in the gearing. The wheels were geared 5-1, which reduced the robot's speed tremendously. We decided that a faster moving robot, while able to complete the course more quickly, would experience substantial slippage on the wheels. Since our primary goal was accuracy and not speed (the robot still completed each lap in approximately one minute), this geartrain worked admirably. In addition, the simplicity of the one stage reduction from an 8 tooth gear to a 40 tooth gear provides little opportunity for errors to be introduced in the geartrain. This gearing is what produces the superimposition of gears in Figure 1.1. Specifically, the 8-tooth gear is directly above the 40-tooth gear.

The third and final design decision was to use a four-wheel drive system. We chose to use four wheel drive for the same reason we chose to make the design decisions above. By powering both front and rear wheels, we were assured that they were moving at precisely the same speed.

Chapter 2

Software Design and Algorithms

2.1 Introduction

The software algorithm which we chose was extremely simple. In short, our robot drives straight for a set distance, finds the edge of the box, moves into the box, and turns. It then moves forward until it finds the edge of the goal square and then aligns itself with that edge. This cycle repeats until the robot has moved around the course three times. Because this simple algorithm permits little room for error and coordinates well with the hardware design, the robot is able to move smoothly through the course.

2.2 Behavior Model

Although the actions which the robot takes are not behaviors in the sense that Murphy uses the term, we will refer to them as such. Each behavior is triggered by the completion of the previous behavior, so in reality, these behaviors form a kind of fixed action pattern. However, since each depends on sensor data (whether as a taxis or a reflex) we cannot consider them as an action pattern either. We thus use the term behavior ambiguously and leave it at that. The robot has five basic behaviors. We will refer to them in this document as: Long Move, Move into Box, Turn, Align, and Search, connected as shown in Figure 2.2. One of these behaviors, Search, was not used in the demonstration, but will still be detailed in this document.

2.2.1 Long Move

The move behavior is the simplest of the five. This behavior takes in data from the shaft encoders only. While the shaft encoders tell the robot that it has moved less than a certain distance, it continues to move forward. The behavior is initially triggered by the completion of the Align behavior.

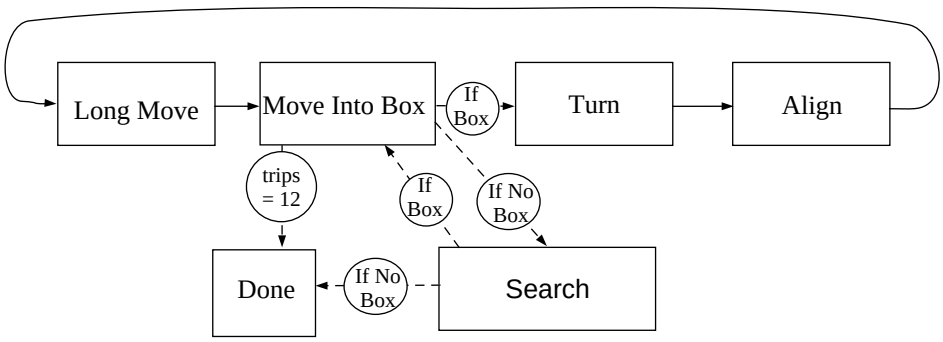

Figure 2.1: The interconnection of individual modules in the software design.

2.2.2 Move into Box

This behavior is triggered by the completion of the Long Move. In this behavior, the robot moves forward slowly until it finds the box, and then moves into the box. If it does not find the edge of the box (using the reflectance sensors) after a certain number of turns of the shaft encoder, it initiates the Search behavior. If it does, it moves into the square and initiates the turn behavior.

2.2.3 Turn

This behavior causes the robot to turn 90 degrees. The exact value of distance turned is inputted as a calibration value. Since we desire for the robot to turn in a very small area, this turn is completed by driving one motor forward and the other backward. This, together with the robot's short wheelbase, causes the robot to turn in a very small area.

2.2.4 Align

This behavior is triggered by the completion of the Turn. The robot moves forward until one of the reflectance sensors reads black. The robot then turns (in very small increments) until the other sensor reads black. Once both sensors read black, the Long Move behavior is initiated.

2.2.5 Search

This behavior is only initiated in the event that the edge is not found during the Move into Box behavior. This behavior is very simple, because the robot has very little time to find the box if it has missed. The robot first searches to the left of where it is by turning and moving forward (our robot has a slight pull to the right - if the box is missed, it should be on that side). If it still does not find the box, it moves backwards by twice the distance that it just traveled, effectively moving just as far to the right, again checking for the box along this whole route. If it still cannot be found, the robot stops moving. If the box has been

missed by this much, there is clearly a calibration error that must be repaired by the team. If the box is found, the robot returns to the Move into Box behavior.

2.3 Conclusion

Overall, this strategy worked very well. Since the robot never went off course, the Search behavior was never actually used, although it worked well in test. The key to the performance of the software (and specifically performance on the extra credit course) was the robots selective perception. Although the line on the floor would have been sensed by the reflectance sensors, the Long Move behavior ignored that data because it was unnecessary. Rather than constantly checking for errors, the robot relied on its data being correct and trusted to the strength of the hardware design. Because of our iterative hardware design (see Section 3.2), this software design was indeed robust enough to earn that trust.

Chapter 3

Team Organization Evaluation and Plans

3.1 Introduction

Overall, our team organization worked extraordinarily well. Each team member completed his required individual tasks and communicated well with the others. However, we did experience some difficulties with our milestones (several of the larger tasks taking longer than expected) and were forced to adjust our fallback plan.

3.2 Individual Efforts

Although our initial team organization document called for each team member to work independently for most of the time, the actual organization reflected a much more balanced approach to the project. Two team members, specifically Amit and Matt, ended up working closely together to ensure that the hardware and software designs would mesh. Rather than working independently, they met as a team and produced iterated designs of both software and hardware components. This worked well for this project because of the simplicity of both of these components.

Since this model worked so well for this project, though, we are considering keeping it for the next project. Specifically, this model would call for the hardware and software manager to work as a design task force to revise the initial design agreed upon by the team. As the projects become more computationally complicated, though, this model will likely become less useful for the software component and more useful for the hardware component. Specifically, as the software component grows, the hardware component can be iteratively constructed to match the needs of the software component. We currently plan to adopt this strategy for at least the second project, and then to reevaluate its usefulness.

Another individual effort which ended up being distributed amongst the entire team was the testing phase. Each team member tested his individual component, and the final integration testing was completed at a team meeting. Although this worked well for this project, a more structured environment may be more appropriate for later projects, where individual robot tasks will need to be tested independently. In these later projects, the need

for one person to organize the integration testing will be paramount. Our current model, despite the fact that it emphasizes teamwork and cooperation, would become difficult to organize on a more complicated project.

3.3 Team Structure and Supervision

Our team, which uses a floating supervisor position rather than a concrete team leader, worked very well. The floating supervisor, Mike on this project, became a coordinating figure rather than an authoritarian leader. Rather than making decisions about group activities, he coordinated efforts to plan activities and make decisions amongst team members. The presence of a team supervisor as an equal and a coordinator reduced friction among team members. By reducing team friction, we improved the product that we created and worked more efficiently.

In general, we plan to continue using the existing team structure and supervisory system. Because of the overall success of this project, we have been extremely pleased with our team structure. All team members reported a feeling of satisfaction and are looking forward to using this structure in the future.

3.4 Conclusions

We have found the team organization to be an overwhelming success. Because of our success with this system and team member satisfaction, we have decided to retain this system in whole, with the few changes mentioned in section 3.2. Of course, as with any model of this type, we will remain constantly in search of methods to improve the overall team structure and organizational model.