Final Design Documentation

Project 3

Team 3:
Mark Branson
Amit Mathur
Matt Roman
Mike Taylor

April 30th, 2003

Section 1

Hardware Description

1.1 Design

The general focus of the hardware design for this project was for the following basic principles:
-Accuracy in movement
-Lots of precision-placed sensors
-Geared for accuracy, then speed
-Zero turn radius

Each of these goals were ultimately met and we were very pleased with the end result.

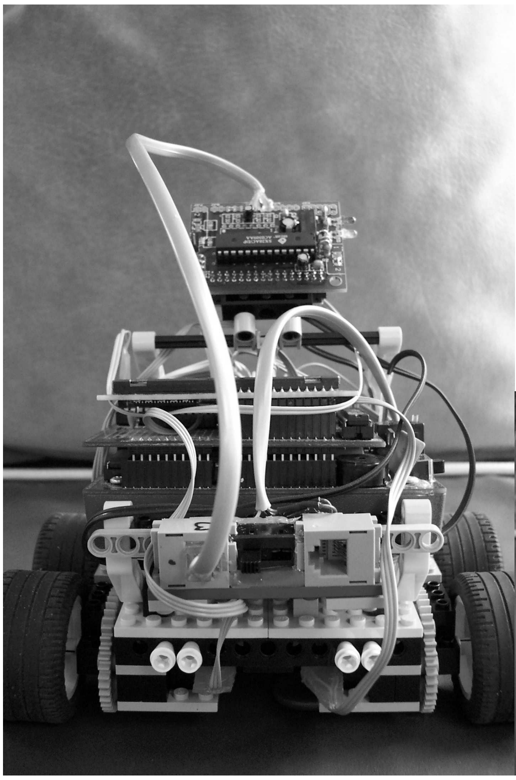
The accurate movement was needed to facilitate precise movement between coordinates in the virtual grid. We had no way of determining our location dynamically, so we had to rely on accurate encoder readings.

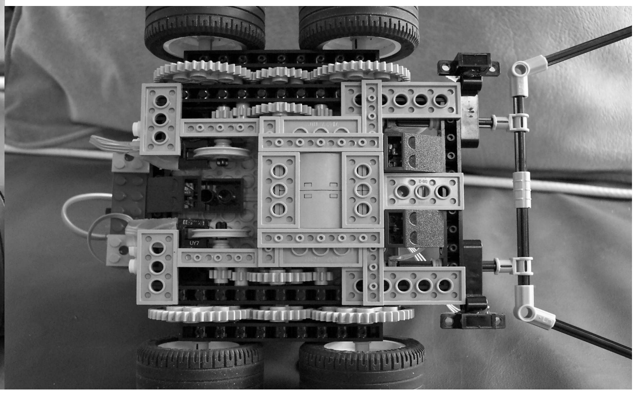
The gearing for accuracy concept meant we had to design our gears to allow as many encoder ticks per revolution as possible, while still allowing for decent speed. The speed was needed to compensate for restarts that might be required due to any software troubles.

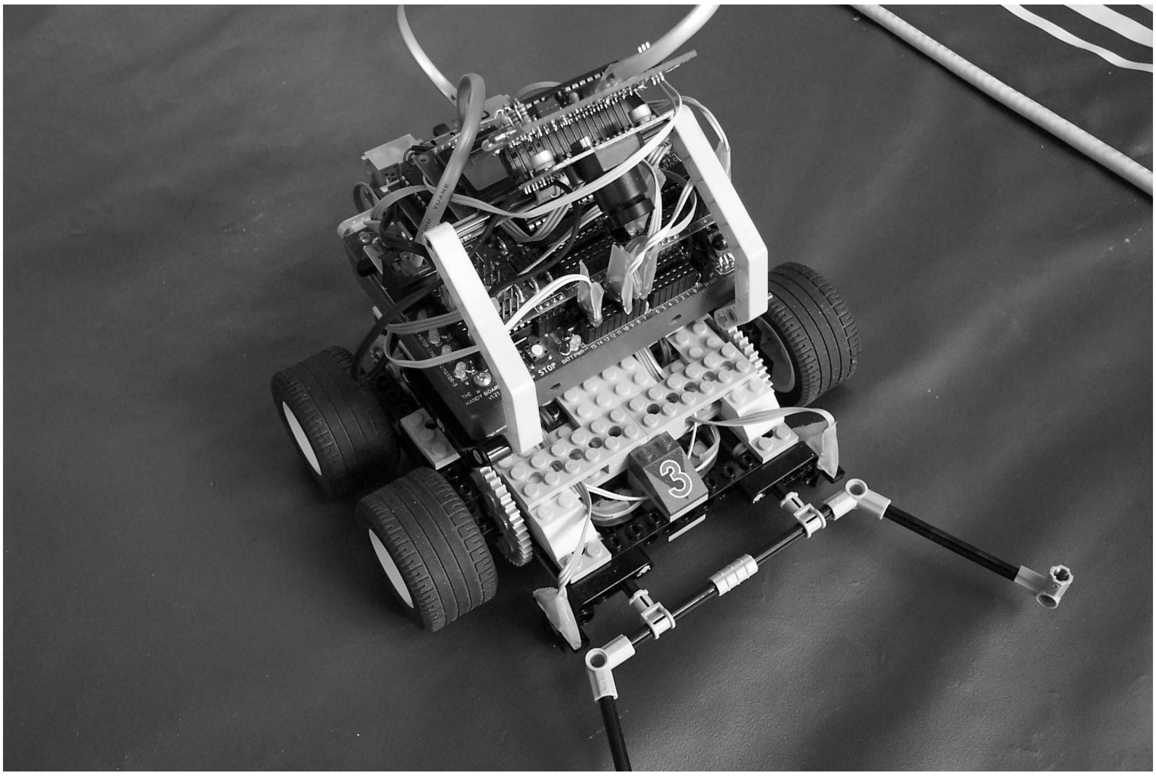
Finally, the zero turn radius was key to not accidentally pushing around any orange blocks when turning.

1.2 Wheels and Gears

We opted for the same basic design as in our first project. The torque for forward motion and turns was provided by two motors attached to the rear wheels. The gear ratios were 5:1. This allowed for reasonable speed with a small loss in torque. The wheels were then geared to the front wheels at a ratio of 1:1. This forced the front wheels into lockstep with the real wheels and significantly cut back on slippage.







1.3 Sensors

Our hardware and software design teams came up with the following list of needed sensors. We needed two bump switches attached to the collecting apparatus to handle running into an outside border, and to detect a bump into the dynamic object. We needed two reflectivity sensors near the front of the collecting apparatus to allow us to sense the precise location of the goals. Finally, we needed exact placement of the CMU cam so we could view a block in any part of the collection area.

Section 2

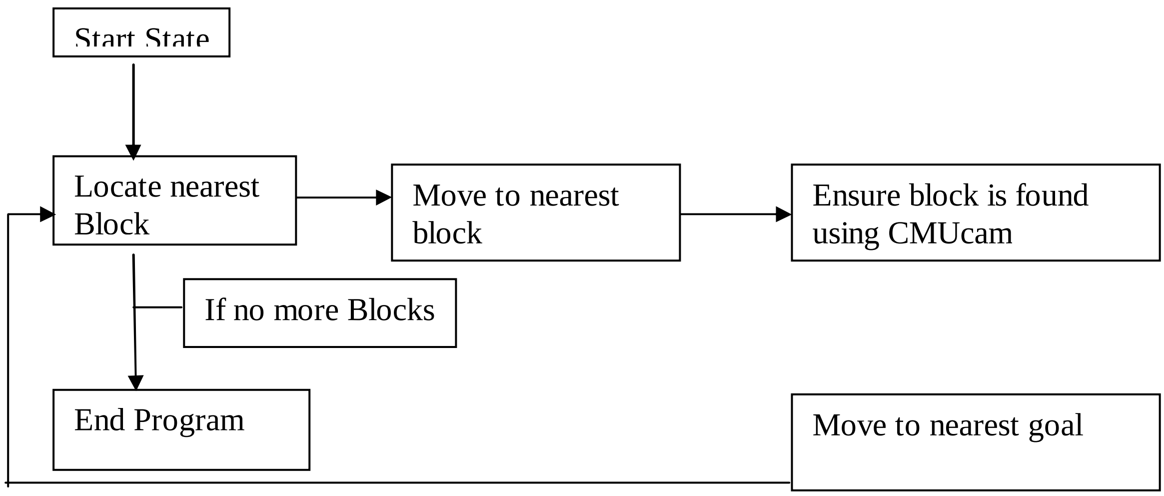
Software Design

2.1 Introduction

The software model is a hybrid reactive-deliberative architecture with a focus on the deliberative side. Tasks are generated and path plans are generated by the list of coordinates input before the demo starts. Once the robot gets a block, she attempts to push it into the nearest goal. We had code for avoiding extra blocks and chasing the blue bot, but in both cases the complexity and debugging issues outweighed the minor improvements added by them.

2.2 Code diagram

The following is a rough diagram of the state transitions undertaken by our robot as it navigates the course.



2.3 Conclusions

Our experience with coding this robot was that we should focus entirely on the main goal and ignore anything not totally required. The blue robot is a perfect example of this. It is not worth losing track of current position just to take a shot at the pipe dream that is the blue bot. We stuck to the main goal of getting blocks and moving them to a goal. The decision not to worry about multiple block pickups was a tough one, and only came after realizing that the potential for error outweighed the 11 to 21 point swing induced by moving an extra block into the goal. The code we had that did avoid tended to send us into an indeterminate location and recovery of current position was impossible, which would cost us all the rest of the blocks.