**DE2Bot Coded for IR Remote Control Manual Parking and Semi-Autonomous Parking Modes**

Ashwin Vadivel

David Thompson

Jeremy Carter

Jonathan Hernandez

Pratik Kunapuli

Georgia Institute of Technology

School of Electrical and Computer Engineering

Submitted

April 25, 2017

# Abstract

This design report outlines the design process, implementation, and results obtained from our software implementation on the DE2 Robot enabling it to perform manual parallel and perpendicular parking as well as fully autonomous perpendicular parking. The document outlines the procedures undertaken to successfully program the robot’s movement and parking capabilities. An overview of the project management plan with a summary of the results is also presented. Additional proposed solutions to fine tune the robot’s motion are also presented to make the final deliverable more robust.

**DE2Bot Coded for IR Remote Control Manual Parking and Semi-Autonomous Parking Modes**

# Introduction

The document presents a software solution to be able to autonomously and semi-autonomously park a DE2 robot, or “DE2Bot” for short. DE2Bots expand upon the DE2 boards with motors and I/O devices including IR receivers and sonar capabilities among others. These additions enable a well-programmed DE2Bot to move and detect its surroundings via remote control. This implementation adds three sets of SCOMP assembly code: basic movement commands, specific combinations of movements, and IR-based remote initialization and direct guidance. The fully-programmed DE2Bot met all three requested project goals of semi-autonomous parking in a numbered perpendicular space and manual remote control for a ‘driver’ to park in each of a parallel and a perpendicular parking spot within the mock parking lot.

# General Methodology

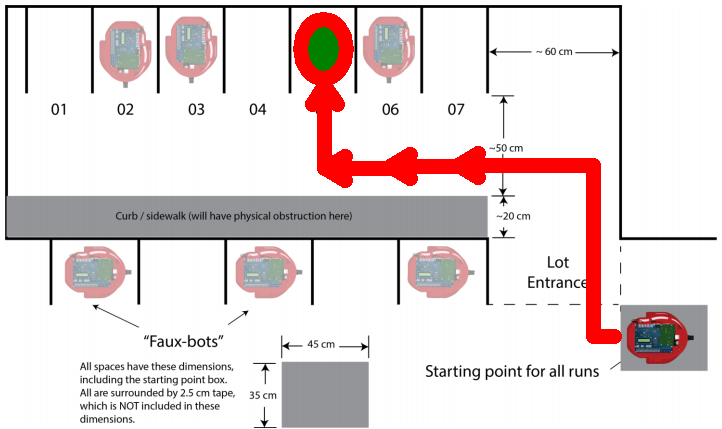
## Initial Planning

The team analyzed the mock parking lot layout to explore possible paths that the DE2 robot could take to reach the parallel or perpendicular parking spots. A detailed project requirement analysis simplified robot movements for the entire lot into three simple maneuvers:

1. move forward a set distance
2. rotate 90 degrees clockwise (CW)
3. rotate 90 degrees counterclockwise (CCW)

These three movements became basic building blocks that, when strung together in functions, direct the robot to move and park in any selected space. The fully-autonomous parking mode could be satisfied using three combinations of the simple maneuvers:

1. an initial upward “S-Curve”
   1. forward movement → rotate 90° CW → forward movement(s) → rotate 90° CCW
2. traverse to the desired space
   1. a. ( 7 - <space number> ) forward movements, e.g. 2 movements to reach space #05
3. enter the perpendicular space
   1. a. rotate 90° CW → forward movement(s)

A combination of these three maneuvers directing the robot to space #05 is depicted in Figure 1.

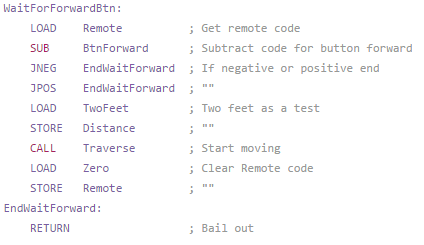
## Figure 1. DE2Bot semi-autonomous path to space #05 using an “S-Curve” function, 2 forward movement calls, and an “enter space” function.

As the project requirements call for both manual and semi-autonomous control, the team decided to utilize only the IR receiver paired with a remote control for movement commands. Initial setup uses the supplied code to disable the safety mechanism following which the DE2Bot will only check for IR commands. This decision further simplifies the code required and the user interface.

An additional decision to use the DE2Bot’s sonar sensors was approached as a contingency to be finalized during the final stages of the project. Near the final stage, sonar was implemented to modify the s-curve code to re-align the DE2Bot with the side wall. This modification significantly improved final parking results for parking space numbers 1-3.

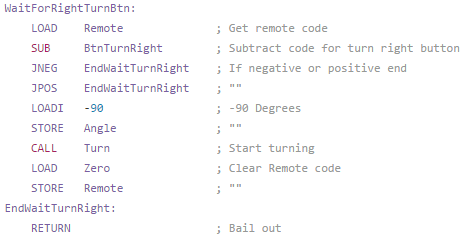
## Movement Implementation

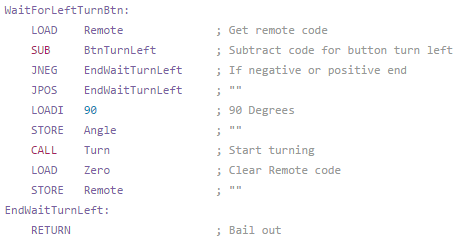
The robot's movement was decomposed into three primary subroutines which direct it to move forward, turn 90 degrees counterclockwise, and turn 90 degrees clockwise. Each subroutine was written in assembly to match the entered remote code and then proceed to turn or move forward a set distance amount using DE2Bot provided odometry readings. The initial implementation for these subroutines - WaitForForwardBtn, WaitForRightTurnBtn, WaitForLeftTurnBtn - are presented in Figure 2 and 3.



**Figure 2.** Assembly code that controls the robot's movement forward. Double checks the code received

by the remote and if correct code was received proceeds to move the robot forward a set distance.

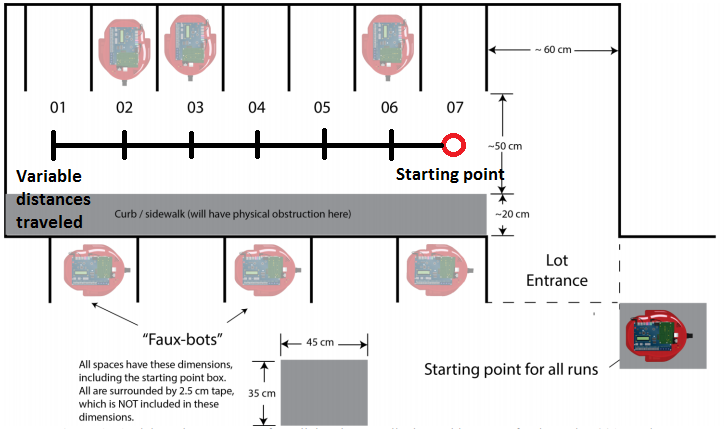




**Figure 3.** Assembly code subroutines that control the robot's turns. Upon receiving the correct IR remote code the robot will rotate to the right (CW) or left (CCW) by 90 degrees.

Construction of function subroutines to reach each parking spot will utilize set numbers of calls to these three movement subroutines. Each subroutine also has a specific margin of error in which the robot will remain within after turning or moving forward or correct for with automatic minor adjustments. Such corrections aids with the accuracy of the robot. One of the main concerns with moving relatively long distances is that a slight error in the turning degree will be amplified the longer the robot travels. With this implementation, most or all error introduced when turning will be immediately resolved thus lowering the chance of the robot veering off course.

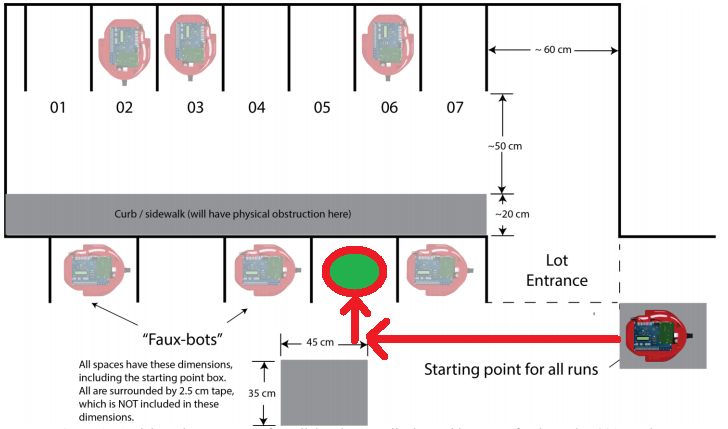
For the autonomous routines, the three building block subroutines were combined to direct the DE2Bot to a “known starting point” in front of the seventh perpendicular parking spot in the parking lot as shown in Figure 4. Based on what spot is chosen to be parked in, the robot will move forward a variable amount and execute a perpendicular parking spot routine which aligns the robot with the spot and pulls into the space. Execution of this task will be controlled with the numbered buttons “1” - “7” on the remote with each number being assigned to the correspondingly numbered spot in the lot.



**Figure 4.**  Starting position for each perpendicular parking subroutine is marked as circle. Depending on the parking spot that is chosen, the DE2 bot will move a variable number of set distances (shown in tick marks above) until it is in front of the desired spot and will then pull in.

Breaking up the movement into sections sped up the debugging process. Early verifications had shown that refinements to the basic movements quickly reduced combinational errors found during semi-autonomous testing such as compounded heading error during forward movements. As one example, the specific distance assigned to the basic forward movement was modified several times in early implementation.

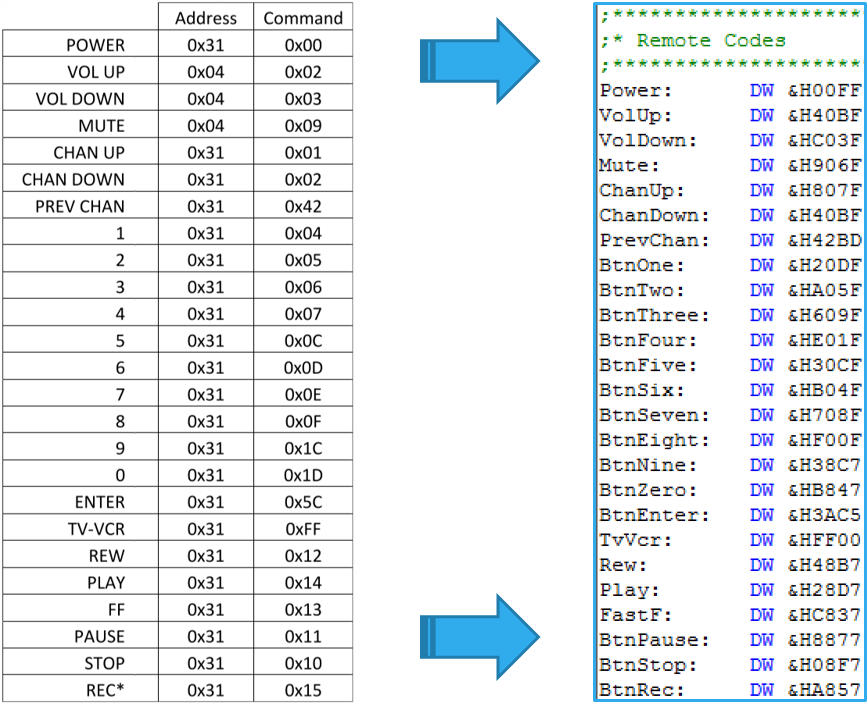
An extension of this approach was used to guide the DE2Bot manually to the parallel parking spots. Using the same three building block subroutines, the “driver” of the DE2bot would use the remote to command the robot to move forward. Once the robot is aligned with the space, the driver would then initiate the self-parallel parking subroutine used in the fully-autonomous mode that directs the DE2Bot to turn 90 degrees clockwise, pull into the parking spot, and then turn 90 degrees counterclockwise for proper orientation within the space. This satisfies the requirements for semi-autonomous parallel parking. Figure 5 diagrams an example path the DE2bot takes for this semi-autonomous task.



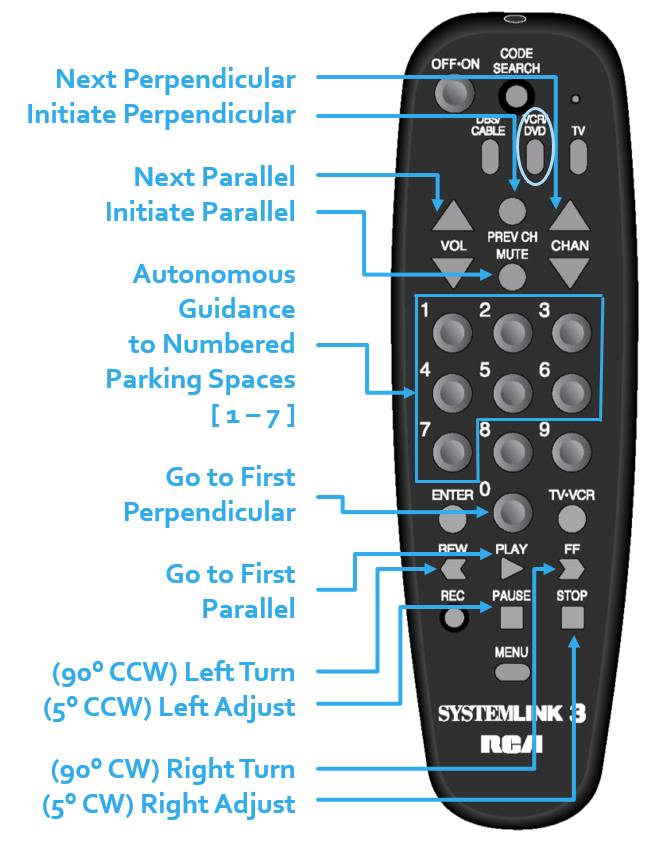
**Figure 5.** An example path that the DE2Bot can take towards a parallel parking space. Manual control of the robot is used to guide the robot along a straight path to align with a chosen spot. Then, execution of the parallel parking subroutine completes the task.

## Robot Control

Project requirements required the use of a universal remote for manual control of the DE2Bot via the IR receiver found on top of the robot housing. This design utilizes the remote control as the primary input for both manual and fully-autonomous modes, forgoing additional DE2 push-button or switch use. For the project, every remote IR code was converted through a C++ script to assembly code for potential use as seen in Figure 6.



**Figure 6.** IR command translation from remote manual to assembly code via C++ program.

With access to all of the available IR codes, the team was able to pivot the original proposed control scheme when initial selections proved less than optimal in testing. The UI as demonstrated uses the VCR buttons (e.g. REW/PLAY/STOP) for manual “driving” control to direct the robot to its assigned parking space as diagrammed in Figure 7. The PREV CH button is assigned to execute the semi-autonomous perpendicular parking function whereas MUTE executes the parallel parking function.

**Figure 7.** Universal remote mapping for communication between the user and the DE2Bot.

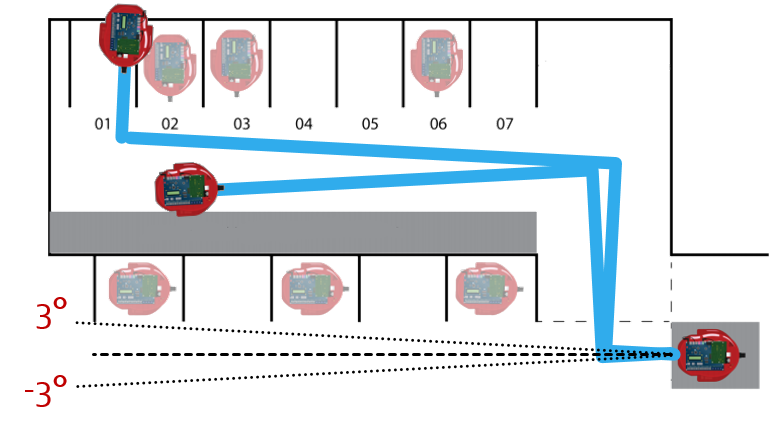
Pressing a number key [1-7] commands the DE2Bot to initiate fully autonomous mode and proceed to the selected space number.

## Project Management

Implementation milestones tasks and dates were broken down under Design Proposal, Robot Motion Tasks, Sonar Optional Tasks, and the Final Demo sections as listed in the Gantt chart included as Appendix B.

## Contingency Plan

While the design decisions selected came from detailed project analysis, the potential remained for issues to arise during implementation. One such potential concern came from relying on the odometry readings alone in the movement subroutines. In parallel to the movement code, the team worked on a sonar-based contingency plan. Mid-stage testing indicated an issue with the human placement of the bot at its initial starting point. A deviation of about 1-3° in either direction would end in a failed parking attempt with the DE2Bot colliding with the simulated curb or the rear wall behind the perpendicular parking spaces, depicted in Figure 8.



**Figure 8** Minor initial placement error propagates into failed semi-autonomous parking attempts.

To address this issue, the contingency code became a set of subroutines combining two sonar range readings off the rear lot walls which then directed fine directional adjustments during the S-curve routine.

# Technical Results

The results of the final demonstration were successful as per the predictions gathered from testing prior to the demonstration trial. The DE2Bot successfully completed the three parking procedures in three trials, leaving the fourth trial run unused. The total time taken to perform the three parking maneuvers was under 4 and a half minutes. When performing the manual perpendicular parking, the DE2Bot ended up on the lines of the parking space, thereby not completing the procedure perfectly.

# Conclusions

Overall the results of testing in addition to the final demo indicate that our designed solution to the proposed problem was successful. The three parking procedures were successfully completed in the allotted time in the mock lot. To future designers who may undertake this project, it may prove useful to know what we learned upon completion of the project. For one, the on-board odometry proved to be quite reliable for the purposes of re-alignment. Our implementation improved the odometry based alignment while using sonar data to improve human placement alignment. Another useful method was outputting sensor data to LCD and 7-segment displays. This sped up troubleshooting by making it easier to identify errors, enabling faster code prototyping and iteration. The simplicity in design of the project allowed tasks to be delegated easily to individual group members. Only assembly code needed to be modified, thereby negating the need to modify the SCOMP architecture on board.

Should this task be continued in the future, the team would make two notable additions to the implementation. First, sonar ranging would be implemented into the parking subroutines by using both far and near wall distances to fine tune the parking procedure. Another addition would to reprogram the remote and incoming IR code list. This would reduce the chance of errant IR signals affecting the operation of the DE2Bot. An additional code to be added would be a stop command when in manual parking mode as well. Overall, the DE2Bot implementation presented by our group was successfully able to accomplish the required tasks. Further additions would be required in order to ensure successful operation in spaces other than the mock parking lot that was used.

Appendix A: Gantt Chart of Project Timeline