Frank Ni
Kevin Hsiue
Project Propsal

Self-Parking Car

For the final project, we aim to implement a Self-Parking Car that can parallel park itself in a predefined space. The goal of this project is to provide a control system for a RC car that has rack-and-pinion steering system so as to simulate a real life parking situation. The task will begin with being able to drive the car around using an on-board FPGA and an base-station FPGA that can communicate wirelessly. Similar to a real driver, the next step would be to attempt to park in a very comfortable space. The final step is to determine a difficult but reasonable parking space that will push the limits of our parking system. The on board FPGA will control the vehicle while reading input from IR distance sensors that will allow the vehicle to gauge its position relative to the parking space. In addition, we will be displaying on the screen the space that the car sees. This project will demonstrate the ability of the FPGA to host a very precise control system to navigate a vehicle into a desired position.

In order to break the project into manageable pieces, we devised a pipeline of the input from the sensors to a representation that is understandable by the vehicle as well as humans. However to begin, we will provide a simple description of parallel parking by breaking it down into steps.

The first step requires the driver to know that there is a parking spot available. The second step will be to pull the car next to the car in front of the desired parking space. The third step is to turn the wheel towards the curb and back up at a 45-degree angle. The fourth step is once the car is into the spot a certain amount, straighten the wheel and continue. The fifth step requires the driver to turn the wheel away from the curb and check to make sure not to hit the car behind.

These steps detail a perfect scenario in which every step is done precisely. Of course this isn't always possible, so the key is to have a FSM control system that will cause the car to do certain maneuvers when its within a certain distance of a obstacle. As such, a proper representation of the obstacles will be required. First the we will position 4 IR distance sensors on the chassis of the car, which during the first step of the parking procedure, will detect and record the obstacles it sees to its right. Upon detecting a reasonable space in step two, the car will pull a car's length forward beyond the space. During these past two procedures, the Obstacle Detect block will send the data to the Obstacle Map block, which will convert the raw data into a matrix representation that will show where the walls are. This is the ultimate representation that we will use to allow us to see and navigate the car into the parking space.

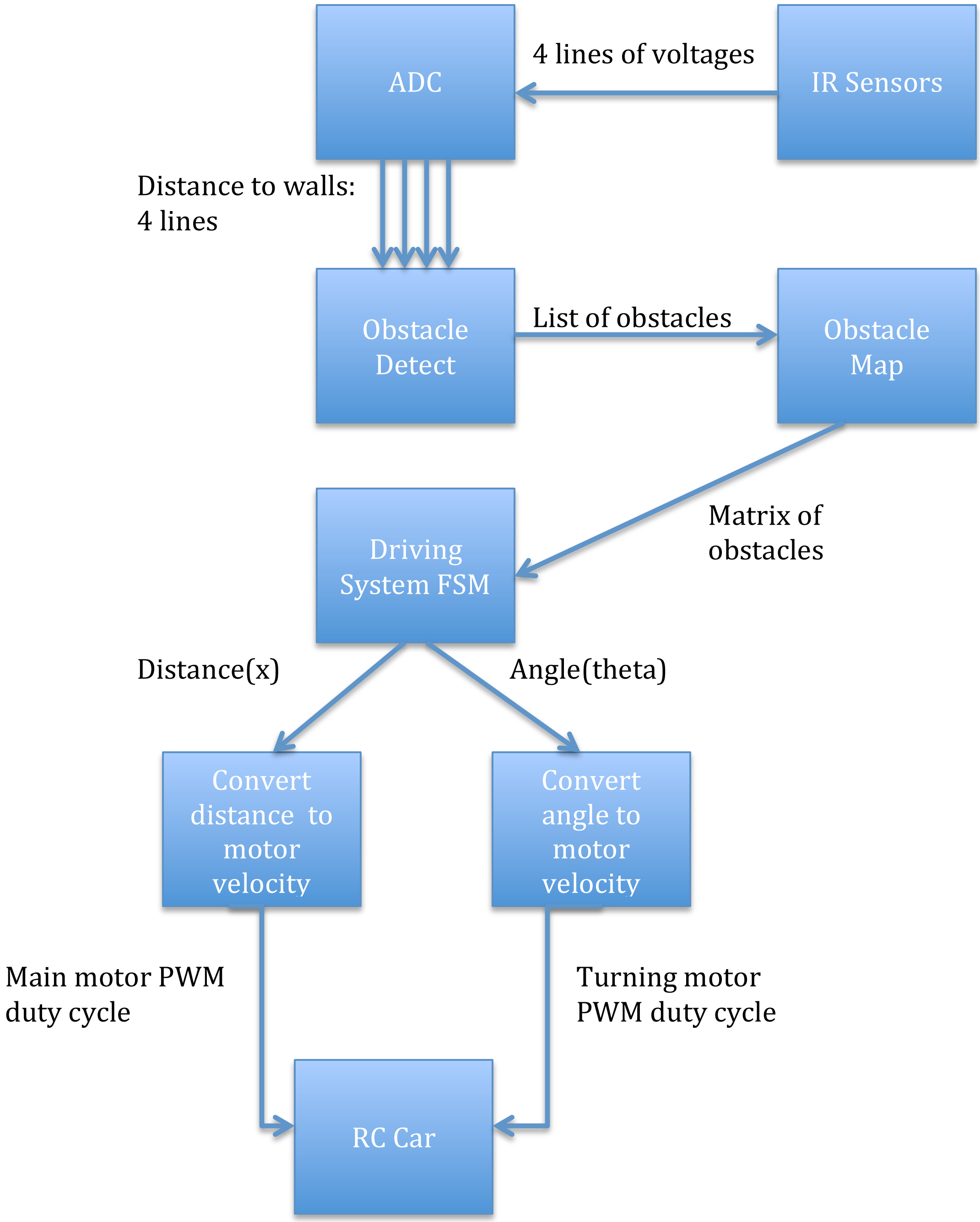
ADC
4 lines of voltages
IR Sensors
Distance to walls:
4 lines
Obstacle Detect
List of obstacles
Obstacle Map
Matrix of obstacles
Driving System FSM
Distance(x)
Angle(theta)
Convert distance to motor velocity
Convert angle to motor velocity
Main motor PWM duty cycle
Turning motor PWM duty cycle
RC Car

Figure 1

The first block, Obstacle Detect, will require a memory that is sized at (sampling_rate) *(adc_resolution)*(speed of vehicle)*(distance until finding space). This will store a sample for every wall to the right it sees. The second block, Obstacle Map will require a memory that will be experimentally determined. It is needed to store the memory map, which will be represented by a matrix of 0s and 1's, where 1's represent the obstacle. In addition to determining the initial map, these two blocks will create a real-time map of where the car itself is located with respect to the obstacles. Frank Ni will implement these two blocks, including the input sensors.

The Driving FSM block will require arithmetic modules that can calculate based on the input map, an angle to turn and a velocity to move at. It will require creating a state diagram that covers all the possible situations the car can be in. By navigating between the different states based on the input map and the current location, we can control the car very precisely. Ultimately the motors will be controlled by PWM waveforms so that we can control their speed. The two conversion blocks will convert a velocity output and an angle output to a duty cycle and create PWM waveforms with such duty cycles. Kevin Hsiue will implement these blocks.

One of the issues that need to be dealt with is tracking the relative location of the car with respect to the obstacles. We will modify our parking situation in order to simplify this by placing a wall 2 car length in front of the parking space so that our car can use IR distance sensors to detect its location. This may not work, and we may need to modify our control system in order to accommodate not knowing where the car is but rather how far it is away from a certain obstacle and in which directions there are obstacles.