|  |
| --- |
| HAROLD's Mechanics |
| Iteration One Report |
| Embedded Systems |
|  |
| **Jenny Reiman** |
| **Andrew Kirkham**  **Jason Laqua** |

**9/26/2013**

|  |
| --- |
|  |

Table of Contents

Table of Contents 2

Specifications 4

Design 5

Structural Design 5

Hello World Robot 5

Hello World Android 5

Robot and Android Not Damaged 5

Robot able to go Straight Forward, Straight Backward, Turn Right and Left 5

Obstacle Avoidance 5

Robot Handles 10% Grade 5

Communication between Android and Android 5

Communication between Android and VEX 6

Remote Emergency Stop Mechanism 6

Review 7

Structural Design 7

Hello World Robot 7

Hello World Android 7

Robot and Android Not Damaged 7

Robot able to go Straight Forward and Backward and turn Left and Right 7

Obstacle Avoidance 7

Communication between Android and Android 7

Communication between Android and VEX 7

Robot Handles 10% Grade 7

Remote Emergency Stop Mechanism 7

Test 8

Structural Design 8

Hello World Robot 8

Hello World Android 8

Robot and Android Not Damaged 8

Robot able to go Straight Forward and Backward and turn Left and Right 8

Obstacle Avoidance 8

Communication between Android and Android 8

Communication between Android and VEX 8

Robot Handles 10% Grade 8

Remote Emergency Stop Mechanism 8

Risk Mitigation 9

Sources 10

# Specifications

The specifications for this iteration of the project involved designing a robot structure, running programs and commands through the microcontroller, and establishing communication through Android devices. More specifically the robot had to be designed to move forward and backward in straight lines, as well as turn left and right. The robot also had to be able to handle climbing a 10% grade. Steps in programming for the VEXpro and Android based devices included hello world programs for all the platforms involved. For obstacle avoidance, bumper sensors needed to be mounted to the robot so it could perform the proper action when it collided with another object or barrier. Another feature that the robot must have is an emergency stop button, which can be used remotely from either an Android smartphone or tablet. Part of this also includes the requirement of Android to Android communication, and Android to VEX communication.

# Design

### Structural Design

The robot was built in a rectangular shape with large wide wheels so it would be able to gain better friction on the ground.

### Hello World Robot

This was implemented by establishing an ad-hoc network on VEXpro and connecting to it on a computer. Then the Terk IDE was opened and a simple hello world program was built and run through the VEXpro.

### Hello World Android

This was accomplished by first installing the correct drivers on both the phone and table. Then, a hello world program was written through the Android development SDK and ran on both of the devices.

### Robot and Android Not Damaged

The robot has been securely stored in a locked project locker in Engineering hall whenever it is not being worked on. All team members have additionally been careful when handling the Android devices.

### Robot able to go Straight Forward, Straight Backward, Turn Right and Left

A simple motor program was written and ran though the Terk IDE to test the movement capabilities of the robot. The program tests the motors running at full speed in each set of the directions for the desired movement. The robot performed well in moving forward and backwards, but struggled to turn in both directions due to unscrewing nuts in the structural design.

### Obstacle Avoidance

Bump sensors were added to the robot and plugged into the VEXpro ports for digital I/O. The program uses polling to constantly check if the bumper sensors have received any input. If they have, the robot will back up, turn, and then try to move forward again. If the robot was to bump into something while it is backing up from the initial bump it will head forward for half of the time that it was backing and try to turn at that new location.

### Robot Handles 10% Grade

The robot was built with 3 wire motors and was tested running at max speed to ensure it could be supplied enough power to climb the 10% grade.

### Communication between Android and Android

To establish Android to Android communication the tablet sends a UDP packet which the phone listens for. Messages are designed to popup on the phone and display the contents of the packet.

### Communication between Android and VEX

An Android device will put out a UDP packet and the server /client programs running on the VEX side will capture and interpret it.

Remote Emergency Stop Mechanism  
 The remote emergency stop feature was implemented by sending a stop command in the form of a string from an Android based device. Client and server programs running on the VEX side were made to capture and interpret the command.

# Review

### Structural Design

In the original design the robot was shaky from the nuts unscrewing all the time, which also inhibited the movement of the robot after a while. The nuts were replaced with lock nuts to solve this issue.

### Hello World Robot

The hello world robot program worked well after learning how to project the ad-hoc network from the VEXpro and connect to it in the IDE. A router was later used to accomplish the same goal.

### Hello World Android

The hello world Android programs on the tablet and phone were successful right away and did not require any modifications.

### Robot and Android Not Damaged

The project locker has worked well to keep the robot and all of our tools safe.

### Robot able to go Straight Forward and Backward and turn Left and Right

The programs that were made to drive the robot were later optimized through the use of a helper class that defines forward, backward, and turning functions. Options for speed settings were also defined.

### Obstacle Avoidance

The obstacle avoidance program was modified a few times. The first time was to have accurate timing of how long certain movements need to be performed for based off of the current system time. The second time involved adding definitions from the motor helper class to optimize the way the program was written.

### Communication between Android and Android

Since TCP/IP caused problems with application stability, UDP was used instead.

### Communication between Android and VEX

A client and server example program was modified for this purpose. After debugging the program, there were no further communication issues.

### Robot Handles 10% Grade

The robot did not have any problem climbing the 10% grade, so this issue was considered closed after testing.

Remote Emergency Stop Mechanism  
 The remote emergency stop button was further improved by building a GUI based application for the Android tablet and smartphone which allow the user to press a big stop button.

# Test

### Structural Design

This was tested by creating motor programs to drive the robot around and see how well it moved.

### Hello World Robot

This was tested by running the hello world example program in the Terk IDE while connected through the VEXpro and ensuring that string of “Hello World” was printed out in the console.

### Hello World Android

This was tested by running the hello world application and making sure that “Hello World” was printed out on both devices.

### Robot and Android Not Damaged

This was tested by looking over all of the equipment to make sure that it had not been damaged.

### Robot able to go Straight Forward and Backward and turn Left and Right

We tested this by observing the robot perform while running the motor test program.

### Obstacle Avoidance

The obstacle avoidance program was tested by having the robot drive into the wall and observing that it corrected its path as specified.

### Communication between Android and Android

This was tested by connecting both of the devices to the network and debugging through sending packets back and forth until it worked.

### Communication between Android and VEX

This was done by sending packets from the Android and ensuring that they were received through the VEX.

### Robot Handles 10% Grade

This was testing by having the robot drive up an angled table lifted to various grades that were at least 10% and greater.

Remote Emergency Stop Mechanism  
 This was tested by making sure that when the “stop” string was sent by the Android application it was received on the VEX side

# Risk Mitigation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requirement** | **Estimated Time To**  **Completion (hours)** | **Actual Hours to Completion** | **Points** | **Risk** | **Final Value (points/hour)** |
| Hello World Robot | 2 | 4 | 5 | .5 | 1.25 |
| Hello World Android | 1 | 1 | 5 | 1 | 5 |
| Robot/Android not damaged | N/A | N/A | 10 | 1 | N/A |
| Robot goes forward/backwards  Turns left/right | 5 | 2 | 10 | .5 | 1 |
| Obstacle Avoidance | 1 | 2 | 10 | .8 | 8 |
| Communication Android to Android | 10 | 10 | 10 | .1 | 0.1 |
| Communication Android to VEX | 8 | 3 | 10 | .2 | 0.25 |
| 10% Grade | 1 | 1 | 10 | .9 | 9 |
| Remote emergency stop | 3 | 2 | 15 | .8 | 4 |
| Totals: | 31 | 25 | 85 | N/A | N/A |

The low risk associated with the Hello World Android, 10% grade, and not damaged robot requirement were correctly calculated to not cause any problems or hold our schedule back. The hello world robot task did not include as much risk as figured into the equation and was easily completed by its due date. Turning the robot was given an appropriate risk since we ended up having to take time to rebuild the robot with lock nuts to make it more stable. Obstacle avoidance was an easy program to write and there were great tutorials on how to interface with the sensors. Even though this was different from our original interpretation of obstacle avoidance, there was no increased risk with the task. Communication from Android to Android was rated as risky and turned out to be rather complicated. Our original idea did not end up working through TCP/IP, and with the redesign it works, but still encounters several bugs. Communication from the Android to VEX was rated as a high risk task, but using a helpful client and served tutorial it ended up not posing any issues. The remote emergency stop in itself was not hard to implement, but it depended on Android communication which slowed its progress down.

For our total hours put into the project we came in under by six hours from the original estimation. The amount of time put into 10% grade, communication Android to Android, and hello world Android, was correctly estimated. The hello world robot took slightly longer than estimated; however moving the robot took less time than expected. The hours that were cut in our project time were from the communication Android to VEX task, which ended up being five hours.

# Sources

1. <http://www.abc.se/~m6695/udp.html> (client/server tutorial code)
2. <http://content.vexrobotics.com/docs/VEXnet_Cortex_UserGuide_081811.pdf>
3. [http://developer.android.com/reference/android/net/wifi/p2p/package-summary.html](http://developer.android.com/reference/android/net/wifi/p2p/package-summary.html" \t "_blank" \o )
4. <http://developer.android.com/training/connect-devices-wirelessly/wifi-direct.html>
5. [http://en.wikipedia.org/wiki/Wi-Fi\_Direct](http://en.wikipedia.org/wiki/Wi-Fi_Direct" \t "_blank" \o )
6. <http://www.education.rec.ri.cmu.edu/products/teaching_robotc_cortex/reference/VEXnet_setup_guide.pdf>
7. <http://www.vexforum.com/wiki/index.php/VEXpro_Programming_Resources>
8. <http://www.vexforum.com/wiki/Bumper_Switch>