# Process Journal for ImpulseV1

15/10/14

I have started the program for the second robot. Currently, it only contains a user control function to make the robot move, but more will definitely be added as time goes on. I have chosen the same file structure as it was in TimeV5 and TimeV6, but the code itself will be completely different and not based off said versions.

15/10/14

I have now completed the autonomous macros needed to move the robot automatically by simply calling a preset function which does a specific thing like move forwards or backwards, or turn left or right. I have used the same strategy in TimeV6, which is defining a method which moves the robot, and adding in #definitions which call that method. Take a look at the example below

#define drive(x, y) (move(x, abs(y), abs(y), abs(y), abs(y)))

#define driveBack(x, y) (move(x, -abs(y), -abs(y), -abs(y), -abs(y)))

#define turnLeft(x, y) (move(x, -abs(y), -abs(y), abs(y), abs(y))

#define turnRight(x, y) (move(x, abs(y), abs(y), -abs(y), -abs(y)))

16/10/14

There is now an arcade mode within the program, which means that you can switch driving control layouts with the push of a button. I also have programmed a cooldown timer, so that you have time to take your finger off the button before the button value is read again.

23/10/14

I have implemented the framework for the autonomous routine for the robot. Currently, the major functionality so far is a calibration method for calibrating the line sensors and finding the threshold between them for easy identification if the robot is currently not on the line.

28/10/14

I have programmed the basic autonomous routine (or so I hope) that will make the robot follow a line (hopefully). Currently, the code is not executed during the autonomous routine, but rather at the press of a button on the controller in usercontrol.c. The while loop is quite simple, here it is below for reference:

if (isValid())

{

drive(10, 127);

}

else

{

correctRobot();

}