IEEE Ottawa Robotics Competition
Compétition de robotique d'Ottawa d'IEEE

Library Documentation for the LRT Detour Challenge

Last Revised: 13 November 2016

Table of Contents

IRControl 2

UltrasonicControl 2

MotorControl 3

IRControl

This class represents the infra-red sensor that will be used to detect the line

```
class IRControl{
        /*
         Constructor to create "IR sensor" object
         @parm pin: the analogPin that the sensor uses
        */
        IRControl(int pin);

        /*
        Function that determines if the sensor is reading black
        @returns: true if black
        */
        boolean isBlack();
};
```

UltrasonicControl

This class represents the sensors used to detect the obstacle. It handles the specifics of triggering the sensor so that the user can simply get a distance back.

```
class UltrasonicControl{
        /*
                Constructor, creates sensor object
                @param trigPin: the trigger pin on the sensor
                @param echoPin: the echo pin on the sensor
        */
        UltrasonicControl(int trigPin,int echoPin);

        /*
                determines the distance
                @return: distance in cm, -1 on error
        */
        int getDistance();

        /*
                Determines if an object is detected within the next square
                @return: true of an object is detected
        */
        boolean detect();
};
```

MotorControl

This class represents the motors. It interfaces to the motor through the motor shield.

```
class MotorControl
{
	/*
		Constructor, takes the pins used and sets the pinmode
		@param pwmPin: the pin used to control the motor speed
					for motor one pwmPin = 5
					for motor two pwmPin = 6
		@param dirPin: the pin used to control the direction
					for motor one dirPin = 4
					for motor two dirPin = 7
	*/
	MotorControl(int pwmPin,int dirPin);

	/*
		Constructor, just takes what motor to use and sets up the pin
		One problem with using this is that if they don't realize the
		pins are already in use they might try to use them for something
		else
		@param motorNum: what motor the object will represent. Must be
					one or two
	*/
	MotorControl(int motorNum);

	/*
		moves the motor forward
		@param speed: how fast the motor should go
	*/
	void forward(int speed);

	/*
		moves the motor backwards
		@param speed: how fast the motor should go
	*/
	void reverse(int speed);

	/*
		Halts the motor
	*/
	void halt    ();
};
```