#include <FEHLCD.h>

#include <FEHIO.h>

#include <FEHUtility.h>

#include <FEHServo.h>

#include <FEHMotor.h>

#include <FEHRPS.h>

void initializeRPS();

void checkHeading(int heading);

void start();

void stopMotors();

//turn for a certain time and adjust for heading

void right(float time, float heading);

void left(float time, float heading);

//turn in reverse for a certain time and adjust for heading

void backRight(float time, float heading);

void backLeft(float time, float heading);

void straight(float percent, float time); //goes straight for amount of time if time is >0, otherwise just sets percent

//get to foosball station

void toFoosball();

//set up foosball arm

void setFoosballArm();

//remove arm

void removeFoosballArm();

//declare motors, servos, microswitches, and cdsCell

//front motor in motor port 0

FEHMotor rightMotor(FEHMotor::Motor0, 7.2);

//back motor in motor port 1

FEHMotor leftMotor(FEHMotor::Motor1, 7.2);

//arm servo in servo port 0

FEHServo armServo(FEHServo::Servo0);

//cds cell in pin P1\_7

AnalogInputPin cdsCell(FEHIO::P1\_7);

//sets servo for arm horizontal rotation

FEHServo rotServo(FEHServo::Servo4);

//sets microservo for arm

FEHServo microServo(FEHServo::Servo2);//Need to find min and max for microServo

//set bump switches for front and back of bot and on arm

DigitalInputPin rightBump(FEHIO::P0\_0);

DigitalInputPin leftBump(FEHIO::P0\_6);

DigitalInputPin armBump(FEHIO::P0\_7);

DigitalInputPin frontBump(FEHIO::P0\_4);

DigitalInputPin backLeftBump(FEHIO::P0\_7);

DigitalInputPin backRightBump(FEHIO::P0\_2);

//declare encoders

DigitalEncoder right\_encoder(FEHIO::P1\_2);

DigitalEncoder left\_encoder(FEHIO::P1\_0);

//Set inputs for line readers

AnalogInputPin rightOpto(FEHIO::P2\_0);

AnalogInputPin midOpto(FEHIO::P2\_1);

AnalogInputPin leftOpto(FEHIO::P2\_3);

#define LINE\_ON\_RIGHT 0

#define ON\_LINE 1

#define LINE\_ON\_LEFT 2

int main(void)

{

LCD.Clear();

//set mins and maxes for all the servos

armServo.SetMin(1200);

armServo.SetMax(2200);

rotServo.SetMin(700);

rotServo.SetMax(2350);

microServo.SetMin(1470);

microServo.SetMax(2270);

armServo.SetDegree(10);

microServo.SetDegree(50);

rotServo.SetDegree(124);

initializeRPS();

//check to see if light is on and start moving once detected

start();

toFoosball();

}

//function to stop motors

void stopMotors(){

rightMotor.Stop();

leftMotor.Stop();

}

void straight(float percent, float time){

rightMotor.SetPercent(percent);

leftMotor.SetPercent(-percent);

if(time>0){

Sleep(time);

stopMotors();

}

}

void backRight(float time, float heading)

{

int motorPower=40;

leftMotor.SetPercent(motorPower);

rightMotor.SetPercent(0);

Sleep(time);

if(heading>=0){

checkHeading(heading);

}

}

void backLeft(float time, float heading)

{

int motorPower=40;

leftMotor.SetPercent(0);

rightMotor.SetPercent(-motorPower);

Sleep(time);

if(heading>=0){

checkHeading(heading);

}

}

void right(float time, float heading){

int motorPower=40;

leftMotor.SetPercent(-motorPower);

rightMotor.SetPercent(-motorPower);

Sleep(time);

if(heading>=0){

checkHeading(heading);

}

}

void left(float time, float heading){

int motorPower=40;

leftMotor.SetPercent(motorPower);

rightMotor.SetPercent(motorPower);

Sleep(time);

if(heading>=0){

checkHeading(heading);

}

}

//function to move to foosball

void toFoosball()

{

float movementSpeed = 40;

float heading;

float time;

float ninetyDegreeTime=0.9;

//go forward a little

time=0.9;

straight(movementSpeed, time);

//turn right till robot faces right wall

time=0.5;

heading=0;

right(time, heading);

//go straight until left and right bumpers are pressed

straight(movementSpeed, -1);

while(leftBump.Value() || rightBump.Value());

stopMotors();

//move back and then turn left

heading=90;

time=0.6;

straight(-movementSpeed, time);

left(ninetyDegreeTime, heading);

//go straight until any bumper is pressed

straight(movementSpeed, -1);

int i=1;

while(i==1)

{

if(!leftBump.Value() || !rightBump.Value() || !frontBump.Value())

{

i=0;

}

}

stopMotors();

//move back while turning right

heading=-1;

backRight(ninetyDegreeTime, heading);

//go straight back until left and right bumpers are pressed

straight(-movementSpeed, -1);

while(leftBump.Value() && rightBump.Value());

stopMotors();

//set time to move forward to foosball station

time=0.3;

straight(movementSpeed, time);

setFoosballArm();

}

void setFoosballArm()

{

//variables for final angles

float finalArmAngle=180;

float finalMicro=135;

//set arm servo to be vertical

float armAngle=90;

armServo.SetDegree(armAngle);

//set rot servo to be to the right of the robot

float rotAngle = 180;

rotServo.SetDegree(rotAngle);

//set micro servo to 45,

float microAngle=45;

microServo.SetDegree(microAngle);

//lower arm servo slowly

while(armAngle<finalArmAngle)

{

armAngle++;

armServo.SetDegree(armAngle);

//lower slowly by sleeping inbetween

Sleep(20);

}

//clamp on foosball using microservo

while(microAngle<finalMicro)

{

microAngle++;

microServo.SetDegree(microAngle);

Sleep(20);

}

}

void removeFoosballArm()

{

//variables for final angles

float finalArmAngle=90;

armServo.SetDegree(finalArmAngle);

}

//function to navigate to and press final button

void foosballToFinal()

{

float movementSpeed = 20, time, heading;

//move until robot bumps into lever station

straight(movementSpeed, -1);

while(rightBump.Value());

stopMotors();

//turn for a second to a heading of 225

time=1.0;

heading=225;

backRight(time, heading);

//move until robot bumps into wall, then turn and go down ramp to final button

straight(movementSpeed, -1);

while(rightBump.Value());

stopMotors();

heading=270;

straight(movementSpeed, -1);

while(rightBump.Value() && leftBump.Value() && frontBump.Value());

stopMotors();

}

//function to check heading

void checkHeading(int heading) //using RPS

{

float startTime = TimeNow();

if(heading!=0){

while((RPS.Heading()<(heading-3) || RPS.Heading()>(heading+3)) && TimeNow() - startTime < 6.0){

if(RPS.Heading() < heading){

leftMotor.SetPercent(20);

leftMotor.SetPercent(20);

Sleep(20);

stopMotors();

Sleep(200);

}

else if(RPS.Heading() > heading){

rightMotor.SetPercent(-20);

leftMotor.SetPercent(-20);

Sleep(20);

stopMotors();

Sleep(200);

}

}

}

else if(heading==0){

while((RPS.Heading()<357 || RPS.Heading()>3) && TimeNow() - startTime < 6.0)

{

if(RPS.Heading() < 360 && RPS.Heading()>180){

rightMotor.SetPercent(20);

leftMotor.SetPercent(20);

Sleep(20);

stopMotors();

Sleep(200);

}

else if(RPS.Heading() > heading && RPS.Heading() <180){

rightMotor.SetPercent(-20);

leftMotor.SetPercent(-20);

Sleep(20);

stopMotors();

Sleep(200);

}

}

}

stopMotors();

}

void initializeRPS(){

//Initialize RPS and declare any variables you might need

float touch\_x, touch\_y;

//Call this function to initialize the RPS to a course

RPS.InitializeTouchMenu();

LCD.Clear();

}

//function to detect start light

void start()

{

//declare maximum value for light being off

float thresholdValue=1.0;

//keep checking to see if start light is on

bool check=true;

while(check)

{

//check to see if light is turned on

if(cdsCell.Value()<thresholdValue)

{

check=false;

}

}

}

NOTES:

48 counts in one rotation

60 counts to travel 10 inches