#include <iostream>

#include <memory>

#include <string>

#include "WPILib.h"

#include <CANTalon.h>

#include <IterativeRobot.h>

#include <LiveWindow/LiveWindow.h>

#include <SmartDashboard/SendableChooser.h>

#include <SmartDashboard/SmartDashboard.h>

class Robot: public frc::IterativeRobot {

public:

CANTalon \*l1, \*l2,\*r1, \*linAct;

Joystick \*joy;

PowerDistributionPanel \*pdp;

cs::UsbCamera camera;

Compressor \*compressor;

DoubleSolenoid \*TPSA ,\*TPSB ,\*MPS;

DigitalInput \*limit1, \*limit2;

Servo \*servo1, \*servo2;

bool pressed = false;

int i = 0;

Robot()

{

compressor = new Compressor(0);

linAct = new CANTalon(53);

l1 = new CANTalon(52);

l2 = new CANTalon(51);

r1 = new CANTalon(50);

joy = new Joystick(1);

pdp = new PowerDistributionPanel();

TPSB = new DoubleSolenoid(4,5);

MPS = new DoubleSolenoid(2,3);

TPSA = new DoubleSolenoid(0,1);

limit1 = new DigitalInput(0);

limit2 = new DigitalInput(1);

servo1 = new Servo(1);

servo2 = new Servo(0);

}

void RobotInit() override

{

compressor->SetClosedLoopControl(true);

cs::UsbCamera camera = CameraServer::GetInstance()->StartAutomaticCapture();

}

void AutonomousInit() override

{

printf("0");

for(int i = 0; i > 5; i++)

{

printf("1 + %d ,test %d", limit2->Get(), limit1->Get());

while(limit2->Get() == false)

{

linAct->Set(1);

}

while(limit1->Get() == false)

{

linAct->Set(-1);

}

}

}

void AutonomousPeriodic() {

}

void TeleopInit() {

}

void TeleopPeriodic() {

TankDrive(-joy->GetRawAxis(5), -joy->GetRawAxis(1));

printf("%d\n", limit1->Get());

camera.SetResolution(640, 480);

servo1->Set(joy->GetRawAxis(2));

servo2->Set(joy->GetRawAxis(3));

if(joy->GetRawButton(7))

{

printf("0");

while(pressed == false)

{

printf("1 + %d ,test %d", limit2->Get(), limit1->Get());

while(limit2->Get() == false)

{

linAct->Set(1);

}

while(limit1->Get() == false)

{

linAct->Set(-1);

}

if(joy->GetRawButton(8))

{

pressed = true;

}

i++;

printf("%i",i);

if(i == 5)

{

MPS->Set(DoubleSolenoid::kForward);

}

else if(i == 10)

{

MPS->Set(DoubleSolenoid::kForward);

i = 0;

}

if(i == 2 || i == 5)

{

TPSA->Set(DoubleSolenoid::kForward);

}

else if(i == 4 || i == 9)

{

TPSA->Set(DoubleSolenoid::kReverse);

}

else if(i == 3 || i == 6)

{

TPSB->Set(DoubleSolenoid::kForward);

}

else if(i == 5 || i == 8)

{

TPSB->Set(DoubleSolenoid::kReverse);

}

}

if(joy->GetRawButton(7))

{

pressed = false;

}

}

if(joy->GetRawButton(1))

{

TPSA->Set(DoubleSolenoid::kForward);

}

else if(joy->GetRawButton(4))

{

TPSA->Set(DoubleSolenoid::kReverse);

}

if(joy->GetRawButton(3))

{

TPSB->Set(DoubleSolenoid::kForward);

}

else if(joy->GetRawButton(2))

{

TPSB->Set(DoubleSolenoid::kReverse);

}

if (joy->GetRawAxis(2) > .25)

{

MPS->Set(DoubleSolenoid::kReverse);

}

else if (joy->GetRawAxis(3) > .25)

{

MPS->Set(DoubleSolenoid::kForward);

}

else

{

MPS->Set(DoubleSolenoid::kOff);

}

if(joy->GetRawButton(5) && limit2->Get() == false)

{

linAct->Set(1);

}

else if(joy->GetRawButton(6) && limit1->Get() == false)

{

linAct->Set(-1);

}

else

{

linAct->Set(0);

}

}

void TankDrive(double left, double right)

{

//Assigns left side of the drive train to the y axis of the left joy stick of the X-box controller

l1->Set(-left);

l2->Set(-left);

//Assigns right side of the drive train to the y axis of the right joy stick of the X-box controller

r1->Set(right);

}

};

START\_ROBOT\_CLASS(Robot)