















// CSCI360 (Robotics) Line Following Test (Experimental)

//

// Center color sensor over a short piece of white tape placed on the

// flooring upon which the tape line resides. Press the green Textrix

// Start Button once to calibrate the color sensor.

//

// Now move the Tetrix to the tape line starting point, centering the

// color sensor over the tape and the chasis aligned with the tape direction.

// Press the Start Button again to begin the line following.

#include <PRIZM.h> // Tetrix PRIZM and EXPANSION controller library

#include <CSCIUtils.h> // CSCI Library routines

#include <C:\Users\chris\OneDrive\Documents\Arduino\CSCI360\_Tetrix\_Apps\CSCI\_DTrain\_Params.h> // Drive train-specific params

PRIZM Prizm; // Instantiate Tetrix controller object.

EXPANSION Exc; // Instantiate Tetrix expansion controller object.

csci::ColorSensor CSensor; // Instantiate ColorSensor object.

csci::SerialMonitor SMonitor(38400); // Instantiate SerialMonitor object.

// Color of tape line to follow.

// Color will be automatically detected during setup.

csci::TapeColor LineColor = csci::TapeColor::unknown;

// Instantiate Tetrix Mecanum Smart Car object.

csci::TMSmartCar TMSCar(Prizm, Exc, CSensor,

csci::FBMultiplier,

csci::LRMultiplier,

csci::DiagMultiplier,

csci::SpinMultiplier);

// This routine called once at program start.

int count = 0;

int red = 0;

void setup()

{

SMonitor.setup(); // Setup serial monitor.

// Setup the Tetrix Smart Car

if ( !TMSCar.setupCar() )

{

SMonitor.sendText("Tetrix SmartCar setup failed!");

SMonitor.sendNewline();

while ( true ) { }; // Hang here. Don't proceed.

}

SMonitor.sendText("Tetrix Battery Voltage = ");

SMonitor.sendDoubleValue(TMSCar.getBatteryVoltage());

SMonitor.sendNewline();

// Calibrate against white tape.

double cScale, rScale, gScale, bScale;

CSensor.calibrateWhiteBalance(cScale, rScale, gScale, bScale);

CSensor.setWhiteBalance(cScale, rScale, gScale, bScale);

csci::PrizmStartButton startButton(Prizm);

// Wait for Start button click to continue program.

// During this time Tetrix is placed so color sensor

// is over the tape line to follow.

startButton.waitForClick();

// Read color of tape line to follow.

LineColor = CSensor.getTapeColor();

Prizm.setServoSpeed(1,25);

Prizm.setServoPosition(1,45);

}

// Tape line width (in inches).

const double LineWidth = 0.75;

// Tetrix speed fraction.

double SpeedFraction = 0.27;

// This routine called repeatedly until a "reset" is performed.

void loop ()

{

TMSCar.setSpeedFraction(SpeedFraction);

// Calc time to rotate one tape width at current speed.

uint32\_t travelTime =

TMSCar.getInchesTravelTime(csci::MoveState::msRotateCW, LineWidth);

// Arbitrary initial line acquisition rotation direction.

csci::MoveState rotDir = csci::MoveState::msRotateCW;

while ( true )

{

// Start moving forward.

TMSCar.move(csci::MoveState::msForward);

// While moving one line width...

csci::TimerMillis moveTimer(travelTime);

while ( !moveTimer.done() )

{

// Are we close to an obstacle?

do

{

int position = Prizm.readServoPosition(1);

if (position >= 125 )

{

Prizm.setServoPosition(1,45);

}

else if (position <= 55)

{

Prizm.setServoPosition(1,135);

}

double rangeDistance = TMSCar.getRangeSensorDistanceCM();

if ( (rangeDistance > 0.0) && (rangeDistance < 25.0) && (LineColor == csci::TapeColor::red))

{

// Yes, stop and wait for obstacle to be removed.

TMSCar.move(csci::MoveState::msRight);

delay(2750);

TMSCar.move(csci::MoveState::msForward);

delay(6500);

TMSCar.move(csci::MoveState::msRotateCCW);

delay(650);

while(TMSCar.getTapeColor() != LineColor)

{

TMSCar.move(csci::MoveState::msLeft);

}

}

else if ( (rangeDistance > 0.0) && (rangeDistance < 25.0) && (LineColor == csci::TapeColor::blue))

{

// Yes, stop and wait for obstacle to be removed.

TMSCar.move(csci::MoveState::msRight);

delay(2750);

TMSCar.move(csci::MoveState::msForward);

delay(6500);

TMSCar.move(csci::MoveState::msRotateCW);

delay(650);

while(TMSCar.getTapeColor() != LineColor)

{

TMSCar.move(csci::MoveState::msLeft);

}

}

else

{

break; // No, continue on.

}

} while ( true );

}

// Car has moved one line width. If car is not over the tape...

if (TMSCar.getTapeColor() == csci::TapeColor::blue && LineColor == csci::TapeColor::red)

{

TMSCar.move(csci::MoveState::msForward);

delay(300);

if (TMSCar.getTapeColor() == csci::TapeColor::blue)

{

TMSCar.move(csci::MoveState::msRotateCW);

delay(4700);

TMSCar.move(csci::MoveState::msForward);

delay(500);

TMSCar.move(csci::MoveState::msLeft);

delay(950);

LineColor = csci::TapeColor::blue;

}

}

if (LineColor == csci::TapeColor::blue && TMSCar.getTapeColor() == csci::TapeColor::red)

{

if (red < 1)

{

TMSCar.move(csci::MoveState::msForward);

delay(1200);

red++;

}

else

{

SpeedFraction = 0.50;

TMSCar.setSpeedFraction(SpeedFraction);

TMSCar.move(csci::MoveState::msRotateCW);

delay(6000);

TMSCar.stop();

exit(0);

}

}

if ( TMSCar.getTapeColor() != LineColor )

{

// Attempt to re-acquire tape line with short search arc.

if ( !AquireTapeLine(TMSCar, LineColor, 3 \* travelTime, rotDir) )

{

// Attempt to re-acquire tape line with longer search arc.

if ( !AquireTapeLine(TMSCar, LineColor, 21 \* travelTime, rotDir) )

{

// \*\*\* Currently stop car and wait for human assistance. \*\*\*

}

}

// Successfully relocated the tape line.

// We're at the edge of the tape, still moving.

// Travel 1/5 tape width further so definitely over tape.

csci::WaitMillis(travelTime / 5);

}

}

}

// Routine to aquire the tape line.

// Assumes the car is off the tape line, but we don't necessarily

// know which side of the tape the sensor is over. "rotDir" is

// the initial rotation direction (msRotateCW or msRotateCCW) to sweep.

// "sweepTime" is the time (in milliseconds) to sweep looking for

// the tape line.

//

// Returns "true" if we successfully aquired the tape.

// Returns "false" if unsucessful.

//

// Note: In either case, car is still moving.

//

// Upon return, "rotDir" will contain the last rotation direction.

// (This is a good candidate for the initial direction the next

// time the routine is called.)

bool AquireTapeLine(csci::TMSmartCar& car, csci::TapeColor lineColor,

uint32\_t sweepTime, csci::MoveState& rotDir)

{

if (TMSCar.getTapeColor() == csci::TapeColor::blue && LineColor == csci::TapeColor::red)

{

TMSCar.move(csci::MoveState::msForward);

delay(300);

if (TMSCar.getTapeColor() == csci::TapeColor::blue)

{

TMSCar.move(csci::MoveState::msRotateCW);

delay(4700);

TMSCar.move(csci::MoveState::msForward);

delay(500);

TMSCar.move(csci::MoveState::msLeft);

delay(950);

LineColor = csci::TapeColor::blue;

}

}

if (LineColor == csci::TapeColor::blue && TMSCar.getTapeColor() == csci::TapeColor::red)

{

if (red < 1)

{

TMSCar.move(csci::MoveState::msForward);

delay(1200);

red++;

}

else

{

SpeedFraction = 0.50;

TMSCar.setSpeedFraction(SpeedFraction);

TMSCar.move(csci::MoveState::msRotateCW);

delay(6000);

TMSCar.stop();

exit(0);

}

}

//

//Rotate initial direction looking for the tape...

if (LineColor != csci::TapeColor::blue)

{

TMSCar.move(csci::MoveState::msForward);

delay(60);

}

else if (LineColor == csci::TapeColor::blue)

{

TMSCar.move(csci::MoveState::msForward);

delay(80);

}

if ( !LookForTape(car, lineColor, rotDir, 1 \* sweepTime) )

{

rotDir = ReverseRotation(rotDir);

// Rotate opposite direction looking for the tape...

//int position = Prizm.readServoPosition(1);

/\*\

if (position == 155)

{

Prizm.setServoPosition(1,25);

}

else if (position == 25)

{

Prizm.setServoPosition(1,155);

}

\*/

if ( !LookForTape(car, lineColor, rotDir, 2 \* sweepTime) )

{

rotDir = ReverseRotation(rotDir);

/\*

if (position == 155)

{

Prizm.setServoPosition(1,25);

}

else if (position == 25)

{

Prizm.setServoPosition(1,155);

}

\*/

// Rotate back to original position while looking for tape.

if ( !LookForTape(car, lineColor, rotDir, sweepTime) )

{

return false; // Did not find the tape.

}

}

}

return true;

}

// Look for tape by rotating for a specified amount of time.

// "rotDir" must be one of tmRotateCW or tmRotateCCW.

// "travelTime" is in milliseconds.

//

// Returns "true" if tape detected.

// Returns "false" if not found.

//

// Note: In either case, car is still moving.

bool LookForTape(csci::TMSmartCar& car, csci::TapeColor lineColor,

csci::MoveState rotDir, uint32\_t travelTime)

{

if (TMSCar.getTapeColor() == csci::TapeColor::blue && LineColor == csci::TapeColor::red)

{

TMSCar.move(csci::MoveState::msForward);

delay(300);

if (TMSCar.getTapeColor() == csci::TapeColor::blue)

{

TMSCar.move(csci::MoveState::msRotateCW);

delay(4700);

TMSCar.move(csci::MoveState::msForward);

delay(500);

TMSCar.move(csci::MoveState::msLeft);

delay(950);

LineColor = csci::TapeColor::blue;

}

}

if (LineColor == csci::TapeColor::blue && TMSCar.getTapeColor() == csci::TapeColor::red)

{

if (red < 1)

{

TMSCar.move(csci::MoveState::msForward);

delay(1200);

red++;

}

else

{

SpeedFraction = 0.50;

TMSCar.setSpeedFraction(SpeedFraction);

TMSCar.move(csci::MoveState::msRotateCW);

delay(6000);

TMSCar.stop();

exit(0);

}

}

// Start car moving in the specified direction.

// Will either time out or locate the line.

car.move(rotDir);

// Start timer running for max travel time.

csci::TimerMillis maxTimer(travelTime);

// Continue to rotate as long as sensor is not over tape

// AND timer has not expired.

while ( ( car.getTapeColor() != lineColor) && !maxTimer.done() )

{ }

// If timer expired without finding the tape first...

if ( !maxTimer.isActive() )

{

return false;

}

// Indicate tape was found.

return true;

}

// Pass in a rotation direction (msRotateCW or msRotateCCW).

// Returns the opposite rotation direction.

csci::MoveState ReverseRotation(csci::MoveState rotDir)

{

int position = Prizm.readServoPosition(1);

if (position >= 125)

{

Prizm.setServoPosition(1,45);

}

else if (position <= 55)

{

Prizm.setServoPosition(1,135);

}

double rangeDistance = TMSCar.getRangeSensorDistanceCM();

if ( (rangeDistance > 0.0) && (rangeDistance < 20.0) && (LineColor == csci::TapeColor::red))

{

// Yes, stop and wait for obstacle to be removed.

TMSCar.move(csci::MoveState::msRight);

delay(2750);

TMSCar.move(csci::MoveState::msForward);

delay(5000);

TMSCar.move(csci::MoveState::msRotateCCW);

delay(650);

while(TMSCar.getTapeColor() != LineColor)

{

TMSCar.move(csci::MoveState::msLeft);

}

}

else if ( (rangeDistance > 0.0) && (rangeDistance < 12.0) && (LineColor == csci::TapeColor::blue))

{

// Yes, stop and wait for obstacle to be removed.

TMSCar.move(csci::MoveState::msRight);

delay(2750);

TMSCar.move(csci::MoveState::msForward);

delay(3200);

TMSCar.move(csci::MoveState::msRotateCW);

delay(750);

while(TMSCar.getTapeColor() != LineColor)

{

TMSCar.move(csci::MoveState::msLeft);

}

}

return ( rotDir == csci::MoveState::msRotateCW ) ?

csci::MoveState::msRotateCCW : csci::MoveState::msRotateCW;

}