**Project Description**

Let the robot find and knock over all columns in a predefined area.

**How we will achieve it**

We will define the area by making sure the surface is white and it will be surrounded by a black surface. Once the robot leaves the white area and reads black it will turn around to stay in the area it's supposed to be in. The robot finds the columns by turning in a circle until it reads shorter distance to the next object and goes towards it until it's close enough at which point it will turn a bit, lower it's arm and turn again to knock the column over with its claw, than raise the claw again and turn to find the next column. The robot can be stopped by pressing a button on the remote controller or if it crashes into an obstacle with an attached button.

**What we need**

Robot with a claw.

Remote controller : To run the robot and have an emergency stop button.

3 motors, port 2, 3 & 7: 2 motors To get the wheels spinning, 1 to get the claw arm to go up and down.

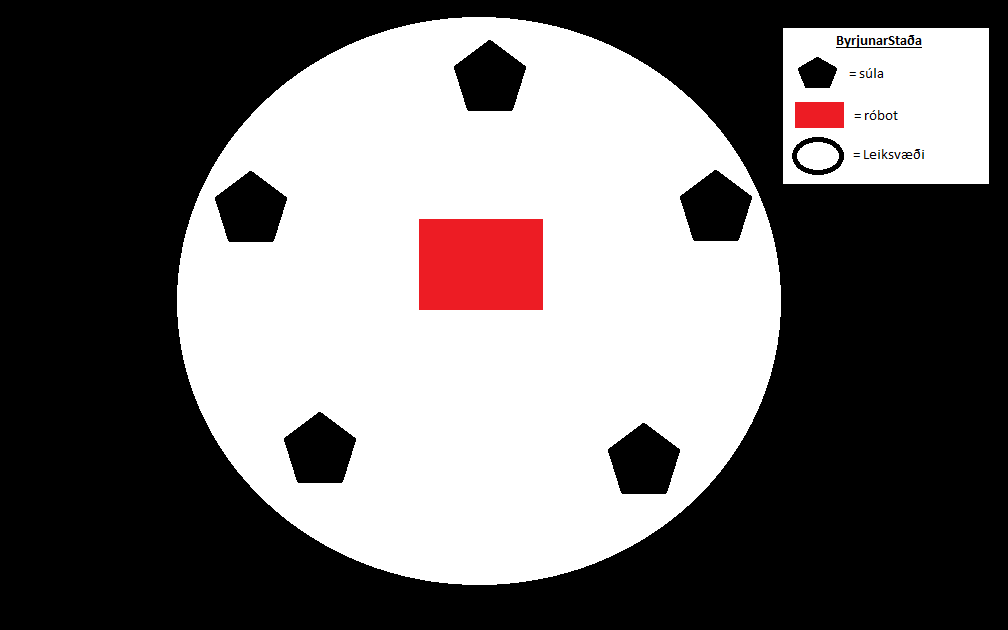
Line Follower, analog sensors 2: To read if the robot is within the designated area (reads whether it's on black or white surface).

Sonarsensor, Digital sensor port 11: To determine if the robot sees an obstacle and how far it is from it (In this case the columns).

Touch button, digital sensor port 19: If the button is hit the robot will stop.

Potentmeter, digital port 4: Determine whether the arm is up or down.

**Sketch**



**Final Project**

**The code**

|  |  |
| --- | --- |
|  | #pragma config(Sensor, in1, lineFollowerRIGHT, sensorLineFollower) |
|  | #pragma config(Sensor, in2, lineFollowerCENTER, sensorLineFollower) |
|  | #pragma config(Sensor, in3, lineFollowerLEFT, sensorLineFollower) |
|  | #pragma config(Sensor, in8, mypot, sensorPotentiometer) |
|  | #pragma config(Sensor, dgtl5, leftEncoder, sensorQuadEncoder) |
|  | #pragma config(Sensor, dgtl7, rightEncoder, sensorQuadEncoder) |
|  | #pragma config(Sensor, dgtl10, touchSensor, sensorTouch) |
|  | #pragma config(Sensor, dgtl11, sonarSensor, sensorSONAR\_cm) |
|  | #pragma config(Motor, port2, leftMotor, tmotorServoContinuousRotation, openLoop) |
|  | #pragma config(Motor, port3, rightMotor, tmotorServoContinuousRotation, openLoop, reversed) |
|  | #pragma config(Motor, port7, armMotor, tmotorVex269, openLoop, reversed) |
|  | //\*!!Code automatically generated by 'ROBOTC' configuration wizard !!\*// |
|  |  |
|  | /\*---------------------------------------------------------------------------------------------------\*\ |
|  | |\* This program makes the robot turn around itself until it the sonar sensor pickups up a signal \*| |
|  | |\* of an object. It drives toward the object tries to knock it down. \*| |
|  | |\* After knocking down the object it tries to find other objects that are near the robot. \*| |
|  | |\* If the robot drives over the border it turns around back to the arena. \*| |
|  | |\* MOTORS & SENSORS: \*| |
|  | |\* [I/O Port] [Name] [Type] [Description] \*| |
|  | |\* Motor - Port 2 leftMotor VEX 3-wire module Right side motor \*| |
|  | |\* Motor - Port 3 rightMotor VEX 3-wire module Left side motor \*| |
|  | |\* Motor - Port 7 armMotor VEX 3-wire module Arm motor \*|6 |
|  | |\* Analog - Port 1 lineFollowerRIGHT VEX Light Sensor Front-right, facing down \*| |
|  | |\* Analog - Port 2 lineFollowerCENTER VEX Light Sensor Front-center, facing down \*| |
|  | |\* Analog - Port 3 lineFollowerLEFT VEX Light Sensor Front-left, facing down \*| |
|  | |\* Analog - Port 4 potentioSensor VEX Quadrature enc. Potentiometer on the side \*| |
|  | |\* Digital - Port 5 sonarSensor VEX Sonar Sensor Front mounted, front facing \*| |
|  | \\*----------------------------------------------------------------------------------------------4246-\*/ |
|  |  |
|  | // Create all the variables used within the code |
|  | int highpower = 35; |
|  | int lowpower = 24; |
|  | int nopower = 0; |
|  | int distance = 45; |
|  | int knockDistance = 15; |
|  | int counter = 0; |
|  | int threshold = 2250; |
|  | float someDegree = 113.097/31.41; // How far the wheels have to go for 1 degree |
|  | float oneDegree = someDegree - 0.5; // fix the distance for one degree |
|  | float degree; |
|  | // Function that turns around until it finds an object |
|  | void find() |
|  | { |
|  | while(SensorValue[sonarSensor] > distance) |
|  | { |
|  | motor[leftMotor] = lowpower; |
|  | motor[rightMotor] = -lowpower; |
|  | } |
|  | } |
|  | void stopMotors() |
|  | { |
|  | motor[leftMotor] = nopower; |
|  | motor[rightMotor] = nopower; |
|  | } |
|  | // function that goes forward towards the object until it's in range to knock it down or it goes over black surface (out of play area) |
|  | void reach() |
|  | { |
|  | while(SensorValue[sonarSensor] > knockDistance && SensorValue[lineFollowerCENTER] < threshold) |
|  | { |
|  | stopMotors(); |
|  | motor[leftMotor] = highpower; |
|  | motor[rightMotor] = highpower; |
|  | } |
|  | // function that turns to the right by the amount of degrees spesified when the function is called |
|  | } |
|  | void turnright(int degrees) //function for turning right |
|  | { |
|  | degree = oneDegree \* degrees; // calcualting how many degrees it needs to turn |
|  | while(abs(SensorValue[rightEncoder]) < degree) // While the right encoder is less than distance: |
|  | { |
|  | // spins right around itself |
|  | motor[rightMotor] = -highpower; |
|  | motor[leftMotor] = highpower; |
|  | } |
|  | } |
|  | //Function that knocks down the object in front of it |
|  | void knock() |
|  | { |
|  | // turns for a bit |
|  | motor[leftMotor] = -lowpower; |
|  | motor[rightMotor] = lowpower; |
|  | wait1Msec(500); |
|  | stopMotors(); |
|  | // Lowers arm |
|  | while (SensorValue[mypot] > 1) { |
|  | motor[armMotor] = -highpower; |
|  | } |
|  | motor[armMotor] = nopower; |
|  | // Turns again to knock the object with the arm |
|  | motor[leftMotor] = highpower; |
|  | motor[rightMotor] = -highpower; |
|  | wait1Msec(1500); |
|  | stopMotors(); |
|  | //Raises the arm again |
|  | while (SensorValue[mypot] < 700) { |
|  | motor[armMotor] = highpower; |
|  | } |
|  | motor[armMotor] = nopower; |
|  | } |
|  | //Create a task that runs alongside the main program and checks whether a button is pressed, if the button is pressed it ends all programs (killswitch) |
|  | task e\_stop() |
|  | { |
|  | while(true) { |
|  | if(SensorValue(touchSensor) == 1) { |
|  | StopAllTasks(); |
|  | } |
|  | wait1Msec(10); |
|  | } |
|  | } |
|  | task main() |
|  | { |
|  | //Let the robot wait for 20 seconds before it starts |
|  | wait1Msec(20000); |
|  | StartTask(e\_stop); |
|  | //Counts 3 objects that it has to knock over before it's mission is done |
|  | while(counter < 3) |
|  | { |
|  | find(); |
|  | //A little fix to have the robot position correctly after finding the target |
|  | wait1Msec(150); |
|  | stopMotors(); |
|  | reach(); |
|  | //While in reach 2 things can happen |
|  | //1. It leaves the play area (leaves white surface) and has to turn around and drive back into the play area |
|  | if(SensorValue[lineFollowerCENTER] > threshold) { |
|  | motor[leftMotor] = nopower; |
|  | motor[rightMotor] = nopower; |
|  | SensorValue[rightEncoder] = 0; |
|  | turnright(120); |
|  | motor[leftMotor] = highpower; |
|  | motor[rightMotor] = highpower; |
|  | wait1Msec(500); |
|  | } |
|  | //2. It reaches close enough to the object to knock it over |
|  | else if(SensorValue[sonarSensor] < knockDistance){ |
|  | stopMotors(); |
|  | knock(); |
|  | //If it manages to knock down an object it counts it |
|  | counter++; |
|  | } |
|  | } |
|  | } |

**Video**

<https://youtu.be/JA89LQrYkWI>

|  |
| --- |
| Pushed the knocked columns out of the way and turned the robot around to speed its finding of 3rd column since that part is only supposed to demonstrate that it turns around when it leaves the play area. |

**Diaries**

**Sveinn**:

**22/11/2017**

Made changes to the robot. Removed the limitswitch and the batterybox. Things that I added or moved are battery straps and moved the sonar sensor to the middle of the robot.

**27/11/2017**

We setup the program file and the flowchart for the assignment. Put header and connected all the components we'll be using.

**29/11/2017**

We relocated the sonar sensor because it read the line follower that was under it. Started coding the program today.

**4/12/2017**

We made some functions for the program, find(), stopMotors(), reach(), turnRight(int degree) and knock().

**6/12/2017**

Changed the code so we use only two tasks, the main task and the stopper task.

**Vytautas:**

**11/22/2017**

Created the project description and added everything we need to solve the project.

**11/27/2017**

Draw the picture of the layout of the project and created the flowchart.

**11/29/2017**

Started coding the program today.

**4/12/2017**

We made some functions for the program, find(), stopMotors(), reach(), turnRight(int degree) and knock().

**6/12/2017**

We finished the final code for the project, Just minor tweaking left

**8/12/2017**

Tweaked the code so it works, Took a video of the robot doing what it's supposed to.

**Final Thoughts**

The Project went exceptionally well, we worked evenly throughout the time period and the final result was very satisfying. Both of us worked well together throwing ideas back and forth so that the final solution would work and both of us did even work on the project.