ZHANNA KLIMANOVA AND SANSITHA PANCHADSARAM

GROUP 33

LAB 1: WALL FOLLOWING TWO-WHEELED ROBOT

ECSE 211 DESIGN PRINCIPLES AND METHODS

FALL 2020

1. Design Evaluation

To create the hardware design for the two-wheeled robot, the LeoCAD robot model provided for Lab0 in MyCourses was used and certain parts were modified to work in Lab1. The color sensor was taken down from the model as it was not required in Lab 1 and the ultrasonic sensor was positioned at an angle of 45 degrees to get a better view of the front and the side. To mount the ultrasonic sensor at an angle of 45 degrees on the robot, one Technic Beam 7 x 0.5, 2 Technic Cross Blocks 1 x 3 (Pin/Pin/Pin) with 4 Pins and one Technic Beam 3 were used as shown in Fig. 1.

The software execution begins in main method of the Lab1 class. As shown in Fig. 2, the performPhysicsStep() method runs for one second synchronizing all threads. When the simulation is ready, the left and right motors start moving. The ultrasonic sensor sends out signals into the surrounding environment to test how far the robot is located from the wall. The filter() method filters the sensor data of invalid signals and the readUsDistance() method outputs the filtered distance in centimeters. Then, the controller determines the speed of the robot's motor in relation to the actual distance from the wall and sets the initial left motor and right motor speeds. At the same time, another performPhysicsStep() is continuously checking whether Webots has indicated for the program to stop. In that case, both motors stop, and the system exits.

Every thread execution, the controller uses the current robot distance (from the wall) to compute the distance error. If the robot is too close to the wall (the distance error is greater than 0), the right motor speed is decreased to make the robot move further away. If the robot is too far from the fall, the distance error is less than 0), the right motor speed is increased to move the robot closer to the wall. Otherwise the left motor speed and right speeds of the motors are set to a constant. The class diagram of the EV3 robot and its components is shown in Fig. 3.

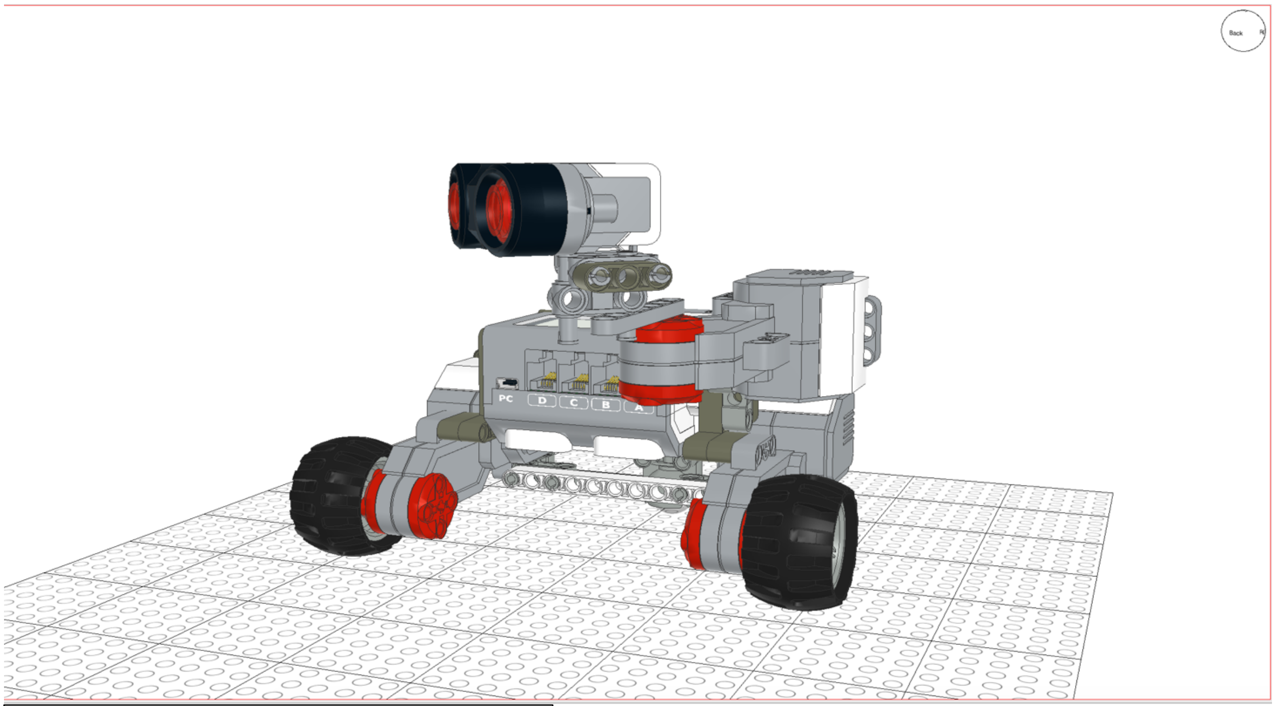

Fig.1 LeoCAD Ultrasonic Robot Model

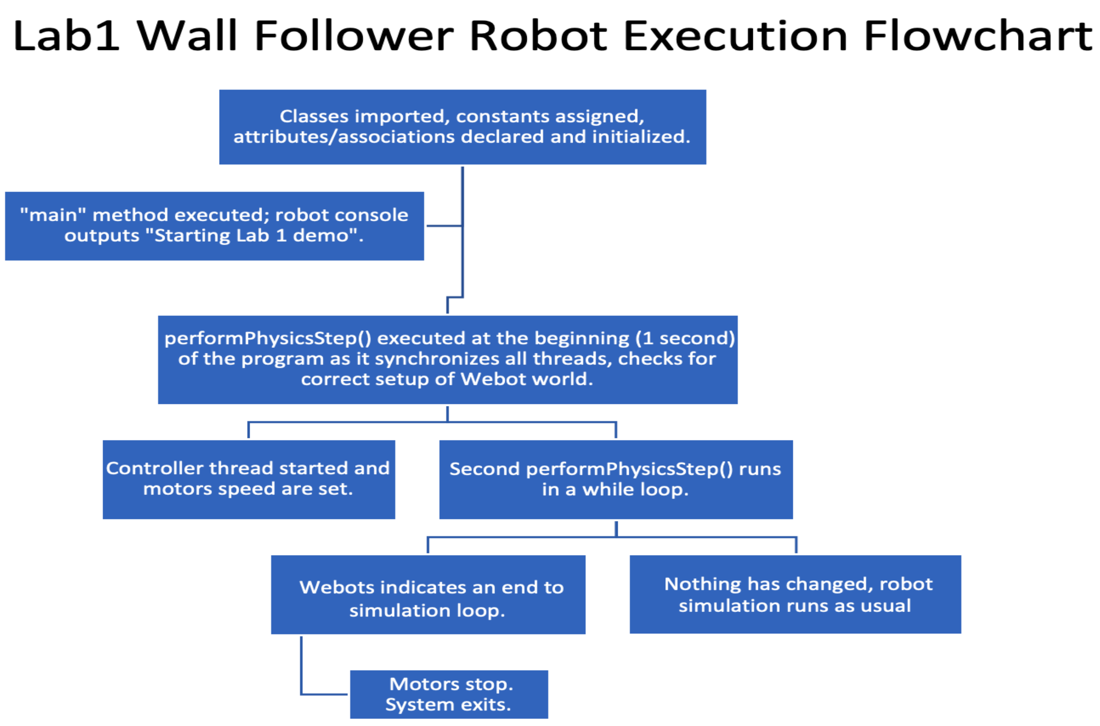
Lab1 Wall Follower Robot Execution Flowchart
Classes imported, constants assigned, attributes/associations declared and initialized.
"main" method executed; robot console outputs "Starting Lab 1 demo".
performPhysicsStep() executed at the beginning (1 second) of the program as it synchronizes all threads, checks for correct setup of Webot world.
Controller thread started and motors speed are set.
Second performPhysicsStep() runs in a while loop.
Webots indicates an end to simulation loop.
Nothing has changed, robot simulation runs as usual
Motors stop. System exits.

Fig.2 Wall Follower Robot Execution Flowchart

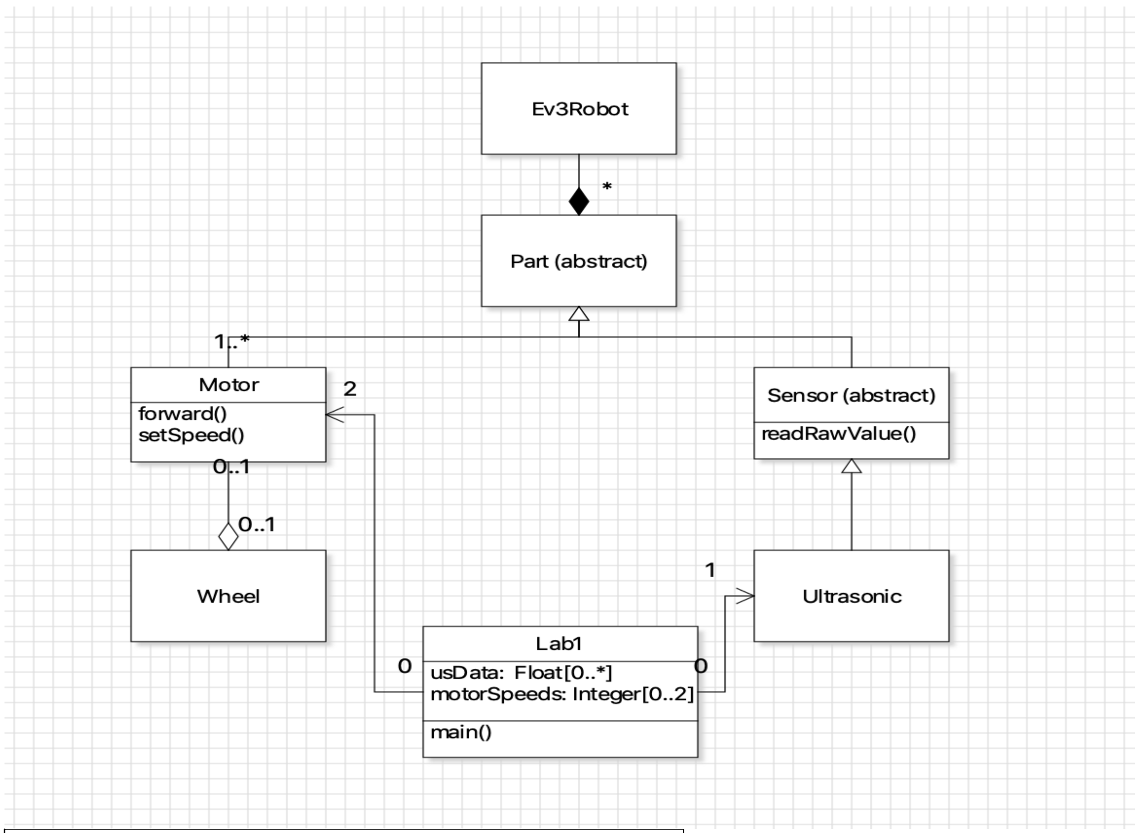

Fig. 3 Class Diagram of Ev3 Ultrasonic Robot

2. Test Data

Test 1:

Problem – The robot did not move forward; it was circulating around in a circular motion.
Observation - The right motor rotation speed was decreased and the left motor rotation speed was increased.
Solution - The slower motor rotation speed (the right motor) was increased slightly.

Test 2:

Problem – The robot moved relatively straight for a short while, then collided with one of the walls.
Observation - When the robot was close to the wall, the speed of the left motor was not high enough.
Solution – The speed of the left motor was increased.

Test 3:

Problem - The robot moved relatively straight, but upon turning a corner it collided with the wall.
Observation - The bandcenter was too low and the robot was initially placed too close to the wall.
Solution - Increased the bandcenter. The robot was then able to easily turn at its first corner without colliding with the wall.

Test 4:

Problem - The robot successfully completed one lap before colliding into a wall corner.
Observation - The cause of the collision may have been due to errors in the ultrasonic sensor.
Solution - The slower rotating motor rotation speed was increased slightly.

Test 5:

Problem – The robot successfully completed two laps but moved very slowly throughout its circulation.
Observation – The velocity of the left and right motors was too low.
Solution - The velocity of the left and right motors was doubled.

3. Test Analysis

How much does your robot oscillate around the band center?

The robot oscillates excessively around the band center. The band center varies from 15 centimeters to 50 centimeters. This excessive oscillation is due to the BANG BANG controller implementation because this type of controller triggers an action on the robot causing it to constantly adjust its distance from the wall. However, if we used a P controller, the robot would have oscillated less around the band center since the speed adjustment is relative/proportional to the error. Hence, a bigger error would cause a greater motor speed adjustment and a smaller error would cause a minor motor speed adjustment.

Did it ever exceed the bandwidth? If so, by how much?

The WALL_DIST_ERR_THRESH of the robot was set to 2 centimeters, the allowed maximum bandwidth between the ideal wall distance and the measured distance. However, throughout the test runs, the robot often exceeded the WALL_DIST_ERR_THRESH by large and small amounts. The largest bandwidth recorded was in the interval [20,30], which exceeds the allowed amount by 20+ cm.

Describe how this occurs qualitatively.

The robot moves forward within the limits of the bandcenter with an error of +/- bandwidth as reported by the ultrasonic sensor. Moreover, the robot will maintain a forward trajectory while the rotation speed of the outer motor equals to the rotation speed of the inner motor. When the ultrasonic sensor reports a bandcenter that is below the error threshold, then the robot is too close to the wall. As a result, the speed of the inner motor must increase for the robot to not touch the wall. When the ultrasonic sensor reports a bandcenter that is above the error threshold, then the robot is too far from the wall. The rotation speed of the inner motor must decrease to move the robot closer to the. The speed adjustment of the one of the motors is what permits the robot to make tight turns around the corners and maintain its trajectory.

4. Observations and Conclusions

Does the ultrasonic sensor produce false positives (detection of non-existent objects) and/or false negatives (failure to detect objects)? How frequent were they? Were they filtered?

The ultrasonic sensor produced false negatives when it tried to complete one lap while following the wall. When the robot was run from one corner of the wall, it successfully turned two corners out of four without touching the wall; however, when it arrived at the third corner, it collided with the wall. These false negatives happened several times during testing and especially when the robot was placed too close to the wall.

When placed at a greater distance from the wall, the robot successfully completed two laps without any collisions. These false negatives were not filtered because the distance measured by the sensor is smaller than the maximum distance detected by the sensor, so the filter was reset to zero in the filter method.

5. Further Improvements

What software improvements could you make to address the ultrasonic sensor errors? Give 3 examples.

1) Splitting robot controller algorithm into two cases: one algorithm would account for the case when the robot is moving against the wall and the second algorithm would account for the case when the robot is turning a corner. These two algorithms would minimize robot oscillation as it moves along the wall and create smoother, more consistent turns at corners.
2) Adjusting starting distance and WALL_DIST constant: by keeping the motor at a further distance from the wall, the motor will make less tight turns at corners.

3) Adjusting motor velocities incrementally: instead of having a constant value added/subtracted to a motor’s rotation speed to change its trajectory, the robot would have run more smoothly if the speed would have been changed relative to the error/bandwidth calculated. The bigger the error, the bigger the change in motor speed and vice versa.

What hardware improvements could you make to improve the controller performance? Give 3 examples.

1) Adding an ultrasonic sensor: instead of having only one sensor pointed 45 degrees to one side, adding an extra 2 ultrasonic sensors would create better controller performance and smoother robot movement as the proximity data from these sensors would give the robot greater “awareness” of its surroundings. One of these extra sensors would be pointed 90 degrees to the side (to measure the side distance between the sensor and the wall) and the other would be pointed straight ahead.
2) Adjusting the location of the ultrasonic sensor: to minimize the number of distance signals received from other parts of the wall (signals that would increase the bandwidth error), it would be better to place the ultrasonic sensor at the same level as the robot wheels/motor.
3) Preventing sensor bounce: by using a more rigid component to hold up the ultrasonic sensor and placing the sensor closer to the middle of that component the amount of sensor up/down oscillation will decrease (this hardware improvement is referring more to the physical system).

What other controller types could be used in place of the controller you used?

The P Controller could be used in place of the BANG BANG controller because the BANG BANG controller constantly triggers an action on the robot. With the BANG BANG controller the robot moves relatively straight if the bandwidth is within the error threshold but if the error is below the error threshold the one of the motor’s rotation speeds increase or decrease a constant amount. This constant quantity added/subtracted is what causes the robot to move around so much during circulation. The P controller the other hand will let the distance from the wall be directly proportional to the speed of the outer wheel. The inner wheel will be kept constant. Proportionally adjusting the motor speeds relative to the error will allow smoother movement.