The Code

Introduction

The code presented here is the implementation of a ripe fruit detector and harvesting robot developed for the National Engineering and Robotics Competition in the Indigenous Category. The robot is designed to follow a predefined path marked by a line and detect ripe fruits at specific junctions. Upon detecting ripe fruits, the robot collects them using servo-controlled harvesting arms and stores them until it reaches the loading area to drop them off. The robot then returns to its initial position and parks itself.

Hardware Configuration

The robot is equipped with various components to facilitate its functions:

1. **Motors:** The robot uses DC motors to move its wheels. Pins L_in1, L_in2, R_in1, and R_in2 control the direction of the motors, while LEFT_MOTOR_ENABLE and RIGHT_MOTOR_ENABLE are used to enable/disable the motor drivers.
2. **IR Sensors:** The robot utilizes Infrared (IR) sensors to detect the line on the ground. Eight IR sensors are positioned on the robot to achieve line-following capability.
3. **Sonar Sensors:** The robot employs two sonar sensors, one on the left and the other on the right, to measure distances and avoid obstacles.
4. **Color Sensors:** The robot integrates color sensors to distinguish between ripe and unripe fruits based on color. The S0, S1, S2, S3, and sensorOut pins are used to control and read the output from the color sensors.
5. **Servos:** The robot uses servo motors for precise and controlled movements. The six servos HR, HL, TL, TR, BL, and BR control the harvesting arms and other movable parts.

Code Structure

The code is written in Arduino programming language and follows the standard setup and loop structure.

Setup Function

The `setup()' function initializes various pins as inputs or outputs and sets the frequency-scaling for the color sensors. It also attaches the servos to their respective pins.

Loop Function

The `loop()` function is the main part of the code that continuously executes once the setup is complete. It is responsible for the robot's primary functionalities, including line-following, fruit detection, harvesting, and parking.

The robot uses IR sensors to follow the line and uses sonar sensors to avoid obstacles and measure distances. When it reaches a junction, it uses specific functions (`handleJunction1()', `handleJunction2()', and `handleJunction3()') to perform the required actions at each junction.

- The `collectLeft()` and `collectRight()` functions are used to control the servo motors and collect ripe fruits at the left and right sides of the robot, respectively.
- The `align()` function is utilized to align the robot to the line during certain maneuvers.
- The `thumka()` function performs a fun dance move when called.
- The `dropOff()` function is responsible for dropping off the harvested fruits at the loading area.
- The `detectColorLeft()` and `detectColorRight()` functions read RGB values from the color sensors to determine whether a fruit is ripe or not.

Conclusion

The ripe fruit detector and harvesting robot code presented here demonstrates an effective and efficient approach to fruit harvesting automation. The robot's capabilities, such as line-following, obstacle avoidance, fruit detection, and servo control, make it a valuable addition to agricultural practices.

It is essential to consider real-world deployment and safety aspects when implementing such robotics solutions in actual farming scenarios. Further optimizations and enhancements can be made to improve the robot's performance and reliability.

The code's success in the National Engineering and Robotics Competition (Indigenous Category) highlights the potential impact of mechatronics engineering in addressing real-world challenges and promoting sustainable practices in the agricultural sector.