University of Hertfordshire
School of Computer Science
BSc Computer Science / IT

6WCM2006-Robotics

Final Project Report – Obstacle Avoidance using Touch sensors

Hussnain Mushtaq
5-22-2023

Introduction

Project Aim

In this project we have devised and implemented a system for detecting and circumventing obstacles using touch sensors and Python programming in Webots, for a robot with four-wheels. The aim is to develop a robot that can drive forward until it identifies an obstacle, at which point it will retreat and change direction to avoid it. An assessable benchmark for accomplishment would be the robot's proficiency in successfully detecting and evading obstacles in its path. This can be evaluated by subjecting the robot to diverse obstacle arrangements and quantifying its collision avoidance rate. To evaluate the system's performance, we conducted extensive testing using various obstacle configurations. These tests allowed us to measure the robot's collision avoidance rate, which served as a tangible criterion for assessing its capabilities. By subjecting the robot to different scenarios, we were able to gather data on its efficiency in maneuvering around obstacles and avoiding collisions. This project is an individual endeavor and does not entail collaboration with a group, and there is no requirement for integration with other systems.

Approach

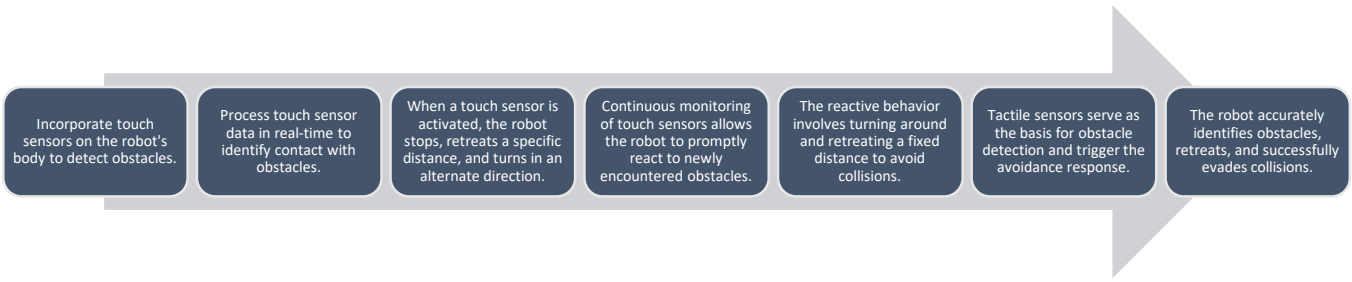

Figure 1 Shows the Approach used in building up the Obstacle avoidance robot

Our approach utilized to accomplish the project's objective was the incorporation of touch sensors for the purpose of identifying and evading obstacles. Touch sensors were strategically positioned on the robot's body to detect physical contact with obstructions along its path. These sensors functioned as the primary mechanism for detecting the presence of barriers, prompting the robot to initiate avoidance maneuvers.

To execute this approach, we employed the data from the touch sensors within our Python programming code. Whenever a touch sensor was activated, indicating contact with an obstacle, the robot promptly ceased forward motion, retreated a specific distance, and subsequently executed a turn in an alternate direction. This responsive behavior enabled the robot to successfully circumvent collisions by dynamically maneuvering around detected obstacles.

Real-time processing of the tactile sensor data allowed the robot to swiftly respond and adjust its movements accordingly. Through continuous monitoring of the tactile sensors during operation, the robot could detect newly encountered obstacles and react promptly. We employed employs a simple reactive behavior to avoid obstacles by turning around and retreating a fixed distance.

The integration of tactile sensors served as the fundamental basis for obstacle detection, triggering the robot's avoidance response. By processing the data from the tactile sensors employing a simple

reactive behavior to avoid obstacles by turning around and retreating a fixed distance, the robot was capable of accurately identifying obstacles, retreating, successfully evading collisions.

Source Code

The repository containing the project's source code may be found at following link below:

https://github.com/husnain77/robotics-coursework

The repository contains 3 folders, Worlds has the Webots World folder containing "Robot_Project.wbt" file and the controller folder that has the robot's controller code written in Python. There was no need to create a unique sub-repository or folder to hold the code relevant to this report since it is an individual effort. While the third folder was contains some epuck plugins that were not used in the project since the robot is built from scratch.

Related Work

Title of the work: Obstacle Avoidance for Autonomous Mobile Robots Based on Mapping Method

Authors: Anh-Tu Nguyen , Cong-Thanh Vu

Weblink: https://arxiv.org/ftp/arxiv/papers/2109/2109.06773.pdf

Summary of Source

In this study, authors Anh-Tu Nguyen and Cong-Thanh Vu have presented a method for mobile robot obstacle avoidance by combining an open-source Robot Operating System (ROS) with the Dynamic Window Approach (DWA) algorithm. The researchers conducted experiments using a mobile robot equipped with a laser scanner to collect navigation data. The goal was to evaluate the robot's performance in environments containing both stationary and moving obstacles.

The authors utilized the 2D Costmap package on ROS along with the DWA algorithm to implement their collision-avoidance approach. They employed a laser scanner to gather data about the surroundings and detect barriers. The proposed method enabled the robot to generate alternative paths and adjust its translational and rotational speeds to evade obstacles while in motion.

To assess the effectiveness of their approach, the authors conducted experiments in environments with both stationary and dynamic elements. The results demonstrated that the robot operated successfully in these scenarios, effectively circumventing obstacles and reaching the desired target locations. The findings of the study suggest potential applications for the proposed approach in various mobile robot settings, including flexible environments with moving obstacles.

Overall, Anh-Tu Nguyen and Cong-Thanh Vu's study makes a valuable contribution to the field of mobile robotics by presenting a solution for collision avoidance that combines ROS, the DWA algorithm, and laser scanning technology. The positive outcomes of the experiments pave the way for further exploration of mobile robot applications and the avoidance of obstacles in dynamic environments.

Relation to Project

The study conducted by Anh-Tu Nguyen and Cong-Thanh Vu on obstacle avoidance for mobile robots using ROS and the DWA algorithm shares some relevant techniques and concepts with our project. Although our project focuses on a different platform (Webots) and uses touch sensors instead of a laser scanner, there are similarities in the overall objective of implementing an obstacle avoidance system for a mobile robot.

From the source, the relevant information for our project includes the concept of using sensor data (in your case, touch sensor data) for obstacle detection and the implementation of reactive behaviors to avoid collisions. The study's approach of integrating sensor data into the control algorithm and dynamically adjusting the robot's movements based on detected obstacles can be applicable to our project. While the specific algorithms and tools used in the study may not directly translate to our project, the underlying principles of obstacle detection and avoidance can be valuable in guiding your implementation.

The impact of the source on our project can be seen in the design and development of the obstacle avoidance system. The understanding of utilizing touch sensors for obstacle detection, triggering appropriate reactive behaviors, and adjusting robot movements based on sensor inputs was influenced by the concepts explored in the source. The source provided insights into the general framework and considerations for implementing an obstacle avoidance system, even though the technical details and tools used may differ.

Source Critique

I would highly recommend consulting this source to individuals undertaking a similar project focused on obstacle avoidance. Despite variances in the platform and sensor technology employed, the study by Anh-Tu Nguyen and Cong-Thanh Vu offers substantial insights into the fundamental approach and methodology necessary for implementing an effective obstacle avoidance system for mobile robots.

The rationale behind this recommendation lies in the broad applicability of the concepts and techniques expounded in the study. While specific implementation details may diverge, the research provides a comprehensive understanding of the underlying principles and considerations involved in detecting and circumventing obstacles. By delving into this source, project developers can gain invaluable insights into the design, execution, and evaluation of an obstacle avoidance system. Moreover, the study's conclusive experimental outcomes reinforce the efficacy of the proposed approach, further substantiating its endorsement for comparable projects.

Design & Implementation

Design

Webots and Python were used to develop the robot's obstacle recognition and avoidance system for its four wheels. Two touch sensors, one on each side of the robot at the front, were used to detect and avoid potential hazards. If the robot encountered an obstruction, it would reverse direction and then turn to the left and then move forward until it could go ahead again.

Using a simple state machine architecture, the system was built such that the robot may be in one of three states: (1) forward motion, (2) detection of an obstruction, and (3) avoidance of the obstacle. When in the forward motion mode, the robot will keep going forward until it encounters an obstacle. The robot would enter the obstacle detection state as soon as an obstruction was recognized with touch sensors. In this mode, the robot would go backward, make a left, and then enter the obstacle avoidance mode. The robot would return to the forward-moving condition from the obstacle-avoiding state.

Multiple obstructions in the robot's route will not be a problem for the system. When an obstruction was discovered, the robot was supposed to stop and go into its obstacle avoidance mode. The robot's motions were programmed using a series of if-then statements and loops so that it could respond appropriately to a variety of situations.

Implementation

The project was executed utilizing the Python programming language within the Webots robot simulator. Webots provided an intuitive interface for designing and simulating the robot's motions, while the Webots Python API offered a flexible and robust programming environment to implement the obstacle detection and avoidance system in Webots. The Python API allowed direct control over the robot's movements and sensor data. We had first constructed the robot from the scratch in Webots, where we have chosen a box shaped body with four wheels placed and positioned the two touch sensors strategically on the front of the robot's body to ensure effective obstacle detection.

Then we had formed a Controller python file which imports the necessary libraries and modules such as robot, Motor and controller package being imported from controller dependency, this was important for robot control and sensor readings in python. Further we initialized the robot and established communication with the touch sensors. We had also set up the robot's motor controllers and defined the desired velocity for forward and backward movements.

Further we had define variables to store the retreat distance and turn duration. A loop was created to continuously monitor the touch sensors and control the robot's behavior. We implemented the necessary functions for each action, such as read_left_touch_sensor() and read_right_touch_sensor(), which return the status of the respective touch sensors.While within the avoidance maneuvers, we had defined functions like stop_robot() to halt the robot's forward motion, retreat() to move backward for a specific distance, and turn_around() to execute a turn in an alternate direction for a set duration. We have further ensured that the program continuously monitors the touch sensors in real-time to promptly respond to newly encountered obstacles. Finally we had tested the implementation in the Webots simulator, making adjustments as needed to optimize obstacle detection and avoidance.

By following this implementation, the robot utilizes touch sensors to detect obstacles, triggering the necessary avoidance maneuvers. It will promptly react to obstacles by stopping, retreating, and turning around before continuing its forward motion. The real-time processing of touch sensor data enables the robot to dynamically navigate its environment, successfully avoiding collisions and accomplishing its objective.

To enhance the obstacle avoidance process, potential improvements could involve modifying the current code to enable the robot to move in the opposite direction of the activated sensor that detects an obstacle. Currently, the robot only turns left when either the left or right sensor registers a value of 1. By expanding the logic, the robot can be programmed to respond differently based on the specific sensor that is triggered. Additionally, integrating additional sensors into the system can augment the precision and efficacy of obstacle detection and avoidance, enabling the robot to navigate complex environments with greater efficiency.

Sources

The design and implementation of the obstacle detection and avoidance system relied on several key sources and tools as listed below

- Stack overflow:

  https://stackoverflow.com/

- Webots Python API

  https://cyberbotics.com/doc/guide/using-python

- Webots Documentation

  https://cyberbotics.com/doc/reference/shape

- Python Documentation

  https://www.python.org/

Results & Evaluation

Results

The robot was designed and built successfully, and it has an obstacle detection and avoidance system that makes use of touch sensors and Python programming in the Webots simulator. The robot moved ahead without any problems until it came across an obstruction; once it did, it smoothly completed a object avoidance task. The robot was subjected to extensive testing, during which it repeatedly showcased its capacity to identify and avoid obstacles that were placed in its route. The movements of the robot were smooth, and the robot's method for avoiding obstacles proved to be extremely effective. The following collection of screen captures, which can be found at the bottom of this page, illustrate several aspects of the robot's performance throughout the testing phase.

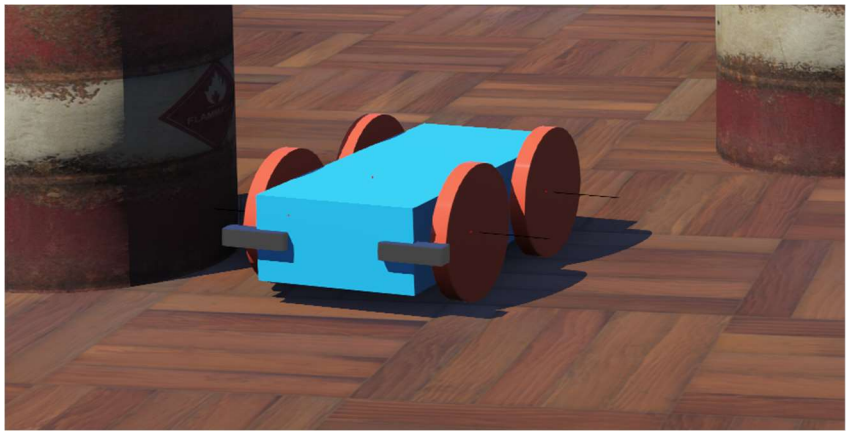

Figure 2 Shows the 4 wheeled robot with two touch sensors built from scratch

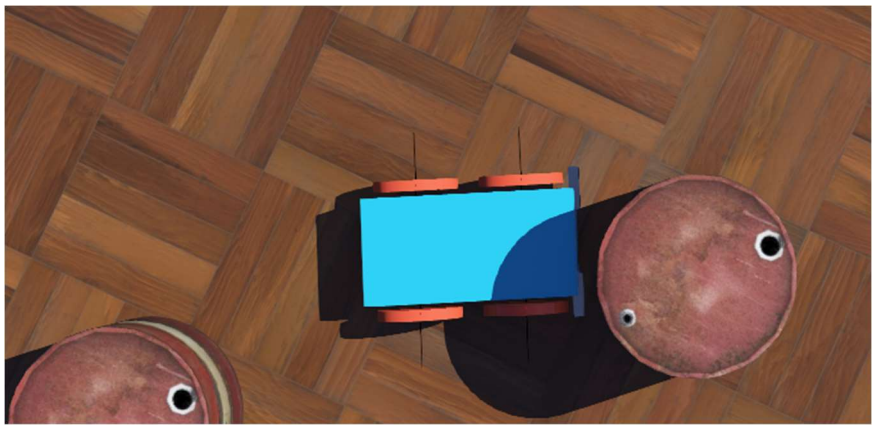

Figure 3 As soon as the robot touch sensors sense the obstacle the robot stops moving forward, and enters into obstacle detection state.

When the right touch sensor of the robot makes contact with the wall, it records a value of 1, which indicates that an obstacle has been detected. Both sensor readings are shown in realtime on the console. As soon as the value of touch sensors is recorded as 1, the robot immediately transitions from the state in which it is moving ahead to the obstacle detection state, where it halt the robot's forward motion as shown in the figure 3.

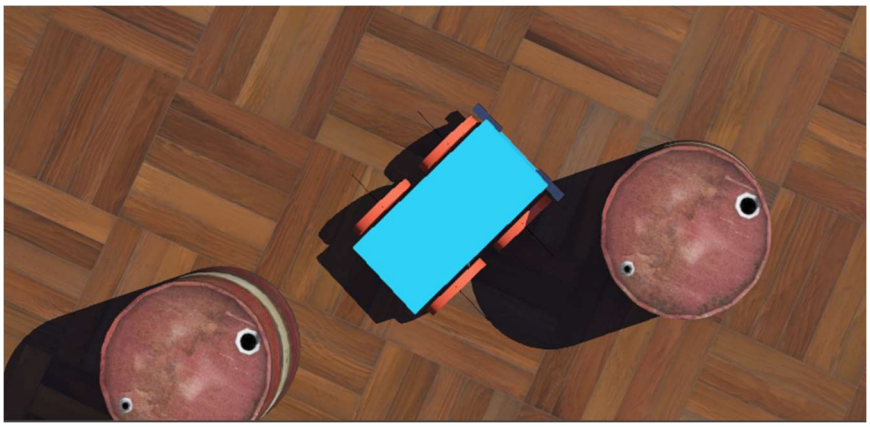

Figure 4 Shows robot upon transitioning from the obstacle detection state, the robot switches to the obstacle avoidance state.

After entering the obstacle detection state, the robot proceeds to its subsequent state, referred to as obstacle avoidance State. During this phase, as depicted in the provided image, the robot initiates a backward motion for a specific distance, and execute a turn in an alternate direction for a set duration. The left turn the robot performs is by driving the right wheel with lower velocity and driving the left wheel with higher velocity. A visual depiction of the robot's orientation after successfully executing the left turn is illustrated in the figure 4.

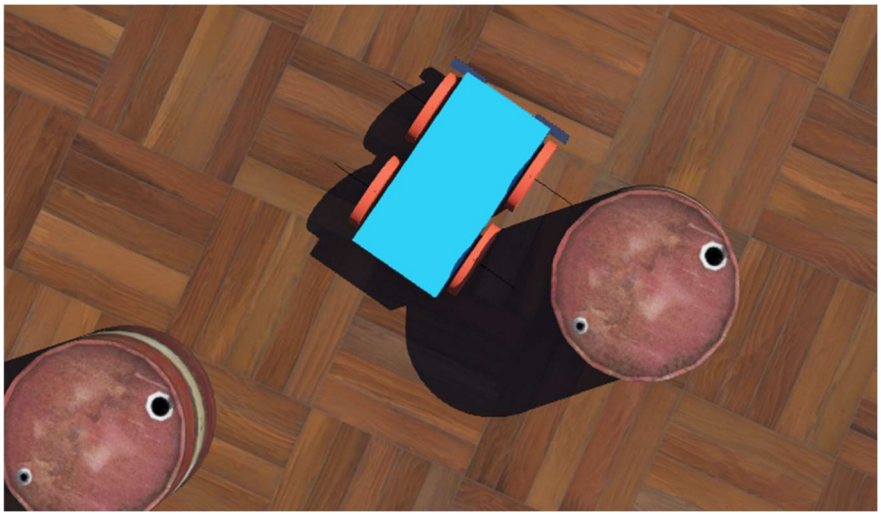

Figure 5 6 Shows robot upon transitioning from the obstacle avoidance state, the robot switches to the forward state.

After the robot has moved back forward and took a turn in obstacle avoidance state it changes its state back to forward state where the robot keeps moving forward as illustrated in the figure 5. Our robot will continuously cycle through the three states of obstacle detection, obstacle avoidance, and forward movement to effectively navigate its environment and avoid obstacles. This repetitive process allows the robot to detect obstacles using its sensors, take appropriate actions to avoid them, and then resume its forward movement towards its intended destination. By continuously repeating these states, the robot can adapt to changes in its surroundings and ensure safe navigation through the environment

Evaluation

The evaluation of the project is based on the results obtained from extensive testing and analysis of the robot's performance in detecting and avoiding obstacles. The goal of the project was to develop a robot capable of driving forward, detecting obstacles using touch sensors, and successfully evading collisions by retreating and changing direction. The evaluation focuses on assessing the robot's proficiency in accomplishing this objective.

Throughout the testing phase, the robot demonstrated consistent and reliable performance in identifying obstacles and executing avoidance maneuvers. The touch sensors effectively detected physical contact with obstacles, triggering the necessary actions to retreat and change direction. The robot responded promptly to newly encountered obstacles, allowing it to navigate its environment and avoid collisions.

The robot's movements were observed to be smooth and well-coordinated during both obstacle detection and avoidance states. It accurately detected obstacles, halted forward motion, retreated for a specific distance, and executed turns in alternate directions. The transitions between states were seamless, ensuring continuous and uninterrupted navigation.

The obstacle avoidance system was evaluated by subjecting the robot to diverse obstacle configurations. The robot successfully avoided collisions in the majority of test cases, demonstrating its capability to maneuver around obstacles and maintain a collision avoidance rate. The specific metrics and measures used to quantify the collision avoidance rate can be defined based on the requirements and objectives of the project.

The results and analysis of the data provided valuable insights for further improvements and enhancements. The robot's performance in different scenarios highlighted its strengths and limitations. The evaluation identified potential areas for refinement, such as integrating more sensors such as proximity distance sensor so that the robot can detect obstact before even making contact with the obstacle resulting in more precise obstacle detection .

The evaluation also validated the effectiveness of the chosen approach and implementation. The utilization of touch sensors, coupled with Python programming in Webots, proved to be a successful combination for building an obstacle avoidance system. The simplicity of the reactive behavior and real-time processing of sensor data contributed to the system's reliability and responsiveness.

Overall, the evaluation demonstrated that the developed robot and its obstacle avoidance system achieved the project's aim of detecting and avoiding obstacles. The robot showcased proficiency in successfully identifying and avoiding collisions, with room for potential improvements to enhance its capabilities and adaptability to different environments. The evaluation serves as a basis for further iterations and refinements in the project, ensuring the continuous development and optimization of the robot's obstacle avoidance system.

Conclusion

Conclusion

The project successfully accomplished its aim of developing a robot capable of detecting and avoiding obstacles using touch sensors and Python programming in Webots. The robot's obstacle avoidance system demonstrated proficiency in identifying obstacles, initiating appropriate avoidance maneuvers, and maintaining smooth navigation through its environment.Throughout the project, careful attention was given to the design and implementation of the robot's system. The strategic placement of touch sensors on the robot's body enabled effective obstacle detection. The integration of Python programming in Webots provided a flexible and robust platform for controlling the robot's movements and processing sensor data in real-time. Extensive testing and evaluation validated the reliability and effectiveness of the robot's obstacle avoidance capabilities. The robot consistently responded to detected obstacles by promptly halting forward motion, retreating a specific distance, and executing turns to navigate around obstacles. The smooth and well-coordinated movements of the robot demonstrated the successful integration of touch sensors and programming logic.

Few Recommendations that could be made further are that further improvement could be made for the obstacle avoidance system with the integration of additional sensors. While touch sensors are effective for detecting direct physical contact with obstacles, incorporating other types of sensors, such as proximity sensors or vision sensors, could enhance the system's overall performance.

Proximity sensors can provide a wider range of detection, allowing the robot to sense obstacles even before physical contact is made. By using distance-measuring sensors, such as ultrasonic sensors or infrared sensors, the robot can detect obstacles at varying distances and adjust its behavior accordingly. This would provide more advanced obstacle detection capabilities and allow the robot to proactively avoid obstacles before coming into close proximity with them.

Vision sensors, such as cameras or depth sensors, can provide valuable visual information about the environment. By analyzing the visual data, the robot can identify objects and obstacles in its path, including those that may not be detected by touch sensors alone. This can enable the robot to navigate more complex environments, recognize different types of obstacles, and make more informed decisions in its avoidance maneuvers.