ROBOTIC APPLICATIONS ASSIGNMENT

Jamaine Christian

DECEMBER 13, 2023

+

ROBOT STRATEGY

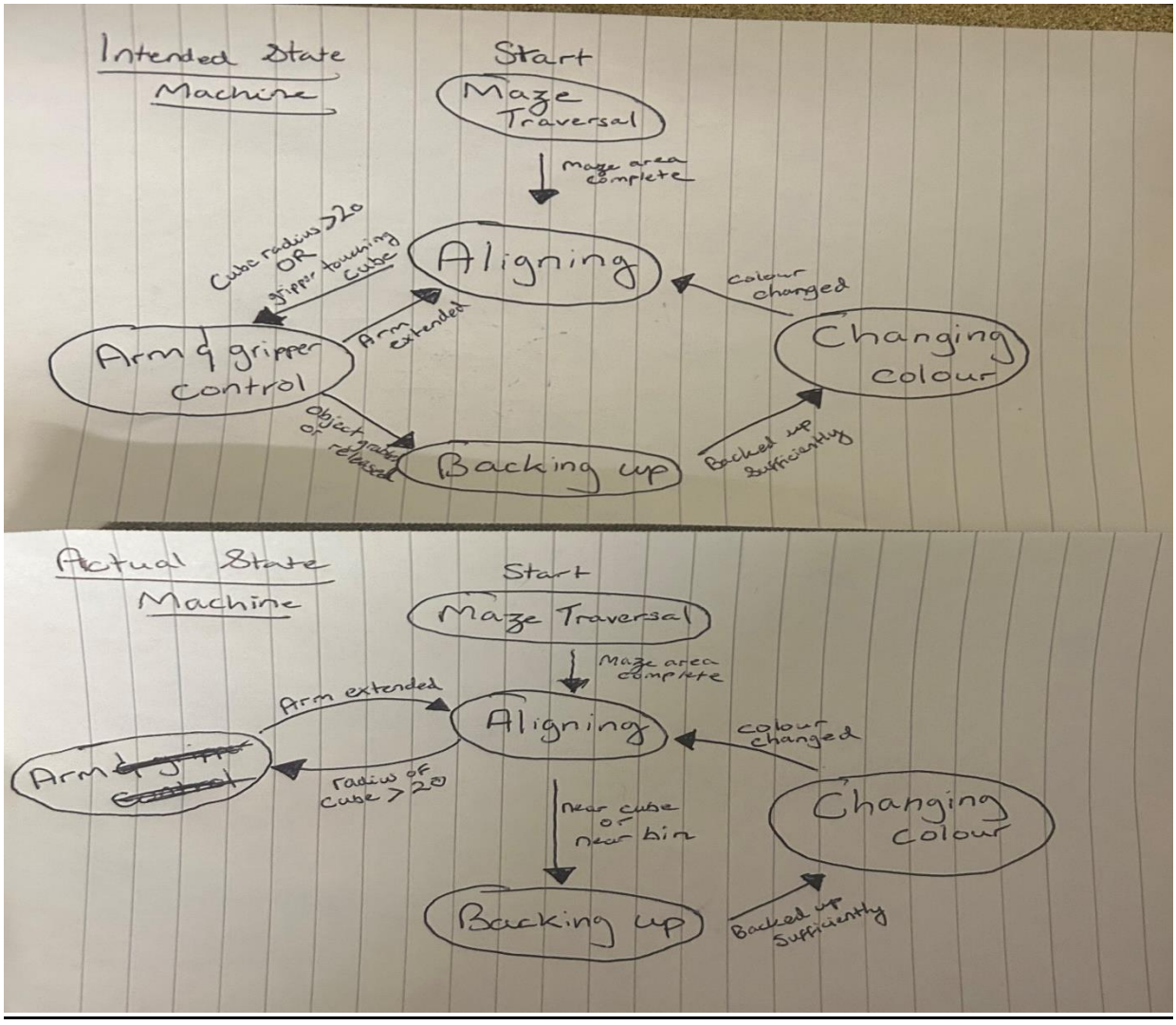
Introduction

This assignment aims to navigate a Tiago Bot, a programmable robot, through a maze-like path, approach a table, pick up one of the blue coloured cubes and then place it in the green bin. This task is performed in a simulated environment by publishing and subscribing to different ros2 topics that are active within the gazebo environment. My approach utilises a C++ script as a state machine for the control mechanism of the robot. Within this report I am going to describe how I have broken down different aspects of the scenario, explain parts of my implementation and then reflect on my approach.

Modelling

I broke down the problem by identifying different aspects of the assignment, maze traversal, aligning to the cube, picking it up, backing up and then taking it to the bin. These tasks were handled separately using a state machine.

State Machine Diagrams



The state machine begins in the maze traversal state. To navigate round the maze area, we set the forwards speed by on the servos, this speed is then reduced if there is a turning ahead. The turning speed is then set based on whether there is a wall or not on the left or right and will turn towards the side without a wall. The robot state transitions to the aligning state when the maze is finished.

In the aligning state, if the robot is looking for the first cube it will just drive straight until it can detect the blue coloured cube. It then centres the cube by turning the robot or adjusting the head tilt. It then continues straight until it reaches a certain distance and then transitions into the arm control state. When returning from the arm control state is continues to align the object with the arm and will now do this for any object. After it transitions back into the arm control state when aligning has finished.

The arm control state would be controlling the arm movement and the gripper. The robot would extend the arm on its first call and then transition back to the aligning state when the arm has finished moving. Every other call it would switch between opening and closing the gripper. It would transition the state into backing up.

The backing up state would use the same principles of the aligning function to keep the object centred, while it uses a negative speed to reverse until backed up sufficiently. It then transitions into the changing colour state.

The changing colour state t changes the target colour based on its previous target. This allows the robot to cycle between targeting a cube and then the bin.

Implementation

This project needs the core ROS 2 C++ client library, “rclcpp/rclcpp.hpp”, for creating ROS nodes and interacting with middleware.

The robot state machine was controlled using an Enum class of robot states. The state machine is originally set to maze traversal and run in the timer callback every 500ms through a switch statement based on the robot’s state and looped different logic for different states.

Maze traversal needed the “sensor_msgs/msg/laser_scan.hpp” library so we could use messages for the laser scanner. It subscriped to the “scan_raw” topic and uses the values to create an average for all the lasers on each side of the robot, so we only deal with one value each for the left, front and right laser sensors in our calculations. The robot sets a moving speed by publishing Twist to the “servoing_cmd_vel” topic and then if the front sensor reads a value less than 3, i.e. there is a turn, it will slow down. This needed the "geometry_msgs/msg/twist.hpp" library so we could use messages for the servos. The turning speed is set to the difference between the right and left sensor averages, this proportional turning speed allows for the robot to smoothly turn corners and self-align. The state then becomes the aligning state when both the left and right sensor averages read more than 5m, i.e. the maze has been completed.

Aligning uses subscribed data from the “/colour_detection/” topic to get the x-coordinate, y-coordinate and radius of a detected object on each callback. To do this we need to include the "geometry_msgs/msg/point32.hpp" library. It will set a linear speed until the radius receives from the call-back is 20cm. If an object has already been detected the robot has an extended arm it will align until the radius is a certain value this is different for each object and is set by checking the current colour that is being targeted. It does the alignment by creeping forwards with a constant

linear speed and set proportional turning speed to the error of the x coordinate from the centre of the colour detection screen. It also uses head tilt that is adjusted by creating an action goal and setting the values of the head position by incrementing or decreasing the tilt value we send in the goal request. This is varied if based on the y coordinate error. It then transitions to the backing up state. These libraries are needed for this to be able to occur: "control_msgs/action/follow_joint_trajectory.hpp", "trajectory_msgs/msg/joint_trajectory.hpp", "trajectory_msgs/msg/joint_trajectory_point.hpp", "rclcpp_action/rclcpp_action.hpp", "rclcpp_components/register_node_macro.hpp"

The arm control state needs the "geometry_msgs/msg/pose.hpp" and "std_msgs/msg/bool.hpp" libraries to send and receive messages to the pose of the arm and to detect if an arm position has been reached or not. In this state we publish to the "/ros360_movit/request" topic. The arm control will attempt to reach out in front of itself by publishing the goal to the "/ros360_movit/request" topic until the message from the "/ros360_movit/result" topic subscription reads true. It then transitions back to aligning.

The back up state uses the same publishing and subscribing as aligning but sets the linear speed of the Tiago Bot to a negative value and then backs up until the radius of the object is of a certain threshold this is determined by which colour it is that is being targeted. The state is then transitioned into changing colour.

The changing colour state creates a client to the service "/colour_detector/setParameters" which changes the RGB values of what the colour detection topic is looking for based on the previous colour it was targeting and then transitions the state back into Aligning.

Discussion and Conclusion

I think my approach was a good and efficient. If I could get the gripper functionality to work, then it would be clear that my approach was viable. It performs very well in all other functionalities; however, it could be improved by making it more flexible to different situations by using values from world space by incorporating the depth camera and using the offsets of each coordinates for arm extension instead of a preset value. Doing the same on the gripper would allow me to grab different sized objects at different distances. The state machine is utilised efficiently to make sure too many topics are not being subscribed and published to at the same time and to avoid conflict.

References:

I used ros360/moveit (2023) package from blackboard, given by Patricia Shaw and Fred Labrosse

ETHICAL CONSIDERATION

Robot architects develop new approaches to building construction to ease housing shortage.

The robot architect becomes a saviour in the fight against housing scarcity. These marvellous machines have the potential to completely revolutionise the building industry and provide an answer to the ongoing housing crisis. Their accuracy, speed and capability to work at them standards for long durations foreshadow an era of faster more efficient construction. However, as these robot architects emerge, so do questions on the ethical decisions and societal impacts. The integration of robot architects creates many advantages and ethical considerations that need to be examined.

Robot architects give promise to try and ease the house shortage and this will reduce homelessness, which currently plagues most major cities. According to the big issue “The most recent annual count showed 10,053 rough sleepers spotted on London’s streets between April 2022 and March 2023” [1]. With robot architects having the ability to work on projects faster and longer without the need for breaks, we can assume that projects will be completed a lot quicker, and people will be housed at a faster rate. These robot architects can also replace humans in dangerous environments and reduce injuries within the industry. According to a study done by Herts Tools Co. “there are 69,000 non-fatal injuries to construction workers each year” [2]. Therefore, if we replaced human workers for robot architects in dangerous situations it would dramatically reduce the number of injuries within the construction field. Automation in construction could also optimise resource utilisation and minimize wastage, which would reduce the cost of the project and therefore allow house prices to fall consequently helping reduce the number of homeless.

On the other hand, we have ethical considerations we must address. These robots are cost-effective and consequently many workers will lose their jobs. Not properly preparing people for this or implementing strategies to help displaced workers could lead to boycotts like what happened with Nokia when it closed one of its factories in 2008 to shift the work to Romania to cut costs, a head-line reads “Anti-Nokia anger in Germany for closing a factory is growing with politicians publicly ditching the firm's phones and joining calls for a national boycott in Europe's largest economy” [3] and shows how large of an impact a decision like this could have, that even politicians are publicly boycotting firms. In construction, ethical decision making arises in different scenarios, such as ensuring worker safety or resolving unanticipated challenges. A robot architect would not be fit to make these types of ethical decisions. If a robot oversaw worker safety, and then something went wrong, and the worker got injured. Who would be to blame? The robot? The CEO? The programmer? The Robotics engineer? There are too many variables to figure out who is to blame for the mistake, and this rises concern for people. Thes robot architects may just increase house prices made by them instead of decreasing them due to making technologically advanced homes, which could become inaccessible to the poor. This would further increase housing inequality.

Robot architects have clear positive impacts such as reducing homelessness, creating a safer work environment and optimising resource utilisation that make the robot look like a hero, however, if we do not have ethical considerations such as considering the impact of workers that will be laid-off,

some guidelines on usage and liability and to ensure low house prices for all. They could easily be seen as the villain.

Self-assessment

I believe my overall grade for this assignment should be 80% because my program is almost completely functional except the gripper not working and the arm not being optimised (23/30), my screen cast clearly describes and explains the robot's behaviour (8/10), and my ROS usage was good to be able to allow that much functionality (16/20)

In addition, I believe my assignment was very well documented within the robot strategy section (17/20) and my article on ethical considerations was good and used plenty of references (16/20)

References

[1] Geraghty, L. (2023, November 2). Homelessness facts and statistics: The numbers you need to know in 2023. The Big Issue. Retrieved from https://www.bigissue.com/news/housing/britains-homelessness-shame-cold-hard-facts/ [Accessed on 12th December 2023]

[2] Herts Tool Co. (2023). Accidents in the construction industry report. Retrieved from https://hertstools.co.uk/accidents-in-the-construction-industry-report/ [Accessed on 12th December 2023]

[3] Agencies. (2008, January 21). Anti-Nokia backlash grows in Germany. China Daily. Retrieved from https://www.chinadaily.com.cn/world/2008-01/21/content_6408183.htm [Accessed on 12th December 2023]