Product: TransportED
Team: AutomatEd
Student: Jiaqing Xie

Abstract

TransportED is a service that helps you automate warehouses and aims to remove the biggest barrier to entry-large-scale infrastructure investment. The main functionality is to pick up, reserve parcels automatically, while moving and routing paths intelligently by interaction with other robots. I mainly contributed to navigation part, which can help the robot move in the right direction and provide path finding function in simulation. In addition, I made a lot of contributions in the writing part, such as the general part of the report, user guide and website. The most important thing that I learned is to arrange the time reasonably and divide the big task into several smallest sub-tasks. From the technical side, I learned a lot in Webots simulation and server-bot interactions. From the writing point of view, I learned to summarize the work of each member into a smoother and more consistent report and also make the user guide more user friendly. From a communication perspective, I understand that I sometimes misunderstand others, so I need better communication skills.

1. Contribution

At the beginning of the SDP project, I was assigned to the hardware and simulation team as a hardware engineer. The most time I have invested in this project is the navigation part, which aims to make the robot move in the correct way. This part can be divided into several sub-parts, which are detecting straight lines and intersections on the ground, avoiding collisions with stuff in the warehouse, moving the robot from one node to another, and finally getting rid of hard coding and developing routing algorithms.

From mid January to early February, I have been familiar with the interface of Webots in python. Since I have no prior knowledge about robot and python interface before at that time. I completed the simulation tutorials on webots(WT). for several days and created a similar four-wheeled robot to become our first demo robot. The *first task* the robot needs to perform is to **track lines** on the ground. I integrated a **distance sensor**, which is placed on the front of the robot to detect the white line with black ground, because the distance sensor will return a larger distance when the robot deviates from a straight line . The difficulty is that when we change the warehouse scene, the straight lines on the ground also need to be changed. So I shouldn't hard code the lines myself. A member of the software team provides me with a program that can generate lines based on specific scenarios, which is helpful for my testing. The measurement of success is to observe whether the robot follows the lines or not. It is one important and basic task that should be performed on the robot. Without following lines, the robot will need to move in a random direction, which might be be in collision with other robots since our project is based on multi-agent interaction.

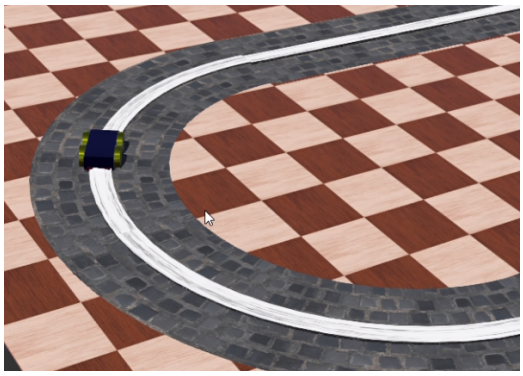

Figure 1. line following mechanism

I also placed two sensors in the front to see if the robot would collide with things in the warehouse. I also realized that only one distance sensor cannot detect the **intersection** of lines because there is only one angle for each sensor. Then I added another two sensors on the both sides of the robot body to detect whether there's left or right branch at the intersection point. The measurement of *intersection* of the lines is to test for each situation, such as when it is necessary to turn left, it should detect the intersection and immediately turn left. The measurement for *avoiding collision* tests is that we find a box(or other similar things) on the robot's forwarding path and to see whether it will collide with that box. Succesfully all the tests are passed. It is very important for testing our multi-agent algorithm. which is done by another team member, because we don't want the two robots to collide together. Therefore, it's the subtask of the path-finding task and multi-agent algorithm task.

Throughout February, I completed the task of successfully integrating a person's A-star algorithm to navigate our robot from one node to another in webots simulation. This is completed by the Finite State Machine(*FSM*) and the linked data graph structure. A member of the software team provided the A* algorithm to solve the path-finding problem for the robot so that I can easily get the result of route that robot should go through. But the difficulty that left me was that the robot didn't know the **direction** at all. From our perspective, we know that the robot is heading east or south

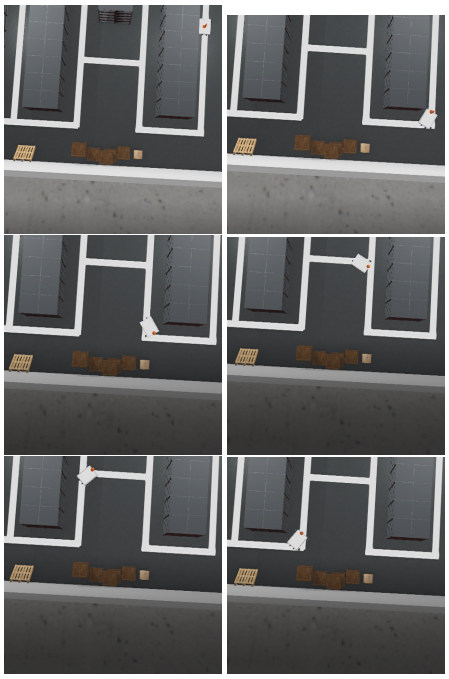

Figure 2. An example : from node 17 to node 22, robot should turn tight, turn right, turn left, turn left and finally turn right.

or west. However, the robot cannot know this information because it has no sense of direction, instead of turning left, turning right, walking straight or backward. So I has designed the linked graph structure that includes coming in node information, current node information, coming out node information and the operation information(such as from path 1 changing into path 2, go straight and then go right). The server should reserve the data structure like: [17: 18, 19, 20, 21](like a fowarding table in packet transfer in network layer). The navigation encoding information is: 1,2,3 and 4 given to forwarding east, south, west and north. If a a two-path information is: path 1 to path 2, which means the robot should first go east and then go south. From the robot's perspective, it should first go forward and then turn right. It's straightfoward and easy to understand. The FSM states includes the *stop* state, *running* state and *chaning direction* state, so when the robot is following a line, it's in running state. It should change to *changing direction* state when it is told to change its direction by the server. It's hard coded and is the subtask of routing algorithm.

From the end of February to the beginning of March, I am supposed to write the routing algorithm to avoid hard coding but I find it very difficult for me to write a new interface which should consider the server's interactions with the robot. It's chanllenging because I am not skilled at this. Also I need to submit and revise my ICLR workshop paper(now accepted) at that time, so instead I wrote the general part of the demo report(one and a half pages) as well as organizing people to fill in their corresponding part in the report which has a lower difficulty and can also manage it by myself. It has been agreed by team leader Fredrik. I sum up overall what specific parts we have done according to our plans since the last demo, what our weekly meeting is about (everyone shares their progress on a daily routine) and what every person has completed specifically, together with the modifications and future works. I was repsonsible for the same part for the last demo report too. Besides the last demo report, I also contributed to the user guide, including the pictures' preparation, introduction part,the description of how it works part, part of the installation part and revising the team members' expressions. I also contribute to the budget part of the website. Summarizing part is very important both for user guide and demo reports because it will represent our outcomes to the specialists and markers explicitly. Otherwise they will not keep on with our progress.

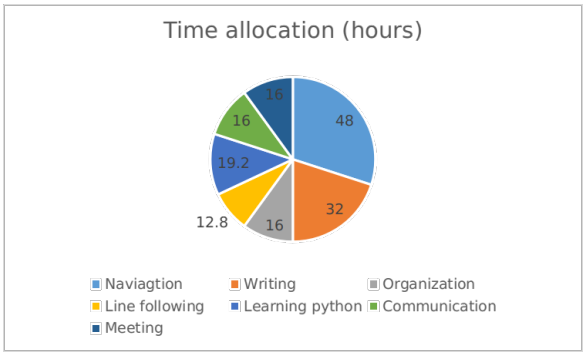

Figure 3. recall: time allocation for this project

2. Lessons learned

In terms of technology, the most important thing I learned is the process of solving super complex problems. Do the smallest things first, and then gather them together. Because our project consists of many complex parts, such as hardware verification and design, webots simulation, database design, navigation, and server-host interaction. Therefore, we need to break them down into small tasks. For me, I should first build a demo robot to make sure that at least we have a model and it's not possible to design navigation without a physical robot. Based on the model, I can then add tasks to be completed by the robot. Similarly, the navigation part also needs to complete the process from hard coding to self-generated. Not if we directly generate the routing algorithm, it is very easy to understand.

In addition, I also learned how to simulate in Webot. I lack the interface knowledge and python package for the robot, which always prevents me from further developing the program. I spent a month learning python and applying what I learned to the project, for example, by using the while statement to implement the finite state machine problem, we need to write each state function to switch between states. I also learned network and front-end knowledge and combined what I have learned in the computer communication course this semester to understand codes about our network interface. For example the tags on the ground needs to send a signal to the server. Server have received the information and extracts the data structure information from database

(which is described earlier) and then returns to the robot. The bot is required to perform the operation or task it has been sent to. This really helps me understand the work of other members and try my best to help them when they need help.

In terms of report writing and organization, I learned to summarize the work of different people together in a concise and clear way. My several conference and journal paper writing experiences told me to write more comprehensive contents . However it sometimes takes up a large space in the demo report. I didn't realize this problem until I wrote too much and wordy in the report which left a little to the technical details part. One team member helps polish the sentences and then I learned from his writing skills when it comes to a 4 page limited report. I improved a lot in the 4th demo report to make sure that I will not increase the burden on others on revising my part. All team members are satisfied with this improvement.

From the experience of organizing people to fill in the report and give the feedback on their progress, I learnt to get contact with everyone in the group instead of just mentioning them in the general channel. This not only allows me to know their personal progress, but also improves my communication skills. Both in the group meeting and team meeting, I learnt to report my progress explicitly to other group members. In the first few weeks, I often let others misunderstood my navigation work and made it overlap with another person's work. So I expressed my aims and current works more clearly in the meeting and also took notes to reduce misunderstandings.

Figure 4. self reflection

Overviewing our whole project, there are two parts I think we can do much better. One is navigation from one node to another. I first thought of the line tracking mechanism, because we need to avoid conflicts with other robots in the warehouse. However, a more automatic routing mechanism is much better. For example, there's a demo (ST) which provides a automatic path finding function, under the scene of a supermarket which is similar to our warehouse robot. And we do not need NFC tags and high-way algorithm any more. This task is more chanllenging which needs a more complex multi-agent algorithm therefore we give up this idea. But it deserves a try, which saves the budget spent on the tags and saves time as well. Another part is about the report part. We should add more pictures in the report instead of expressing too much in the report to show up everybody's contributions. For example in the first page of each our report, we need to add an image of our robot, which can express a progress of the robot construction from zero to one. Also, we can design more evaluations to make sure of the robot's robustness. The descriptions of the evaluations also are not comprehensive. Also in the first two demo report design details part, we say too little about why we choose this algorithm or this electrical component which does not make the markers convincing and confident about the correctness of our design.

The most regrettable thing is that I did not finally realize my routing algorithm. It also reflects that I do not have arranged time too well between paper submission and algorithm designing during that time period. However I have known the principle and reproduced the other's algorithm by myself. I think it would be very helpful to my future research work. And also I am very grateful to all of my group members, who are always activating me when I met with difficulties and always let me come up with new ideas on the project side. The last thing about the reflection is that I need to do more chanllenging task. The amount of my load is not too much and also not too chanllenging so I have known that I should make my own request to increase the difficulty of work.

3. Acknowledgment

Thank Fredrik for being our team leader and Divy as our co-leader. Thank other group members for helping me with the report and algorithm, especially Reece. Thank Timmy for being our supervisior and giving us useful advice weekly.

References

Super transbot. URL https://github.com/sszxc/Super_Transbot. Online, accessed 28 March 2021.

Webots tutorials. URL https://cyberbotics.com/doc/guide/tutorials. Online, accessed 28 March 2021.