Towards Autonomous Navigation and Assembly using Visual Recognition

Will Curran, Daniel J. Leal, Jesus D. Leal, Brandon Martinez, Daniela Puente, Leonel Pena, Marcos Pena, Bryan Rodriguez, Martin Rivero, Eli Zamora, Read Sandstrom, Saurabh Mishra, Nancy M. Amato

Parasol Lab, Department of Computer Science and Engineering, Texas A&M University,
College Station, TX, USA.
(willecurran, daniel.leal04, david.leal63, dannypuente15, leopena117, mpena16123 bryanrodz15) @gmail.com
martin_rivero@outlook.com (brandon.martinez02, eli.zamora01)@utrgv.edu
(jdenny, readamus, amato)@cse.tamu.edu

***Abstract*— Motion planning is the problem of finding a collision-free path for a robot from a start to goal configuration. State of the art techniques rely on sampling- based planning which samples and connects configurations until a valid path is found. Although most of these methods have been kept strictly computational, we implemented motion planning to physical robots to demonstrate their efficiency in real world scenarios. In "A Machine That Learns" by W. Grey Walter, he used light in his experiments with similar robots as a means to get the machine to perform a certain task. In this work we explore a new method for physical robots, allowing them to push boxes from an initial to a final position. Our novel method allows the robot to read pre-coded markers, which contain a set of data specifying the location in the global map there are supposed to be in. In our experimental results, we saw that the final position the robot left the box in was well within an acceptable margin of error. Additionally, the robot was completing the task ninety-six percent of the time.**

I. Introduction

The world we live in is constantly changing, we either construct or destruct buildings. We began using machines to assist in the construction or demolition projects as they proved to be extremely helpful and much more efficient than one man. With this in mind, we wanted to create something similar to possibly one day in the future affect the construction industry. Having robots do the heavy lifting greatly reduces the chances of having human casualties. Before this can happen, a great amount of work has to be put towards making this possible. One key aspect of making this construction robot efficient and reliable is to make sure it follows instructions and executes them precisely.

Having precise machines involves a great deal of planning prior to having the robot do what it is told. These vehicles simplify the problem of navigation by restricting their paths to predertermined routes. [1] This is why we use motion planning to assist the process of finding the best and most accurate path for the robot. The state of the art solution to this problem is sampling-based planning, which generates a roadmap of valid configurations and then extracts the best path from the start to end. Most planners are designed for the exclusive use of robots with no motion contraints, but a monotonal forward and turn speed. Additionally, these robots only move in straight lines which makes the computational planning less complicated.

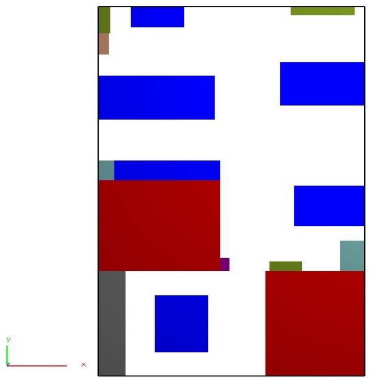

Fig. 1. Virtual Environment of lab

In this work, we were tasked with coming up with a construction site system of robots that were tasked to push boxes from a depot to a construction site. The machines also switched out whenever their battery was running low. Although we were unable to reach the ultimate goal of the project, our results were in fact excellent and we plan to pursue any future work possible in this field of research.

II. Related Work

In this section, we address related work that is most relevant with our project.

A. Localization

Localization is the problem of determining the position of a mobile robot from sensor data. [2] In our case, we used the laptop's camera as a means to get localized. It was extremely important for us to be able to get localized so that communication between the robot and the virtual planner collaborated together. Initially we were localizing the robot globally, which was causing us problems. Instead, we switched to localizing the robot locally which was much more accurate to where it actually was. It localizes itself by looking at the markers it is around, which have their unique location in the environment. The robot then calculates where

it is. Much work has gone into good localization algorithms as the ones described in *Active Markov localization for mobile robots.*

B. Robot Navigation

There have been many Autonomous Robot Navigation papers that describe so much work that has been put in towards making robot navigation a necessary implementation. Some of the robots already being used to test autonomous navigation have been designed for use in a structured office or factory environments. This is a significant restriction, as compared to have one designed for outdoor use, nonetheless it still allows for many potential applications. Robot navigation in our experiment was a large portion of the project. Our job was to get the robot to move from one place to another, without proper navigation methods the robot would have never completed the task we wanted it to complete.

Fig. 2. Some of the markers that were set throughout the environment

III. Main method

Our method consists of using the law of cosines and basic trigonometry to find the h, y and m values. This allows the robot to accurately compute where it should move to be centered with the object.

- Able to successfully acuire data from the markers r.g., position of marker, and distance and angle to the robot
- Robot can successfully push a box forward a given distance with a margin of error under 5%
- Robot can accurately compute the distance between two markers and wall
- Robot uses trigonometric functions to center itself in front of the box, facing towards it

When the robot reads a marker, it gets information about the marker e.g., x position, y position, and marker orientation. This helps the robot localize itself in its environment. [3] Once the robot reads the information, the marker has precoded instructions that the robot will follow. We are able to push a box any given distance with precision.

IV. Experimental Analysis

We wanted to see how our results varied with several different factors in the picture. We had two sets of experiments,

Algorithm 1 Algorithm

Input: Environment env, $dist$
Output: Box Pushing $mvmt$

```
map ← markerInfo
rotate 180
map ← markerInfo
return pushDist and dist
LocateboxMarkers
calculate m, h, &Y
Rotate Y to align
Move to center
Rotate 90
Push box dist
```

one that included a plow on the robot, while the other did not have one. Additionally we also changed variables that would've shown if the plow proved to be more effective.

A. Setup

The methods were all implemented in a C++ motion planning library developed in the Parasol Lab at Texas A&M University. It uses a distributed graph data structure from the Standard Template Adaptive Parallel Library (STAPL), a C++ library designed for parallel computing. All experiments were run on Dell Optiplex 780 computers running Cent OS with Intel Core 2 Quad CPU 2.83 GHz processors with the GNU gcc compiler version 4.7.

Additionally, all experiments were conducted in Parasol Laboratory with boxes made from recycled cardboard. Create connection with desktops were made possible by ASUS Eee PC computers running Fedora 23 with Intel Atom CPU 1.60 GHz processors.

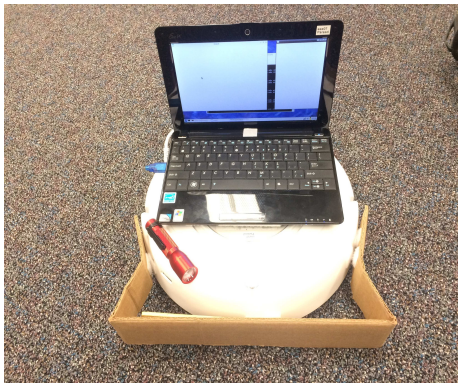

Fig. 3. Create robot with mini laptop mounted on top

For our experiments we set a total of 158 markers throughout the environment and boxes. There was a marker put in every corner of our lab, and two on each side of every box. Each marker had a unique marker ID, and they were individually measured from the origin to get their x & y coordinates and marker orientation. Their information was inserted into a C++ map for easy retrieval of information and instructions. The robot had an ASUS Eee PC mounted on top which was in charge of seeing the markers and sending back

their information. It also assisted the robot in guiding it in the environment. We finally created a virtual environment as a representation of our lab where the planning and experiments took place.

The virtual environment we created is shown in Figure 1.

- 2D Environment (Figure 1):
  - Red boxes are offices
  - Blue boxes are tables and desks
  - Grey box is a counter
  - Green boxes are shelves
  - Light blue boxes are drawers
  - Purple Box is our lab's trash can
  - Environment was created to scale of actual lab

B. Results

For our experiments we tested how well the robot was able to center itself with a given tolerance level. The tolerance level being the maximum error percentage allowed. We calculated this by calculating where the robot should be and where it actually is. This in turn gave us the &Error. We found that the best performing tolerance level was 0.25 because it balanced hardware and software error.

We were also tasked with gettint the robot to dock by itself whenever it was running on low battery. We had a docking device that emitted three different infrared signals, we were able to decifer which was in the left, center and right. Depending on where the robot was, it would move accordingly to the center of the docking device, and move forward until it docked itself. We were glad this system of docking had proper functionality.

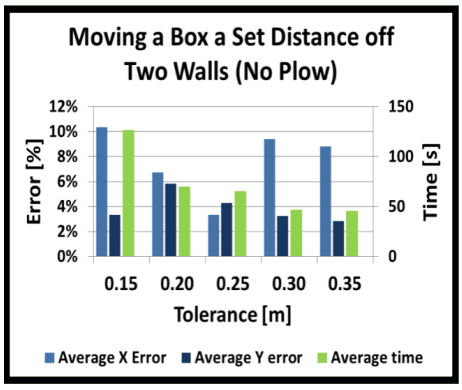

Fig. 4. Shows results of tests with no plow

Additionally we ran 130 iterations in total comparing the robot with the plow against itself without one. There were 10 iterations for each of the 0.15 & 0.30 tolerance levels and 15 for each other tolerance level. We expected to see that the lower the tolerance the less &Error we would get, which seemed only reasonable. What we saw with our results was that the less strict the program got the better results we got. The X error descreased on the robot with the plow the less strict it got. There was an unexpected inverse relationship between the tolerance and error. Additionally, we saw several problems arise for each tolerance level. For the levels more on the strict side we saw multiple hardware issues arise, whereas in the relaxed level we saw that the planner compromised accuracy to save time. The optimal level was the best because it found the balance between being fast and not compromising accuracy.

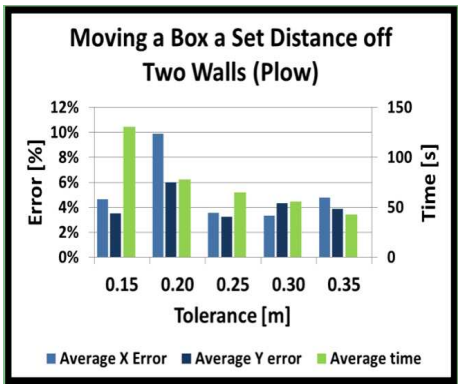

Fig. 5. Shows results of tests with a plow

V. Discussion

Our experiment demonstrates that a robot with a plow increases the precision of the box placement and decreasese the time taken to complete the task. The findings clearly suggest that a plow is beneficial for a robot to execute the task precisely. Since we just scratched the surface of this project's true potential, any further implementation of our algorithm in more complex environments will be left to future work.

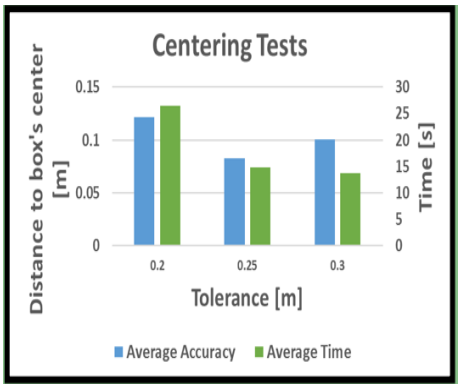

Fig. 6. Shows results of centering tests

VI. Conclusion

In this paper, we introduced a method for robots to assist in manipulating an environment e.g., pushing boxes from a start to a goal. We were able to allow the robot to localize itself, push boxes and dock so that it can recharge itself whenever it is running low on battery. We also found that performance was improved by adding a plow and refining the tolerance level.

VII. Acknowledgements

We would like to thank all the people who helped us and funded our research for the summer, especially the DREU program, which is the reason all of this was possible. Also, thanks to everyone who mentored us and taught us more about programming and working as a team: Jory Denny, Read Sandstrom, Saurabh Mishra, Mukulika Ghosh, Irving Solis, and everyone who helped with our research poster. Supported in part by an NSF Graduate Research Fellowship,

This research was supported by the Department of Computer Science and Engineering of Texas A&M University College Station.

References

[1] R. C. arkin and R. R. Murphy. Autonomous navigation in a manufacturing environment. *IEEE Transactions on Robotics and Automation*, 6(4):445–454, 1990.

[2] D. Fox, W. Burgard, and S. Thrun. Active markov localization for mobile robots. *Robotics and Autonomous Systems*, 25(3):195–207, 1998.

[3] L. Ross and D. K. Bradshaw. Fiducial marker navigation for mobile robots, 2012.