Huan-Yang Chang
Team G: Excalibr
Teammates:
Man-ning Chen, Yiqing Cai, Sambuddha Sarkar, Siddharth Raina

EXCALIBR

ILR08
03/02 2017

Individual Progress

1.Path Optimization

First, I wanted to distinguish what Cece was working on and what I was done. Cece's part was to optimize the number of trigger positions that the robot should traverse basing on the evaluation the scale and distribution and coverage. The main purpose of Cece's part was to make sure the quality of calibration quality. What I did here was to optimize the path length of traveling all points that Cece provided.

For finding the shortest length when traversing the fixed point set is so called the Travelling Salesman Problem (TSP). This problem had been proofed as an NP-hard problem which means that the cost of finding the shortest path in my situation (hundred of points) would not be practical for us. Thence there are several algorithms had been used to find the suboptimal solution. After searching, I chose the Nearest Neighbor and genetic algorithm to test and compared the result. The genetic algorithm was using the Matlab package[1] and the nearest neighbor method was made from scratch. The nearest neighbor method would connect the nearest one that didn't connect yet when every time it try to connect the next one. By that implementation, it was also the greedy algorithm which means choosing the locally optimal choice at each stage. On the other side, the genetic algorithm on TSP would initialize the possible route and choose the better one to conduct crossover method to generate the spring (part of route combination) and then add the random swap on the mature route elements and repeat the process. This method was inspired by the process of natural selection and it worked well to solve the complex problem[2].

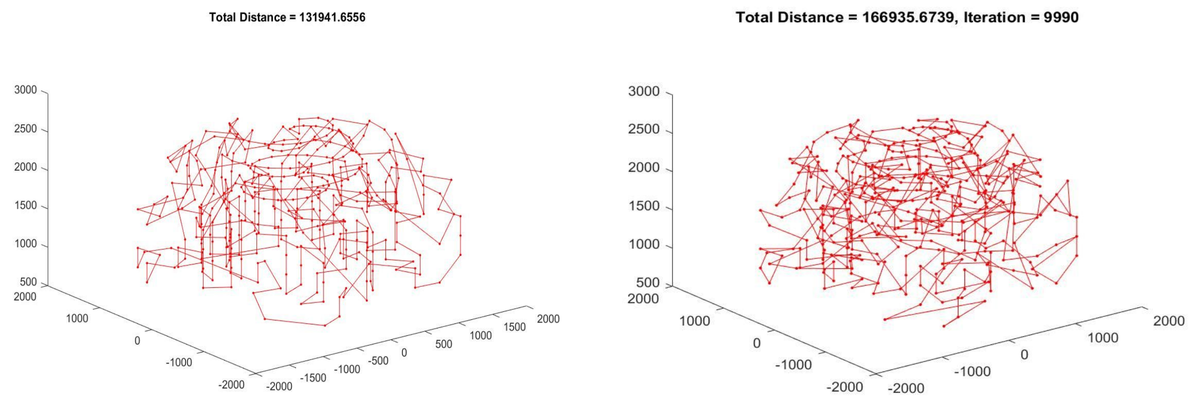


Fig 1.The result of nearest neighbor and genetic algorithm, 47% and 34% improvement respectively.

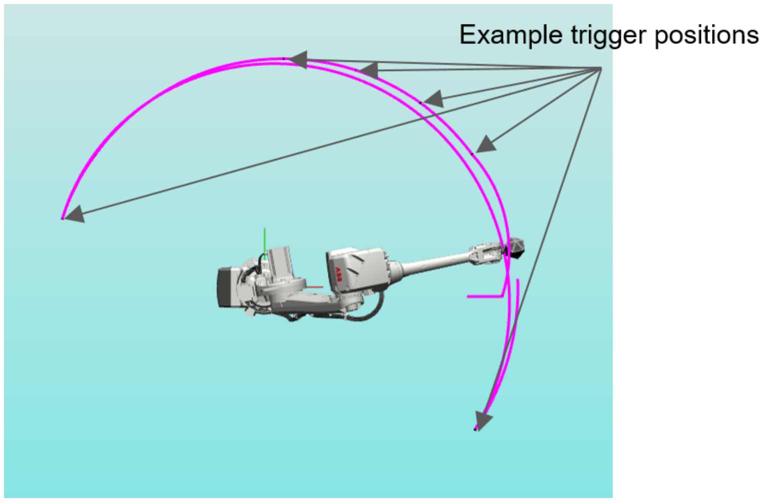
[1] http://www.theprojectspot.com/tutorial-post/applying-a-genetic-algorithm-to-the-travelling-salesman-problem/5

[2] https://www.mathworks.com/search/site_search.html?c%5B%5D=entire_site&q=Travelling+salesman&page=2

In the current 420 point path, the nearest neighbor was working better than the genetic algorithm (Fig 1). But due to there was an absolute advantage of the nearest neighbor than genetic and the current points was not the final points set, hence I would not get rid of genetic algorithm right now. I would keep both of them and until I got the final points set from Cece. Then we could conduct the test to decide to use which one.

2. Signal generation

For the signal generation, I used the ABB's pulse function to generate the output signal. The basic process to was to stop the arm and then trigger the output signal to prevent the motion blur (Fig 2).



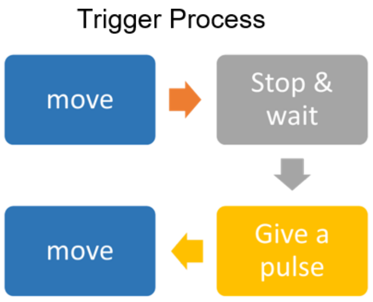


Figure 2. The example trigger position and the trigger process

Challenges

The challenge of this progress review was to automatically generate RAPID code. The original process, we have to manually type the points into the Robotstudio, it wasted a lot of time on just importing the points. Hence I thought writing a Matlab script for automatically transfer the points set directly to workable RAPID code would be really helpful. The problem was that we have understood the syntax of RAPID code and then tried to use the Matlab script to create one. I first checked the exist RAPID code in the Robotstudio to understand basic syntax. Then I found that the RAPID code was quite like the G code, hence the pattern and the usage was very straightforward which also means that using the Matlab to do a script is workable. Finally, I refer the online source and finished the script. And the script was work well and I could get the valid RAPID code immediately, which could help me to increase the speed of testing.

Teamwork

Work	Work is done from last PR	Name
Sensor noise	Sensor noise data collection	Sid
Color calibration	Test color mapping function with linear model	Mandy
Robot simulation and control	Path optimization and finish the simulation part of robot arm.	Peter
Path planning	Camera model evaluation function : scale and distribution	Cece
3D Scene generation	Blender Pipeline & Target modeling	Sam

For the sensor noise, the data collection of two cameras was finished, the data would be analyzed by Sid to calibrated the four type of sensor noise. Mandy would start to use the polynomial model to test the color mapping function. The robot simulation part is finished the required the function is achieved. For the path planning function, Cece finished the image evaluation function, hence the next step would be to generate the optimized point set. Sam was working on the target modeling, which needs to align the face normal and direction between the virtual object and the real one.

Future Plans

In the Robot simulation and Control part, The simulation part is almost finished. Hence, the next step would be on transfer the simulation result to the real robot. Due to controlling ABB robot arm is dangerous and might causing the collision, hence the basic manual control and system familiarization were necessary. Besides, in the first beginning, we set the safe work region to prevent the collision basing on CAD model, so we need to validate it by the real robot arm. And for signal generation, we have to validate that the trigger position is the designed position.

In summary, the future plan of Robot simulation and control will be

1.Manual / program control practice

2.Safe workspace validation

3.Signal generation validation

Ps. Due to the confidential issue of the Oculus Dome, I couldn't take picture of it. I think that would be a problem to show our progress of the real robot arm and further integration part, I would try to depict them using the simulation model.