UNIVERSITÀ DEGLI STUDI
DI GENOVA

RESEARCH TRACK 2

STATISTICAL ANALYSIS

ROBOTICS ENGINEERING

SUBMITTED BY

Ammar Iqbal, 5183355, Robotics

SUBMITTED TO

Prof. Carmine Recchiuto

Abstract

This report is about the statistically analysis to determine the superiority between two distinct algorithms (my algorithm and professor's algorithm) in accomplishing a specific task. The specific task in this case is a simulation of a robot in a pygame environment with the objective of completing laps through the designed track (circuit), while occasionally moving silver tokens through its lifting mechanism to a spot behind it, as it makes its way through the circuit. For this analysis, Jupyter Notebook is used to compute the statistical analysis of the behaviour of two different algorithms for the first assignment of Research Track I based on Python Robotics Simulator.

Índice

1. Introduction

1.1. Objective

This analysis aims at comparing and testing the behaviour of the robot considering two different implementations of the code:

My code : My own algorithm turned in at the end of the project and published on GitHub repository: https://github.com/AmmarIqbal48/RT1assignment1

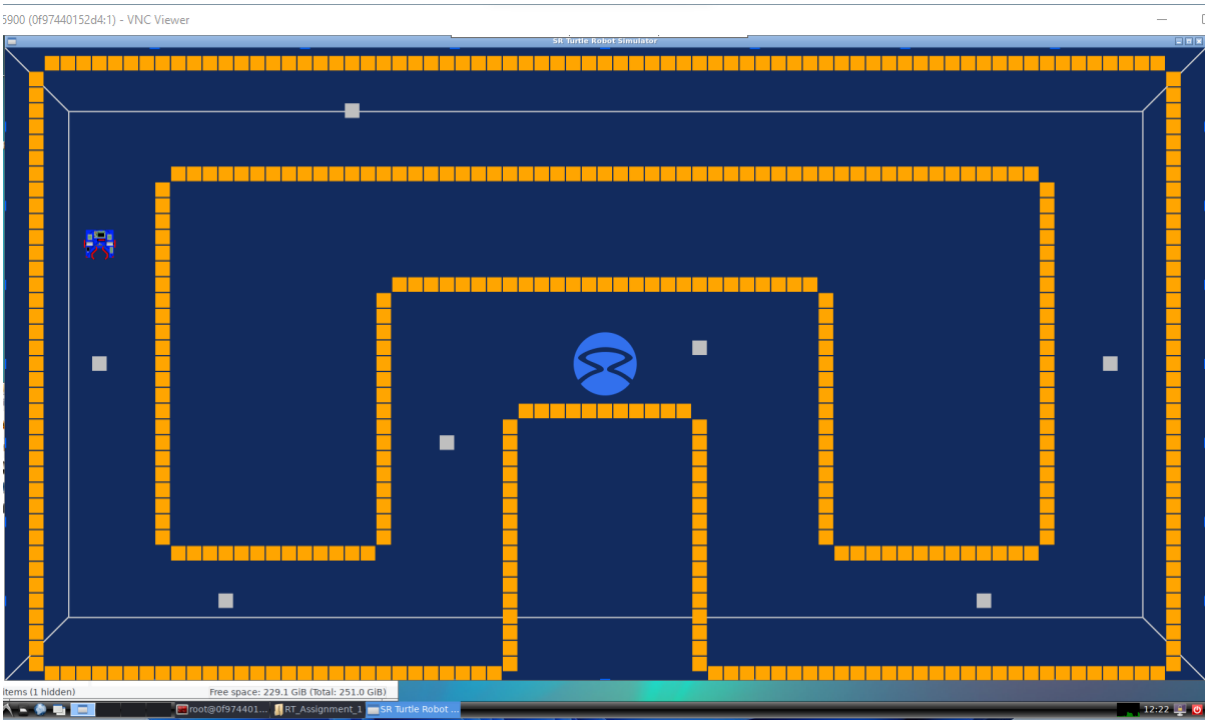
Professor code : The Professor algorithm cloned from the following GitHub repository: CarmineD8/pythonsimulatoratrt2(github.com)

1.2. Robot's Environment

This is a simple, portable robot simulator developed by Student Robotics. Some of the arenas and the assignment has been modified for the Research Track 1 course. In this simulator the robot will spawn inside of an arena composed of squared tokens of two different colors Silver token and Golden token.

1)Golden tokens: Walls of the simulator are made by Golden token.

2)Silver tokens: Silver tokens are randomly placed in the simulator which robot has to collect.



2. Design and Methodology

Given the python environment as shown in figure 1, the two algorithms – One given by the professor, and one from a student (Which I will refer to as Professor Code and My Code respectively) - compared pass instructions to the robot in the simulated environment to move thought the circuit while grabbing and dropping silver tokens and avoiding golden tokens. A clear indication of performance – considering all environment variables remain constant (token positions and robot starting point) will be the time factor. A measure of the time between each silver token grabbing and releasing events can be measured, as they distinctively define the amount of distance travelled due to their being equally spaced within the environment, and the efficiency of task execution, and overall time taken to lap the circuit. For this experiment, we will consider the results with a level of significant of and above 5 %.

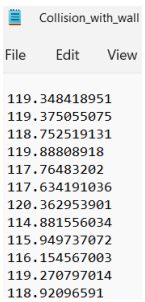
3. Implementation

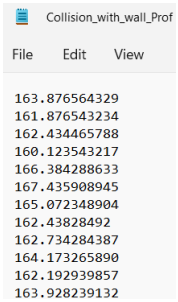
To evaluate some differences between the two algorithms, the following parameters are taken into consideration: (i) Robot's change in direction (ii) Time taken to complete the circuit (iii) Number of time simulation failed

3.1. Data Preprocessing

3.1.1. Data for the first analysis Collision with wall

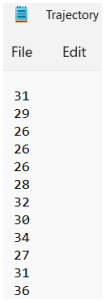
To obtain data for the first analysis parameter, main objective is to focus on when the robot is close to a wall. Moreover, I modified both the codes where the lines manage the change of direction close to a wall (therefore close to the golden tokens), simply by adding a + 1 each time the running program goes in that part of the code.

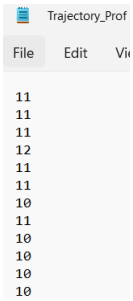




3.1.2. Data for the second analysis Trajectory

To obtain data for the second analysis parameter, main motive is to monitor the time as soon as the running program enters the robot movement part, which is usually the main() function. The code is set to take the end time when the robot finishes to grab the 8 the silver token (because I added one). So as you can think it is not a real circuit, but as far as the final token is not randomly placed, it makes the time a good parameter to be analyzed.



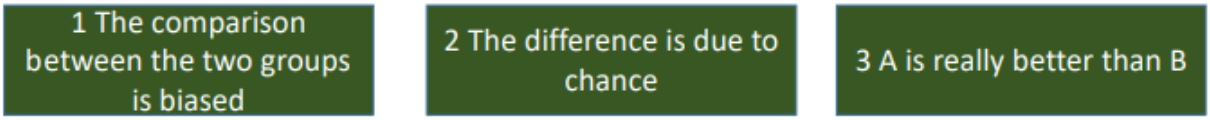


3.1.3. Additional Tokens

Considering the two algorithms (My Code and Professor Code), followed the path .../sr/robot there was a file called sunny_side_up_arena.py. This file contains the creation of the map, and even the position and the number of tokens (7) in the arena. I added one more to see if it was still working and I changed the position of all others. The position is set to make the token "slideïn the direction of the tunnel, in order to not set silver tokens too close to the walls.

3.1.4. Hypothesis

There are three hypotheses to be considered:



Hypothesis 1:

Hypothesis 1 is due to a bad choice of participants made by the experimenter, who selected a sample that is not representative of the population under analysis.

• This is referred to as a biased sample. Suppose that we want to draw conclusions about the whole population of «any Italian students», and we selected a sample entirely composed of «evil» Robotics students, typically making any possible effort to find bugs in the algorithm.

Hypothesis 2:

Hypothesis 2 is due to random variability. Suppose to make the experiment again, but with a different sample of the same population: Will we not get the same results, since there will be factors (known and unknown) that are due to chance.

• To isolate the effect of chance, we shall refer to statistical tests to check the significance of results.

Hypothesis 3:

Hypothesis 3 can be accepted onlt after excluding 1 and 2.

There are several ways to compare the algorithms based on their performance by using two hypotheses: Ha (alternative hypothesis) and Ho (null hypothesis). To differentiate one hypothesis from another, it is better to understand which nature of each of them: Ho:The robot from my code and professor code complete the circuit in a particular time, so their mean are more or less equal Ha: The robot from my code and professor code no able to complete the circuit in a particular time, so the mean is not equal.

3.1.5. Observation Data

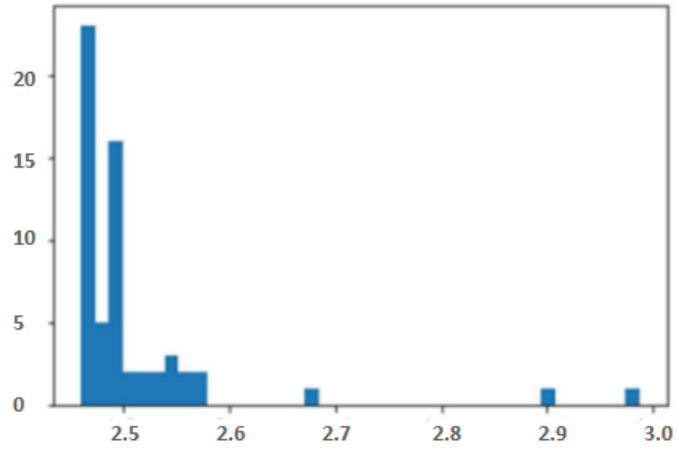
60 observations of each code were monitor with the time(s) i.e time in seconds taken to complete the robot tasks. Each observation was obtained automatically through embedding instructions in the python script to save time data. The following are the first 5 observations of My Code and Professor Code, respectively.

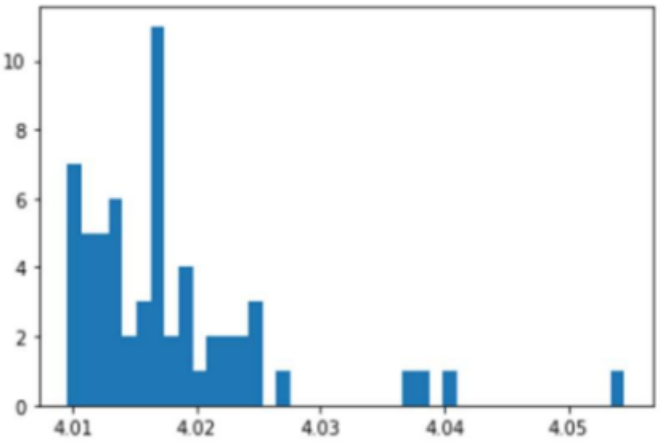
```
collision_with_wall_mean = np.mean(collision_with_wall)
timestamp_mean = np.mean(timestamp)
collision_with_wall_Prof_mean = np.mean(collision_with_wall_Prof)
timestamp_Prof_mean = np.mean(timestamp_Prof)

# Calculating the standard deviation
collision_with_wall_std = np.std(collision_with_wall)
timestamp_std = np.std(timestamp)
collision_with_wall_Prof_std = np.std(collision_with_wall_Prof)
timestamp_Prof_std = np.std(timestamp_Prof)
```

	time(s)
0	2.556254
1	2.514890
2	2.658965
3	2.591450
4	2.538406

	time(s)
0	4.009535
1	4.019174
2	4.054525
3	4.010038
4	4.014200

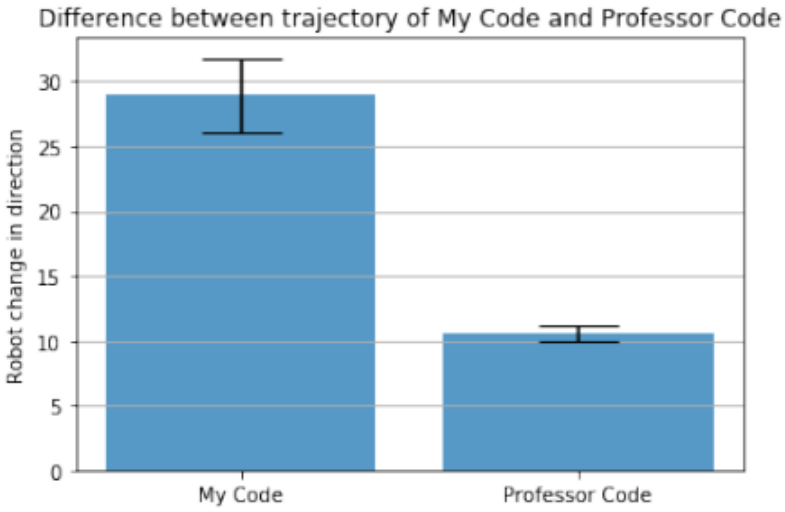
It is immediately apparent that my code has time superiority over professor code, exhibiting lower run times and hence completing the circuit faster.

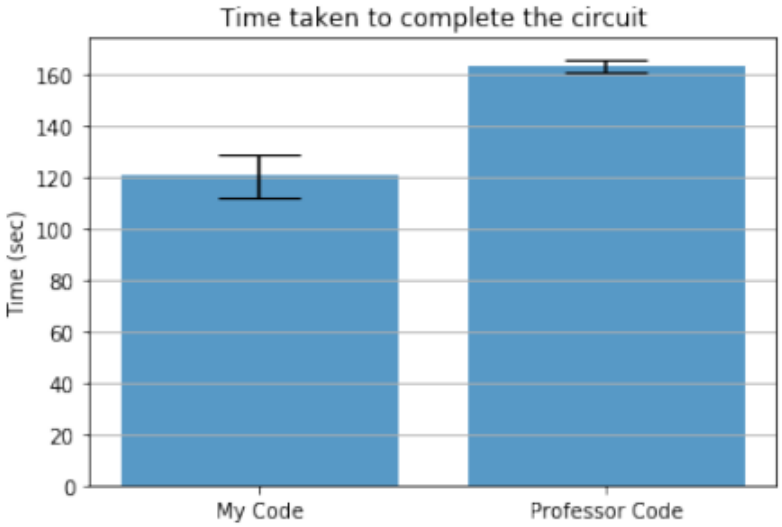


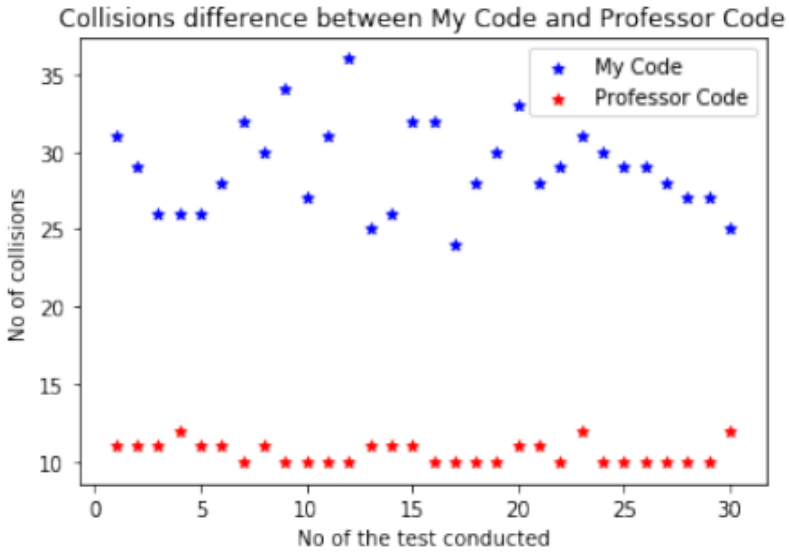


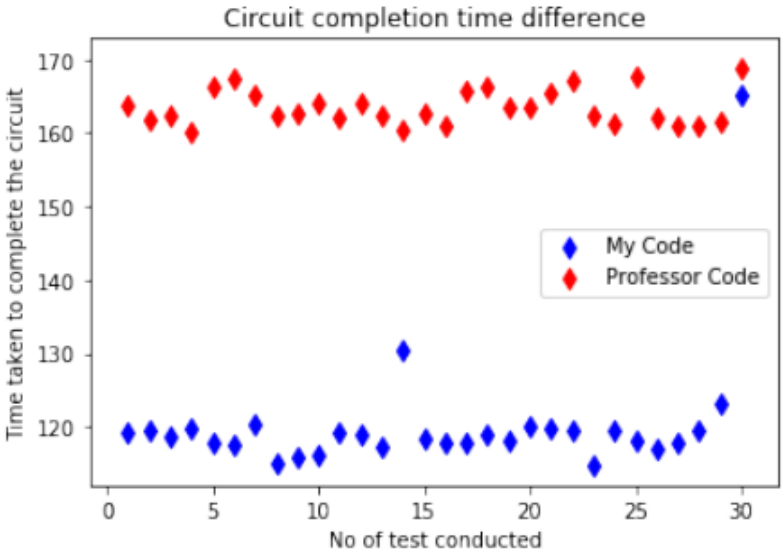
4. Results

This section will describe the experimental results which shows the analysis of the two algorithms i.e. My Code and Professor Code. Figure below shows the trajectory difference between my code and professor code.




Time taken to complete the circuit
Time (sec)
0
20
40
60
80
100
120
140
160
My Code
Professor Code


Collisions difference between My Code and Professor Code
No of collisions
10
15
20
25
30
35
0
5
10
15
20
25
30
No of the test conducted
My Code
Professor Code


Circuit completion time difference
Time taken to complete the circuit
120
130
140
150
160
170
0
5
10
15
20
25
30
No of test conducted
My Code
Professor Code

The figure below shows the Graphing Representation of the collision difference of both the two codes (my code professor code).

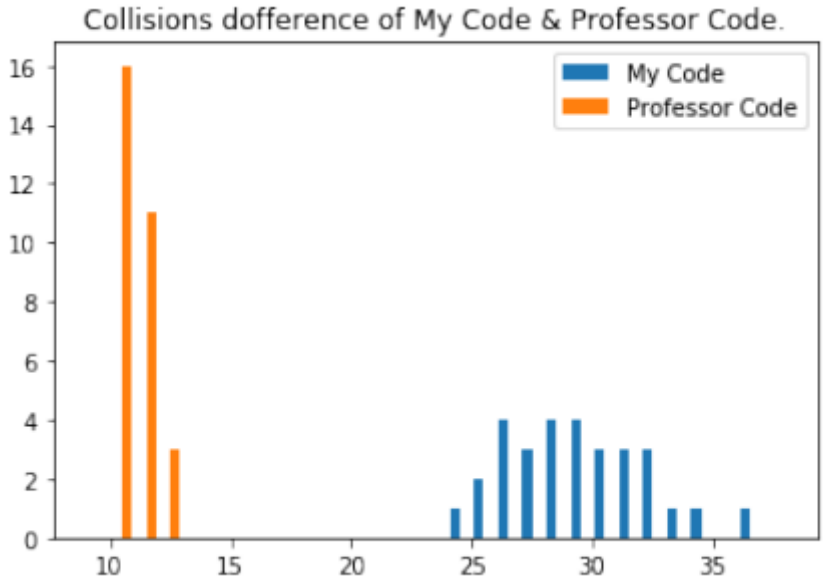
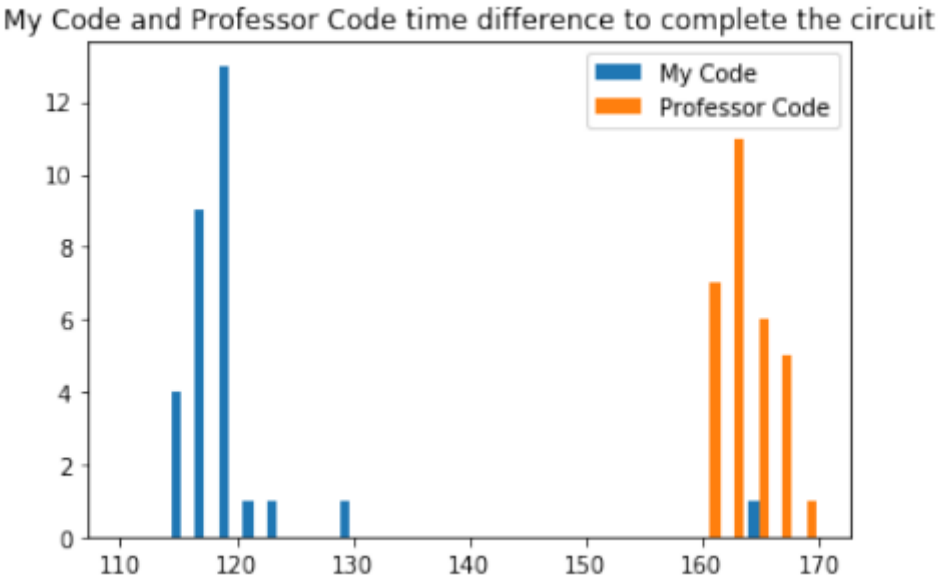


Figure below shows the Graphing Representation of Time taken to complete the circuit by my code professor code.



4.1. Analysis

4.1.1. Chi-Square Test

We compare two percentages obtained in a controlled experiment, to check if the difference is due to chance or not.

We start by formulating the so called null hypothesis, or H0.

• H0: the difference between percentages in the population is 0 (and then results are due to chance)

• Then we implement a statistical test to accept or reject H0

• More complex null hypothesis can be formulated, and the formulation of the hypothesis may depend on many factor. However, we always assume this kind of hypothesis in the following.

The text to accept or reject the null hypothesis depends on the nature of the dependent variable: when comparing two percentages, it is appropriate to use a Chi-square test

4.1.2. T-Test Analysis

T-Test, also known as Student's Test, is based on t-distribution and is considered an appropriate test for judging the significance of a sample mean or for judging the significance of difference between the means of two samples in case of small sample(s) when population variance is not known (in which case we use variance of the sample as an estimate of the population variance). The relevant test statistic, t, is calculated from the sample data and then compared with its probable value based on t-distribution at a specified level of significance for concerning degrees of freedom for accepting or rejecting the null hypothesis (source from https://2021.aulaweb.unige.it/pluginfile.php/422414/mod_resource/content/0/04_Statistics. In my Jupyter Code, two T-test analysis is performed between the two data and expected that they're going to be deeply different. This is a test for the null hypothesis that two independent samples have identical average values. This test assumes that the data have identical variances by default. This is to understand how different they are. The outcomes are:

Analysis I:
Collision_Static = 34.031085935665956
p_Values = 5.05537028942152e-40
Analysis II:
Collision_Static = -25.589230325578303
p_Values = 2.750043113035198e-33

Where the p_value is less than our significance value and with a negative t-value, the null hypothesis will be rejected and the alternative sustained.

5. Conclusion

The main objective of this statistical part of the project is to find particular differences between my code and professor code that "guide" the robot inside the environment. I have taken lots of data, making sure to have the same condition during the acquisition, since I know results are non-deterministic, so I wanted to decrease the inconsistencies due to the performances of the PC (even if there are). In the end I found that there are substantial differences about lap times and distances from golden tokens. In fact my robot keeps an higher distances from the walls; this is very important since many wall-crush can be avoided. My robot also, despite of the number of the silver tokens, is faster than the robot's professor in concluding laps. All these assumptions that I made at the beginning of the project were con validated due to the statistical tests that I have done.