



## RESEARCH TRACK II

STATISTICAL ANALYSIS  
ON

## Python Robotics Simulator

*Ankur Kohli (5160903)*  
*Master of Science Robotics Engineering*

Submitted to  
*Prof. Carmine Tommaso Recchiuto*

Dipartimento di Informatica, Bioingegneria, Robotica e  
Ingegneria dei Sistemi (DIBRIS)  
Universit a degli Studi di Genova (UniGe)

## Preface

The work is based on Statistical Analysis which involves the analysis of two code i.e My Code and Professor Code. This deals with Python Robotics Simulator environment.

## Acknowledgements

I take this opportunity to express my sincere thanks to *Prof. Carmine Tommaso Recchiuto* for his guidance and support. I would also like to express my gratitude towards him for showing confidence in me. It was a privilege to have a great experience working under him in a cordial environment. I am very much thankful to the University of Genoa, for providing me the opportunity to pursue M.Sc in Robotics Engineering in a peaceful environment with ample resources. In the end, I would like to acknowledge my parents, friends. Without their support, this work would not have been possible.

## 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**.*

# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Approach</b>	<b>6</b>
<b>3</b>	<b>Methodology</b>	<b>6</b>
3.1	Data extraction . . . . .	6
3.2	Additional Tokens . . . . .	7
3.3	Hypothesis . . . . .	7
3.4	Observation Data . . . . .	7
<b>4</b>	<b>Analysis Results</b>	<b>8</b>
4.1	T-Test Analysis . . . . .	11
<b>5</b>	<b>Conclusion</b>	<b>12</b>

# 1 Introduction

In this report, the statistical analysis of two implementations (my code and professor code) are going to be studied. The main objective of this analysis is to drive a robot autonomously by grasping the *silver token*, and putting them behind itself, and last but not the least robot should avoids the obstacles, which are *golden tokens*. In Figure 1, the map of the circuit has been presented as well as the robot itself and tokens are shown.

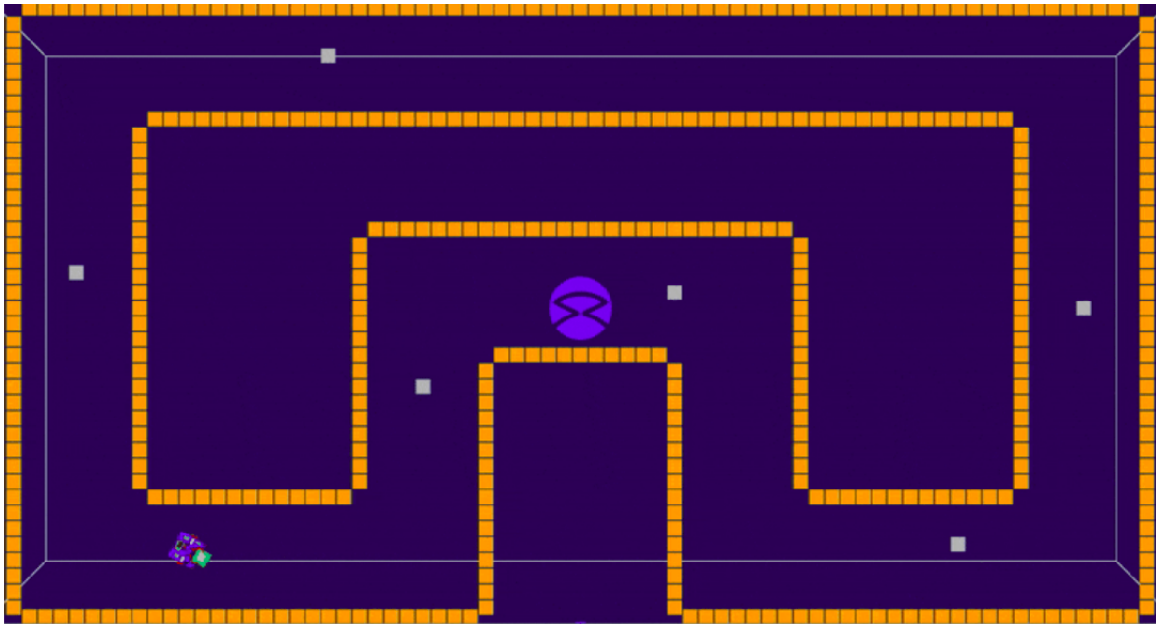


Figure 1: Robot's Environment

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* at the following repository: [https://github.com/ankurkohli007/Research\\_Track\\_I\\_Assignment\\_1.git](https://github.com/ankurkohli007/Research_Track_I_Assignment_1.git)

**Professor code** : The Professor algorithm cloned from the following *GitHub* repository: [CarmineD8/python\\_simulator](https://github.com/CarmineD8/python_simulator) at [rt2 \(github.com\)](https://github.com)

## 2 Approach

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 through 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 Methodology

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 extraction

**(i) Data for the first analysis parameter:** 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.

**(ii) Data for the second analysis parameter:** 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<sup>th</sup> 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.

### 3.2 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 "slide" in the direction of the tunnel, in order to not set silver tokens too close to the walls.

### 3.3 Hypothesis

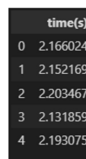
There are several ways to compare the algorithms based on their performance by using two hypotheses:  $H_a$  (*alternative hypothesis*) and  $H_o$  (*null hypothesis*). To differentiate one hypothesis from another, it is better to understand which nature of each of them:

$H_o$ : The robot from my code and professor code complete the circuit in a particular time, so their mean are more or less equal

$H_a$ : The robot from my code and professor code no able to complete the circuit in a particular time, so the mean is not equal.

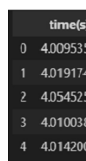
### 3.4 Observation Data

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.



	time(s)
0	2.166024
1	2.152169
2	2.203467
3	2.131859
4	2.193075

Figure 2: My Code Time



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

Figure 3: Professor Code Time

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

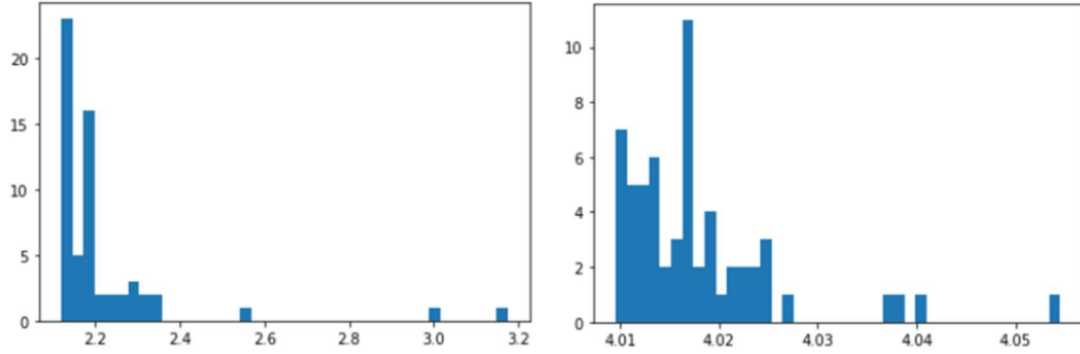


Figure 4: My Code & Professor Code Time Lap

Observing the histogram which shows the distribution of lap times, it is also apparent that my code maintains most observations in the lower timings, and is less sparsely distributed, while professor code distributes more over several bins, and can be described as less precise.

## 4 Analysis 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.

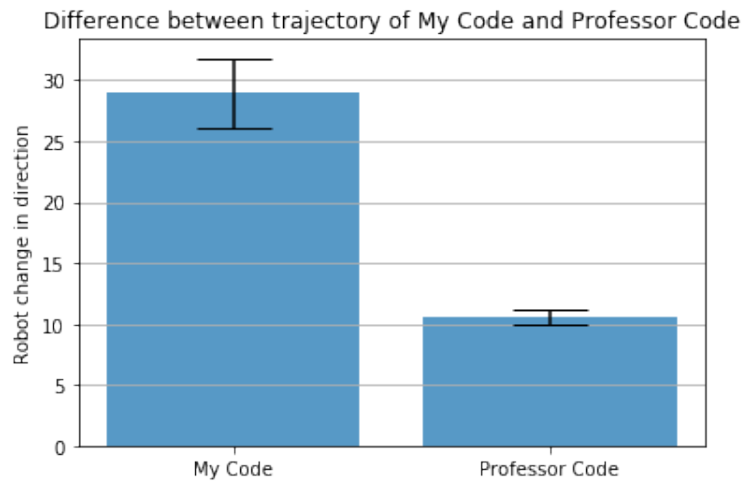


Figure 5: Difference between trajectory of My Code and Professor Code

Figure below shows the time taken to complete the circuit by my code and professor code.



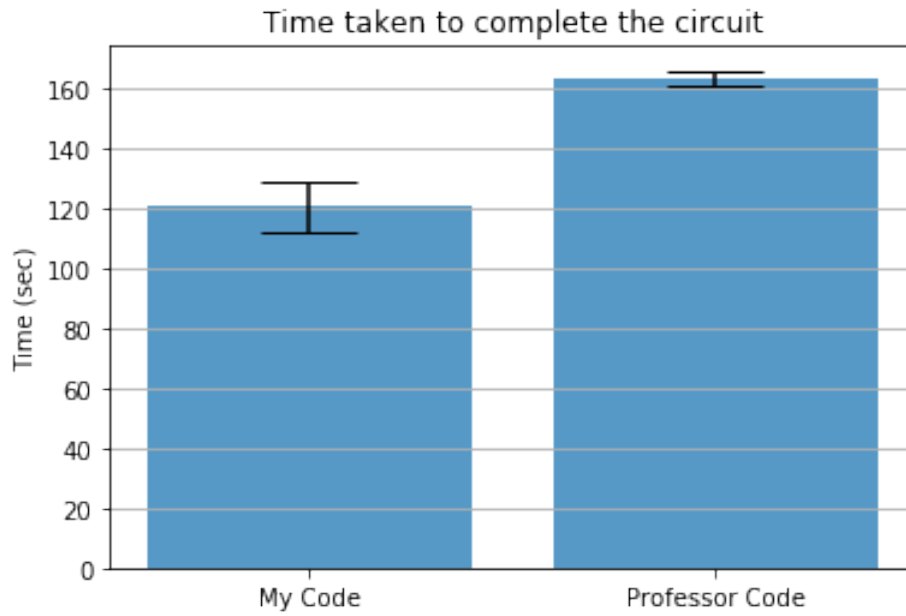


Figure 6: Time taken to complete the circuit

Figure below shows the collision difference between my code and professor code.

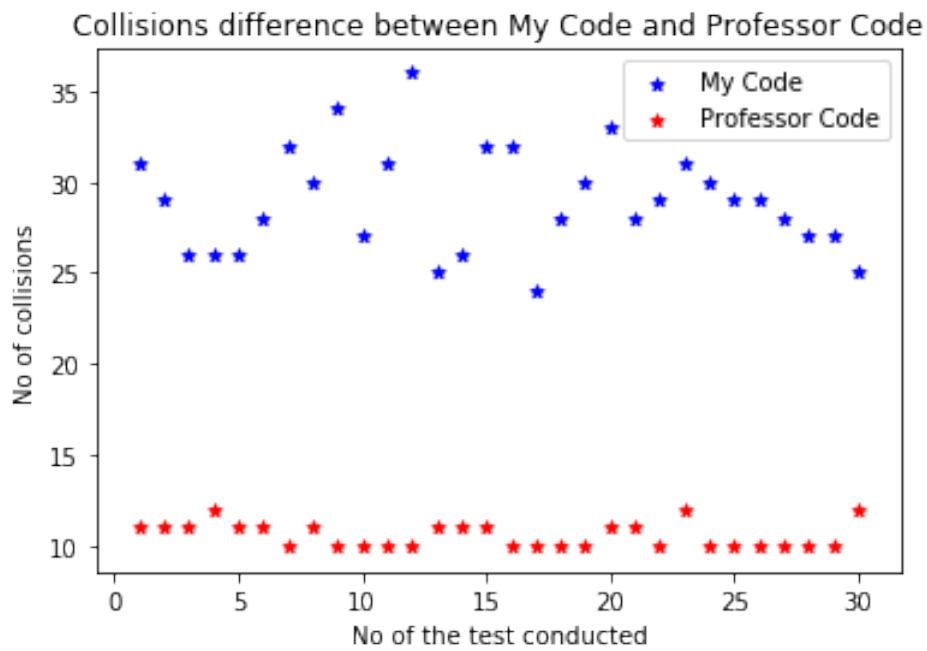


Figure 7: Collisions difference between My Code and Professor Code

Figure below shows the graphical representation of the circuit completion time difference by my code and professor code.

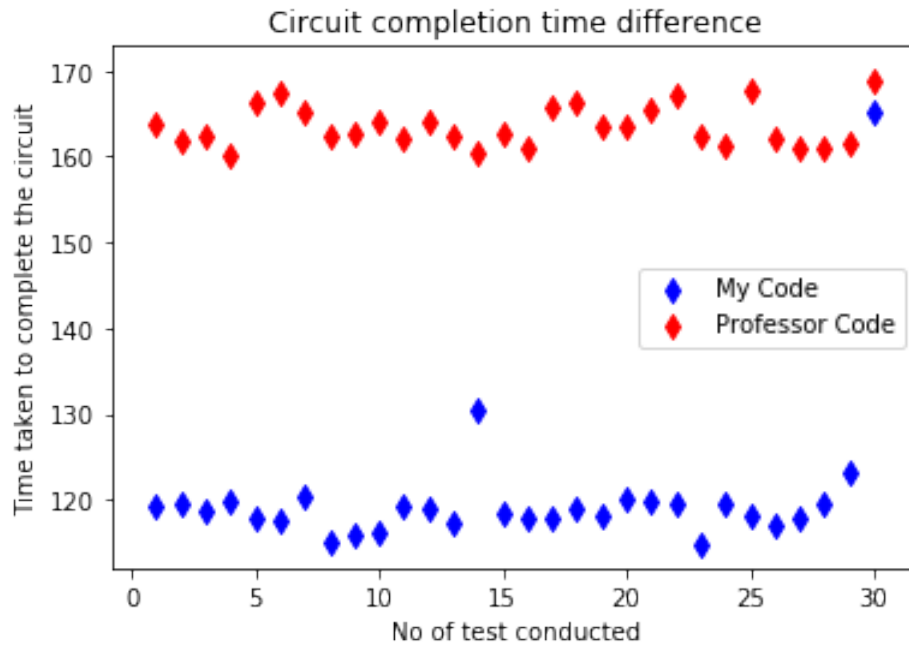


Figure 8: Circuit completion time difference

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

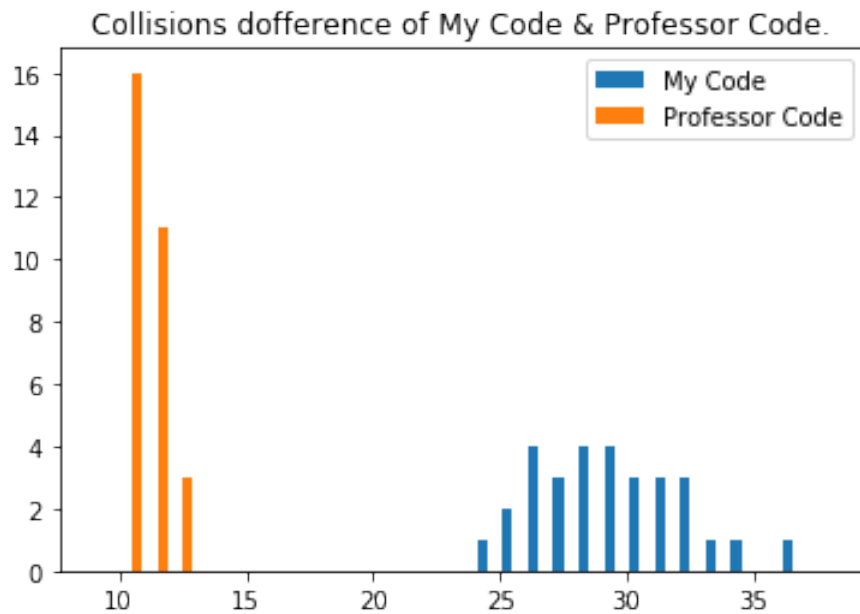


Figure 9: Collisions difference of My Code & Professor Code

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

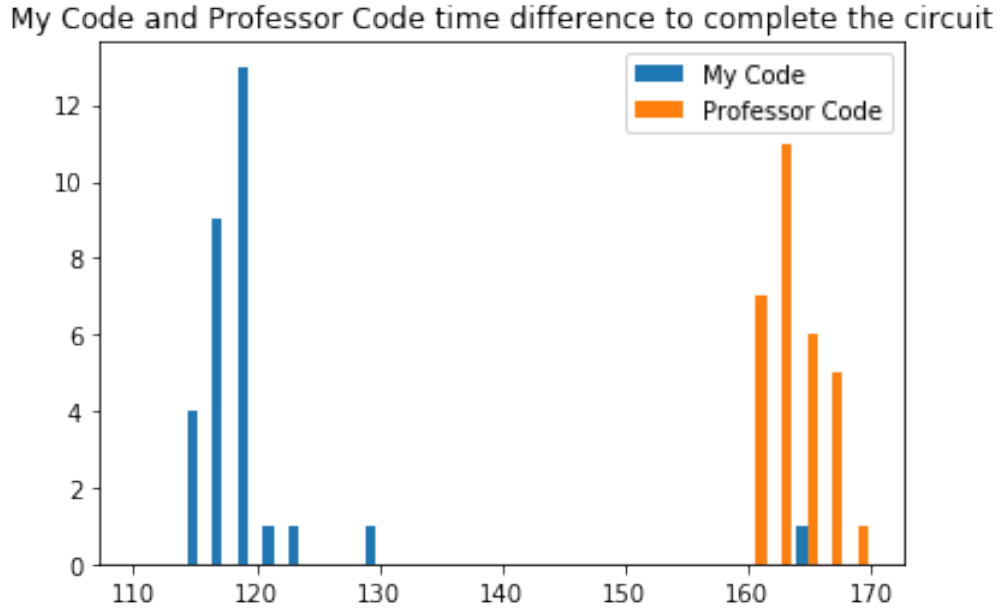


Figure 10: Collisions difference of My Code & Professor Code

For Statistical Analysis and evaluation of our results, the T-test has been selected. Using the SciPy library, the `ttest_rel` method was used to give required results corresponding to a two-tailed test.

#### 4.1 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.pdf](https://2021.aulaweb.unige.it/pluginfile.php/422414/mod_resource/content/0/04_Statistics.pdf)).

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-crash 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.