Research Track II Assignment. Statistical Analysis.

Author: Francesco Pagano

Student ID: S4636579

INTRODUCTION.

The Research Track II Assignment final part was about a statistical analysis about two code performances:

1. My solution of the first assignment from Research Track I course
2. A Professor's solution of the first assignment from Research Track I course.

We were asked to assess which one performs better in the circuit given. Moreover, we were supposed to change the silver tokens position by placing them randomly in the circuit. I considered two scenarios, one with silver tokens standardly placed and one with silver tokens randomly placed. For each of them I applied the **same statistical analysis** with the same rules. Since it would have increased the simulation time, I decided to avoid adding new silver tokens in the circuit, but only to change their position.

These are the two considered scenarios:

Figure 1: First scenario, Silver Token Standardly Placed

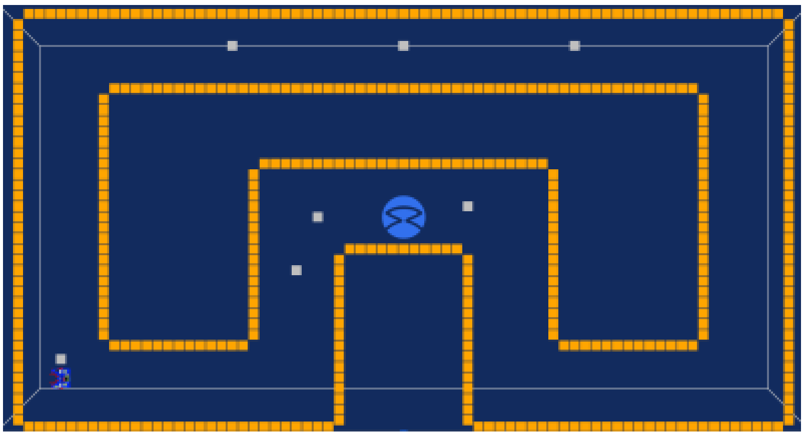


Figure 2: Second scenario, Silver Tokens Randomly Placed

As performance evaluators I considered:

- The **average time** to complete a lap with silver tokens standardly and randomly placed inside the circuit.
- The **frontal distance from gold tokens** every time that the robot decides to make its turn decision method with silver tokens standardly and randomly placed inside the circuit.

My statistical analysis has been performed on the "RT_Test.mlx" Matlab live script that you can find in this repository.

METHODS.

First, I modified the two scripts by adding a few lines of code to leave them as similar as possible to the original ones. These changes make the two scripts to create a log file in which all the data to be analyzed are printed. I made them to create two text files:

- One for storing data about frontal distance from gold tokens.
- One for storing data about the number of completed laps and the time taken for each lap.

I run both the two codes ten times and for each time I waited for the robot to complete at least 5 laps. This procedure has been the same for collecting data with both standardly and randomly placed silver tokens. So, in the end I got:

1. 10 files named "*distances from_gold(i)*" *(i = 0, ..., 9)* that store data about frontal distance from gold tokens for silver tokens standardly placed.
2. 10 files named "*time(i)*" *(i = 0, ..., 9)* that store data about the number of completed laps and the time taken for each lap for silver tokens standardly placed.
3. 10 files named "*distances from_gold_solution(i)*" *(i = 0, ..., 9)* that store data about frontal distance from gold tokens for silver tokens randomly placed.
4. 10 files named "*time_solution(i)*" *(i = 0, ..., 9)* that store data about the number of completed laps and the time taken for each lap for silver tokens randomly placed.

DISTANCE BARPLOTS.

Once I got all the needed data, I imported them into my Matlab workspace by using the `importdata` function and for each scenario I computed the mean distance from gold tokens for each code execution, obtaining the following plots:

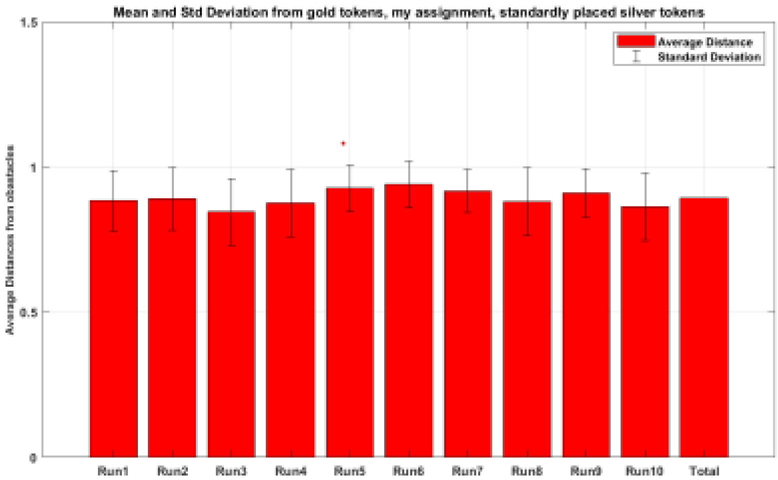


Figure 3: My solution Means and Standard Deviations from Gold Tokens with ***standardly*** *placed Silver Tokens.*

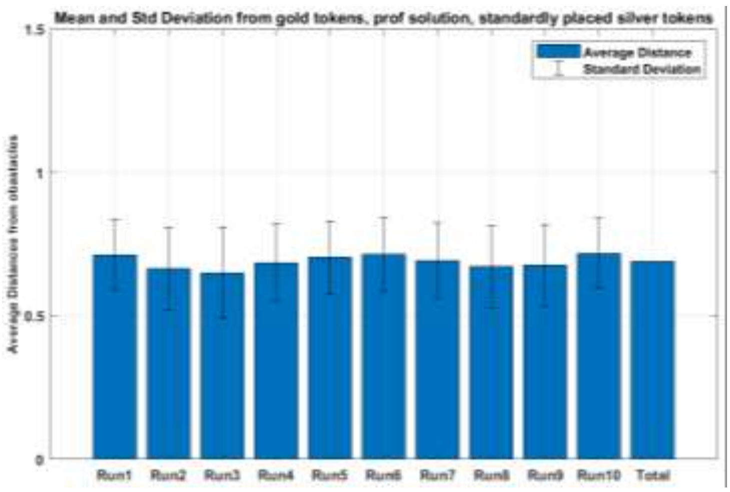


Figure 4: Professor's solution Means and Standard Deviations from Gold Tokens with ***standardly*** *placed Silver Tokens.*

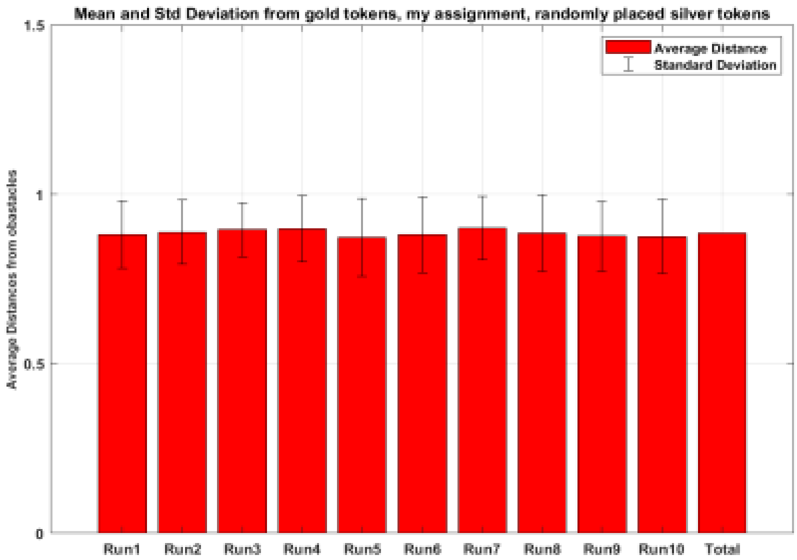


Figure 5: My solution Means and Standard Deviations from Gold Tokens with ***randomly*** *placed Silver Tokens.*

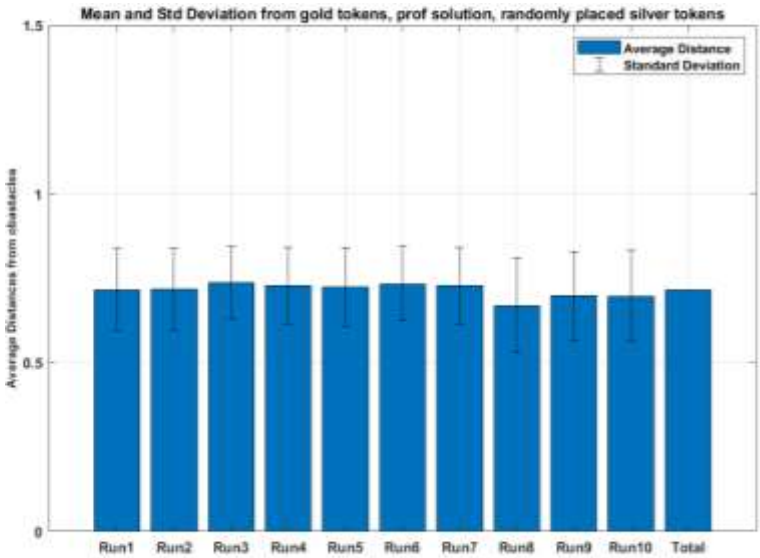


Figure 6: Professor's solution Means and Standard Deviations from Gold Tokens with ***randomly*** *placed Silver Tokens.*

TIME PLOTS.

Moreover, I computed the mean time taken from the robot for each lap for every code execution, obtainig the following plots:

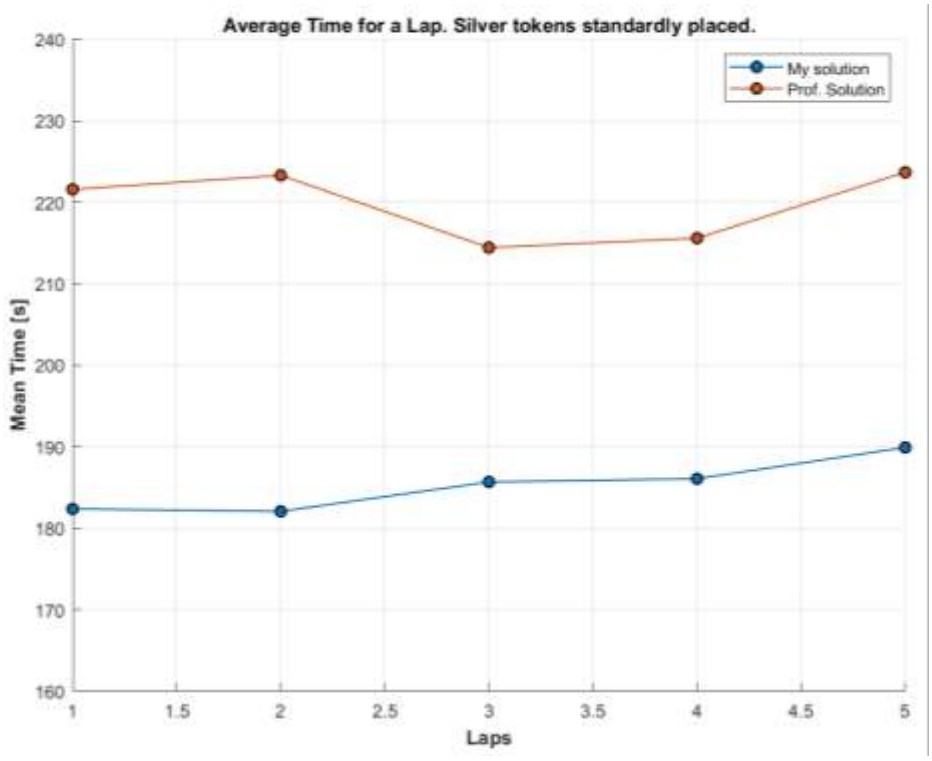


Figure 7: Mean time taken for each lap with ***standardly*** *placed Silver Tokens.*

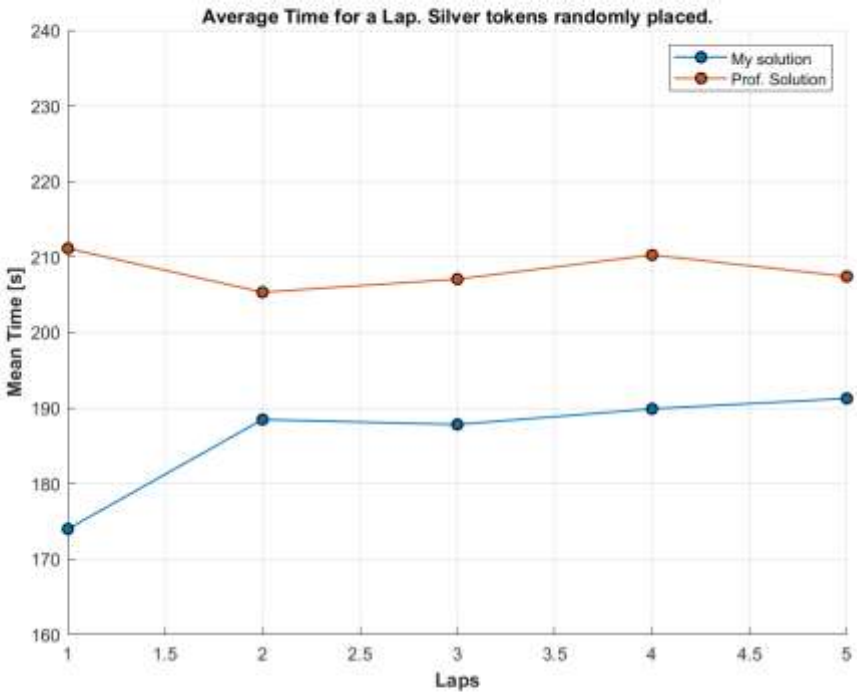
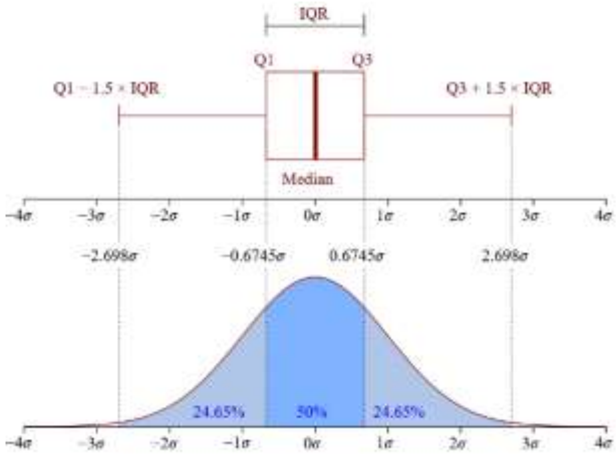


Figure 8: Mean time taken for each lap with ***randomly*** *placed Silver Tokens.*

BOXPLOTS.

To have a better data visualization I also made boxplots. These kind of plots give a nice idea of a sample data distribution. A boxplot is composed of two parts, a box, and two whiskers. The central line inside the box indicates the median (50th percentile), and the bottom and top box edges indicate the 25th and 75th percentiles, respectively. The plot between these two edges is denoted as IQR (Interquartile range). The lowest point on the boxplot (i.e. the boundary of the lower whisker) is the minimum value of the data set and the highest point (i.e. the boundary of the upper whisker) is the maximum value of the data set, excluding outliers that are the green points outside the whiskers that differs significantly from other observations. The plot inside the whiskers corresponds approximately *+/–2.69σ* and *99.3%* coverage of a normal distribution.



The boxplots that I obtained with my data sets are the following:

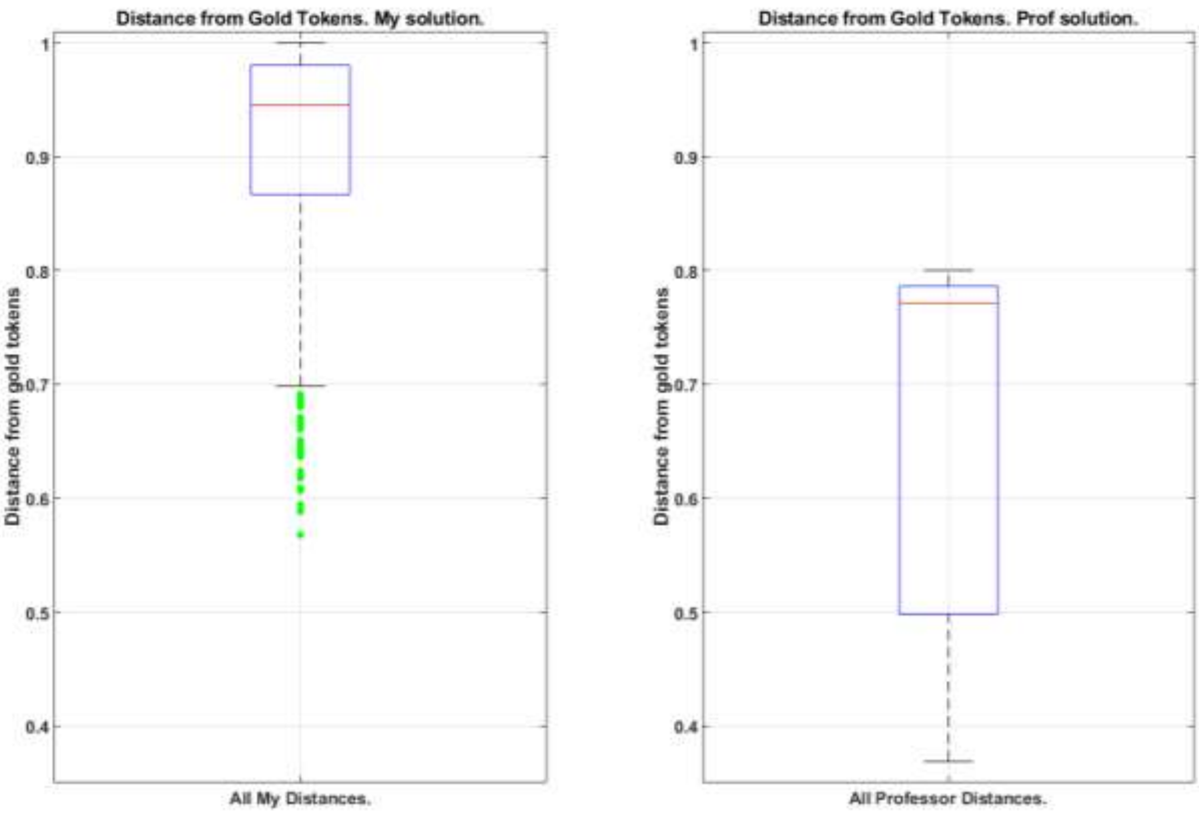


Figure 10: Boxplots about distribution of distances from Gold Tokens with ***standardly*** *placed Silver Tokens.*

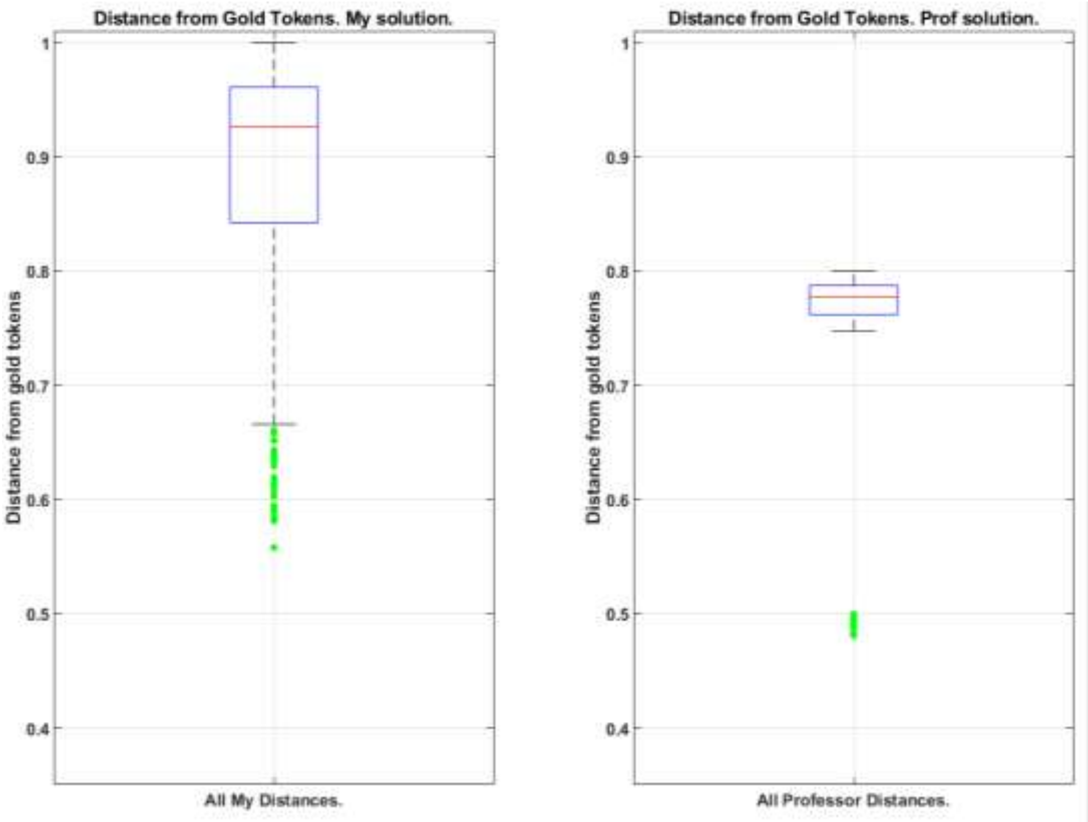


*Figure 11: Boxplots about distribution of distances from Gold Tokens with **randomly** placed Silver Tokens.*

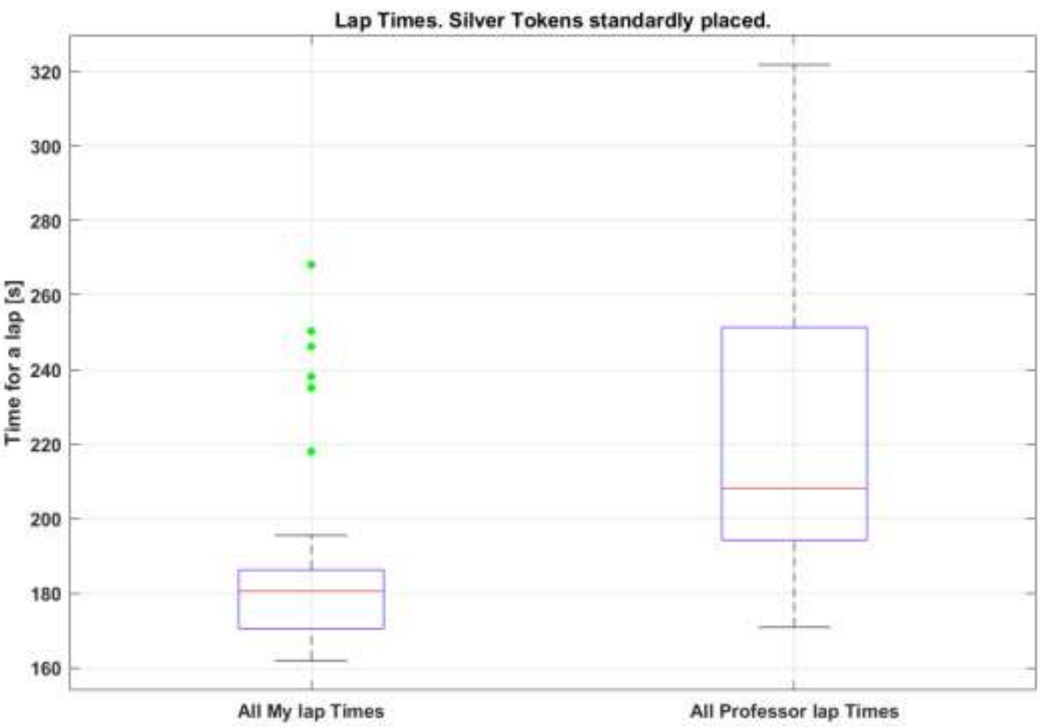


*Figure 12: Boxplots about distribution of time laps with **standardly** placed Silver Tokens.*

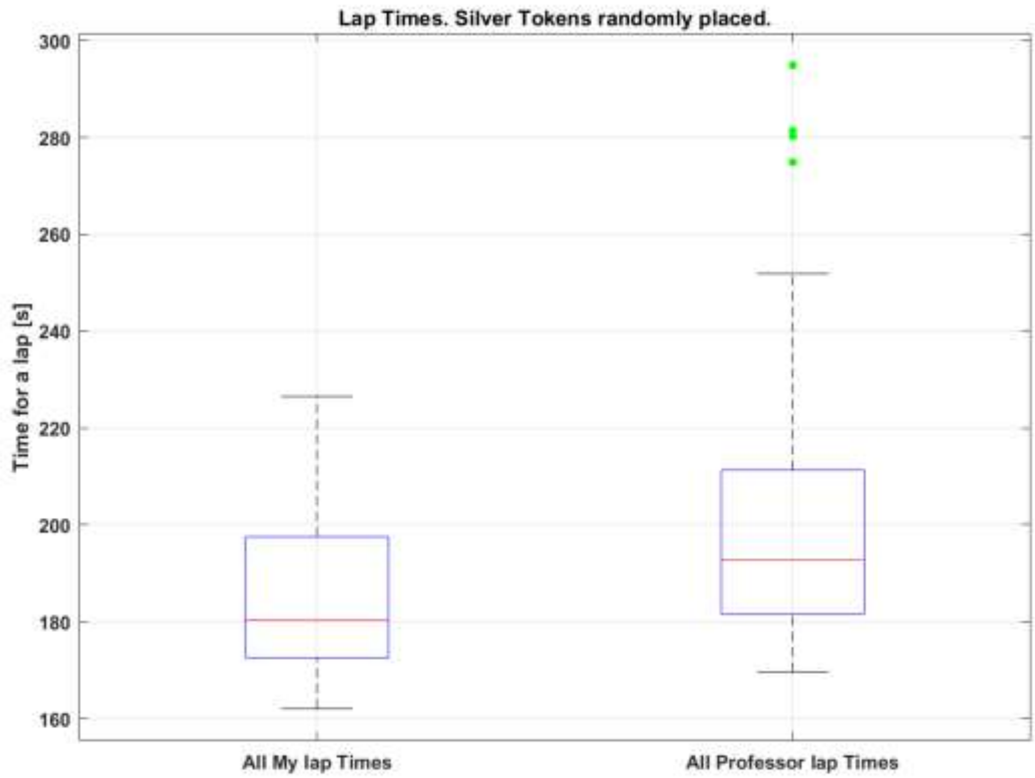


Figure 13: Boxplots about distribution of time laps with ***randomly*** *placed Silver Tokens.*

T-TEST.

All these graphs show how, in both scenarios, my solution keeps a smaller distance to obstacles and takes slightly less time with respect to the professor's solution. However, for a valid statistical proof I needed to perform a one tail T-test between samples of both distances and time laps. Before executing the T-test I performed a Lilliefors test, to kwow if my data samples came from a normal distrubution family. Then, I performed the right tail T-test with the `ttest2` Matlab function passing to this funciton the data-set about the **mean distance and average lap times** coming from every code execution, for both scenarios. The degree of freedom of the tests I performed are 18 (10 + 10 - 2). The null hypotesys is always that the data passed to the function comes from indipendent random samples from normal distributions with equal means and equal but unknown variances. This function returns *h=1* if the test rejects the null hypothesis at the **5% significance level**, and *0* otherwise.

Test / Scenario	Standardly Placed Silver Tokens	Randomly Placed Silver Tokens
T-test for distances data-set. **Null hypotesys Ho:** The mean distance from obstacles of the professor's solution is equal to the mean distance from obstacles of my solution. **Alternative hypotesys Ha:** My solution distance mean is grater than professor's solution mean.	**Result:** h = 1, therefore `ttest2` rejects the null hypothesis at the default 5% significance level. So, I can state that my solution robot keeps a grater distance from obstacles with respect to the professor robot when silver tokens are standardly placed.	**Result:** h = 1, therefore `ttest2` rejects the null hypothesis at the default 5% significance level. So, I can state that my solution robot keeps a grater distance from obstacles with respect to the professor robot **even when silver tokens are randomly placed.**
T-test for time laps data-set. **Null hypotesys Ho:** The mean time of the professor's solution is equal to the mean time of my solution. **Alternative hypotesys Ha:** Professor's solution mean is grater than my solution mean.	**Result:** ***h = 1***, therefore `ttest2` rejects the null hypothesis at the default 5% significance level. So, I can state that professor's solution takes a greater amount of time in completing a lap with respect to my solution when silver tokens are standardly placed.	**Result:** ***h = 1***, therefore `ttest2` rejects the null hypothesis at the default 5% significance level. So, I can state that professor's solution takes a greater amount of time in completing a lap with respect to my solution **even when silver tokens are randomly placed.**