Scilab case study project on

Optimal Minimum Snap Trajectory Generation for Quadcopter Navigation

Thabsheer Jafer

Mea Engineering College

Robotics and Autonomous Systems

August 29, 2024

Abstract

This case study explores the generation of a minimum snap trajectory for a quadcopter using Scilab, with the orientation of the vehicle not being considered. The goal is to create a smooth, continuous path through predefined waypoint positions by minimizing the snap, the fourth derivative of position, ensuring the trajectory's smoothness in terms of velocity, acceleration, and higher-order derivatives. The trajectory generation is achieved through the use of a spline-based approach, resulting in a seventh-order polynomial that satisfies the necessary boundary conditions. The calculus of variations is employed to derive the Euler-Lagrange equations, leading to the construction of the minimum snap polynomial. The study demonstrates how the desired trajectory can be accurately replicated by rounding off computational errors. The method is validated through simulations, and the results confirm the effectiveness of the approach in generating optimal trajectories for quadcopter navigation without considering vehicle orientation.

1. Introduction

This case study focuses on designing a smooth and efficient trajectory for a quadcopter, an unmanned aerial vehicle, using advanced mathematical techniques. The main objective is to create a "minimum snap" trajectory that ensures smooth transitions between predefined waypoints by minimizing abrupt changes in acceleration and jerk. The trajectory is represented by a seventh-order polynomial, designed through spline-based methods to meet stringent smoothness criteria, specifically targeting the minimization of the fourth derivative of position, known as snap.

Waypoints are specific locations that the quadcopter is programmed to pass through during its flight. These waypoints serve as intermediate targets, guiding the quadcopter along its intended path. By defining these waypoints, we can ensure that the quadcopter follows a planned route with precise navigation. The trajectory generation process involves designing a path that smoothly connects these waypoints, facilitating seamless transitions and efficient movement between them.

To achieve this, the study employs calculus of variations to derive the optimal trajectory by solving the Euler-Lagrange equations. This approach allows for precise control of the quadcopter’s motion, ensuring that the generated path is smooth and continuous. The integration of these techniques demonstrates how a well-designed trajectory can enhance the quadcopter's flight performance, ensuring stable and efficient navigation through the designated waypoints.

2. Problem Statement

The goal of this case study is to generate a smooth and efficient trajectory for a quadcopter using spline-based methods. Specifically, the objective is to develop a "minimum snap" trajectory, represented by a seventh-order polynomial, that smoothly connects a series of predefined waypoints. This involves:

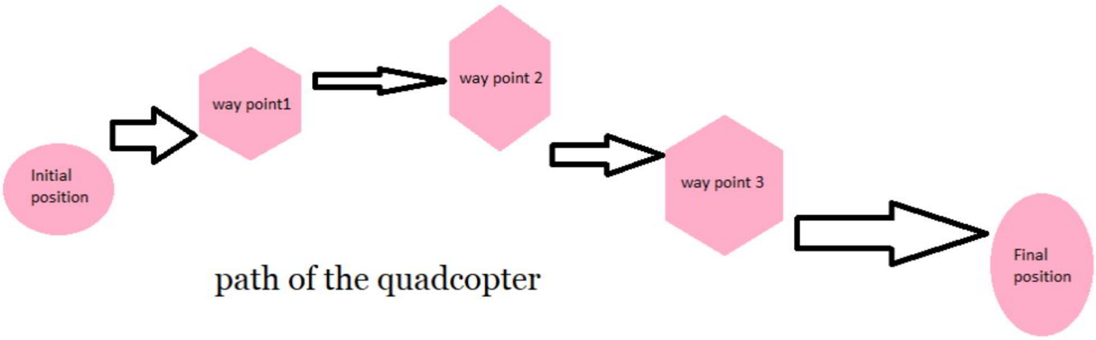
- Using splines and the Euler-Lagrange equations to design a trajectory that minimizes abrupt changes in acceleration and jerk.
- Generating a polynomial path that accurately follows the waypoint positions.
- Rounding off errors to ensure the resulting trajectory matches the desired path precisely.

The approach aims to enhance the quadcopter’s flight performance by ensuring stable and . Start,goal positions 2. waypoint positions 3. smoothness criterion 4. order of the system (n=4)

3. Basic concepts related to the topic

I. General setup:

1. Start,goal positions
2. Waypoint positions
3. Smoothness criterion
4. Order of the system (n=4)



path of the quadcopter

II. Calculus of Variation

$$x^*(t) = argminx(t) \int_0^T L(x^\cdot, x, t)dt[\mathbf{1}] \tag{1}$$

Calculus of variation is a generalization of calculus and it seeks to find the path, curve ,surface etc. for which a given function has a stationary value(we find minimum value). Using equation(1), find a value that gives the minimum value for the functional L. We use euler-lagrange equation to resolve the conditions and find the polynomial for minimum snap trajectory.

$$\frac{\partial L}{\partial x} - \frac{d}{dt}(\frac{\partial L}{\partial x^\cdot}) + \frac{d^2}{dt^2}(\frac{\partial L}{\partial x^\cdot.}) - \frac{d^3}{dt^3}(\frac{\partial L}{\partial x^\cdot..}) + \frac{d^4}{dt^4}(\frac{\partial L}{\partial x^\cdot...}) = 0 \tag{2}$$

Here

$$L = (\frac{d^4}{dt^4(x)})^2 \tag{3}$$

Solving equation(2) using (3) we get

$$\frac{d^8}{dt^8}(x) = 0 \tag{4}$$

integrating this differential equation till position, gives a polynomial of 7th order representing displacement. For each of the 4 paths, it is as follows:

$$p1(x) = c11 + c12t + c13t^2 + c14t^3 + c15t^4 + c16t^5 + c17t^6 + c18t^7 \tag{5}$$

$$p2(x) = c21 + c22t + c23t^2 + c24t^3 + c25t^4 + c26t^5 + c27t^6 + c28t^7 \tag{6}$$

$$p3(x) = c31 + c32t + c33t^2 + c34t^3 + c35t^4 + c36t^5 + c37t^6 + c38t^7 \tag{7}$$

$$p4(x) = c41 + c42t + c43t^2 + c44t^3 + c45t^4 + c46t^5 + c47t^6 + c48t^7 \tag{8}$$

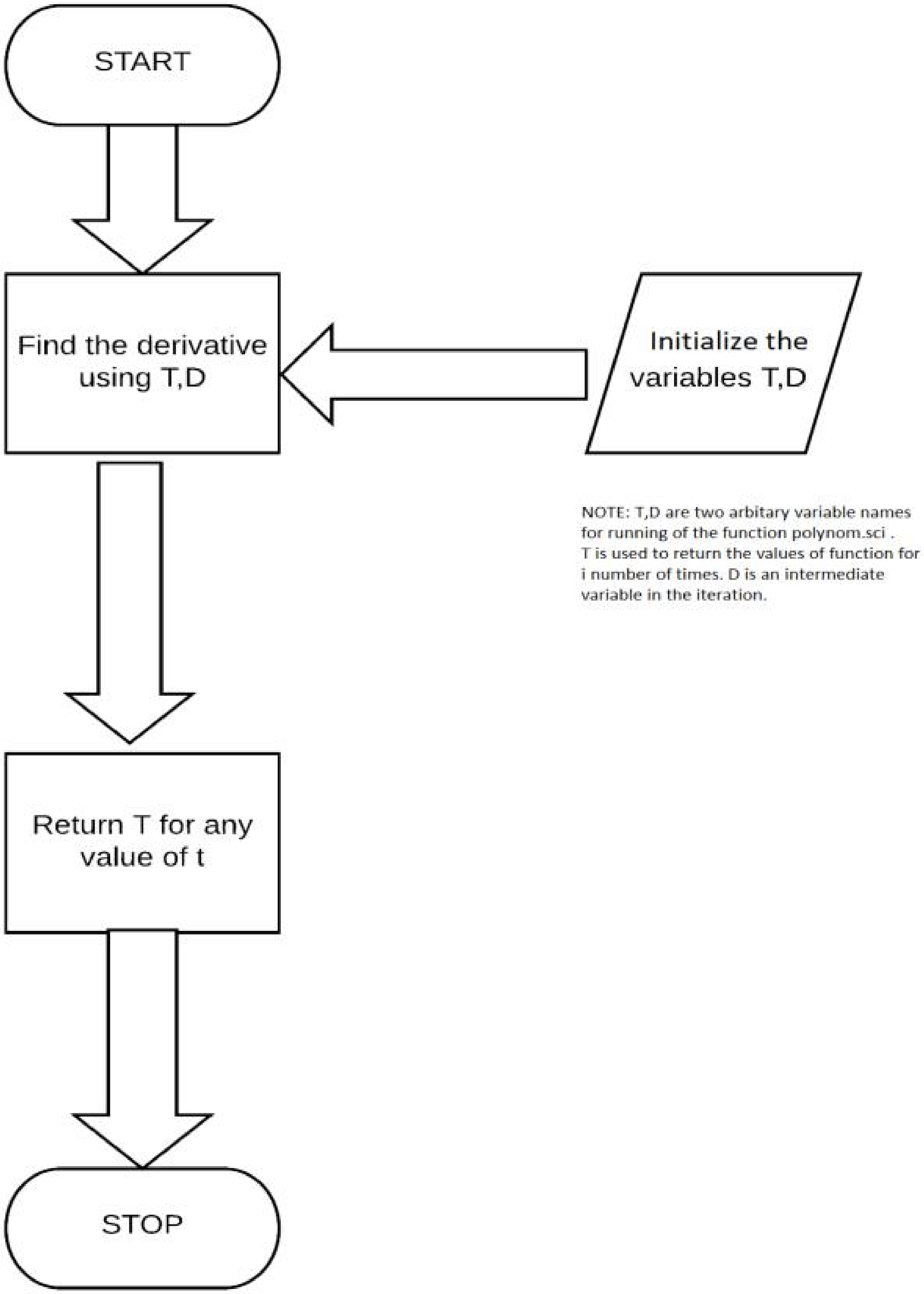
Here cij is an element of a 32X32 constant matrix. Similarly there will be four such polynomial for each y and z directions. Calculations for each of the 3 directions are separately done in this case study. In order to solve these polynomials, we need boundary conditions. since there is 3 waypoints, there will 4 paths between initial and final position. Thus 8 times 4 equals 32 coefficients and we need 32 boundary conditions to solve the problem.

1. Position at the beginning and ending of each paths is defined- that gives 8 boundary conditions
2. Velocity, acceleration and jerk(3rd derivative of position) is zero at the beginning and ending- that gives 6 boundary conditions
3. Velocity and up to 6th derivative of position at intermediate points are equal- that gives 18 boundary conditions

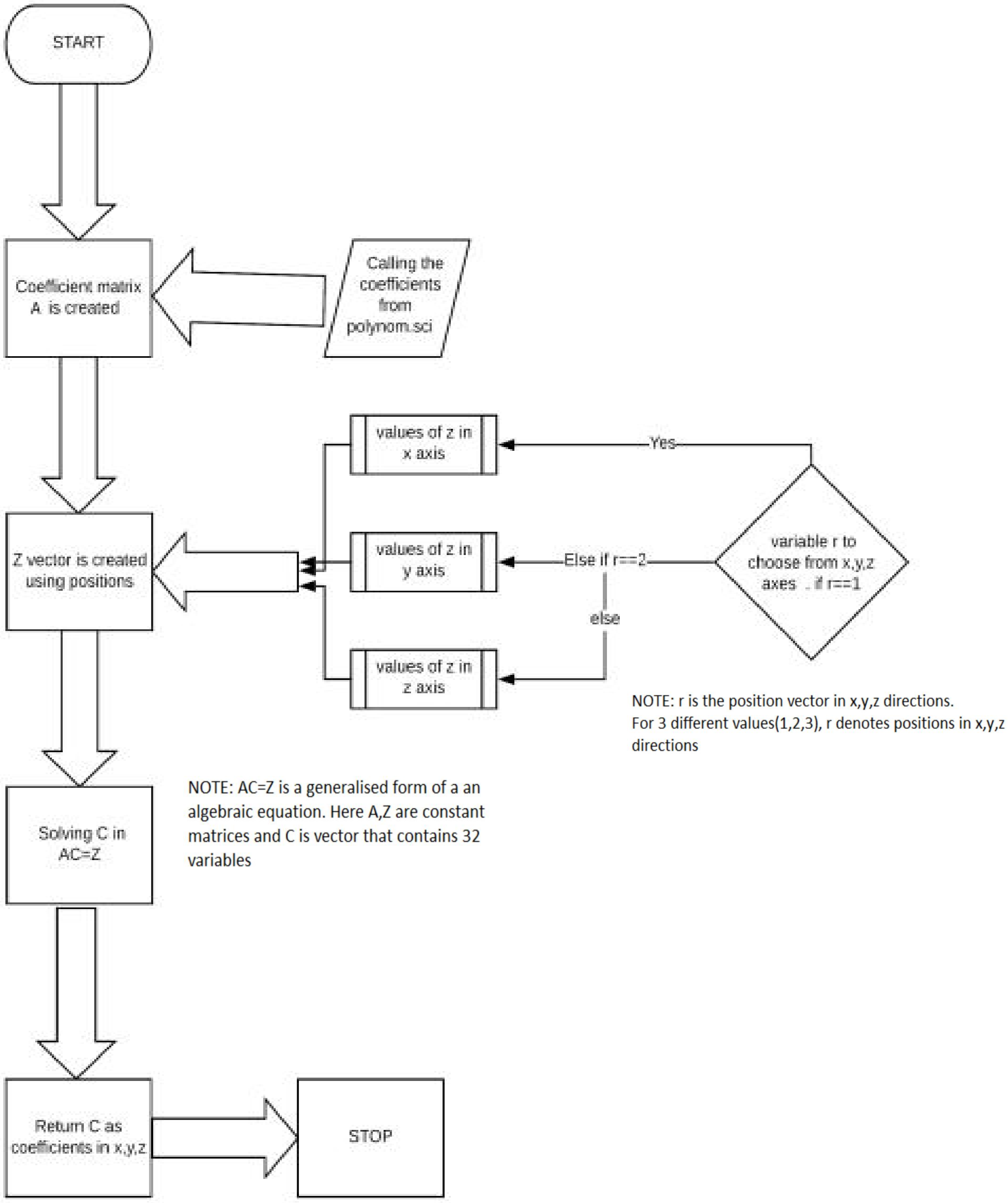
By applying all the 32 boundary conditions we get a 32X32 coefficient matrix, which contains constant values. When this matrix is put together with variables in equations 5,6,7,8; we get the equations describing the given path.Switching the values of time in these 4 equations we get the necessary points to plot the trajectory. The plot is not exact match to the desired trajectory, by rounding off we obtain the exact copy of desired trajectory.And this is to demonstrate that removing errors actually yields the desired trajectory. Hence we can confirm that the spline generation algorithm produced desired results.

4. Flowchart

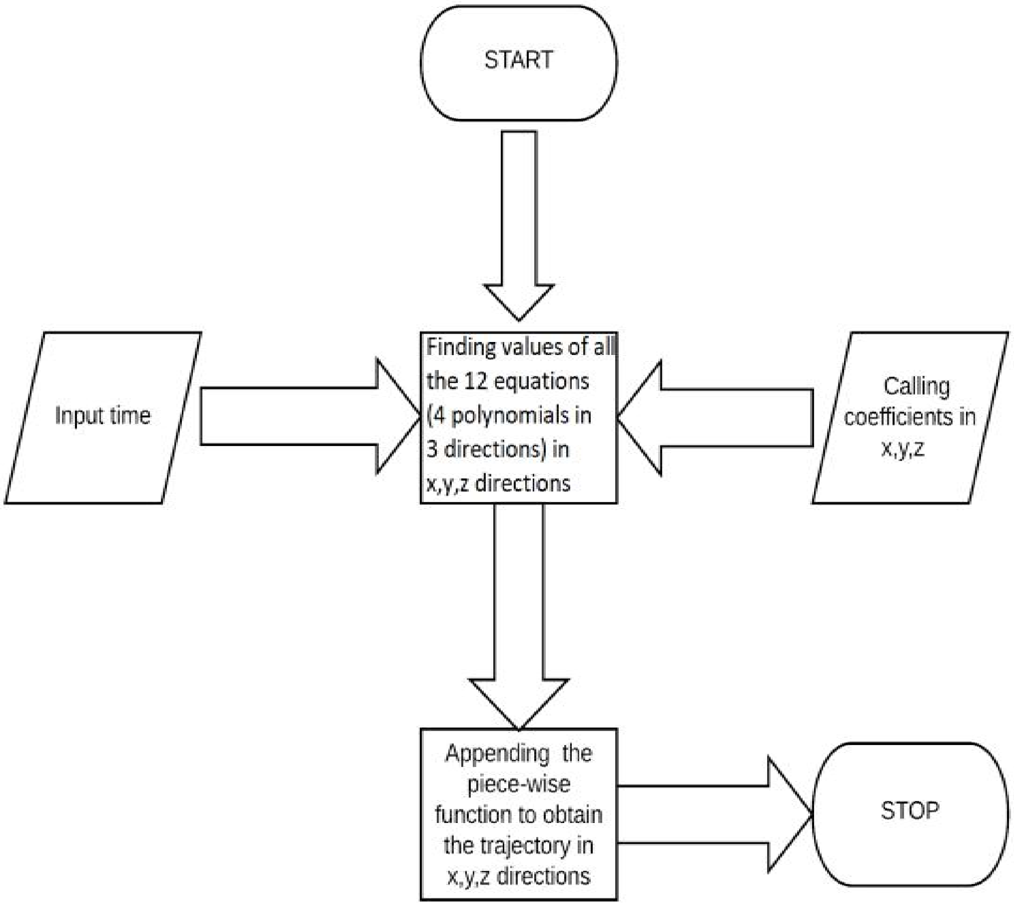
4.1 Polynom.sci-function to create a vector of polynomial coefficients



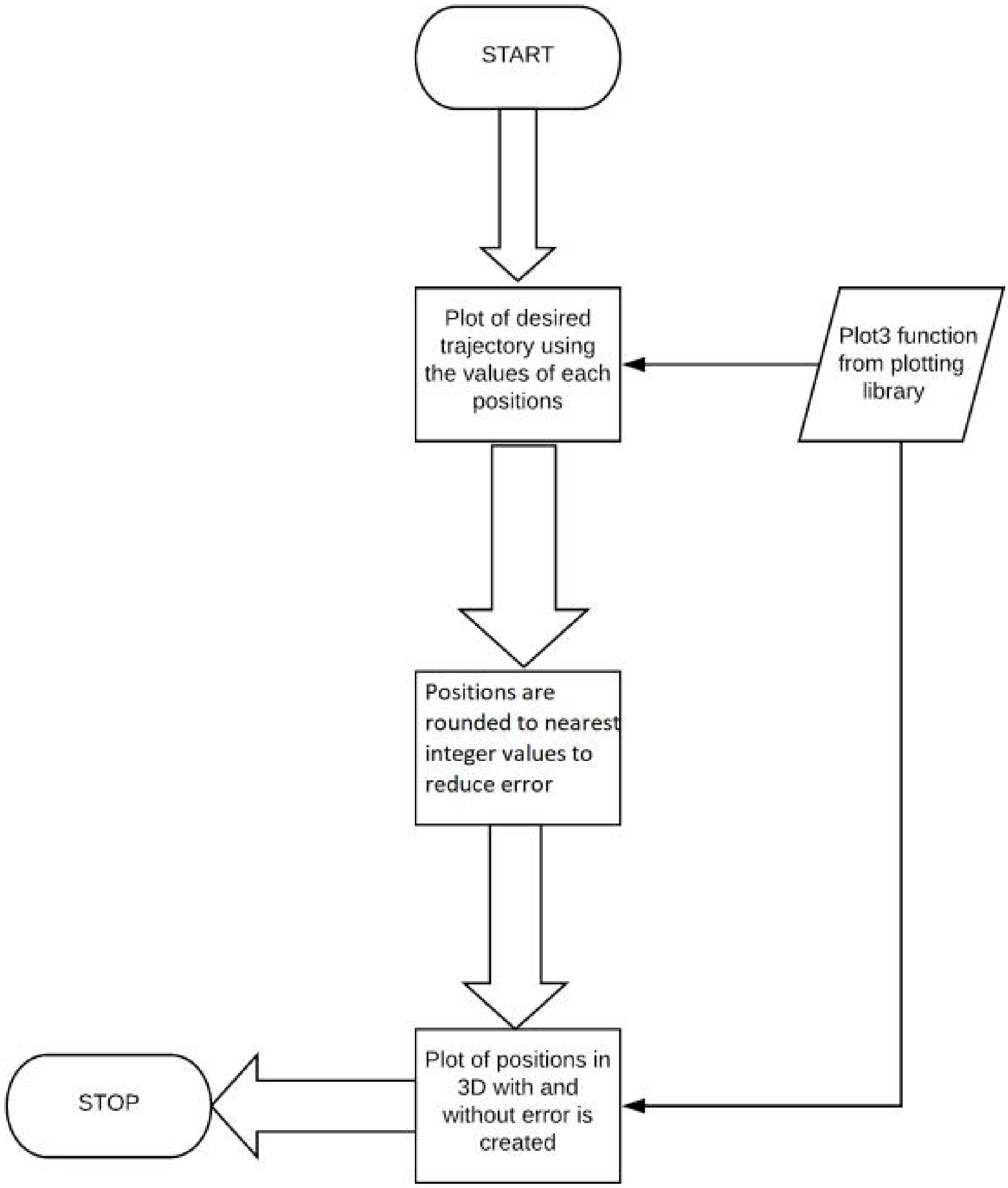
4.2 Const.sci-returns the coefficients in x,y,z



4.3 traj.sce-function generates trajectory



4.4 Run.sce-Plot the trajectory and approximate the values without a controller and obtain exact trajectory



5. Software/Hardware used

a. Windows 10 OS

b. Scilab 5.5.2

c. Plotting library toolbox version 0.46

6. Procedure of execution

a. Go to the Applications menu and click on ATOMS.

In the ATOMS window:

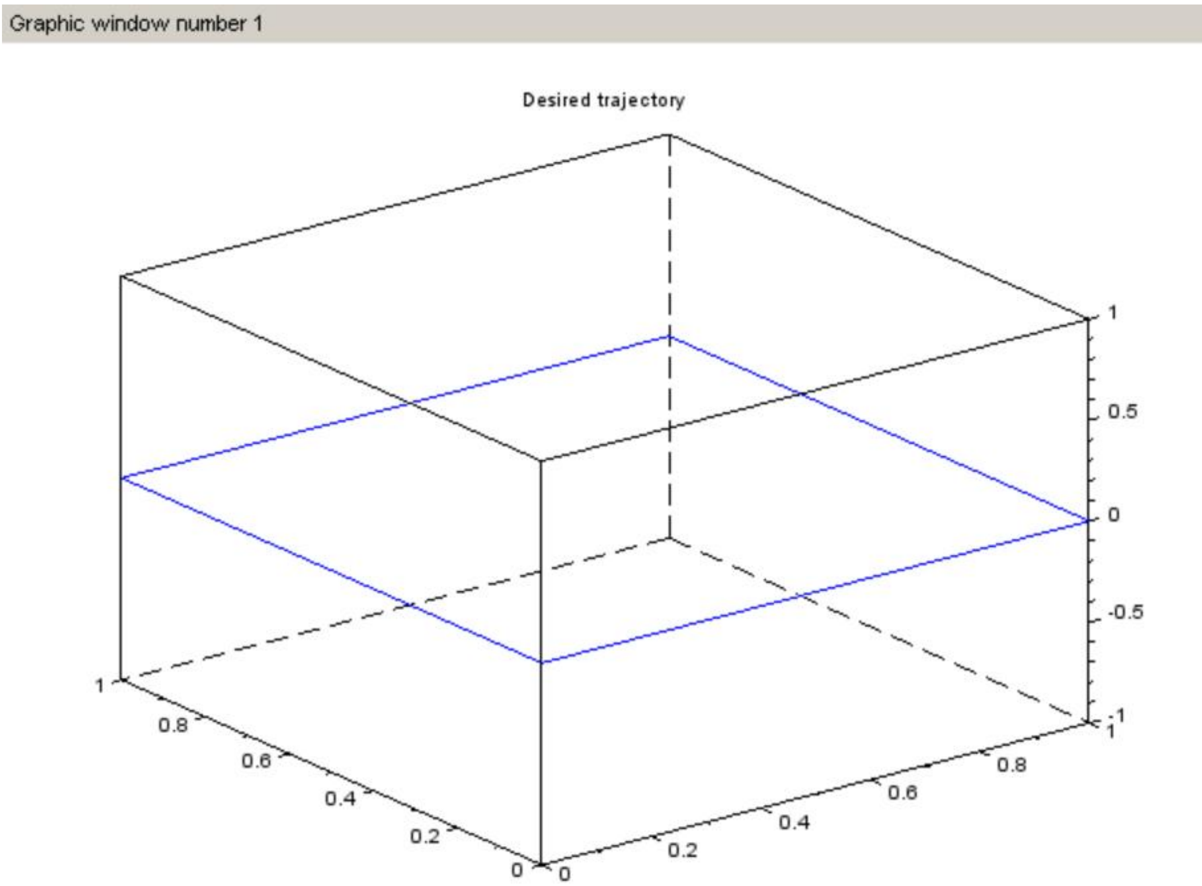
- Select the **Graphics** module from the left-side panel.
- Locate the toolbox named **Plotting library**.
- Click on Install on the right side to install the toolbox.

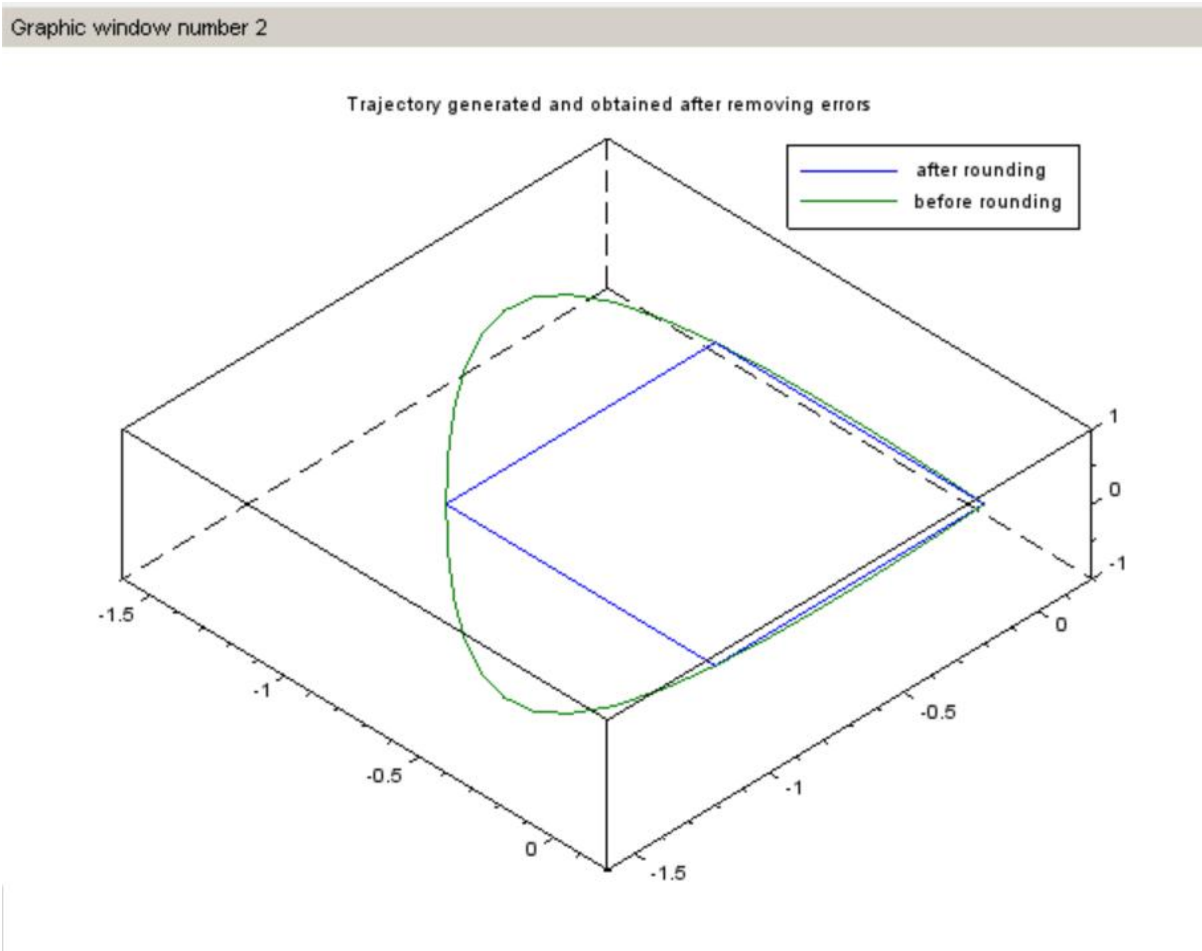
b. After the installation is complete, restart Scilab to ensure the toolbox loads successfully.

c. Change the current working directory to the folder containing the code.

d. Execute run.sce file.

7. Result





8. References

[1] Rotations and translations: University of pennsylvania https://prod-edxapp.edx-cdn.org/assets/ courseware/v1/834346628b86ff76fe98a975b10f51cf/asset-v1:PennX+ROBO3x+2T2017+type@asset+ block/Robo3x-Week11-final.pdf

[2] Trajectory generation using spline method S. Lai, M. Lan and B. M. Chen, "Optimal Constrained Tra jectory Generation for Quadrotors Through Smoothing Splines," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, 2018, pp. 4743-4750, doi: 10.1109/IROS.2018.8594357. https://ieeexplore.ieee.org/document/8594357

[3] To download plotting library:http://atoms.scilab.org/toolboxes/plotlib/0.46