Faculty of Engineering and Science

MAS 416

Modeling and Simulation of Mechatronic Systems – Industrial Robot Project

Group 5

CĂTĂLIN-MARIAN ORZAN

PEDRO SALCEDO BEISTEGUI

Supervisors

M. Ali Poursina

Michael R. Hansen

2023

List of Figures

List of Tables

Contents

1. Introduction

This project is a component of the MAS416 - Modeling and Simulation course, at the University of Agder, which serves as an introductory program focusing on the modelling and simulation of mechatronic systems. Our team, represented by two exchange students, Orzan Cătălin-Marian and Pedro Salcedo Beistegui, has the main objective to utilize knowledge gained throughout the course and personal research, in order to achieve the simulation of a dynamic model in Simulink/Simscape of a complex system. Developing skills in formulating mathematical models for integrative systems, constructing simulation models using specialized software, utilizing these models as tools for designing and assessing functionality, stability, and performance, the system in question presented a significant challenge, offering a valuable opportunity to expand our knowledge and take on a new task.

The robot to be simulated has six degrees of freedom and includes a parallel linkage system with an electric machining spindle as the tooltip. Our main goal is to compute the reference position and velocity for the spindle's tip during a specific operation using Matlab code, integrated into Simulink/Simscape. The simulation involves geared servomotors as drivers for each axis. Figure 1: Visual representation of the robot system shows a representation of our robot system. Notably, our project doesn't require extensive optimization efforts, allowing us to focus on dynamic modeling, code implementation, and simulation.

To facilitate the simulation of the robot, the project is divided into three main phases: modeling the robot in Simulink/Simscape, solving the kinematics, and generating the trajectory.

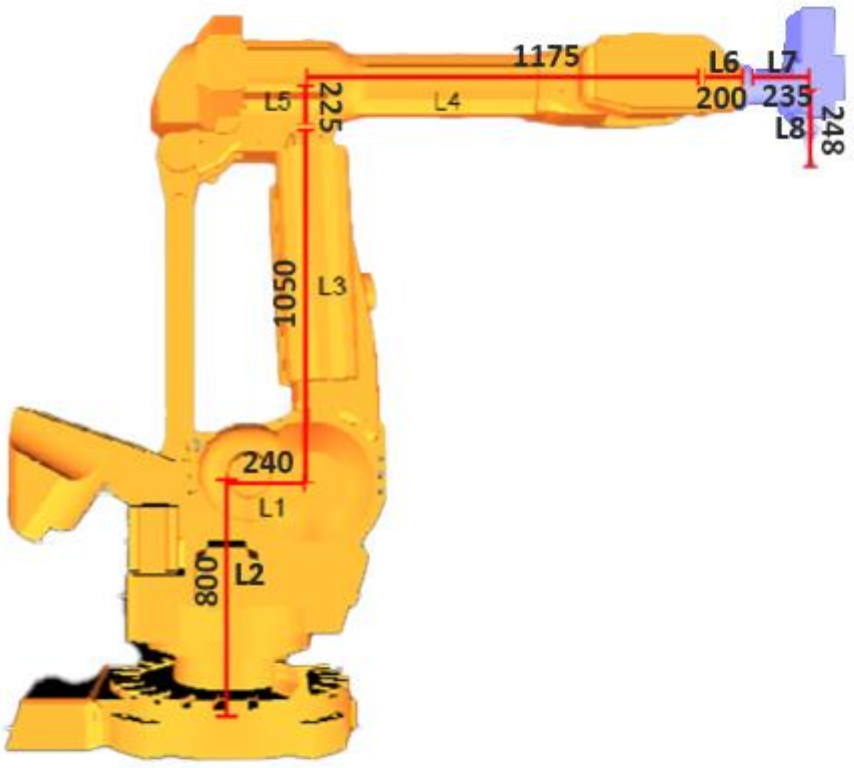


Figure 1: Visual representation of the robot system

2. Methodology

The project was systematically approached through collaboration, starting with a review of relevant course materials and exercises. Attendance at the project lecture provided valuable guidance. The following steps were established: initial research, gathering information and documentation, task distribution, development of the robot model, calculation of robot kinematics (Hand of God) and trajectory and path planning.

During the development phase, the tasks mentioned above, including a final task of report writing, were evenly distributed between the two team members. To facilitate project management, a series of tools and software apps were utilized. Noteworthy tools include SolidWorks 2022-2023 for generating the 3D model, Matlab R2023b with Simulink/Simscape for simulation and control, and ChatGPT 3.5 and Google Bard AI services for debugging and final project report inspection.

The project organization was maintained using Gantt's chart method, implemented in a shared Excel spreadsheet, Figure 13: Gantt Chart. Communication tools like WhatsApp assured team connectivity, and Microsoft OneDrive hosted project files. Additional tools such as Adobe Photoshop for photo editing and Microsoft Word 2021 for report writing were also utilized.

2.1. Research and Documentation

The project started with a thorough research for all the materials needed. We began by collecting useful documentation, valuable for any of the already established steps.

Provided by the teacher, the Robot Project Description |2| includes information regarding the robot type and linkage system, together with a simplified kinematic model (also visible in Figure 2: Robot kinematic model - ABB 6400) |7|, a linkage table (Table 1: Linkage table - revolute rotations), lengths of the kinematic model (Figure 12: Robot kinematic details) and oher information, such as motion limitations for specific joints (BE ±80° vertically, HI ±90° from the center line of GH) and presence of springs (CE, 750mm, 100 kN/m).

Other requirements are the tooltip, that must remain horizontal lengthwise and vertical heightwise. There are 5 points that the tooltip must touch, with the maximum deviation for the first 4 points being 3mm and the last one of 1mm. The geared servomotors with the mass of intertia J = 0.002 kg•m^2 and maximum current I_{max} = 16 A. The motor constant (K_m) can be 0.5/1.0/1.5 Nm/A, with a specific cost for each and the gearbox ratio can be 20/50/100/200 with a specific cost for each (see Table 6: Servomotors torque and cost and Table 7: Gearboxes ratio and cost for reference, part of the 7. Appendix chapter)

Linkage	Details	Bodies involved
A	Rotation along reference z axis	Base (blue)
B	Rotation out of the plane	Base (blue) to green
B	Rotation out of the plane	Base (blue) to red
G	Rotation along GH line	Cyan to gray
H	Rotation out of the plane	Gray to black
I	Rotation along HI line	Black to light blue (tooltip)

Table 1: Linkage table - revolute rotations

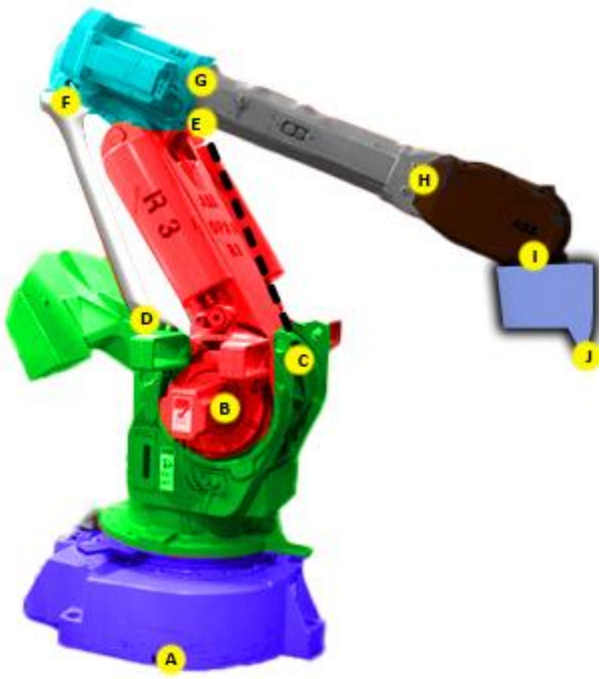
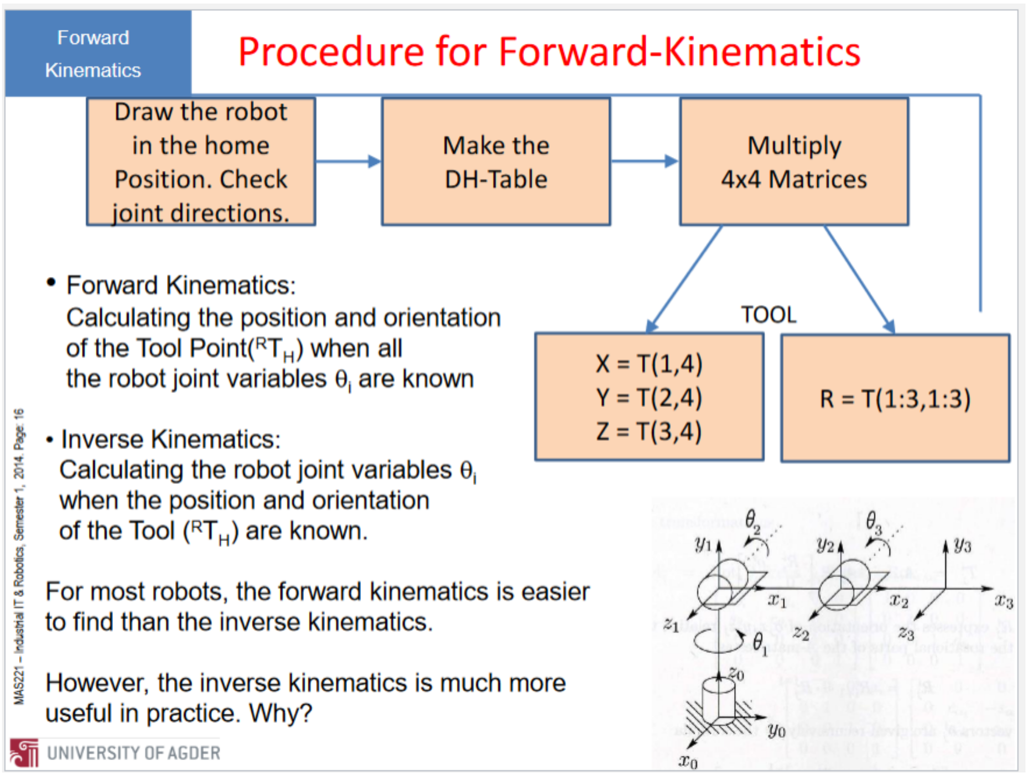


Figure 2: Robot kinematic model - ABB 6400

Another useful course utilized was Tyapin, I, MAS248 Robotics – 2023 UiA |3|. This course presents us valiable information regarding robotic arm kinematics (Figure 3: MAS248 Robotics course guidance) and the DH (Denavit-Hartenberg) table convention, together with the Forward and Inverse kinematics notions.



Forward Kinematics

Procedure for Forward-Kinematics

Draw the robot in the home Position. Check joint directions. → Make the DH-Table → Multiply 4x4 Matrices

TOOL

X = T(1,4)
Y = T(2,4)
Z = T(3,4)

R = T(1:3,1:3)

- Forward Kinematics:
  Calculating the position and orientation of the Tool Point($^{R}T_{H}$) when all the robot joint variables θ_i are known

- Inverse Kinematics:
  Calculating the robot joint variables θ_i when the position and orientation of the Tool ($^{R}T_{H}$) are known.

For most robots, the forward kinematics is easier to find than the inverse kinematics.

However, the inverse kinematics is much more useful in practice. Why?

MAS221 – Industrial IT & Robotics, Semester 1, 2014. Page: 16

UNIVERSITY OF AGDER

Figure 3: MAS248 Robotics course guidance

We gained important information about the PID controller and how to implement it in our project from the MAS247 – Industrial IT Classes |12|, which we both attended. This source offered useful details about using the PID controller in a closed loop (Figure 4: Closed Loop PID Controller). We can use this as a control system to manage data from various sources (inputs and disturbances) and produce an output to control our motors as needed.

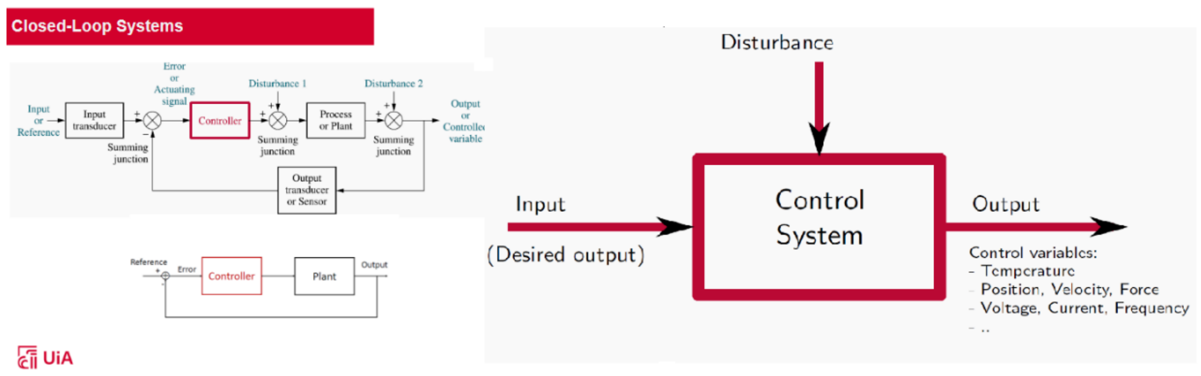


Figure 4: Closed Loop PID Controller

Other resources valuable for our project were the Simscape MAS416 Problems 8th-9th and several YouTube tutorials. |4| |5|

2.2. Kinematics

Our robot arm uses a parallelogram linkage mechanism, along with springs and a counterweight, to handle the force of gravity. The counterweight at the back of the arm is sized to balance the forces when the robot is not moving, which is a clever solution to initiate even the standing position of the robot.

The robotic arm electric motors are placed directly on the element to be moved, and the motor for the elbow is considered to be in the elbow joint in order to make the kinematics easier, allowing the mechanism to be considered a series of connected parts. As vizible in the figures, the transformation frames were rotated and adjusted accordingly to the size of the links.

2.2.1. Forward kinematics

Forward kinematics refers to the use of the kinematic equations of a robot to compute the position of the end-effector from specified values for the joint parameters.

To determine the position and orientation of the tooltip, the forward kinematics equations must be used. Our robotic system consists of rotational joints only, therefore the x and z rotational matrices were involved, together with the translational ones. A rotation matrix (Figure 5: Rotational matrix) is a mathematical object that represents a rotation in three-dimensional space |9|.

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Figure 5: Rotational matrix

The translation matrix is a key component representing the linear movement in a transformation. It's a 4x4 matrix with zeros everywhere except for 1s on the diagonal. The distance of travel is specified either in the (1,4) position for x translation or in the (3,4) position for z translation.

With this elements, a homogeneous matrix (H) can be anticipated (1), being an all-in-one package for transformations. This matrix, usually 4x4 in size, combines both rotation and translation information. The top-left part deals with how things rotate, and the right column handles how they move around. Having everything in a single matrix makes it simpler to work with and understand how an object changes in both position and orientation. In the homogeneous matrix, the final position data is in the last column (x, y, z). In our scenario, this matrix results from multiplying the rotational Z matrix, translational Z, translational X, and rotational X matrices.

The next step was organizing the data into a DH table, process that involved the use of an online genereator for the table |8|, based on the visual representation of the robot. Θ represents the joint number, while the lenghts are visible in Figure 1: Visual representation of the robot system, part of the introduction.

Link	RotZ	TransZ	TransX	RotX
1	Θ_1	L2	L1	π/2
2	$\pi/2 + \Theta_2$	0	L3	0
3	$\Theta_3 - \Theta_2$	0	L5	π/2
4	Θ_4	L4	0	- π/2
5	Θ_5	0	0	π/2
6	$\pi + \Theta_6$	L6 + L7	L8	0

Table 2: DH Table

The ultimate data results from multiplying all homogeneous matrices, starting from the base and extending up to the tooltip, in Matlab.

2.2.2. Inverse Kinematics

Inverse kinematics is the use of kinematic equations to determine the motion of a robot to reach a desired position. Given the desired robot's end-effector positions, inverse kinematics can determine an appropriate joint configuration for which the end-effectors move to the target pose. |10|

To solve the Inverse kinematics for our robotic system, the Hand of God system came in handy. This method supposes the calculation of joint angles based on forced motion, computing the joints as needed to follow the path in real time. Having the need to maintain the tooltip horizontal lengthwise and vertial heightwise and no rotation of the tooltip relative to the world frame, our robot can be considered a 3DOF system, with the joints remaining relevant being the base rotation, the parallel linkage rotation and the final rotation. Provided for this was a Hand Of God example file, that we could directly apply to our project in order to control it.

2.3. Development of Solidworks Model

The robot arm parts were designed using SolidWorks, a computer-aided design software. The dimensions were based on Figure 12: Robot kinematic details, in the appendix. Each part was created using the "Sketch" function, with thickness added using the pad feature. Color was used to distinguish between different parts, and mass and inertia were assigned appropriately for realistic representation.

The assembly process involved connecting the parts using the "Mate" function in SolidWorks. Cylindrical components were included for the joints, allowing for a specific type of movement - revolute.

An interesting aspect is the compatibility with Matlab. Using an addon, the entire model can be exported to Matlab, where it can be converted into Simscape library elements as part of Simulink, extending its usability for simulation and analysis in engineering applications.

2.4. Transition to Simulink Model

The robot, as described earlier, was transferred into a Simulink file. With a solid foundation in the form of the robot simulation, adjustments could be applied, like incorporating the C-E spring system. The outcomes of these adjustments are presented in the Results chapter. The modifications align with the guidance provided in Frontier's article on Modeling and Simulation of Robotics in Simulink Through Simscape |11|. A clear visual representation of these changes is depicted in Figure 6: Simscape Object module.

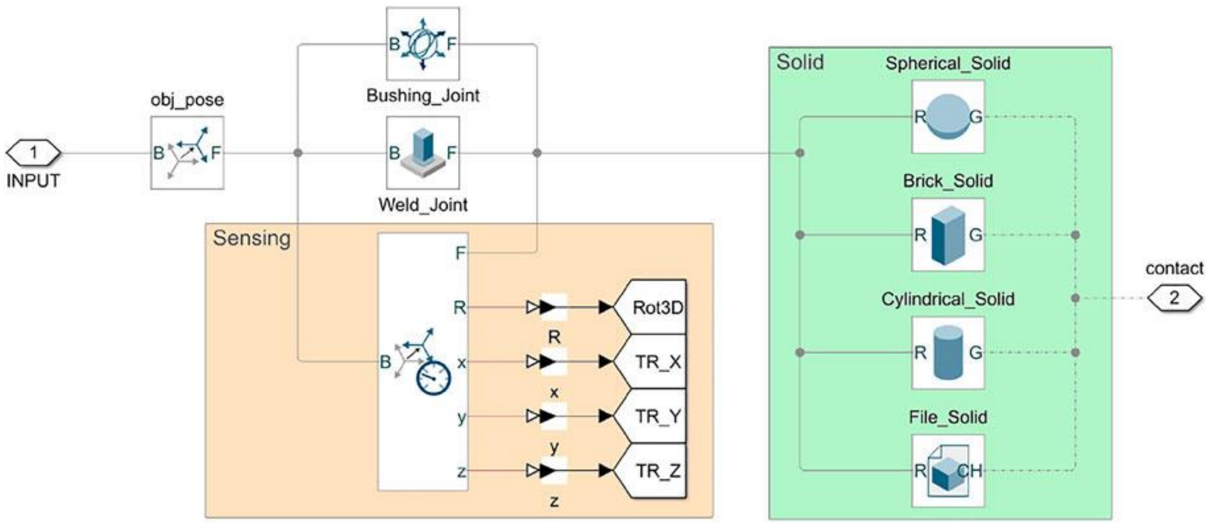


Figure 6: Simscape Object module

2.5. Motor and gear selection

In order to run the model, motors and gearboxes are needed in the joints to make the robot move. In this case, there are 6 joints in our model, each one handling a different amount of torque. To select both gearbox and servomotor for each of the joints, it is needed to known the amount of maximum torque they can stand. Then, the cheapest torque that stands those quantities will be selected among the others to be settled in that specific joint.

The maximum torque is calculated using the equations provided in the MAS416 – Robot Project Description provided PDF |2|. The mass moment of inertia (J) is the same for all the servomotors, with a value of 0.002 kg•m^2. Also maximum current is given (I_{max}=16 A). With this data, and the data given about each servomotor and gearbox, we can obtain the maximum torque allowed.

Gear\Motor	A		B		C	
	Cost	MaxT	Cost	MaxT	Cost	MaxT
1	2	160	2.25	320	2.5	480
2	2.25	400	2.5	800	2.75	1200
3	2.75	800	2.75	1600	3	2400
4	3.5	1600	3.75	3200	4	4800

Table 3: Cost and maximum torque per Motor/gear

While working on plotting the torque for each joint over a 10-second period, we faced some obstacles. In response, we decided to assign arbitrary numerical values to each joint. This approach allowed us to overcome the challenges and move forward with the selection process for gearboxes and motors for each individual joint.

Joint	A	B1	B2	G	H	I
Max T	2300	1900	3700	200	700	100

Table 4: Maximum torque for T=10 seconds

Having the maximum torque supported, then the motors and gearboxes are chosen (Table 4).

2.6. Path planning. Control of the Simulink Model

The process of planning the path was carried out by following the instructional videos provided for this particular task. Mathematical process of this procedure is vizible in Figure 14: Path calculation, attached to the 7. Appendix chapter. The outcome can be utilized to establish a sequence of waypoints for the robot's end tip to track. The objective is for the end tip to adhere to these points with a straight movement, avoiding any horizontal or vertical deviation of the tooltip. This path will serve as a MATLAB function block to govern our simulated robotic system. To manage this system, we opted for a PID control system.

The easiest form of control for our system is the Closed-Loop PID. This implementation is crucial for control in our system due to its versatility, robustness, and adaptability. PID controllers excel at maintaining the desired output of a system by minimizing the error between the setpoint and the actual output. They achieve this by incorporating three control actions: proportional, integral, and derivative. PID controllers are particularly well-suited for our project due to their ability to handle a wide range of system dynamics and disturbances. They are relatively simple to implement and tune, making them a practical choice for real-world applications. Also, PID controllers are highly adaptable to changing conditions, allowing them to maintain control even as the system's characteristics evolve.

Therefore, the arrangement of the robot model is as follows: the produced trajectory controls the inverse kinematics, yielding six outputs. Each of these outputs is independently regulated by its own PID controller. Afterwards, the PID controller produces an output that influences the gear, actuating the particular motor connected to it as a component of our simulated robot, which gives feedback information back to the controller PID.

Adjusting the PID controllers for each joint is essential for meeting translation and rotation requirements. By the simulations, we fine-tune proportional, integral, and derivative gains. The process involves minimizing oscillations with proportional gains, addressing steady-state errors with integral gains, and dampening oscillations with derivative gains. This iterative approach continues until the joints meet specifications.

3. Results

Our initial outcome is visible in the SolidWorks project, showing our simulated robot model. This marks a successful step, but with some minor issues that will be addressed in the Simulink version.

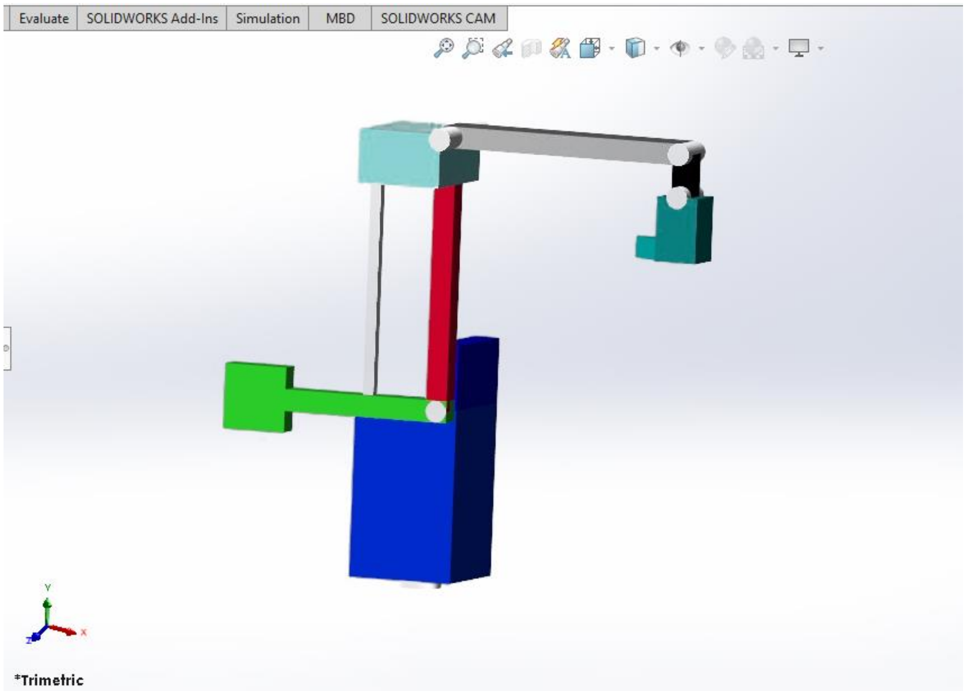


Figure 7: First Simulation - SolidWorks 2022

After completing that part, we used a plugin in SolidWorks for Simulink to change our robot into a simscape-made elements system. This generates a series of connected systems. For some parts, we had to adjust their settings following the project guide or what was important for our project. We did this by clicking on the part and changing the values if needed or adding components.

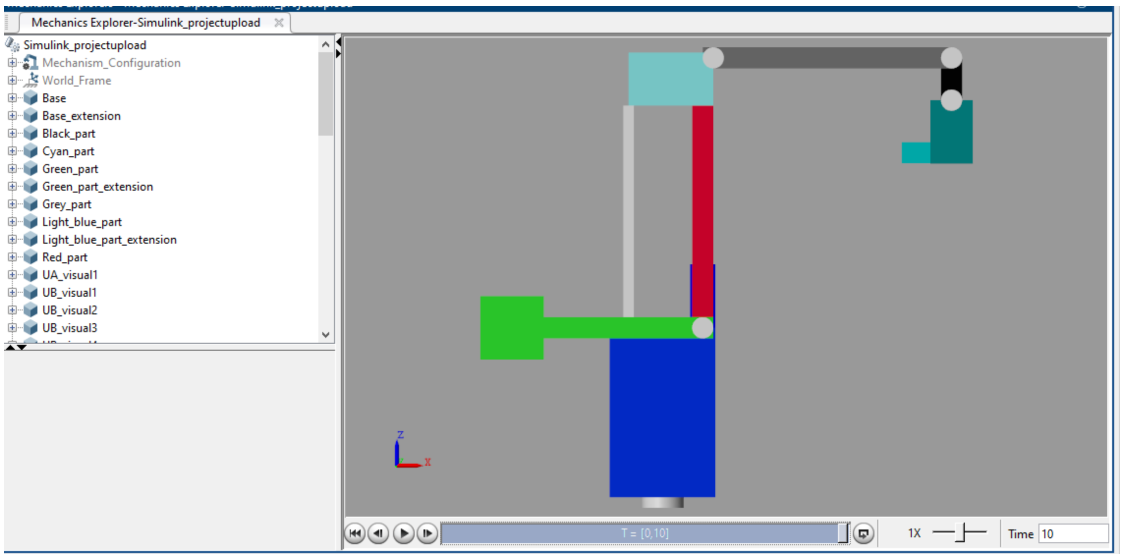


Figure 8: Matlab Simulation

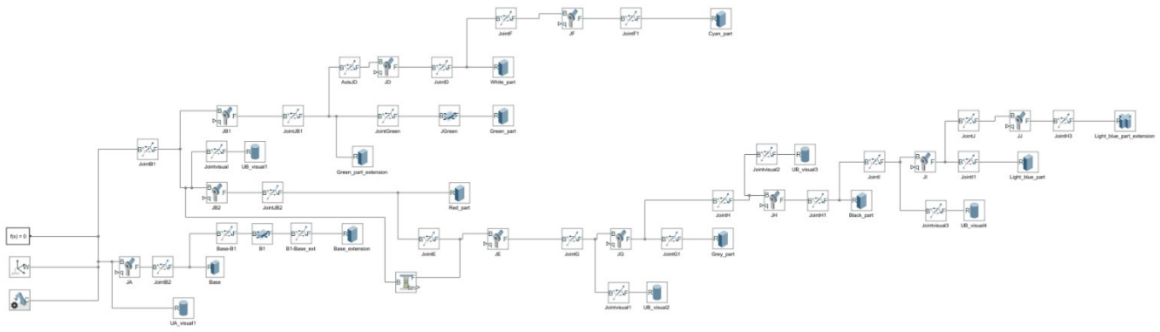


Figure 9: Simulink configuration from SolidWorks - Spring included

The next phase involved the comprehensive integration of the entire system. The path generation, which graphs are also visible in Figure 11: Path waypoints plots (right: over time), resulted from the following mathematical expressions:

P1(t) = a1+a2*t+a3*t^2+a4*t^3+a5*t^4 (2)

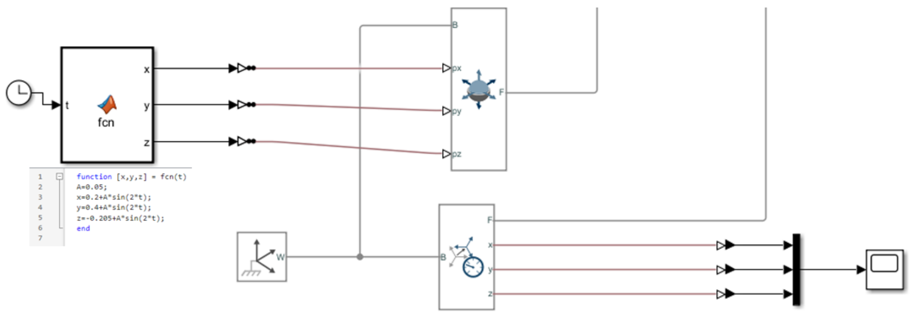
P2(t) = b1+b2*t+b3*t^2+b4*t^3 (3)

P3(t) = c1+c2*t+c3*t^2+c4*t^3 (4) and

P4(t) = d1+d2*t+d3*t^2+d4*t^3+d5*t^4 (5)

We established multiple subsystems in MATLAB, representing distinct components such as the trajectory, inverse kinematics (Hand of God) visible in Figure 10: Hand of God model, and PID control systems for each. Additionally, the gears came with their own pre-existing gain element in MATLAB, while the robotic system was formulated through SolidWorks and was a subsystem with six inputs to control – the motors to actuate every joint, together with outputs.

The project was simulated in time, with every state variables being considered properly and the trajectory simulated being followed by the end tip of the robot.



```
function [x,y,z] = fcn(t)
A=0.05;
x=0.2+A*sin(2*t);
y=0.4+A*sin(2*t);
z=-0.205+A*sin(2*t);
end
```

Figure 10: Hand of God model

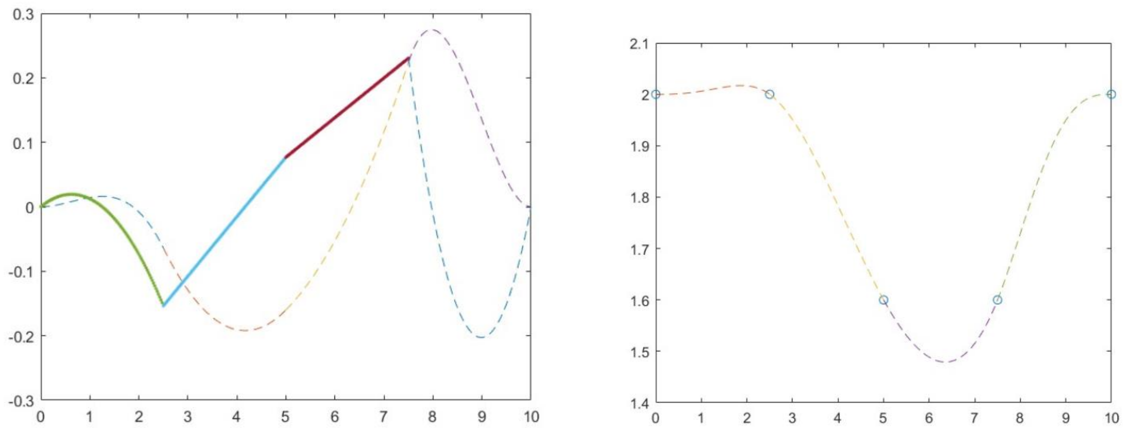


Figure 11: Path waypoints plots (right: over time)

Based on the results achieved, we chose a series of motors and gearbox, together with their cost, in order to achieve optimal performance of our system.

Joint	A	B1	B2	G	H	I
Motor	C	C	C	A	B	A
Gear	3	3	4	2	2	1
Cost	3	3	4	2.25	2.5	2

Table 5: Configuration of joints

4. Discussion

The simulation results offer valuable insights into how the system behaves in different situations. As expected, the system showed the trends we predicted and responded as anticipated to various inputs. The simulation accurately represented the complex interactions between different parts of the system, showing that it effectively mirrors the real-world system.

One significant finding from the simulation is the identification of areas where the system could be improved. The simulation results highlighted specific parameters and settings that could be fine-tuned to enhance the system's performance and efficiency. These discoveries will be crucial for refining the system's design and how it operates.

Moreover, the simulation proved to be a helpful tool for decision-making. By simulating various scenarios and examining the outcomes, we gained a comprehensive understanding of the system's strengths, limitations, and potential trade-offs. This knowledge will guide our future decisions on how to allocate resources, make design changes, and set operational protocols.

To sum up, the simulation proved to be a valuable tool for comprehending and analyzing the system. The results we obtained will play a key role in making well-informed decisions, ultimately leading to improved system performance and overall effectiveness.

5. Conclusion

The simulation study successfully met its goals, giving us useful insights into how the system behaves and performs. The model effectively replicated the real-world system, demonstrating how the various components interact.

These findings are crucial for designing and operating the system better. Fixing the identified issues will make the system work better and more efficiently.

Additionally, the simulation was a valuable tool for decision-making. By simulating different situations and checking the results, we learned more about the system's strengths, weaknesses, and possible trade-offs. This knowledge will help us make decisions about resources, design changes, and operational plans.

In conclusion, the simulation study provided a good understanding of the system in different situations, helping us make better decisions. The results will improve system performance, and the simulation model itself will be useful for future analysis and improvements.

6. Bibliography

|1| Hansen, M. R., Poursina, M. A. (2023). MAS416 Modeling and Simulation Lectures.

|2| Hansen, M. R., Poursina, M. A. (2023). MAS416 – Robot Project Description.

|3| Tyapin, I. (2023). "MAS248 Robotics" class, University of Agder.

|4|https://www.youtube.com/watch?v=xpA8TKEMpMk&list=WL&index=2&t=191s&pp=gAQBiAQB

|5|https://www.youtube.com/watch?v=pDiwAA1cnb0&list=WL&index=3&t=6s&pp=gAQBiAQB

|6| https://www.rosroboticslearning.com/forward-kinematics

|7| https://industrialpartsrus.com/abb-irb-6400-2-4m-robot-200kg-payload-s4c-m97a-control-system-teach-pendant/

|8| https://org.ntnu.no/intelligentsystemslab/course/DH/index.html

|9| https://en.wikipedia.org/wiki/Rotation_matrix

|10| https://www.mathworks.com/discovery/inverse-kinematics.html

|11| Pozzi, M., Achilli, G. M., Valigi, M. C., & Malvezzi, M. (2022). Modeling and Simulation of Robotic Grasping in Simulink Through Simscape Multibody. *Frontiers in Robotics and AI, 9,* Article 873558. https://www.frontiersin.org/articles/10.3389/frobt.2022.873558

|12| M. F. Aftab, MAS247 industrial it - teacher's class, 2023

|n| OpenAI. (2023). UiO GPT (Version GPT-3.5 Turbo) [Large language model]. University of Agder. (Used mostly for debugging purposes)

|n| Google. (2023). Google Bard [Advanced conversational model]. Independent user. (Applied for a variety of natural language processing tasks and interactions, such as grammar check)

7. Appendix

7.1. Robot dimension marks, mass and inertia

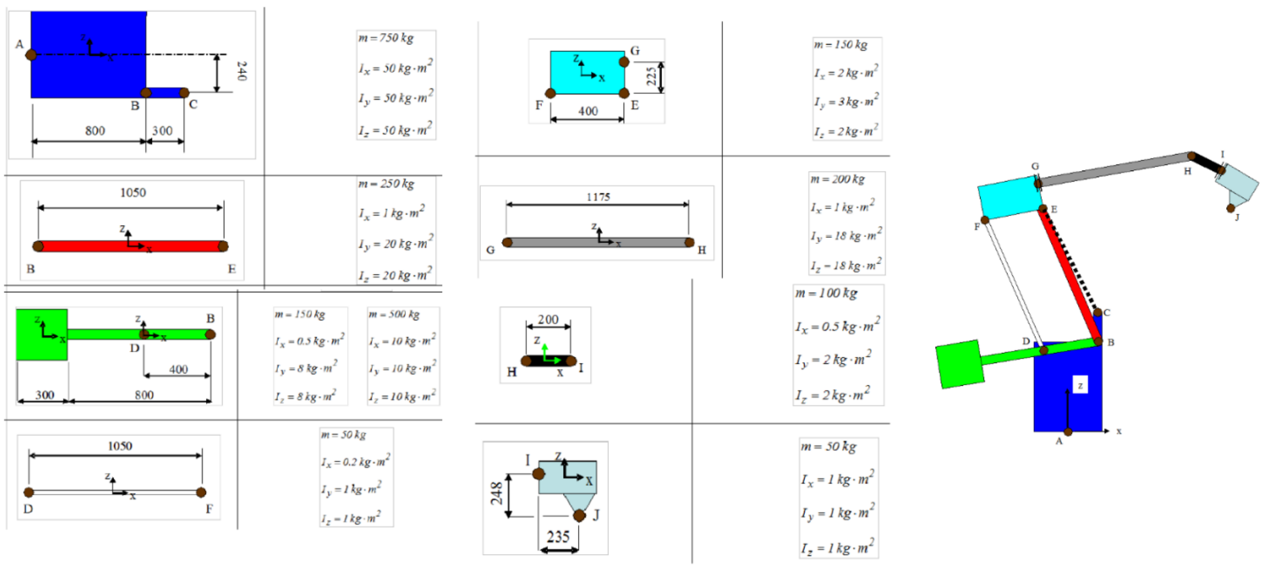
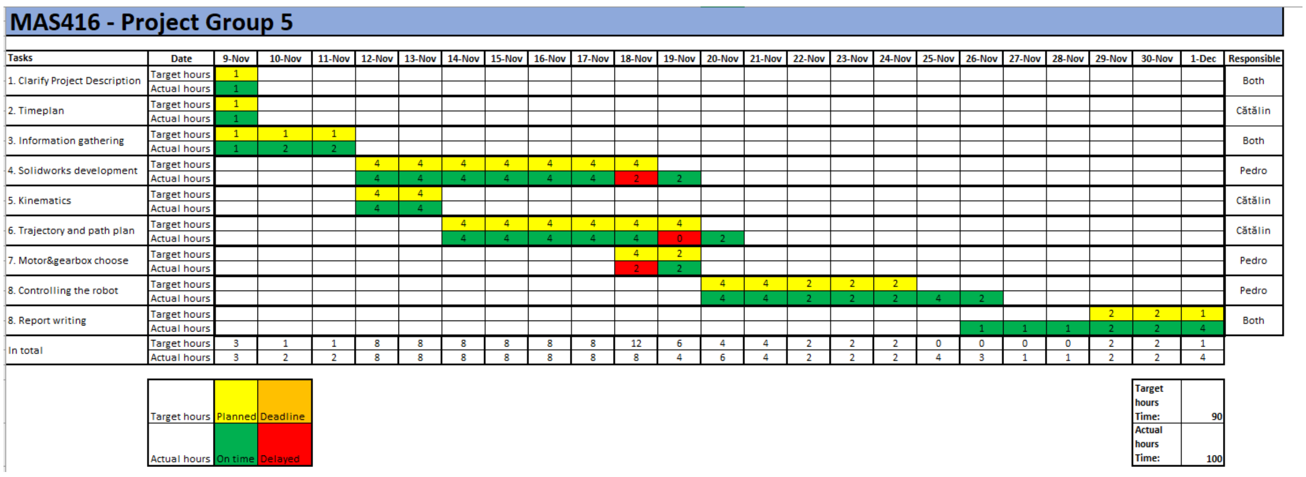


Figure 12: Robot kinematic details

7.2. Gantt Chart – project management



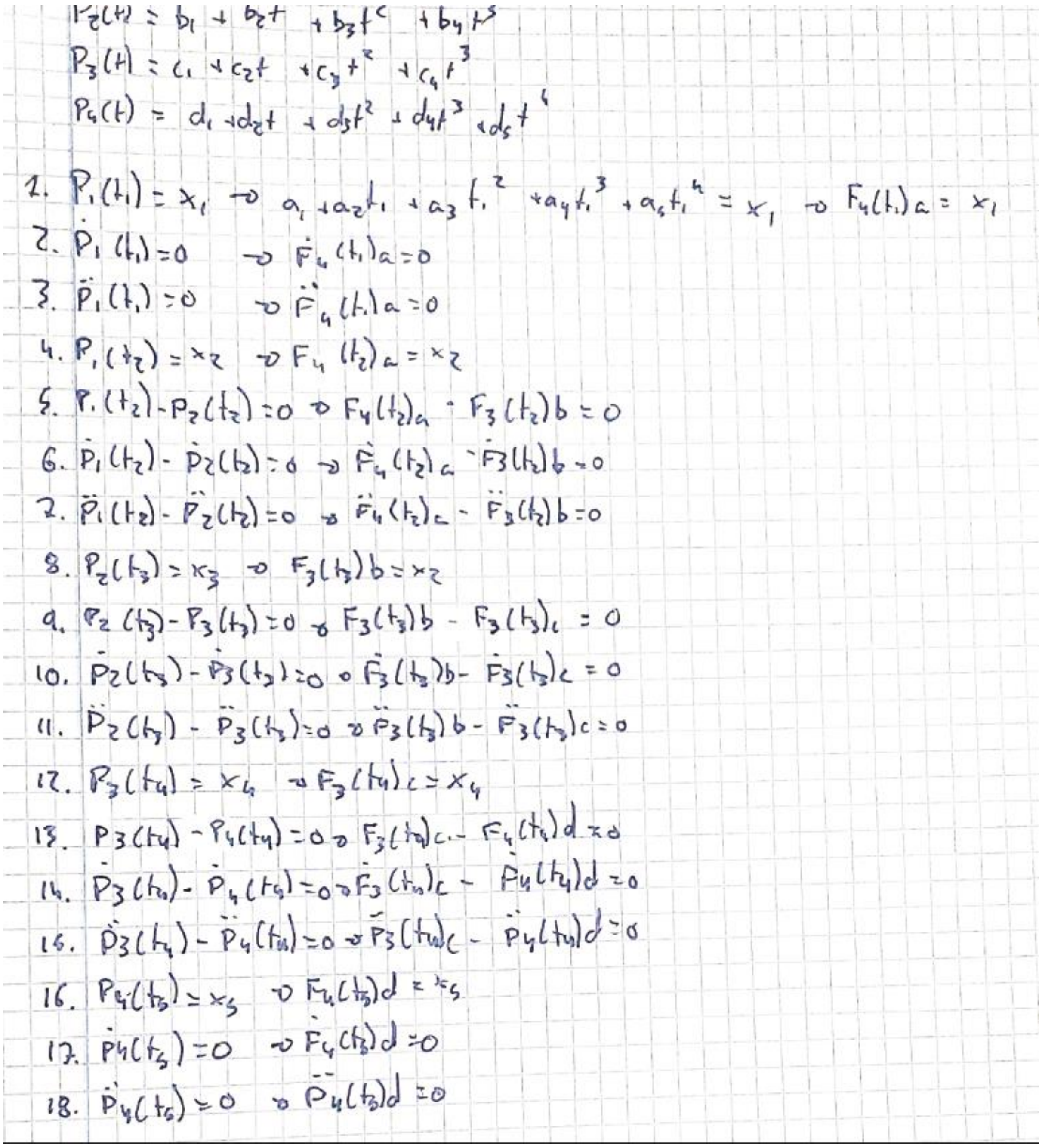
MAS416 - Project Group 5

Tasks	Date	9-Nov	10-Nov	11-Nov	12-Nov	13-Nov	14-Nov	15-Nov	16-Nov	17-Nov	18-Nov	19-Nov	20-Nov	21-Nov	22-Nov	23-Nov	24-Nov	25-Nov	26-Nov	27-Nov	28-Nov	29-Nov	30-Nov	1-Dec	Responsible
1. Clarify Project Description	Target hours	1																							Both
	Actual hours	1																							
2. Timeplan	Target hours	1																							Cătălin
	Actual hours	1																							
3. Information gathering	Target hours	1	1	1																					Both
	Actual hours	1	2	2																					
4. Solidworks development	Target hours				4	4	4	4	4	4	4														Pedro
	Actual hours				4	4	4	4	4	4	2	2													
5. Kinematics	Target hours				4	4																			Cătălin
	Actual hours				4	4																			
6. Trajectory and path plan	Target hours						4	4	4	4	4	4													Cătălin
	Actual hours						4	4	4	4	4	0	2												
7. Motor&gearbox choose	Target hours										4	2													Pedro
	Actual hours										2	2													
8. Controlling the robot	Target hours												4	4	2	2	2								Pedro
	Actual hours												4	4	2	2	2	4	2						
8. Report writing	Target hours																					2	2	1	Both
	Actual hours																		1	1	1	2	2	4	
In total	Target hours	3	1	1	8	8	8	8	8	8	12	6	4	4	2	2	2	0	0	0	0	2	2	1	
	Actual hours	3	2	2	8	8	8	8	8	8	8	4	6	4	2	2	2	4	3	1	1	2	2	4	

Target hours	Planned	Deadline
Actual hours	On time	Delayed

Target hours Time:	90
Actual hours Time:	100

Figure 13: Gantt Chart



$P_2(t) = b_1 + b_2 t + b_3 t^2 + b_4 t^3$

$P_3(t) = c_1 + c_2 t + c_3 t^2 + c_4 t^3$

$P_4(t) = d_1 + d_2 t + d_3 t^2 + d_4 t^3 + d_5 t^4$

1. $P_1(t_1) = x_1 \rightarrow a_1 + a_2 t_1 + a_3 t_1^2 + a_4 t_1^3 + a_5 t_1^4 = x_1 \rightarrow F_4(t_1)a = x_1$
2. $\dot{P}_1(t_1) = 0 \rightarrow \dot{F}_4(t_1)a = 0$
3. $\ddot{P}_1(t_1) = 0 \rightarrow \ddot{F}_4(t_1)a = 0$
4. $P_1(t_2) = x_2 \rightarrow F_4(t_2)a = x_2$
5. $P_1(t_2) - P_2(t_2) = 0 \rightarrow F_4(t_2)a - F_3(t_2)b = 0$
6. $\dot{P}_1(t_2) - \dot{P}_2(t_2) = 0 \rightarrow \dot{F}_4(t_2)a - \dot{F}_3(t_2)b = 0$
7. $\ddot{P}_1(t_2) - \ddot{P}_2(t_2) = 0 \rightarrow \ddot{F}_4(t_2)a - \ddot{F}_3(t_2)b = 0$
8. $P_2(t_3) = x_3 \rightarrow F_3(t_3)b = x_2$
9. $P_2(t_3) - P_3(t_3) = 0 \rightarrow F_3(t_3)b - F_3(t_3)c = 0$
10. $\dot{P}_2(t_3) - \dot{P}_3(t_3) = 0 \rightarrow \dot{F}_3(t_3)b - \dot{F}_3(t_3)c = 0$
11. $\ddot{P}_2(t_3) - \ddot{P}_3(t_3) = 0 \rightarrow \ddot{F}_3(t_3)b - \ddot{F}_3(t_3)c = 0$
12. $P_3(t_4) = x_4 \rightarrow F_3(t_4)c = x_4$
13. $P_3(t_4) - P_4(t_4) = 0 \rightarrow F_3(t_4)c - F_4(t_4)d = 0$
14. $\dot{P}_3(t_4) - \dot{P}_4(t_4) = 0 \rightarrow \dot{F}_3(t_4)c - F_4(t_4)d = 0$
15. $\ddot{P}_3(t_4) - \ddot{P}_4(t_4) = 0 \rightarrow \ddot{F}_3(t_4)c - \ddot{P}_4(t_4)d = 0$
16. $P_4(t_5) = x_5 \rightarrow F_4(t_5)d = x_5$
17. $\dot{P}_4(t_5) = 0 \rightarrow \dot{F}_4(t_5)d = 0$
18. $\ddot{P}_4(t_5) = 0 \rightarrow \ddot{P}_4(t_5)d = 0$

Figure 14: Path calculation

Motor torque contant Km	0.5 Nm/A	1.0 Nm/A	1.5 Nm/A
Motor dimensionless cost	1	1.25	1.5

Table 6: Servomotors torque and cost

Gearbox ratio	20	50	100	200
Gearbox dimensionless cost	1	1.25	1.5	2.5

Table 7: Gearboxes ratio and cost