Trajectory Planning, Kinematics, and Experimental Validation of a 3D-Printed Delta Robot Manipulator

Abu-Alim Ayazbay [1,*], Gani Balabyev [2], Sandugash Orazaliyeva [1], Konrad Gromaszek [3], and Algazy Zhauyt [1]

[1] Department of Electronics Engineering, Almaty University of Power Engineering and Telecommunications named after G.Daukeyev, Almaty, Kazakhstan
[2] Academy of Logistics and Transport, Almaty, Kazakhstan
[3] Lublin University of Technology, Lublin, Poland
Email: work_abu@hotmail.com (A.A.A.); g.balbayev@gmail.com (G.B.); orazalieva-sandugash@mail.ru (S.O.); k.gromaszek@pollub.pl (K.G.); ali84jauit@mail.ru (A.Z.)
*Corresponding author

***Abstract*—This paper presents the implementation, trajectory planning, kinematics, and experimental validation of an open-source 3D-printed delta robot manipulator. The robot's hardware consists of three servo motors, an Arduino Uno microcontroller, and a Pulse Width Modulation (PWM) servo module. The software includes a trapezoidal velocity profile planner and an inverse and forward kinematics solver to calculate the motor angles required to achieve a desired end-effector position in task space. The 3D printed parts were obtained from an open-source Thingiverse project and assembled to form the robot's kinematic structure. The robot's performance was evaluated in terms of its accuracy, repeatability, and maximum speed for a pick-and-place task. Experimental results show that the robot can achieve a positioning accuracy of 2.2 mm, with a top speed of 0.4 m/s, 30 picks per minute, and load carrying capacity up to 200 g. These results are satisfactory, considering the ease of assembly and the cost-effectiveness of the robot. The comparison of the calculated motor angles with actual motor angles obtained through additional potentiometer wires soldering shows promising results. The proposed robot's cost-effective characteristics, utilization of open-source additive manufacturing, and motion planning algorithms make it suitable for various applications, including education, research, and small-scale industrial applications.**

***Keywords*—robotics, delta robot, trajectory planning, kinematics, workspace, additive manufacturing**

I. Introduction

A delta robot is one of the types of parallel manipulators first introduced by Raymond Clavell [1]. Fig. 1 shows an example of a delta robot. The delta robot consists of two triangular platforms, one stationary (1), and the other movable and an end effector (4). The platforms are connected by three kinematic chains, each consisting of two links-active (2) and passive arms (3).

Manuscript received August 25, 2023; revised October 10, 2023; accepted November 8, 2023; published February 15, 2024.

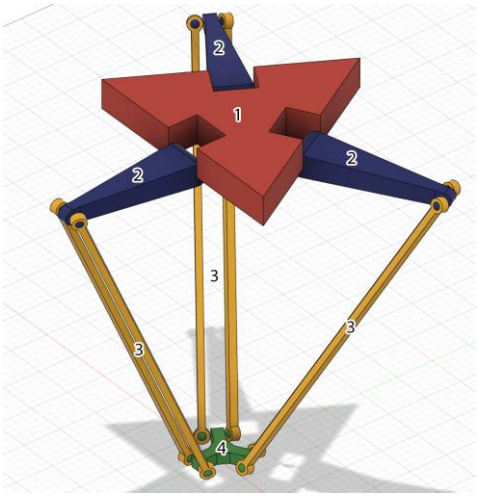

Fig. 1. Delta robot scheme.

Rotary motors drive the active arms, while the passive arms consist of two rods that are connected to the active arm and the movable platform by spherical joints. This design allows the moving platform to move in three axes and remain parallel to the stationary platform. Since the main weight represented by the motors is on the stationary platform, the robot's moving parts have low inertia, which allows high accelerations and speeds to be achieved. These advantages are essential in packaging, sorting, pharmaceutical, and food processing tasks, i.e., pick-and-place movements must be made at high speed [2–4].

In addition to delta robots, various types of robots are used in industrial applications. Industrial robots have been extensively studied and widely employed in manufacturing processes due to their high precision, repeatability, and versatility. These robots are typically large-scale manipulators capable of performing complex tasks in industrial settings. They are crucial in assembly lines, welding, painting, healthcare, aerospace, and other automated processes [5, 6].

doi: 10.18178/ijmerr.13.1.113-125

For instance, Huang *et al.* [5] proposed a method of positioning accuracy reliability analysis of industrial robots. Wang *et al.* [6] presented a study to verify the applicability of novel joint reducers in industrial robots. Xu *et al.* [7] proposed a dynamic model for the bearing-cycloid-pinwheel transmission mechanisms of industrial robots' joint reducers. These studies provide theoretical support to advancements in calibration and optimization techniques for industrial manipulators.

While industrial robots offer exceptional performance and reliability, they are often associated with high costs. Mainly industrial delta robot manipulators are often expensive (from 5000 USD) [8–10], making them less available for small-scale applications and research purposes.

Due to the high cost of commercial solutions, many developers and researchers use additive manufacturing to make prototypes and physical models of manipulators [11–13]. Additive manufacturing allows researchers to create a manipulator by developing their designs or using open-source designs. This method also makes it possible to easily change certain parts of the manipulator during research when deficiencies are detected or to adapt it to specific tasks.

There is no doubt that opportunities for customization and fast and cheap prototyping are significant, but at the same time, manipulators created with the help of 3D-printing technology have several disadvantages compared with commercial products. Commercial manipulators are produced with high-precision equipment with low tolerances, are equipped with quality bearings and joints, and have a ready control system with pre-installed programs for specific tasks. Since commercial manipulators are manufactured with low tolerances, there is no need to calibrate the control system parameters for each instance, while to achieve a sufficient positioning accuracy of 3D-printed manipulators, individual selection of parameters and calibration is required. This research aims to create a cost-effective delta robot using 3D-printed parts and open-source design, capable of sufficient speeds, accelerations, accuracy, and Picks per Minute (PPM) rate.

To achieve the goal of the study, the following tasks were set:

- Printing parts and assembling the delta robot;
- Developing a control scheme;
- Developing a control algorithm;
- Conducting experiments to evaluate the parameters of the robot.

There are many open-source projects of delta robots, with parts files, a description of the assembly process, and a list of necessary components. The EEZYbotDELTA project was used to assemble the delta robot, which is publicly available on the resource Thingiverse [14]. The selected project is characterized by the ease of assembly, rigidity of construction, the use of a small number, and availability of the necessary components. It should also be noted that this project involves using available hobby-grade spherical joints, which allow you to achieve higher accuracy than projects where two cylindrical 3D-printed joints are used instead [15–17].

Since the goal of the work is to create a cheap and affordable manipulator, one of the most common and affordable debug boards Arduino Uno was used as the basis of the control system. To generate the PWM control signal of the servo drives, the PWM/Servo driver board based on the PCA9685 chip was used, which has a higher resolution of the PWM signal compared to the standard Servo library in Arduino, which will achieve greater positioning accuracy.

An integral part of the control algorithm is path planning. Many trajectory planning algorithms can provide continuity of position, velocity, acceleration, and higher position derivatives to produce smooth and efficient movements [18]. One of the most common trajectories planning types is the trapezoidal method (linear section with parabolic blends or LSPB) [19, 20]. This trajectory planning algorithm can optimize manipulator movements by considering the movement's starting and ending points, maximum velocity, acceleration, and duration [15]. Also, to develop the control algorithm, it is necessary to find a solution to the inverse kinematics of the delta robot.

Experiments will be conducted to evaluate the assembled robot's parameters, including accuracy estimates, maximum velocities, and accelerations. The results will allow us to understand the capabilities, limitations, and cost-effectiveness of the 3D-printed delta robot, i.e., the feasibility of creating manipulators for small industrial tasks, research, and study. Also, the obtained characteristics will be compared with those of commercial models.

It should be noted that in addition to industrial models, there are readily available non-industrial low-cost delta robots. The results obtained during the experiments will also be compared with the characteristics of one of the representatives of such solutions, Delta X 1 [21], which costs 399 USD.

In summary, this paper's contributions lie in integrating open-source robotics, cost-effective control systems, advanced kinematics, and trajectory planning on affordable hardware, culminating in a validated robotic system suitable for additive manufacturing and beyond. This work presents a practical and economical solution and contributes to the broader discourse on accessible and innovative robotics development.

II. Materials and Methods

A. Description of the 3D Printed Delta Robot Manipulator Design

The EEZYbotDELTA delta robot comprises 23 parts of 8 different types, illustrated in Fig. 2. The movable platform is represented by part (1), while the fixed platform is constructed by connecting three instances of parts (2) and one instance of (6). Part (2) facilitates the installation of servo drives MG995/MG996. The three active arms are constituted by the combination of parts (3), (4), and two spacers (7) between them. To establish a

supportive framework, aluminum tubes are inserted into parts (5) and (8). Ultimately, these components collectively contribute to the structural configuration of the robot.

All parts were printed on a Creality CR-10 Smart 3D printer. Polylactic Acid (PLA) filament was used as the material. With a print speed of 50 mm/s and a fill factor of 20%, printing all the parts took about 40 h and 20 ms.

PLA is one of the most common, affordable, and highly regarded materials for creating prototypes and producing parts with high dimensional accuracy. Furthermore, it is widely acknowledged for its ease of printing compared to other common filaments and biodegradability [22, 23].

In addition to the aluminum tubes and servos, rolling bearings, spherical joints, M3 and M4 threaded nuts, and screws were needed to assemble the delta robot. A more detailed list of required components and the assembly process is described on the associated webpage [14].

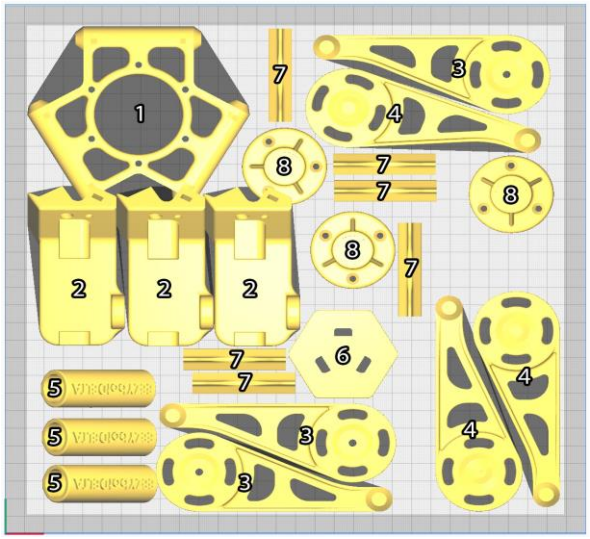

Fig. 2. Delta robot parts.

It took 271 grams of filament to print the parts, which, for 20 USD per kg, was about 5.5 USD. The rest of the components cost about 55 USD. As a result, the cost of the delta robot was 60.5 USD, which stands at about 1% of the commercial delta robot price. The costs of each component are shown in Table I.

TABLE I. DELTA ROBOT COMPONENTS COST

No.	Component	Cost USD
1	PLA filament	5.5
2	MG996R servo motors	14.49
3	M4 × 200 mm threaded rods	10.15
4	M4 spherical joints	2.77
5	604ZZ ball bearings	2.26
6	M4/M3 hardware	5.84
7	16 mm aluminum pipes	10.5
8	Arduino Uno	6.2
9	Servo control module	2.05
10	DC-DC converter	0.7
	Total	**60.46**

The assembled delta robot is shown in Fig. 3. The dimensions of the delta robot are 475 mm × 440 mm × 445 mm in width, length, and height respectively.

Fig. 3. Assembled delta robot.

B. Overview of the Hardware Components

Fig. 4 shows a schematic diagram of the control system of the manipulator. Since the work aims to create a low-cost and affordable delta robot, a cheap and common debug board Arduino Uno was used as the basis of the control system. The Arduino Uno board is equipped with an 8-bit microcontroller, already obsolete today, but despite this, its computing power is enough to calculate the trapezoidal motion trajectory and kinematics.

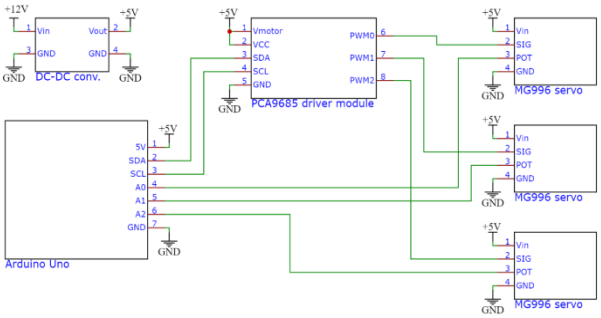

Fig. 4. Schematic diagram of the delta robot control system.

Since the design of the delta robot used is designed for MG995 or MG996 servos, three MG996 servos were used. The MG996 servos are an adapted version of the MG995 and can deliver a torque of 0.92 N.m (9.4 kg.cm in datasheet) and an angular speed of about 350°/s at a supply voltage of 4.8 V.

Such servo drives have only three contacts (*Vin*, *Signal*, and *GND*), i.e., there is no possibility to determine the current shaft position during their operation. To determine the current shaft positions of the servo drives, they have been modified in the course of the work. Each servo drive includes a potentiometer, with the help of which the built-in servo controller detects the shaft positions, so it was only necessary to output the signal from the middle contact of each potentiometer. The structural diagrams of the basic and modified servo drives are shown in Fig. 5.

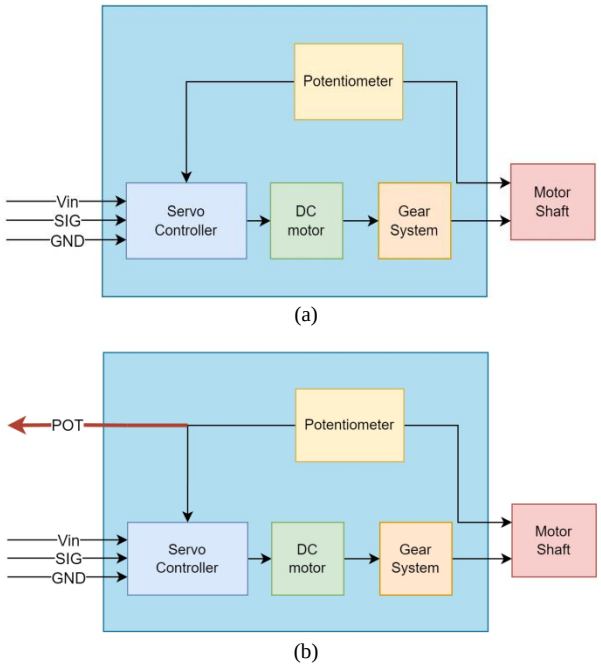

Fig. 5. Schematics of servo drives: (a) Basic; (b) Modified.

The output signal of the potentiometer is analog and varies from 0 V to *Vin*, depending on the position of the servo shaft. To determine the relationship between these values, seven measurements were made in 15° increments. 0° was taken to be the position at which the active arm of the robot was parallel to the stationary platform. The measurement results are shown in Table II.

TABLE II. MEASURED VALUES OF ANGLE Θ AND V_{POT} VOLTAGE

i	θ, deg	V_{POT}, V
1	–45	2.92
2	–30	2.59
3	–15	2.32
4	0	2.02
5	15	1.75
6	30	1.42
7	45	1.12

The data were also used to plot the dependence of V_{POT} on θ. The plot is shown in Fig. 6.

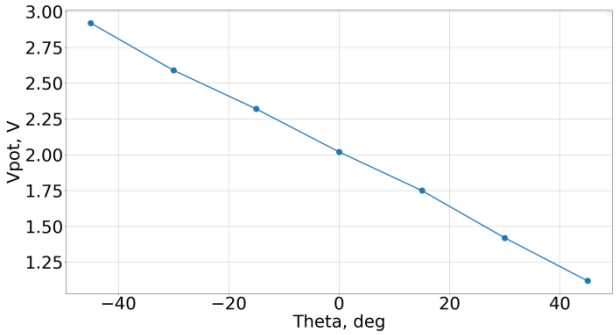

Fig. 6. The plot of V_{POT} voltage versus angle θ.

It is clear from the shape of the graph that the dependence is linear, so the V_{POT} voltage function on the angle θ can be written in the following form:

$$V_{POT}(\theta) = k \cdot \theta + b. \quad (1)$$

Since the angles of the lines between each of the two points on the graph are slightly different, it's necessary to find the coefficient k for each segment and find the average of the 6 (N–1):

$$k_i = \frac{V_{POT_{i+1}} - V_{POT_i}}{\theta_{i+1} - \theta_i}, \quad (2)$$

$$k = \frac{\sum_{i=1}^{N-1} k_i}{N-1}. \quad (3)$$

Now, the value of b can be determined in a similar way:

$$b_i = \frac{V_{POT_i}}{k \cdot \theta_i}, \quad (4)$$

$$b = \frac{\sum_{i=1}^{N} b_i}{N}. \quad (5)$$

As a result of the calculations, it was determined that k = −0.02 and b = 2.02. Then Eq. (1) will take the following form:

$$V_{POT}(\theta) = -0.02 \cdot \theta + 2.02. \quad (6)$$

It should be taken into account that when assembling the delta robot, achieving the same initial position of the shafts for all servo drives is impossible, so the value of b will be different.

To validate the obtained dependence, signal voltage measurements were performed for 10 positions of the active arm of the robot. According to the obtained values, the angles were calculated using the formula obtained from Eq. (6).

$$\theta(V_{POT}) = -50 \cdot V_{POT} + 101.$$

The results obtained are presented in Table III.

TABLE III. MEASURED AND CALCULATED V_{POT} VOLTAGE VALUES AT DIFFERENT Θ ANGLES

i	V_{POT}, V	Calc. θ, deg	θ, deg	*Error*, deg
1	2.815	–39.75	–40	–0.25
2	2.63	–30.5	–30	0.5
3	2.409	–19.45	–20	–0.55
4	2.224	–10.2	–10	0.2
5	2.01	0.5	0	–0.5
6	1.813	10.35	10	–0.35
7	1.618	20.1	20	–0.1
8	1.42	30	30	0
9	1.215	40.25	40	–0.25

As a result of this procedure, it was found that the maximum error in determining the angle of the active arm was 0.55° and the average error was 0.15°.

To control the servos, a board based on the PCA9685 chip is used. The board is controlled via the Inter-integrated Circuit (I2C) interface and allows to generate 16 PWM signals with a bit depth of 12 bits.

C. Trajectory Planning Algorithm

The trapezoidal trajectory planning method combines two methods – parabolic and linear, so it is also called Linear Section with Parabolic Blends. A trapezoidal trajectory consists of three parts—a section with constant positive acceleration (acceleration phase), a section with constant velocity, and a section with constant negative acceleration (deceleration phase) [18].

In the first phase ($0 \le t \le T_a$), the acceleration is constant, a linear equation describes the velocity, and a second-order polynomial describes the position:

$$\begin{cases} q_a(t) = a_0 + a_1 t + a_2 t^2 \\ \dot{q}_a(t) = a_1 + 2a_2 t \\ \ddot{q}_a(t) = 2a_2. \end{cases} \quad (7)$$

The initial conditions determine the parameters a_0, a_1, and a_2: the initial position q_0, velocity v_0, acceleration time T_a, and the value of the velocity constant v_{const}:

$$q_a(0) = a_0 = q_0,$$

$$\dot{q}_a(0) = a_1 = v_0 = 0.$$

The value of velocity by the time T_a must reach the value v_{const}, so:

$$\ddot{q}_a(t) = 2a_2 = \frac{v_{const}}{T_a},$$

Then the parameter a_2:

$$a_2 = \frac{v_{const}}{2T_a}.$$

Rewriting these expressions in Eq. (7), we obtain:

$$\begin{cases} q_a(t) = q_0 + \frac{v_{const}}{2T_a} t^2 \\ \dot{q}_a(t) = \frac{v_{const}}{T_a} t \\ \ddot{q}_a(t) = \frac{v_{const}}{T_a}. \end{cases} \quad (8)$$

In the second phase ($T_a \le t \le T - T_d$), the velocity is constant, the acceleration is zero, and a linear equation describes the position.

$$\begin{cases} q_b(t) = b_0 + b_1 t \\ \dot{q}_b(t) = b_1 \\ \ddot{q}_b(t) = 0. \end{cases} \quad (9)$$

Since the speed and position at the beginning of the second phase must be equal to the speed and position at the end of the first phase, then:

$$\dot{q}_b(t) = b_1 = \dot{q}_a(T_a) = v_{const},$$

$$q_a(T_a) = q_0 + \frac{v_{const} T_a}{2} = q_b(T_a) = b_0 + v_{const} T_a,$$

$$b_0 = q_0 - \frac{v_{const} T_a}{2}.$$

Substituting everything in Eq. (9), we obtain:

$$\begin{cases} q_b(t) = q_0 - \frac{v_{const} T_a}{2} + v_{const} t \\ \dot{q}_b(t) = v_c \\ \ddot{q}_b(t) = 0. \end{cases} \quad (10)$$

The third phase ($T - T_d \le t \le T$) is similar to the first:

$$\begin{cases} q_c(t) = c_0 + c_1 t + c_2 t^2 \\ \dot{q}_c(t) = c_1 + 2c_2 t \\ \ddot{q}_c(t) = 2c_2. \end{cases} \quad (11)$$

The velocity at the beginning of the third phase is v_{const}, and by the time T must reach zero, and $T_a = T_d$, so:

$$\ddot{q}_c(t) = 2c_2 = -\frac{v_{const}}{T_a},$$

Then the parameter c_2:

$$c_2 = -\frac{v_{const}}{2T_a}.$$

Now we can determine the parameter c_1:

$$\dot{q}_c(T - T_a) = c_1 - \frac{v_{const}}{T_a}(T - T_a) = v_{const},$$

$$c_1 = \frac{v_{const} T}{T_a}.$$

The position at time T must be equal to the value of the final position:

$$q_c(T) = c_0 + \frac{v_{const}T}{T_a}T - \frac{v_{const}}{2T_a}T^2 = q_1,$$

Then the parameter c_0:

$$c_0 = q_1 - \frac{v_{const}T^2}{2T_a}.$$

Substituting all expressions in Eq. (11), we obtain:

$$\begin{cases} q_c(t) = q_1 - \frac{v_{const}T^2}{2T_a} + \frac{v_{const}T}{T_a}t - \frac{v_{const}}{2T_a}t^2 \\ \dot{q}_c(t) = \frac{v_{const}T}{T_a} - \frac{v_{const}}{T_a}t \\ \ddot{q}_c(t) = \frac{v_{const}}{T_a}. \end{cases} \quad (12)$$

In the equations of position and velocity, we can remove $\frac{v_{const}}{2T_a}$ and $\frac{v_{const}}{T_a}$, from the brackets, then Eq. (12) will take the following form:

$$\begin{cases} q_c(t) = q_1 - \frac{v_{const}}{2T_a}(T-t)^2 \\ \dot{q}_c(t) = \frac{v_{const}}{T_a}(T-t) \\ \ddot{q}_c(t) = \frac{v_{const}}{T_a}. \end{cases} \quad (13)$$

The values of T_a and v_{const} are determined by different methods, depending on the initial conditions [15]. In this study, the method chosen for trajectory planning is the one in which the initial q_0 and final q_1 positions, the duration of motion T, and the acceleration a are specified. Using these values, the acceleration duration T_a and v_{const} are calculated:

$$T_a = \frac{aT - \sqrt{a^2T^2 - 4a(q_1 - q_0)}}{2a}, \quad (14)$$

$$v_{const} = \frac{q_1 - q_0}{T - T_a}. \quad (15)$$

Fig. 7 shows an example of a trapezoidal trajectory at $q_0 = -26°$, $q_1 = 54°$, $T = 1$ s, $a = 400°/s^2$.

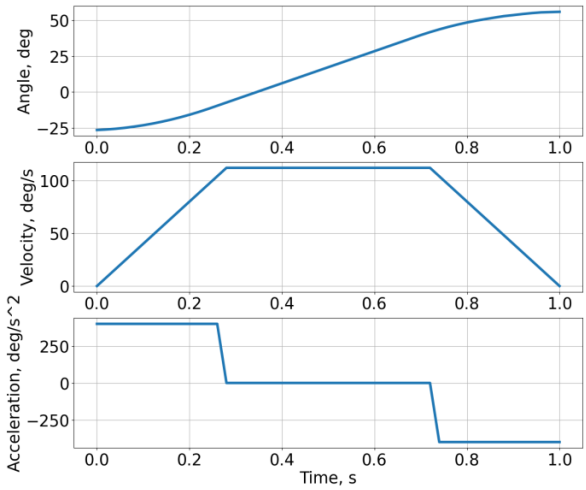

Fig. 7. Position, velocity, and acceleration of the trapezoidal trajectory.

D. Delta Robot Inverse and Forward Kinematics

The forward and inverse kinematics problems allow us to determine the relationship between the coordinates of the end effector and the rotation angles of the active arms of the delta robot. To determine the rotation angles of the active arms for given coordinates of the end effector, it is necessary to solve inverse kinematics, and direct kinematics, in turn, allows one to determine the coordinates of the center of the moving platform for given rotation angles of the active arms. Both problems were solved in [24] with examples of programs in Basic and supplemented in [25] with examples of programs in C.

To solve the forward and inverse kinematics of the delta robot, it is necessary to determine its four parameters: the side length of the upper triangular platform f, the length of the active arm r_f, the length of the passive arm r_e, and the side length of the moving triangular platform e. The listed parameters are shown in Fig. 8.

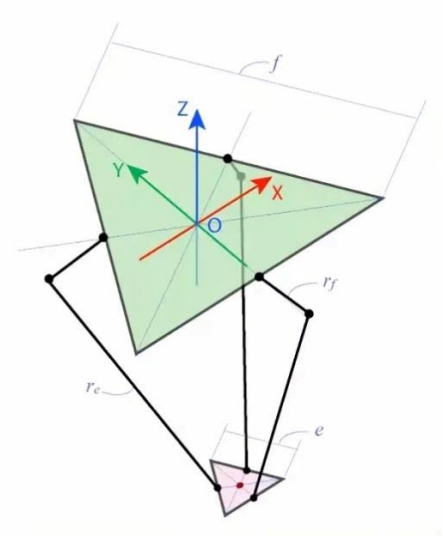

Fig. 8. Parameters of the delta robot [25].

1) Inverse kinematics

To simplify the inverse kinematics problem, it was proposed in [24] first to determine the angle θ_1 of the active arm, which is in the YZ plane, as shown in Fig. 9, where F_1J_1 *is the active arm, and* J_1E_1 *is passive arm.*

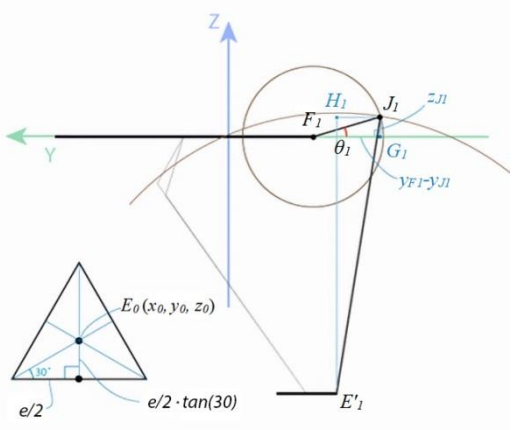

Fig. 9. Projection of the delta robot on the YZ plane [25].

Points F_1, J_1, and G_1 form a right triangle, where angle θ_1 can be defined as the arctangent of the ratio of the opposing cathetus to the adjoining:

$$\theta_1 = arctan\left(\frac{z_{J_1}}{y_{F_1} - y_{J_1}}\right), \quad (16)$$

where the value of y_{F1} is determined by the coordinates of the point F_1, which is located on the fixed base:

$$F_1\left(0, -\frac{f}{2\sqrt{3}}, 0\right),$$

and the values of z_{J1} and y_{J1} can be determined from the system of equations by the Pythagorean theorem for triangles $F_1J_1G_1$ and $J_1H_1E'_1$:

$$\begin{cases} \left(y_{F_1} - y_{J_1}\right)^2 + \left(z_{J_1} - z_{F_1}\right)^2 = r_f^2 \\ \left(y_{J_1} - y_{E'_1}\right)^2 + \left(z_{J_1} - z_{E'_1}\right)^2 = {J_1E'_1}^2. \end{cases} \quad (17)$$

Point J_1 is the intersection of two circles, one of which describes the active arm of the delta robot, and the other describes a projection of the passive arm on the plane YZ with a center at point E'_1. Point E'_1 is a projection of point E_1.

Point E_1 is on the moving platform, and its coordinates are determined by the coordinates of the center of the platform and the length of its side.

$$E_1\left(x_0, y_0 - \frac{e}{2\sqrt{3}}, z_0\right).$$

Then the coordinates of point E'_1 are:

$$E'_1\left(0, y_0 - \frac{e}{2\sqrt{3}}, z_0\right)$$

The passive arm J_1E_1 of length r_e and its projection $J_1E'_1$ form a right triangle, where the length of the cathetus $E_1E'_1$ is equal to the value x_0, then $J_1E'_1$ can be determined by the Pythagorean theorem:

$$J_1E'_1 = \sqrt{r_e^2 - x_0^2}.$$

As a result, to determine the angle θ_1, it is necessary to solve the system of Eq. (17) and substitute the obtained values in Eq. (16). This method can be applied only in the case when the required angle is located in the YZ plane, so to determine angles θ_2 and θ_3, it is necessary to rotate the coordinate system around the Z axis by 120° and -120°, then the required angles will be in the $Y'Z$ plane. To do this, it is only necessary to recalculate the coordinates of point E_0 in the new coordinate system:

$$E_0(x_0, y_0, z_0) \rightarrow$$
$$\rightarrow (x_0\cos(\varphi) + y_0 sin(\varphi), -x_0 sin(\varphi) + y_0\cos(\varphi), z_0),$$

where the angle φ will be 120° and –120° for θ_2 and θ_3, respectively.

2) Direct kinematics

As mentioned at the beginning of paragraph 2.4, to solve the direct kinematics problem, it is necessary to find the coordinates of the movable platform center at given values of θ_1, θ_2, and θ_3.

To solve this problem, it was proposed in [19] to displace points J_1, J_2, and J_3 parallel to the XY plane to the center by a distance $\frac{e}{2\sqrt{3}}$, then the center of the moving platform will coincide with the intersection point of three spheres described by three passive arms with centers at the points J'_1, J'_2, and J'_3.

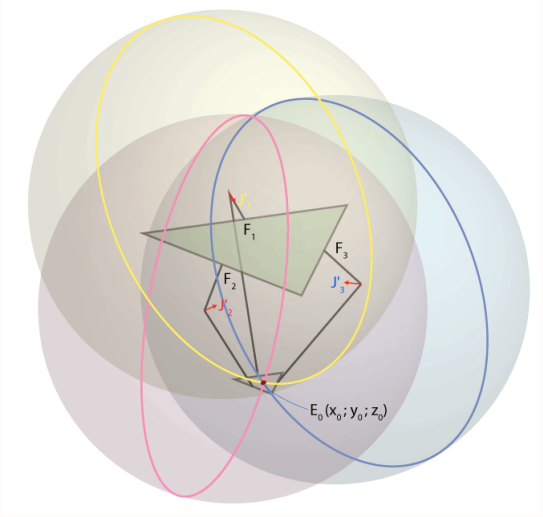

Fig. 10. Solving the direct kinematics problem [25].

Based on the above, to determine the coordinates of point E_0 we need to solve a system of three equations of spheres:

$$\begin{cases} (x_0 - x_{J'1})^2 + (y_0 - y_{J'1})^2 + (z_0 - z_{J'1})^2 = {r_e}^2 \\ (x_0 - x_{J'2})^2 + (y_0 - y_{J'2})^2 + (z_0 - z_{J'2})^2 = {r_e}^2 \\ (x_0 - x_{J'3})^2 + (y_0 - y_{J'3})^2 + (z_0 - z_{J'3})^2 = {r_e}^2. \end{cases} \quad (18)$$

Coordinates of points J_1, J_2, and J_3 are easy to determine, because the coordinates of F_1, F_2, F_3, length of the active arm r_f and angles θ_1, θ_2, and θ_3 are known, and already starting from these, find J'_1, J'_2, and J'_3.

E. Experimental Setup and Procedures

The experimental setup shown in Fig. 11 consists of a 3D-printed delta robot with three servo drives, an Arduino Uno debug board, a PWM servo module, a DC-DC (direct current to direct current) step-down converter, and a computer.

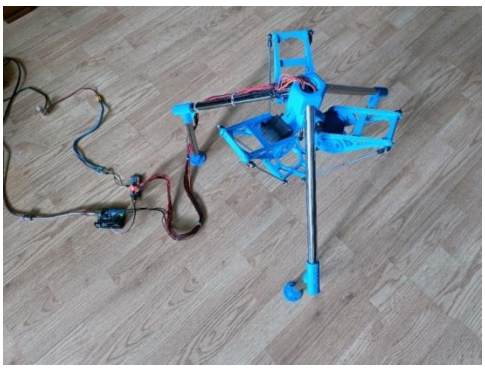

Fig. 11. Experimental setup.

The Arduino Uno board performs trajectory calculation and forward and reverse kinematics, as well as controls the servos via a PWM module. The Arduino Uno also sends to the computer the calculated coordinates and angles of the motors, the measured motor angles, and the coordinates calculated from the actual angles.

To determine the area in which the working body of the assembled delta robot can be located, its working space was calculated. The direct kinematics equations were implemented in Python, and the matplotlib module was used for visualization.

During the experiments, the delta robot made four consecutive three-point motions:

$$P_0 \rightarrow P_1 \rightarrow P_0 \rightarrow P_2 \rightarrow P_0,$$

$$P_0(0, 0, -150),$$

$$P_1(50, 50, -250),$$

$$P_2(-50, -50, -250).$$

The trajectories between the given points were calculated in task and joint space. The duration of each movement was set to 0.5 s, which is a total of 2 s. The trajectories are shown in Fig. 12.

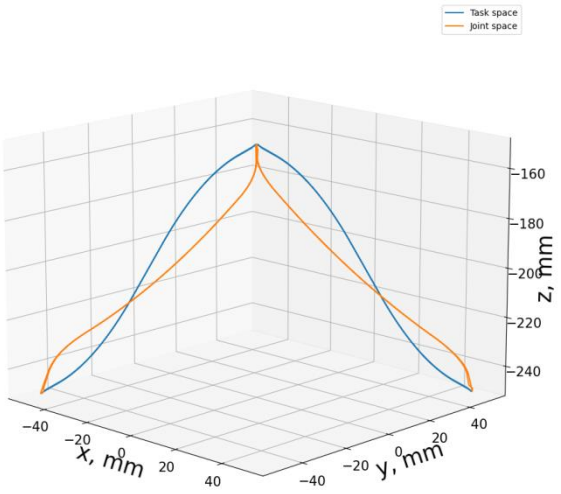

Fig. 12. Task space (blue line) and joint space (orange line) trapezoidal velocity trajectories.

The trajectory planning, inverse and forward kinematics algorithms were implemented in the control system. The calculations and measurements were performed at a frequency of 100 Hz.

In each cycle, the microcontroller calculated the joint space trajectory by the active arms' initial and final positions, and the end effector's calculated coordinates were determined using the forward kinematics. Voltages from three potentiometers were also measured, and the actual positions of the active arms were determined using these voltages. Using inverse kinematics, the actual coordinates of the end effector were determined. All data in each cycle were transferred to a personal computer.

The experiments for the task space trajectory were carried out in the same way. The only difference was that the trajectory was calculated using the end effector's initial and final coordinates, and the active arms' calculated positions were determined using inverse kinematics.

Algorithm flowcharts of both joint space and task space trajectory planning control systems are shown in Fig. 13.

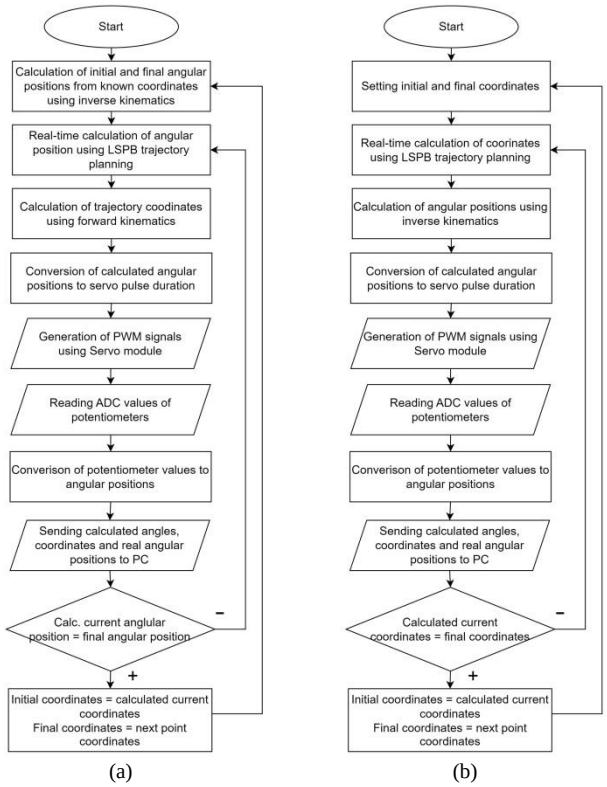

Fig. 13. Control system algorithm flowcharts for joint space (a) and task space (b) trajectory planning.

The developed algorithms of the delta robot control system occupied about 40% of the program memory and 30% of the RAM (random access memory) of Arduino Uno, and the execution time per cycle was 4232 and 9752 microseconds for joint space and task space trajectories, respectively.

The potentiometer voltages shown in Section II.B were obtained in static positions. During the movement of the active arms, the signals obtained from the potentiometers are accompanied by noise. To get correct results, a low-pass Butterworth filter was used, as this filter is known as a maximally flat magnitude filter and is one of the most commonly used filters for motion analysis [26]. To determine the filter's cutoff frequency, a spectral analysis of the signal was performed in a Python environment using the scipy.fftpack module, which is shown in Fig. 14.

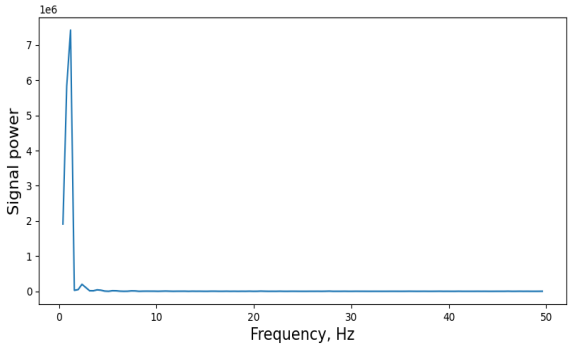

Fig. 14. Spectral analysis of potentiometer signal during motion.

Based on the obtained plot, it can be concluded that the useful signal has frequencies below 10 Hz. This frequency was used as the cut-off frequency.

Signal filtering was also performed in the Python environment using the scipy. signal module. The raw and filtered signal for one of the active arms is shown in Fig. 15.

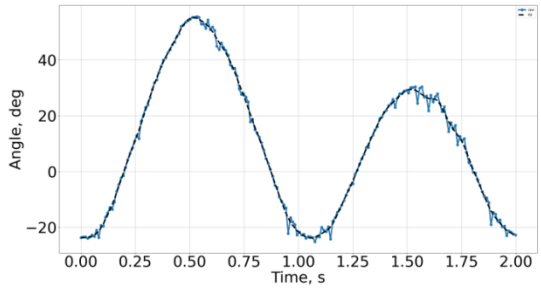

Fig. 15. Raw (line with dots) and filtered (dashed) angular position of active arm.

This procedure was carried out for all active arm angular position data obtained from the potentiometers.

III. Results

A. Workspace Visualization

Fig. 16 shows the workspace of the delta robot, which has a conical shape with a height of about 175 mm and a diameter of about 300 mm.

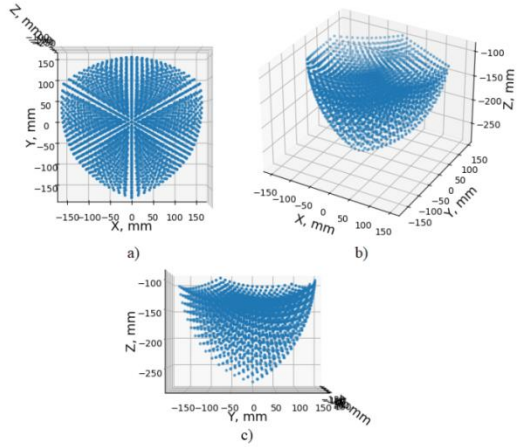

Fig. 16. Delta-robot workspace: (a) YX plane; (b) 3D view; (c) YZ plane.

The workspace diameters for heights (Z axis) in 20 mm increments are shown in Table IV.

TABLE IV. Workspace Diameters for Heights

Height, mm	Diameter, mm
–100	314
–120	320
–140	310
–160	296
–180	280
–200	254
–220	224
–40	186
–260	134
–280	52

B. Joint Space Trajectory Calculations and Experiment Results

Fig. 17 shows the angular positions, velocities, and accelerations of the joint space trajectories of the active arms. The maximum angular velocity and acceleration were 242 °/s and 1600 °/s^2.

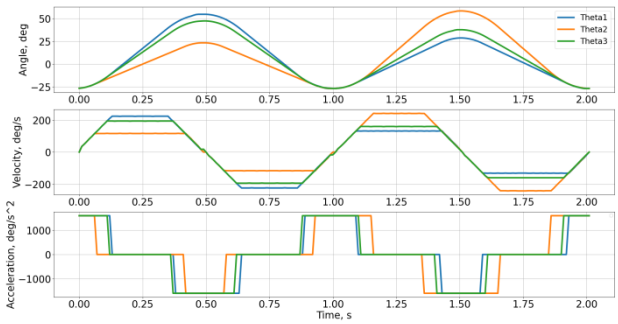

Fig. 17. Joint space trajectories: calculated angular positions, velocities, and accelerations of active arm.

Fig. 18 shows the linear positions, velocities, and accelerations of the joint space trajectories of the end effector calculated using direct kinematics. The maximum linear velocity was achieved in the *Z* axis and was 278 mm/s, and the absolute velocity reached 404 mm/s. The maximum acceleration was achieved in the *Y* axis and was 4000 mm/s^2.

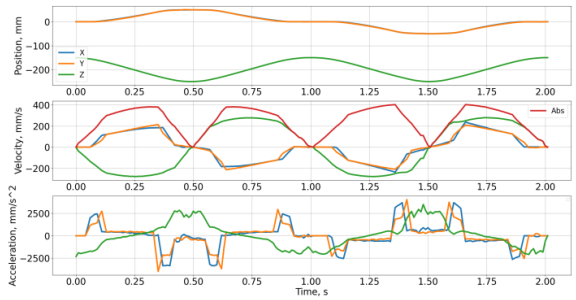

Fig. 18. Joint space trajectories: calculated linear positions, velocities, and accelerations of end effector.

Fig. 19 shows the calculated and actual joint space trajectories in a three-dimensional view.

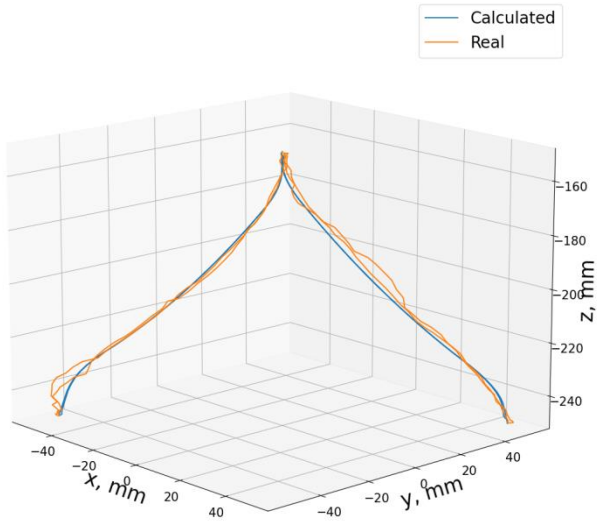

Fig. 19. Joint space trajectories: calculated (blue line) and actual (orange line).

Fig. 20 shows the errors between the specified and experimental active arm angles. The average error for all data was 0.09%.

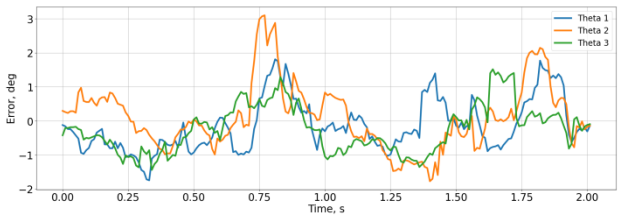

Fig. 20. Angular error by θ_1, (blue), θ_2 (orange) and θ_3 (green).

Fig. 21 shows the positioning error for the three coordinates and the absolute error. The average value of the absolute error was 2.238 mm.

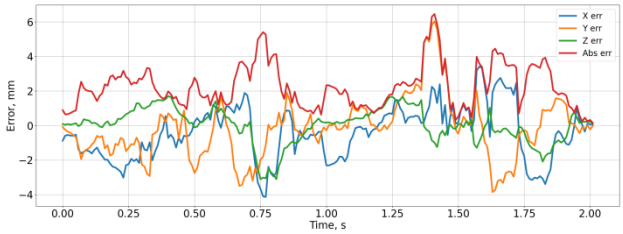

Fig. 21. Position error by X (blue), Y (orange), Z (green), and absolute (red).

C. *Task Space Trajectory Calculations and Experiment Results*

Fig. 22 shows the linear positions, velocities, and accelerations of the end effector trajectories. The maximum linear velocity was achieved in the Z axis and was 355 mm/s, and the absolute velocity reached 391 mm/s. The maximum acceleration was 1600 mm/s^2 in all axes.

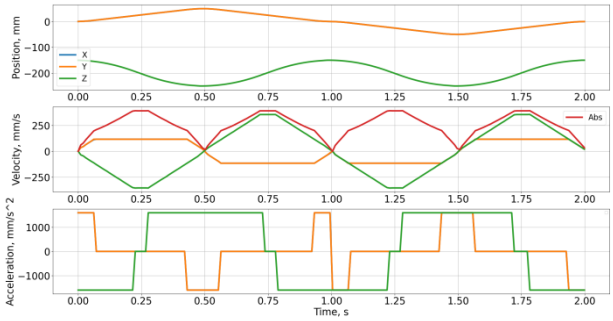

Fig. 22. Task space trajectories: calculated linear positions, velocities, and accelerations of the end effector.

Fig. 23 shows the angular positions, velocities, and accelerations of the active arms of the task space trajectories calculated using inverse kinematics. The maximum angular velocity and acceleration were 255 °/s and 3400 °/s^2.

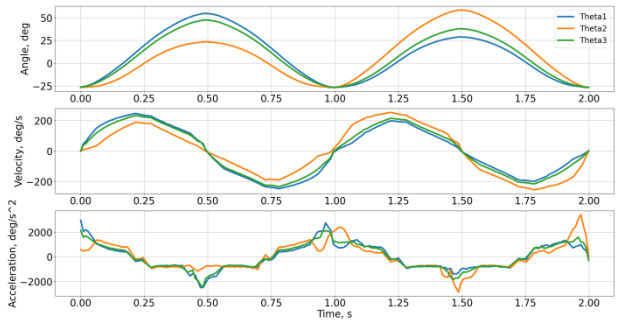

Fig. 23. Task space trajectories: calculated angular positions, velocities, and accelerations of active arm.

Fig. 24 shows the calculated and actual task space trajectories in a three-dimensional view.

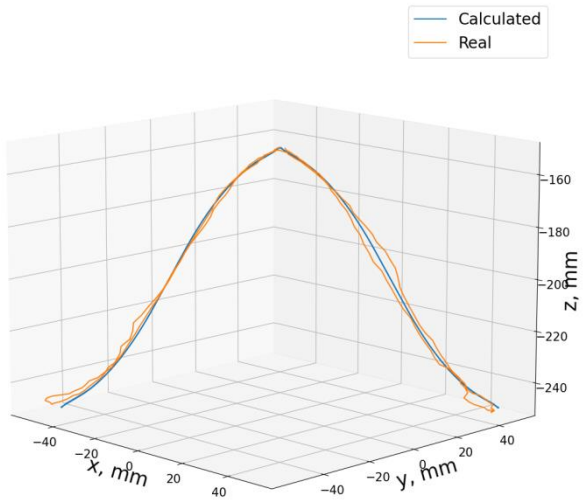

Fig. 24. Task space trajectories: calculated (blue line) and actual (orange line).

Fig. 25 shows the positioning error for the three coordinates and the absolute error. The average value of the absolute error was 2.199 mm.

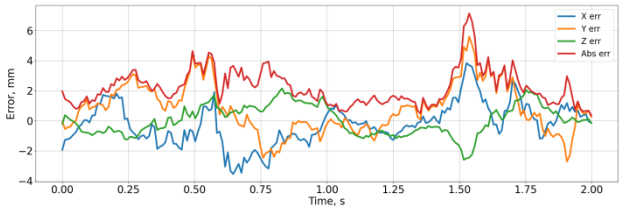

Fig. 25. Position error by *X* (blue), *Y* (orange), *Z* (green), and absolute (red).

Fig. 26 shows the errors between the specified and experimental active arm angles. The average error for all data was 0.11%.

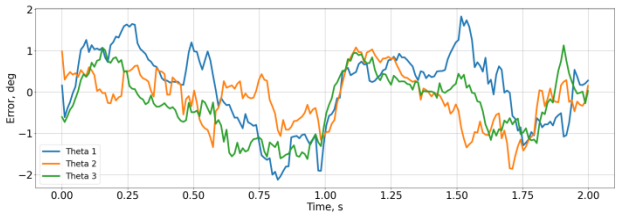

Fig. 26. Angular error by θ1, (orange), θ2 (red) and θ3 (green).

D. Load Carrying Capacity Test

To evaluate the load capacity of the assembled delta robot, tests were performed with different objects attached to the moving platform. The objects used were Halls candies, 18,650, 21,700 li-ion batteries, and 18,650 battery packs. Table IV shows the list of objects used for the tests.

TABLE IV. LIST OF OBJECTS USED FOR THE TESTS

N	Object name	Dimensions, mm	Weight, g
1	Halls candies	18×18×90	27
2	18650 li-ion battery	18×18×65	50
3	21700 li-ion battery	21×21×70	70
4	2s 18650 battery pack	36×18×65	100
5	4s 18650 battery pack	36×36×65	200

During the tests, the delta robot produced pick and place motions with a load of up to 200 g and showed similar results to those obtained in the unloaded tests. It should be noted that the weight of the entire delta robot structure is 1107 g. Thus, the robot can operate objects whose mass is 18.2% of its own.

IV. DISCUSSIONS

During this work, a delta robot was assembled by additive manufacturing. The design of the robot was based on a publicly available design. The design used was designed for MG995/996 servo drives. Some of the most low-cost and affordable components were chosen for the delta robot's control system. Trapezoidal trajectory planning, inverse and forward kinematics were used in the control system algorithm.

To evaluate the performance of the delta robot, experiments with joint space and task space trajectory calculations were performed. During the experiments, the robot simulated pick-and-place movements. The total motion duration was set to 2 s, corresponding to a speed of 0.5 pick-and-place motions per second. The signals of the potentiometers installed in the servo drives were used to obtain experimental data. Also, the calculation and visualization of the workspace were performed using Python and the matplotlib module.

As a result of the experiments, the absolute linear velocities of the end effector reached 404 mm/s for the joint space trajectory and 391 mm/s for the task space, with absolute positioning errors of 2.238 mm and 2.199 mm, respectively.

The mean angular errors were 0.09° and 0.11° for the joint and task space trajectories, respectively. This error is due to the dead bandwidth parameter of the servo drive. The servo used in this work has a dead bandwidth of 5 ms, which allows the motor to be controlled in steps of 0.09°. Thus, it is necessary to select servos with smaller dead bandwidth value to obtain better positioning accuracy.

The positioning errors determined during the experiments for the joint space and task space trajectories were comparable, but the end-effector trajectory was preferable for the joint space trajectory, as it allows for top capture, while for the task space trajectory, the end-effector approaches the endpoint from the side. It should also be noted that for task space, the trajectory does not consider the constraints of the servos, and the calculated speeds and accelerations may exceed the capabilities of the motors, and the joint space trajectory planning is preferable for this reason as well.

The payload capacity tests of the assembled delta robot were also carried out. The robot could perform pick and place movements with a load of up to 200 g, which is 18.2% of its weight.

The workspace of the robot has a traditional for delta robot conical shape, and its dimensions were 175 mm in height and 300 mm in diameter.

Table V compares the parameters of the assembled delta robot with the commercial ones.

TABLE V. DELTA ROBOTS' PARAMETERS AND PRICE COMPARISON

Model	Accuracy, mm	Workspace, mm × mm	Rated payload, kg	Weight, kg	Payload to weight ratio, %	Picks per minute (PPM)	Cost, USD	PPM per USD
Warsonco WSC-600DJ [5]	0.02	600 × 250	3	40	7.5	180	From 7.519	0.024
igus DLE-DR-0005 [6]	0.5	360 × 120	1	15	6.7	60	5.598	0.01
Delta X 1 [21]	0.2	340 × 150	0.5	3.5	14.3	N/A	399	N/A
Proposed delta robot	2.2	300 × 175	0.2	1.1	18.2	30	60.5	0.49

Undoubtedly, commercial models are superior to the delta robot assembled in this work, but its parameters are quite satisfactory considering its ease of assembly, availability of components, and cost-effectiveness. The delta robot made in this paper has 20 times the PPM per USD ratio of the Warsonco model and 49 times the ratio of the igus model, while the PPM rate of Delta X 1 robot is unavailable. Payload to weight ratio of the proposed robot is more than double that of both industrial models, and slightly higher than Delta X 1 robot.

Compared to the cheapest robot in the table, the proposed robot still has a significant price advantage, not considering shipping costs and taxes. Also, the proposed method allows to obtain a prototype in a very short time, while the delivery and clearance time of commercial solutions can reach a significant period.

V. Conclusion

The conducted experiments demonstrate the successful realization and evaluation of the 3D-printed delta robot. Considering the positioning accuracy of the proposed robot, it cannot be applied to tasks where high accuracy is required, but it still has potential for applications that do not demand high positioning accuracy.

One such application is material handling and packaging tasks where high precision is not critical, in particular, the robot described in this paper has been assembled to conduct studies using machine vision for its further application in sorting candies of different flavors.

In addition, researchers and engineers can use such platforms as a rapid prototyping tool to test and validate algorithms and control strategies.

It should also be noted that the robot's smooth and coordinated movements make it suitable for interactive art installations, light painting, or creative displays, where precise positioning may not be the primary concern.

In addition to the above, this platform can be applied to the study and experimentation of Swarm Robotics, where multiple robots are needed, and the cost of each instance becomes even more important [27].

Of course, this approach has disadvantages in the form of relatively lower positioning accuracy and speed. These disadvantages can be mitigated by making more significant investments: using higher-quality filaments, accurate and fast servo drives, and high-performance microcontrollers.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization: A.A. and G.B.; methodology: A.A.; software: A.A.; validation: A.A.; formal analysis: A.A.; investigation: A.A.; resources: A.A.; data curation: A.A.; writing—original draft preparation: A.A.; writing—review and editing: A.A., K.G. and A.Z.; visualization: A.A.; supervision: S.O.; project administration: G.B.; funding acquisition: A.A. All authors have read and agreed to the published version of the manuscript.

References

[1] R. Clavel, "Design of a fast parallel robot with 4 degrees of freedom," *EPFL, Lausanne*, Switzerland, 1991. (in French)

[2] M. López, E. Castillo, G. García, and A. Bashir, "Delta robot: Inverse, direct, and intermediate Jacobians," in *Proc. Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Scienc*e, Jan. 2006, vol. 220, no. 1, pp. 103–109.

[3] N. Cretescu, M. Neagoe, and R. Saulescu, "Dynamic analysis of a delta parallel robot with flexible links and Joint clearances," *Applied Sciences*, vol. 13, no. 11, 6693, May 2023.

[4] Z. Guo, J. Zhang, and P. Zhang, "Research on the residual vibration suppression of delta robots based on the dual-modal Input shaping method," *Actuators*, vol. 12, no. 2, 84, Feb. 2023.

[5] P. Huang, H. Z. Huang, Y. F. Li, and H. Li, "Positioning accuracy reliability analysis of industrial robots based on differential kinematics and saddlepoint approximation," *Mechanism and Machine Theory*, vol. 162, 104367, Aug. 2021.

[6] S. Wang, X. Deng, H. Feng, K. Ren, F. Li, and Y. Liu, "Dynamic simulation analysis and experimental study of an industrial robot with novel joint reducers," *in Review*, vol. 2, 2022.

[7] L. X. Xu, B. K. Chen, and C. Y. Li, "Dynamic modelling and contact analysis of bearing-cycloid-pinwheel transmission mechanisms used in joint rotate vector reducers," *Mechanism and Machine Theory*, vol. 137, pp. 432–458, Jul. 2019.

[8] Delta Robots Manufacturer. [Online]. Available: https://www.warsoncorobot.com/delta-robot/

[9] Drylin delta pick and place robot. [Online]. Available: https://www.igus.com/robolink/drylin-delta-robot?sort=3&inch=false

[10] Industrial Products. [Online]. Available: https://rbtx.com/en-GB/components/robots?lang=ko-KR&filter=range%2Cprecision%2C0%2C400&filter=categories%2Ccategory%2C0990ce3b-e712-4a03-921d-fb7658f8a8e7

[11] S. Patil, S. C. Alvares, P. Mannam, O. Kroemer, and F. Z. Temel, "DeltaZ: An accessible compliant delta robot manipulator for research and education," in *Proc. 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kyoto, 2022, pp. 13213–13219.

[12] P. Mannam, A. Rudich, K. Zhang, M. Veloso, O. Kroemer, and F. Temel, "A low-cost compliant gripper using cooperative mini-delta robots for dexterous manipulation," *Robotics: Science and Systems XVII, Robotics: Science and Systems Foundation*, vol. 3, 2021.

[13] K. Zhou, Z. Meng, M. He, J. Hou, and T. Li, "Design and test of a sorting device based on machine vision," *IEEE Access*, vol. 8, pp. 27178–27187, 2020.

[14] EEZYbotDELTA by daGHIZmo Thingiverse. [Online]. Available: https://www.thingiverse.com/thing:1249297

[15] Delta Robot – Yazzo PolyBot Derivative by Renosis – Thingiverse. [Online]. Available: https://www.thingiverse.com/thing:7554

[16] Delta Robot v2 with three arms by i-make-robots – Thingiverse. [Online]. Available: https://www.thingiverse.com/thing:263500

[17] Servo Delta Robot by tinkersprojects – Thingiverse. [Online]. Available: https://www.thingiverse.com/thing:3223787

[18] L. Biagiotti and C. Melchiorri, *Trajectory Planning for Automatic Machines and Robots*, Berlin Heidelberg: Springer, 2008, pp. 58–76.

[19] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," in *Motion and Operation Planning of Robotic Systems*, G. Carbone and F. Gomez-Bravo, Eds. Mechanisms and Machine Science, vol 29. Springer, Cham., 2015.

[20] M. Massaro, S. Lovato, M. Bottin, and G. Rosati, "An optimal control approach to the minimum-time trajectory planning of robotic manipulators," *Robotics*, vol. 12, no. 3, p. 64, Apr. 2023.

[21] Delta X Basic Kit – Delta X Robot Store. [Online]. Available: https://deltaxstore.com/products/delta-x-basic-kit?variant=35965594239143

[22] A. Milovanović *et al*., "Comparative analysis of printing parameters effect on mechanical properties of natural PLA and advanced PLA-X material," *Procedia Structural Integrity*, vol. 28, pp. 1963–1968, 2020.

[23] B. D. M. Matos *et al.*, "Evaluation of commercially available Polylactic Acid (PLA) filaments for 3D printing applications," *J Therm. Anal. Calorim.*, vol. 137, no. 2, pp. 555–562, Jul. 2019.
[24] P. J. Z. Murray, "Descriptive geometric, kinematic analysis of clavel's delta robot," *Centre of Intelligent Machine*s, McGill University, 2004.
[25] Kinematics of a delta robot. [Online]. Available: https://habr.com/ru/articles/390281/
[26] M. Shouran and E. Elgamli, "Design and implementation of butterworth filter," *IJIRSET*, vol. 9, 2020.
[27] L. Bayindir and E. Sahin, "A review of studies in swarm robotics," *Turk. J. Elec. Engin.*, vol. 15, Jan. 2007.