IE410-Introduction to Robotics

PROJECT 1 – INVERSE KINEMATICS

Kartavya Akabari - 202201213

Divyesh Ramani - 202201241

Janvi Ramani - 202201158

Smruti Parmar - 202201008

1 . Aim of the experiment:

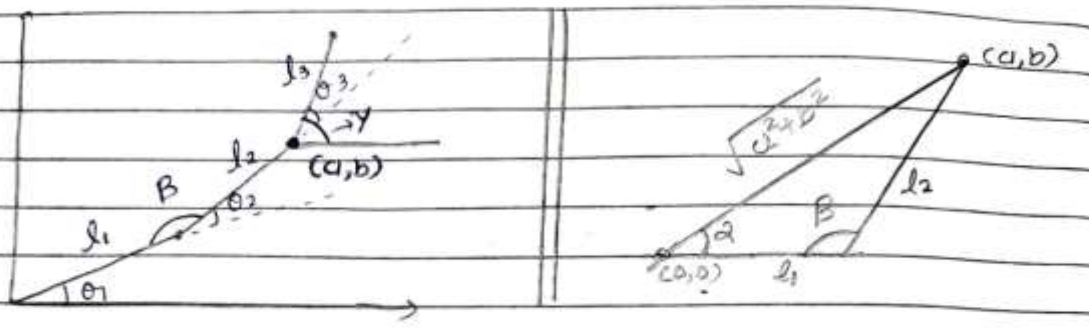
- ❖ How to deal with inverse Kinematics using Robot arm manipulation.
- ❖ Given the point (x, y) (co-ordinate of end effector) and the total angle γ determine the value of $\theta_1,\ \theta_2\ and\ \theta_3$.

2 . Calculation:

❖ Image:

★ We are given the destination coordinates (x, y) and γ.

⇒ Here is the geometric interpritation of 3 link arm



$$a = x - l_3\cos(\gamma)$$
$$b = y - l_3\sin(\gamma)$$

$$\alpha = \cos^{-1}\left(\frac{l_1^2 + (a^2+b^2) - l_2^2}{2l_1\sqrt{a^2+b^2}}\right)$$

$$\beta = \cos^{-1}\left(\frac{l_1^2 + l_2^2 - (a^2+b^2)}{2l_1l_2}\right)$$

$$\boxed{\theta_2 = \pi - \beta}\ ,\ \boxed{\theta_1 = \tan^{-1}\left(\frac{b}{a}\right) - \alpha}\quad \boxed{\theta_3 = \gamma - \theta_1 - \theta_2}$$

❖ ***Explanation of the above calculation:***

- Given: L1,L2,L3 (length of the links), x, y (final co-ordinates), and gamma which is equal to θ_1+θ_2+θ_3.
- So, from the above image we can infer that (a, b) can be determined by the projection of L3 with angle γ on the x-axis and y-axis respectively.
- Now we have the points (a, b) which form a triangle of sides L1, L2 and $\sqrt{a^2 + b^2}$.
- Using the inverse kinematics of 2 degrees of freedom (using the cosine rule) we can determine the angles α and β.
- Now from the geometry (shown in the above image) we can derive

$$\theta_1 = \pi - \beta$$
$$\theta_2 = \tan^{-1}\left(\frac{b}{a}\right) - \alpha$$
$$\theta_3 = \gamma - \theta_1 - \theta_2$$

- We can use these equations in Arduino code to move our robot arm to a desirable position using inverse kinematics.

3 . Codes:

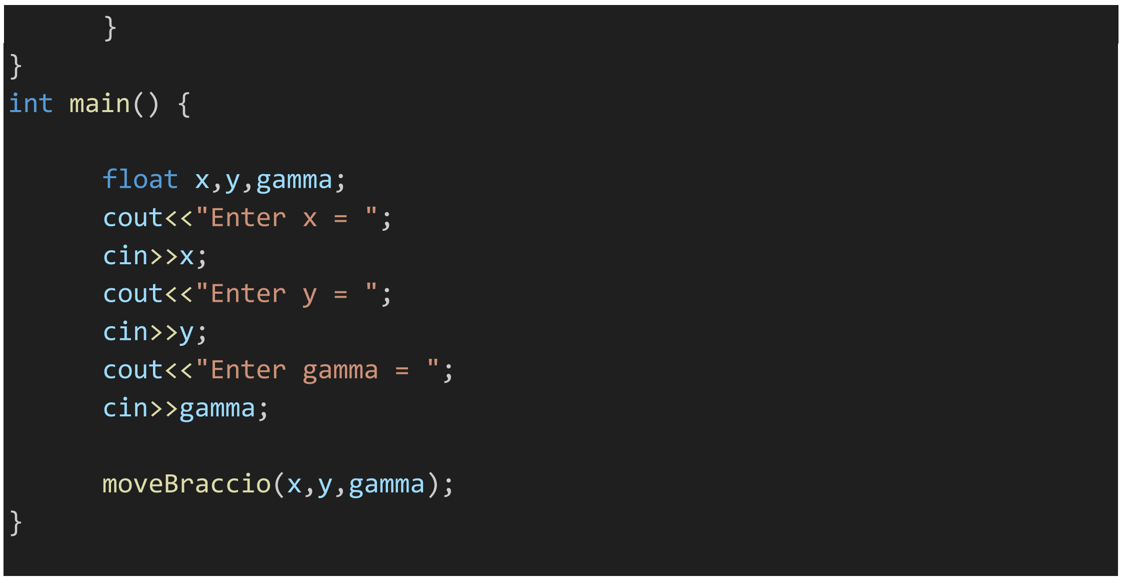
❖ We have written two codes for this project one in the C++ and another in ino.

❖ Cpp code we used for our simplicity and determine the output of $\theta_1,\ \theta_2\ and\ \theta_3$ to verify the position of the robot arm.

❖ Using the ino code we change the movement of the robot arm by assigning adjusting values of $\theta_1,\ \theta_2\ and\ \theta_3$.

❖ ***CPP Code:***

```
#include<iostream>
#include<cmath>
using namespace std;
define pi 3.14159265359

const float L1 = 12.5; // Length of link 1 (shoulder to elbow)
const float L2 = 12.5; // Length of link 2 (elbow to wrist)
const float L3 = 7.15; // Length of link 3 (wrist to end-effector)

float radian(float theta){      //will convert degree to radian
      return theta*pi/180;
}

float degree(float theta){      //will convert radian to degree
      return theta*180/pi;
}

void moveBraccio(float x, float y, float gamma) {
      // Calculate position of point (a,b)
      float a = x - (L3 * cos(radian(gamma)));
      float b = y - (L3 * sin(radian(gamma)));
      //C is distance of point (a,b) to origin
      float C = sqrt(pow(a, 2) + pow(b, 2));
      // Check if position is within reachable workspace
      if ((L1 + L2) >= C) {
   // Calculate angles
      //alpha and beta are mentioned above in the picture
            float alpha = degree(acos((pow(L1, 2) + pow(C, 2) - pow(L2,
2)) / (2 * L1 * C)));
            float Beta = degree(acos((pow(L1, 2) + pow(L2, 2) - pow(C,
2)) / (2 * L1 * L2)));


             //calculating the angles which we really need form inverse
kinematics of 2dof
            float theta1 = degree(atan(b/ a)) - alpha;
            float theta2 = 180 - Beta;
            float theta3 = gamma - theta1 - theta2;

      // printing the values for our simplicity and verifying the final
answer

            cout<<"theta 1 = "<<theta1<<"\n";
            cout<<"theta 2 = "<<theta2<<"\n";
            cout<<"theta 3 = "<<theta3<<"\n";
```



```cpp
    }
}
int main() {

    float x,y,gamma;
    cout<<"Enter x = ";
    cin>>x;
    cout<<"Enter y = ";
    cin>>y;
    cout<<"Enter gamma = ";
    cin>>gamma;

    moveBraccio(x,y,gamma);
}
```

❖ ***<u>Explanation of the above CPP Code:</u>***

- ❖ In the cpp code we globally declared the size of the 3 links L1, L2 and L3 which will remain constant through out the entire experiment.
- ❖ We have taken x, y and gamma as inputs from the user for the cpp code only we have not taken it into the ino code.
- ❖ Then, we called moveBraccio function to calculate the values of theta1, theta2, theta3.
- ❖ We, also globally declared two functions which convert radian angle to degrees and degrees to radian.
- ❖ In, the moveBraccio we first calculate the point (a, b) and distance C from the origin to the point (a, b).
- ❖ Then using the concept of 2-Degree of inverse kinematics, the formula which we have already derived into our calculation and the inbuild cmath library in cpp we determine the values of theta1, theta2 and theta3.
- ❖ Then, we printed out its value to verify the position of robot arm.
- ❖ We also include some comments to make our code understandable.

❖ ***Arduino Code (ino):***

```cpp
#include <Braccio.h> // Include the Braccio library
#include<Servo.h>


Servo base;
Servo shoulder;
Servo elbow;
Servo wrist_rot;
Servo wrist_ver;
Servo gripper;


// Define constants for link lengths (in cm) for the Braccio arm
const float L1 = 12.5; // Length of link 1 (shoulder to elbow)
const float L2 = 12.5; // Length of link 2 (elbow to wrist)
const float L3 = 7.15; // Length of link 3 (wrist to end-effector)

void setup() {
      Serial.begin(9600); // Initialize serial communication
      delay(1000);
      Braccio.begin(); // Initialize Braccio
}

void loop() {
      //  user to enter end-effector coordinates and orientation randomly
      float x = 10, y = 20, gamma=90;

      // Call inverse kinematics function and move the Braccio arm
      moveBraccio(x, y, gamma);

      delay(1000); // Wait for 1 second before next iteration
}

void moveBraccio(float x, float y, float gamma) {
      //we did the same thing mentioned into the picture finding point (a,b)
      float a = x - (L3 * cos(radians(gamma)));
      float b = y - (L3 * sin(radians(gamma)));
      float C = sqrt(pow(a, 2) + pow(b, 2));

  // Check if position is within reachable workspace
  if ((L1 + L2) > C) {
    // Calculate angles
      float alpha = degrees(acos((pow(L1, 2) + pow(C, 2) - pow(L2, 2)) / (2 *
L1 * C)));
```

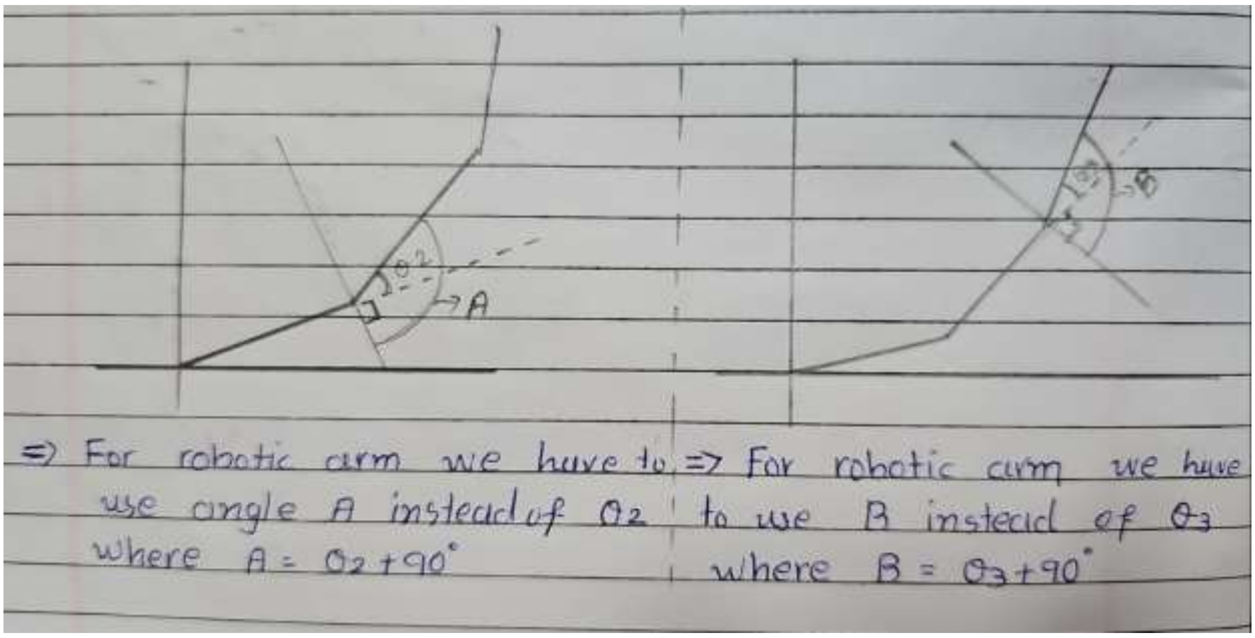
```
      float Beta = degrees(acos((pow(L1, 2) + pow(L2, 2) - pow(C, 2)) / (2 *
L1 * L2)));

      // finding the final theta1, theta2 and theta3 angles of shoulder,
elbow and wrist respectively.
      float theta1 = degrees(atan2(b, a)) - alpha;

      float theta2 = 180 - Beta;
      float theta3 = gamma - theta1 - theta2;

      // Set servo angles and move the Braccio arm
      //why we added 90 degrees angle in M3 and M4 mentioned into the detains
into the explanation of code

      Braccio.ServoMovement(20,0,int(theta1),90+int(theta2),90+int(theta3),0,
73);

    }
}
```

❖ ***Brief Explanation of the above Arduino Code (ino):***

- This code is the same as cpp code.
- But in this code instead of taking the x, y, and gamma from the user we assign it directly into the code and then run the code into the Arduino ide.
- We assign those parameters as arguments of moveBraccio function.
- In, moveBraccio function we did the same thing mentioned in the image and cpp code, found the point (a, b) then using inverse kinematics of two degree of freedom calculated alpha and beta then using that alpha and beta determine the angle theta1, theta2 and theta3.
- In, BraccioMovement function we assign 20 step delay time, angle of M1=90, M2=theta1, M3=theta2+90, M4=theta3+90, M5 and M6 are don’t care values or zero contribution in this experiment. (Not indeed needed in out experiment so we assigned it 0 and 73).

- **Why we used theta2+90 and theta3+90 in the angles of M3 and M4?**
- Refer to the image shown below.

❖ Image for clarification of (θ_1+ 90) and (θ_3+90):



⇒ For robotic arm we have to use angle A instead of θ_2 where $A = \theta_2 + 90^\circ$

⇒ For robotic arm we have to use B instead of θ_3 where $B = \theta_3 + 90^\circ$

- **Explanation of above question:**
    - The robot arm's M3 or elbow servo motor always measures the angle for its next link which is link2 in our case from the perpendicular axis from the link1's end point.
    - Similarly, the same case for the robot arm's M4 or horizontal wrist always measures the angle of its next link (link 3 in our case) from the perpendicular axis from the link2's endpoint.

- In, ServoMovement function M2 measures the angle from the perpendicular axis of the link1 so we need to add 90 degrees to obtain the exact position of the robot arm.

- Similarly, M3 measures the angle from the perpendicular axis of the link2 so we need to add 90 degrees to obtain the exact position of the robot arm.
- After the adjustment of the angles for M2 and M3 we easily move the robot arm to the point (x, y) desired position. Which we will show in the next test cases.
- In the practical experiments we also printed the values of the theta, theta2 and theta3 in the serial monitor for the verification of the angles we obtained by calculations.

4 . ***Material Used:***

- ❖ Arduino Uno
- ❖ Braccio Robot arm
- ❖ Adapter (for power supply)
- ❖ Arduino-compatible shield
- ❖ Programming environment (Arduino IDE)

5 . **Method/procedure:**

- ❖ After writing the above code in the Arduino IDE environment and uploading it into the Arduino UNO, connect the compatible shield with Arduino UNO and supply the power through the adapter to the compatible shield.
- ❖ Now, we randomly generate the points (x, y) and the total angle gamma which are in the range of the 3-link Braccio Arm.
- ❖ Arduino code will calculate the desired parameters theta1, theta2 and theta3 and then move the Robot arm in such a way that we get the end effector of the robot arm at the point (x, y).

- As we displayed all the angles $\theta_1,\ \theta_2\ and\ \theta_3$ in the console window or the serial output window of Arduino ide, we verified these values with cpp code, with calculations and with experiments.

6 . *Test – Cases:*

- We have done 4 test cases and in each test case we have found out the values of $\theta_1, \theta_2\ and\ \theta_3$ using the cpp code Arduino code and with the hand-written analysis.
- Also, verified that all the values are the same for each test-case or experiment number.
- The format of these experiments are following:

Experiment Number-n:

Point of the end-effector: (x, y) *(in centi-meters)*
Value of γ $(in\ degrees)$

- *Obtained Values:*
    - θ_1
    - θ_2
    - θ_3

- Serial Output / Console of the Arduino IDE:
- CPP output:
- Handwritten Analysis:
- Image of the final position of the braccio Robot Arm:

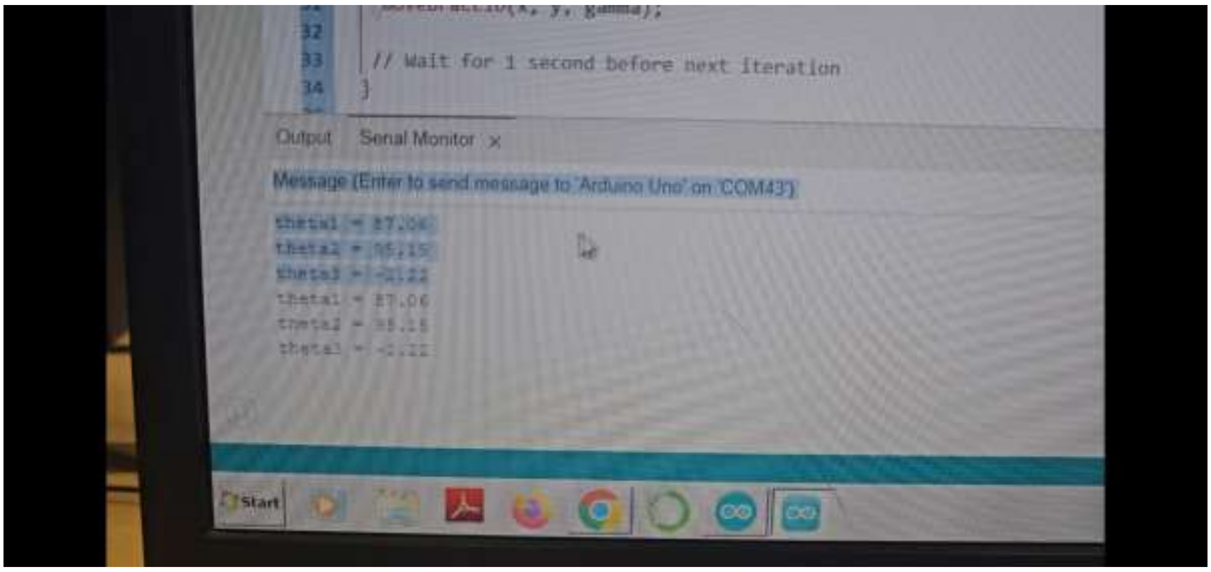
❖ *Experiment Number-1:*

Point of the end-effector: (x, y) = (-19, 12.5) *(in centi-meters)*

Value of $\gamma = 180\ (in\ degrees)$

- *Obtained Values:*
    - θ_1 = 87.02 (in degrees)
    - θ_2 = 92.9032 (in degrees)
    - θ_3 = 0.077 (in degrees)

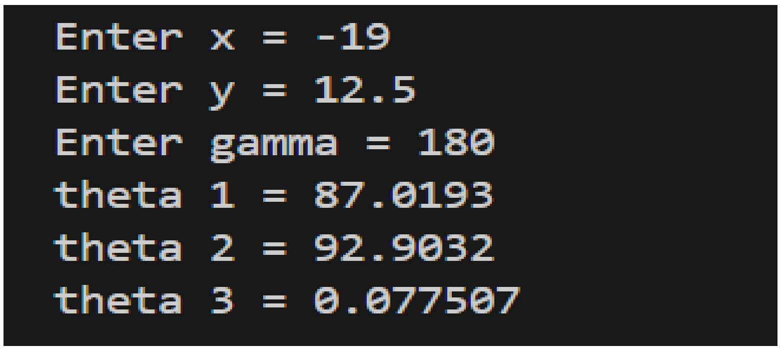
- Serial Output / Console of the Arduino IDE:



```
    [illegible](x, y, gamma);

    // Wait for 1 second before next iteration
}
```

```
theta1 = 87.06
theta2 = 95.15
theta3 = -2.22
theta1 = 87.06
theta2 = 95.15
theta3 = -2.22
```

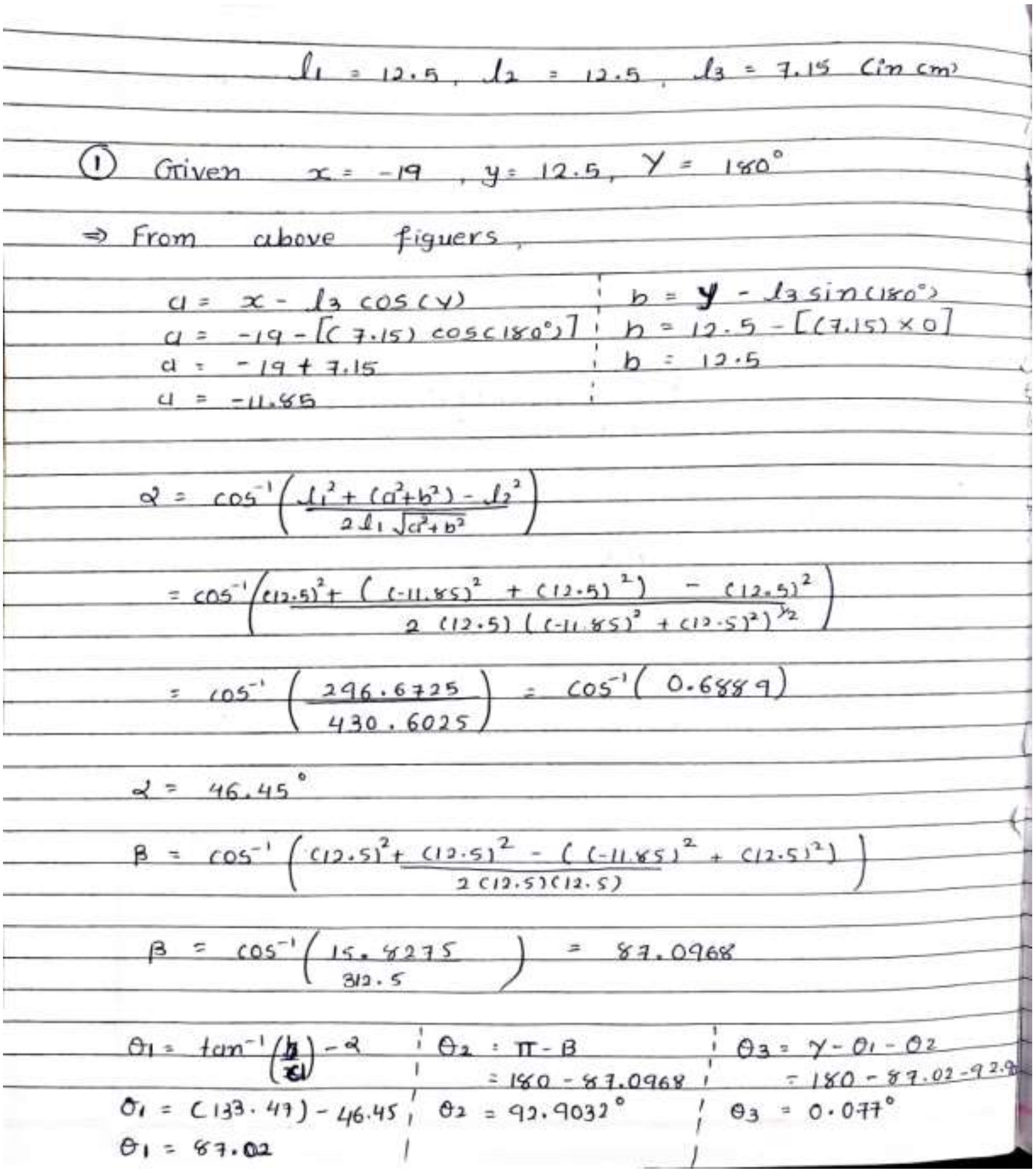
- CPP output:



```
Enter x = -19
Enter y = 12.5
Enter gamma = 180
theta 1 = 87.0193
theta 2 = 92.9032
theta 3 = 0.077507
```

- ➢ Here we see a slight error in the output of the both codes because me mistakenly taken y=12 instead of y=12.5 in Arduino IDE code.
- ➢ However, the resulting values won't affect the position of the robot arm too much.

- <u>Handwritten Analysis:</u>



$l_1 = 12.5,\ l_2 = 12.5,\ l_3 = 7.15$ (in cm)

① Given $x = -19$, $y = 12.5$, $\gamma = 180^\circ$

⇒ From above figuers,

$a = x - l_3 \cos(\gamma)$
$a = -19 - [(7.15)\cos(180^\circ)]$
$a = -19 + 7.15$
$a = -11.85$

$b = y - l_3 \sin(180^\circ)$
$b = 12.5 - [(7.15) \times 0]$
$b = 12.5$

$$\alpha = \cos^{-1}\left(\frac{l_1^2 + (a^2+b^2) - l_2^2}{2 l_1 \sqrt{a^2+b^2}}\right)$$

$$= \cos^{-1}\left(\frac{(12.5)^2 + ((-11.85)^2 + (12.5)^2) - (12.5)^2}{2(12.5)((-11.85)^2 + (12.5)^2)^{1/2}}\right)$$

$$= \cos^{-1}\left(\frac{296.6725}{430.6025}\right) = \cos^{-1}(0.6889)$$

$\alpha = 46.45^\circ$

$$\beta = \cos^{-1}\left(\frac{(12.5)^2 + (12.5)^2 - ((-11.85)^2 + (12.5)^2)}{2(12.5)(12.5)}\right)$$

$$\beta = \cos^{-1}\left(\frac{15.8275}{312.5}\right) = 87.0968$$

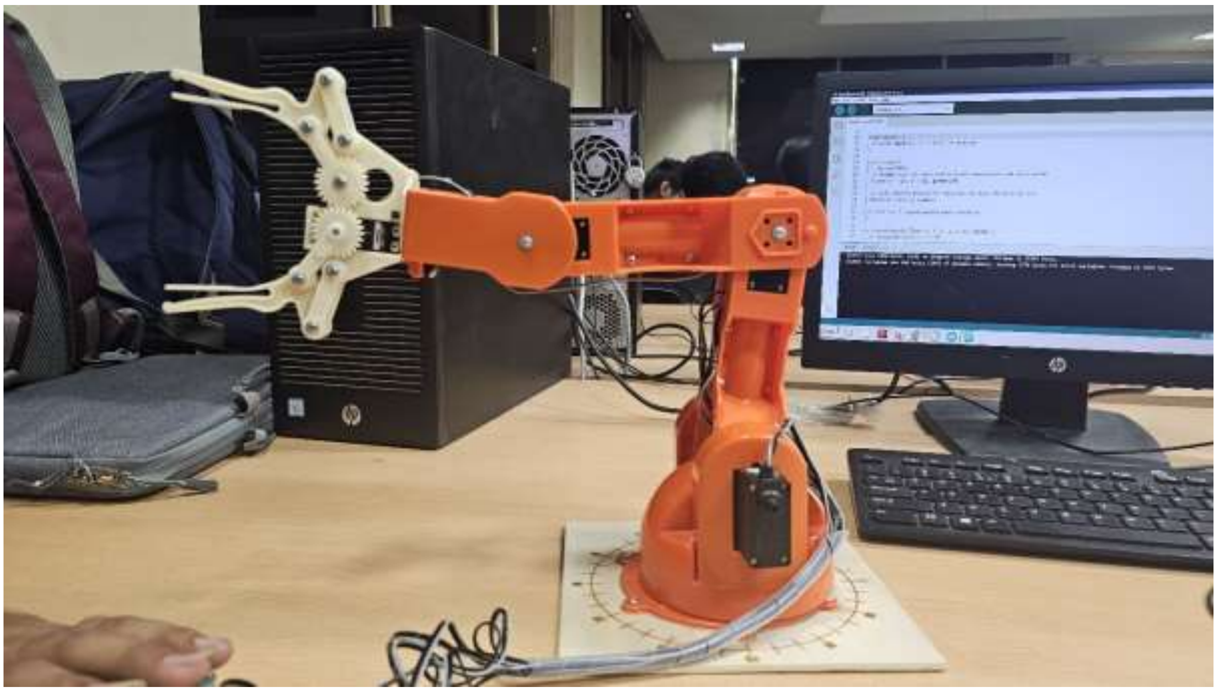
$\theta_1 = \tan^{-1}\left(\frac{b}{a}\right) - \alpha$
$\theta_1 = (133.47) - 46.45$
$\theta_1 = 87.02$

$\theta_2 = \pi - \beta$
$= 180 - 87.0968$
$\theta_2 = 92.9032^\circ$

$\theta_3 = \gamma - \theta_1 - \theta_2$
$= 180 - 87.02 - 92.9$
$\theta_3 = 0.077^\circ$

- We can see that all three values from the three different methods are nearly the same, not identical because of the error mentioned above.

- <u>Image of the final position of the Braccio Robot Arm:</u>



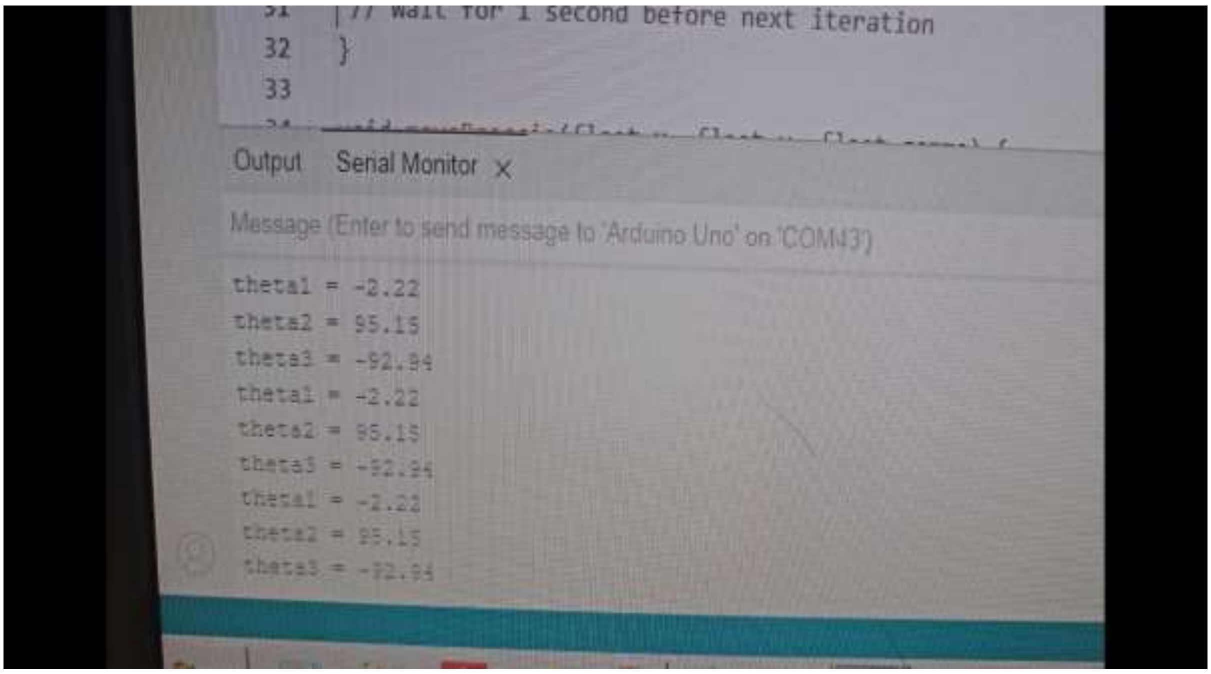
❖ From the above image we can clarify that the obtained angles from the serial output, cpp output and handwritten analysis match in this experiment.

❖ ***<u>Experiment Number-2:</u>***

Point of the end-effector: (x, y) = (19, 12) *(in centi-meters)*
Value of $\gamma = 0\ (in\ degrees)$

- *<u>Obtained Values:</u>*
    - θ_1 = -2.12 (in degrees)
    - θ_2 = 95.15 (in degrees)
    - θ_3 = -92.9378 (in degrees)

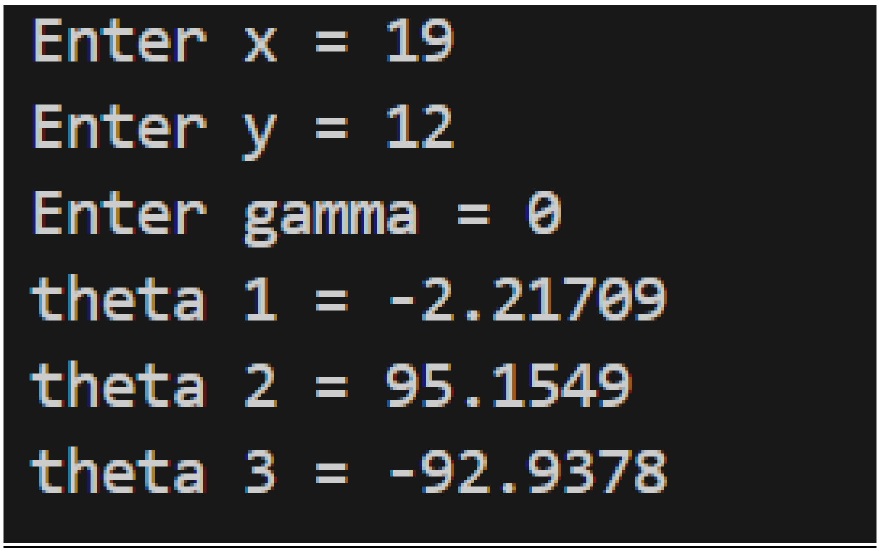
- Serial Output / Console of the Arduino IDE:



```
   // wait for 1 second before next iteration
}
```

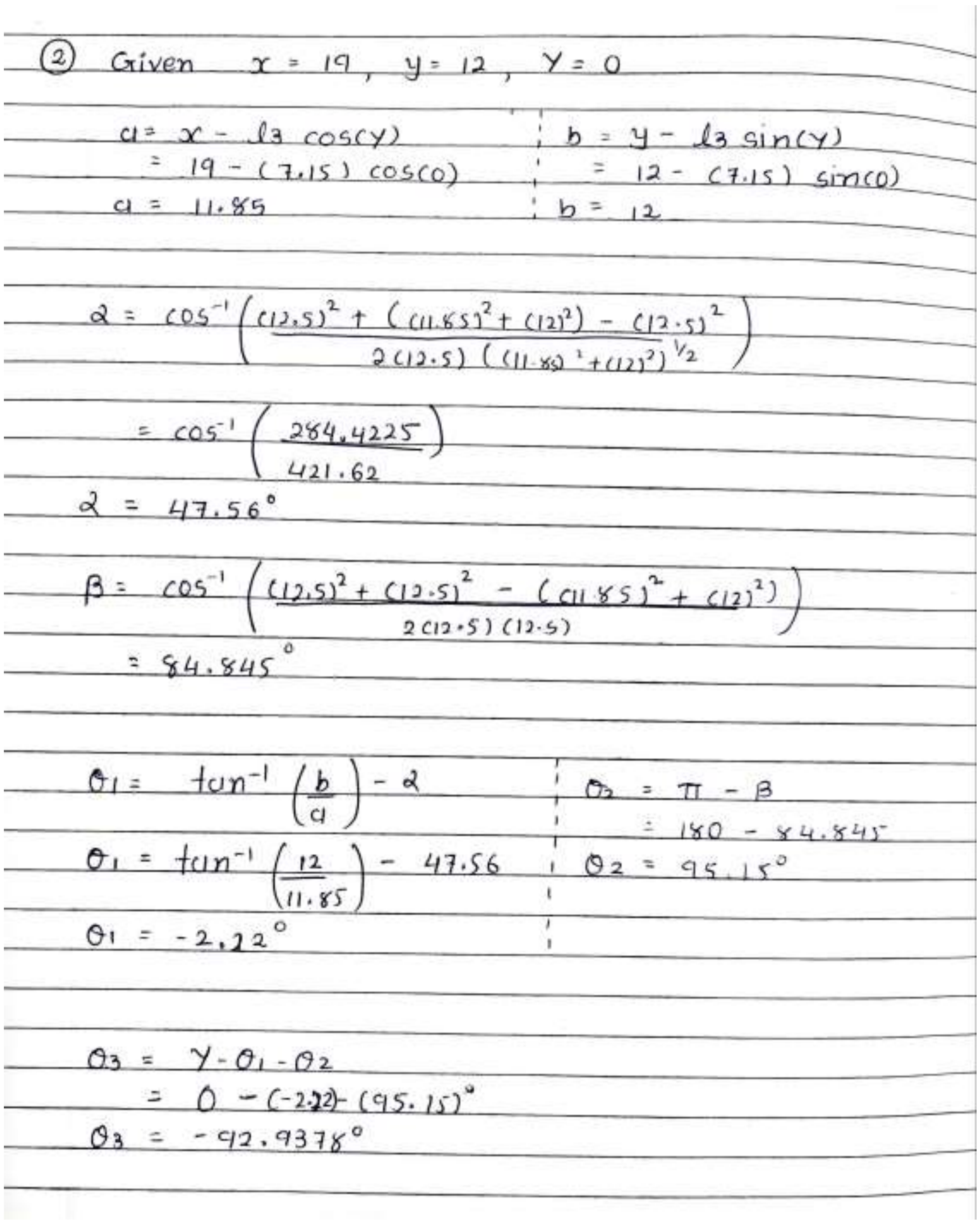
```
theta1 = -2.22
theta2 = 95.15
theta3 = -92.94
theta1 = -2.22
theta2 = 95.15
theta3 = -92.94
theta1 = -2.22
theta2 = 95.15
theta3 = -92.94
```

- CPP output:



```
Enter x = 19
Enter y = 12
Enter gamma = 0
theta 1 = -2.21709
theta 2 = 95.1549
theta 3 = -92.9378
```

- Handwritten Analysis:



(2) Given $x = 19$, $y = 12$, $\gamma = 0$

$a = x - l_3 \cos(\gamma)$
$= 19 - (7.15)\cos(0)$
$a = 11.85$

$b = y - l_3 \sin(\gamma)$
$= 12 - (7.15)\sin(0)$
$b = 12$

$$\alpha = \cos^{-1}\left(\frac{(12.5)^2 + ((11.85)^2 + (12)^2) - (12.5)^2}{2(12.5)((11.85)^2 + (12)^2)^{1/2}}\right)$$

$$= \cos^{-1}\left(\frac{284.4225}{421.62}\right)$$

$\alpha = 47.56^\circ$

$$\beta = \cos^{-1}\left(\frac{(12.5)^2 + (12.5)^2 - ((11.85)^2 + (12)^2)}{2(12.5)(12.5)}\right)$$

$= 84.845^\circ$

$$\theta_1 = \tan^{-1}\left(\frac{b}{a}\right) - \alpha$$

$$\theta_1 = \tan^{-1}\left(\frac{12}{11.85}\right) - 47.56$$

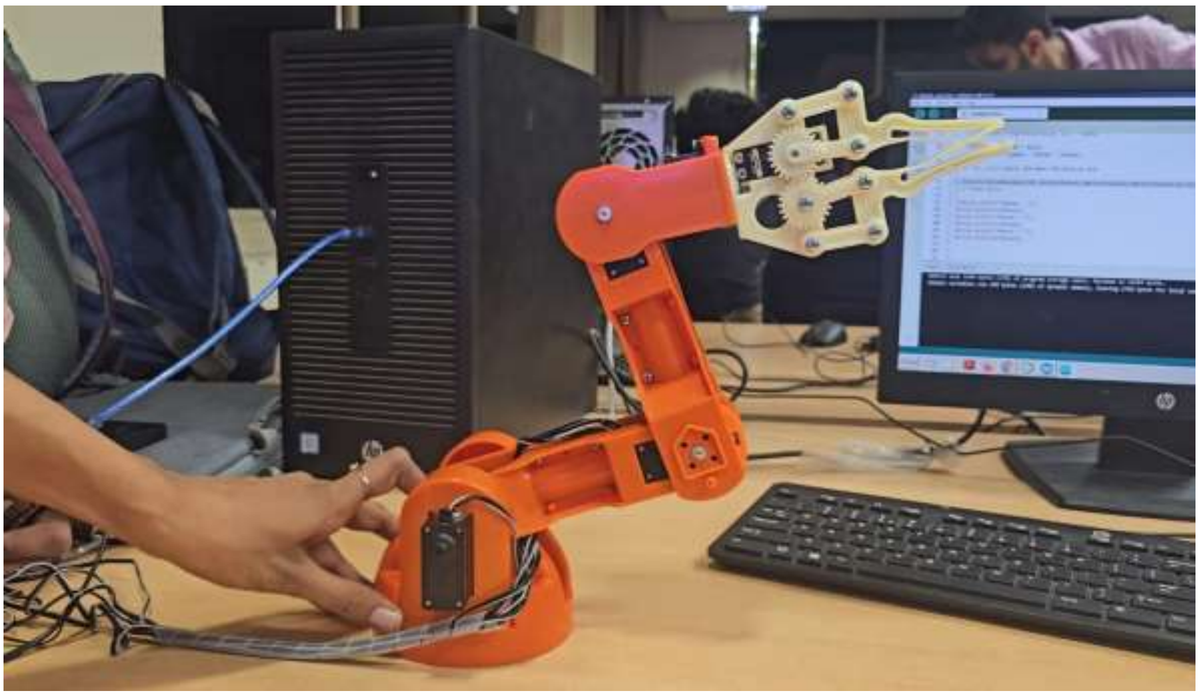
$\theta_1 = -2.22^\circ$

$\theta_2 = \pi - \beta$
$= 180 - 84.845$
$\theta_2 = 95.15^\circ$

$\theta_3 = \gamma - \theta_1 - \theta_2$
$= 0 - (-2.22) - (95.15)^\circ$
$\theta_3 = -92.9378^\circ$

- We can see that all three values from the three different method are same.

- Image of the final position of the braccio Robot Arm:



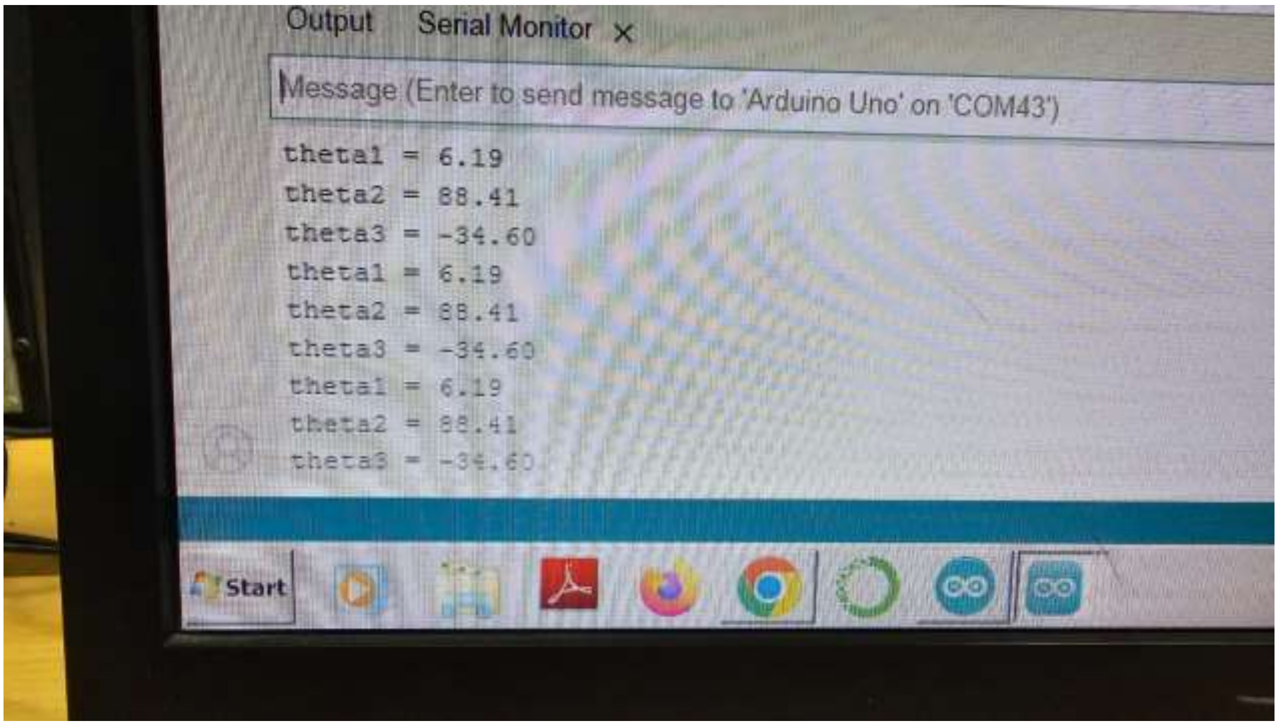
❖ From the above image we can clarify that the obtained angles from the serial output, cpp output and handwritten analysis match in this experiment.

❖ ***Experiment Number-3:***

Point of the end-effector: (x, y) = (15, 20) *(in centi-meters)*
Value of $\gamma = 60$ ($in\ degrees$)

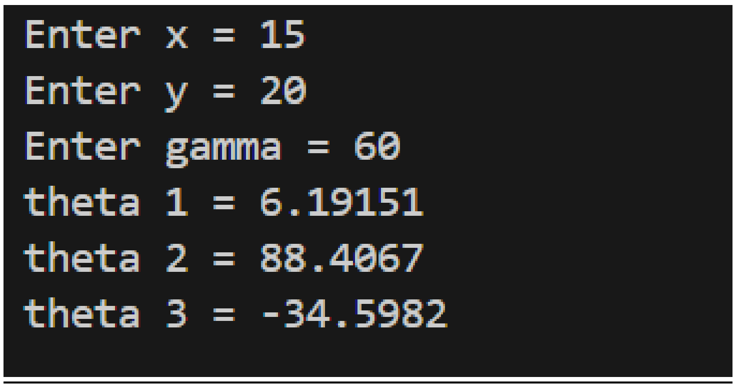
- *Obtained Values:*
    - θ_1 = 6.1915 (in degrees)
    - θ_2 = 88.4067 (in degrees)
    - θ_3 = -34.5982 (in degrees)

- Serial Output / Console of the Arduino IDE:



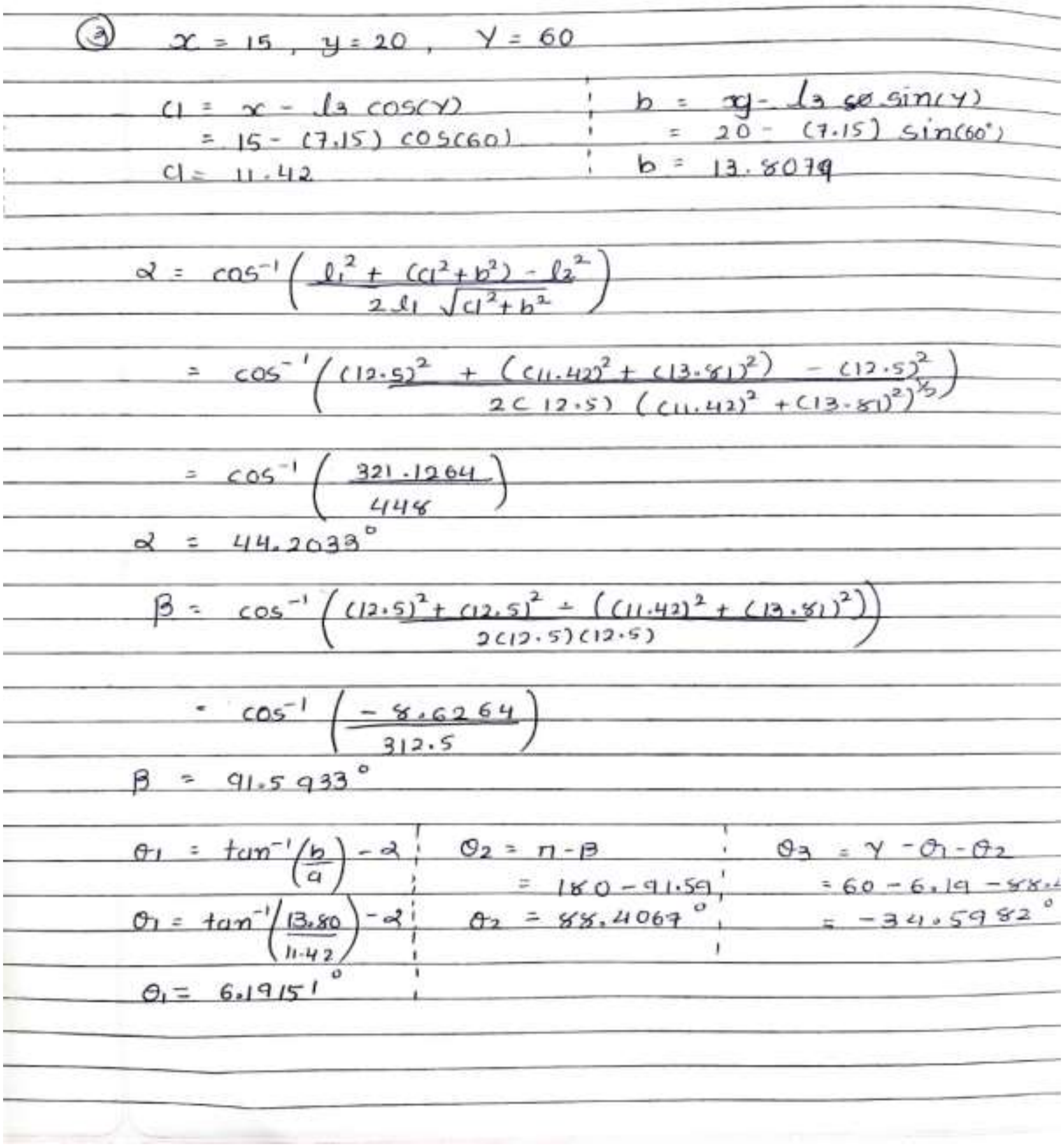
```
Message (Enter to send message to 'Arduino Uno' on 'COM43')
theta1 = 6.19
theta2 = 88.41
theta3 = -34.60
theta1 = 6.19
theta2 = 88.41
theta3 = -34.60
theta1 = 6.19
theta2 = 88.41
theta3 = -34.60
```

- CPP output:



```
Enter x = 15
Enter y = 20
Enter gamma = 60
theta 1 = 6.19151
theta 2 = 88.4067
theta 3 = -34.5982
```

- <u>Handwritten Analysis:</u>



③ $x = 15$, $y = 20$, $\gamma = 60$

$a = x - l_3 \cos(\gamma)$
$= 15 - (7.15)\cos(60)$
$a = 11.42$

$b = y - l_3 \cancel{60} \sin(\gamma)$
$= 20 - (7.15)\sin(60^\circ)$
$b = 13.8079$

$$\alpha = \cos^{-1}\left(\frac{l_1^2 + (a^2+b^2) - l_2^2}{2 l_1 \sqrt{a^2+b^2}}\right)$$

$$= \cos^{-1}\left(\frac{(12.5)^2 + ((11.42)^2 + (13.81)^2) - (12.5)^2}{2(12.5)\left((11.42)^2 + (13.81)^2\right)^{1/2}}\right)$$

$$= \cos^{-1}\left(\frac{321.1264}{448}\right)$$

$\alpha = 44.2033^\circ$

$$\beta = \cos^{-1}\left(\frac{(12.5)^2 + (12.5)^2 - ((11.42)^2 + (13.81)^2)}{2(12.5)(12.5)}\right)$$

$$= \cos^{-1}\left(\frac{-8.6264}{312.5}\right)$$

$\beta = 91.5933^\circ$

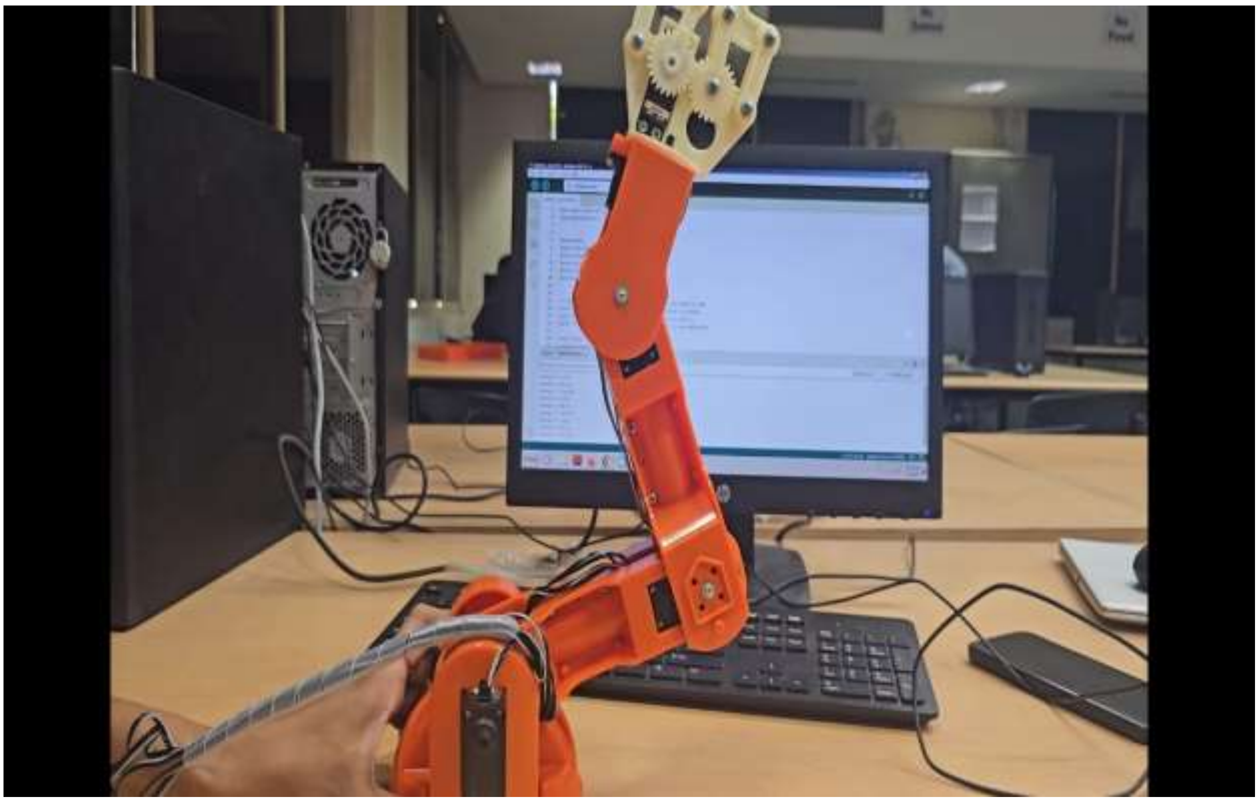
$\theta_1 = \tan^{-1}\left(\frac{b}{a}\right) - \alpha$
$\theta_1 = \tan^{-1}\left(\frac{13.80}{11.42}\right) - \alpha$
$\theta_1 = 6.19151^\circ$

$\theta_2 = \pi - \beta$
$= 180 - 91.59$
$\theta_2 = 88.4067^\circ$

$\theta_3 = \gamma - \theta_1 - \theta_2$
$= 60 - 6.19 - 88.$
$= -34.5982^\circ$

- We can see that all three values from the three different method are same.

- Image of the final position of the braccio Robot Arm:



❖ From the above image we can clarify that the obtained angles from the serial output, cpp output and handwritten analysis match in this experiment.

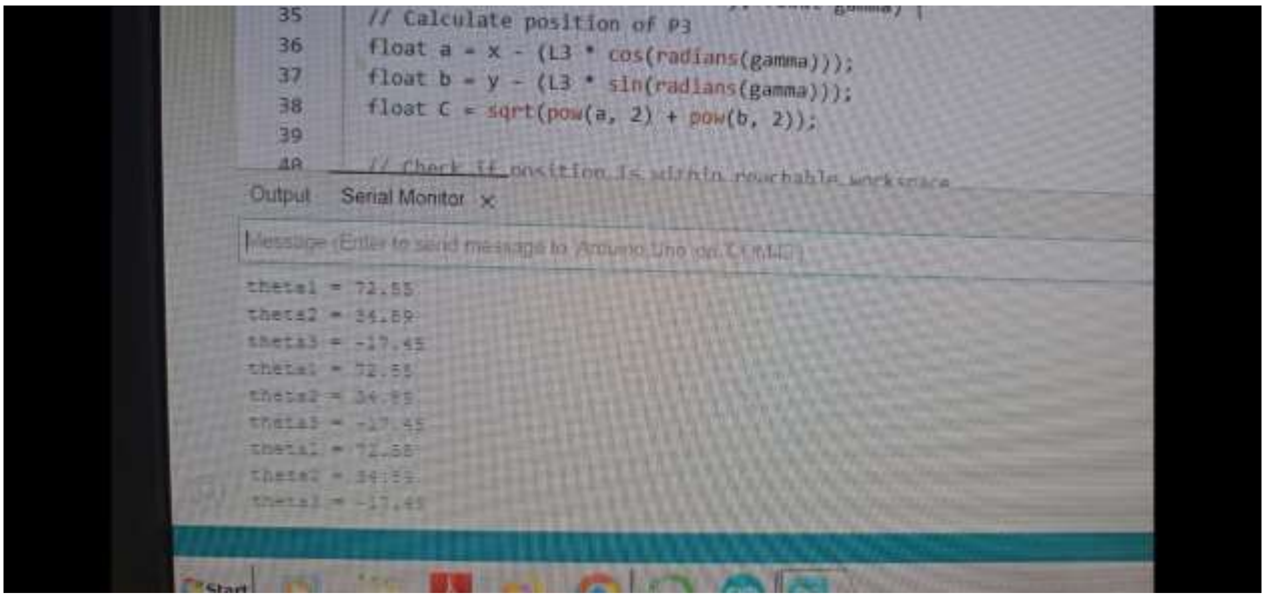
❖ ***Experiment Number-4:***

Point of the end-effector: (x, y) = (0, 31) *(in centi-meters)*
Value of $\gamma = 90\ (in\ degrees)$

- *Obtained Values:*

  - θ_1 = 72.554 (in degrees)
  - θ_2 = 34.892 (in degrees)
  - θ_3 = -17.446(in degrees)

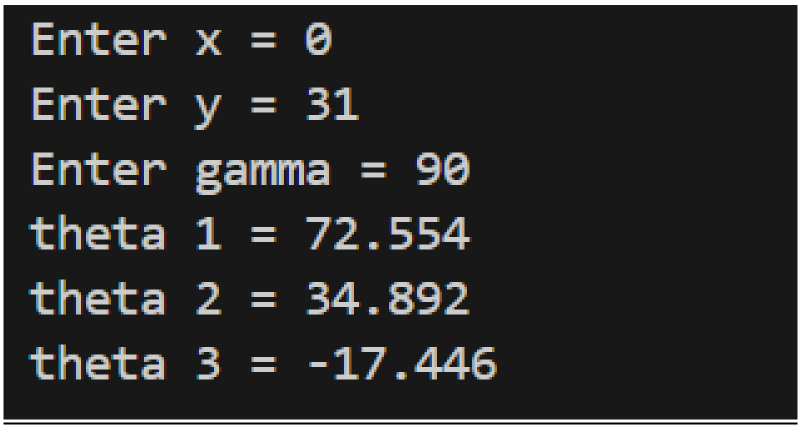
- <u>Serial Output / Console of the Arduino IDE:</u>



```
// Calculate position of P3
float a = x - (L3 * cos(radians(gamma)));
float b = y - (L3 * sin(radians(gamma)));
float C = sqrt(pow(a, 2) + pow(b, 2));

// Check if position is within reachable workspace
```

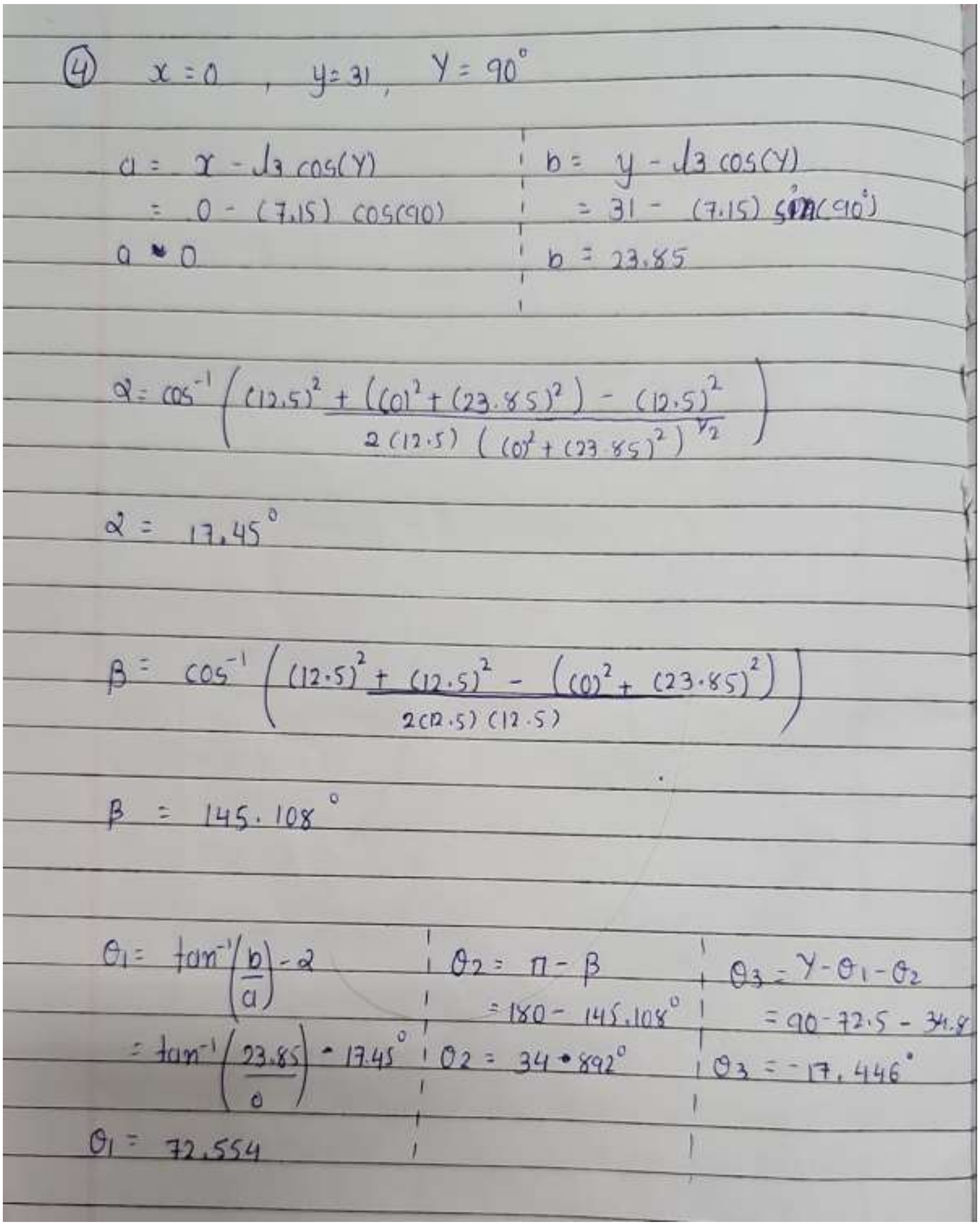
```
theta1 = 72.55
theta2 = 34.89
theta3 = -17.45
theta1 = 72.55
theta2 = 34.89
theta3 = -17.45
theta1 = 72.55
theta2 = 34.89
theta3 = -17.45
```

- <u>CPP output:</u>



```
Enter x = 0
Enter y = 31
Enter gamma = 90
theta 1 = 72.554
theta 2 = 34.892
theta 3 = -17.446
```

- <u>Handwritten Analysis:</u>



④ $x = 0$, $y = 31$, $\gamma = 90^\circ$

$a = x - l_3 \cos(\gamma)$

$= 0 - (7.15)\cos(90)$

$a = 0$

$b = y - l_3 \cos(\gamma)$

$= 31 - (7.15)\sin(90^\circ)$

$b = 23.85$

$$\alpha = \cos^{-1}\left(\frac{(12.5)^2 + ((0)^2 + (23.85)^2) - (12.5)^2}{2(12.5)\left((0)^2 + (23.85)^2\right)^{1/2}}\right)$$

$\alpha = 17.45^\circ$

$$\beta = \cos^{-1}\left(\frac{(12.5)^2 + (12.5)^2 - ((0)^2 + (23.85)^2)}{2(12.5)(12.5)}\right)$$

$\beta = 145.108^\circ$

$\theta_1 = \tan^{-1}\left(\frac{b}{a}\right) - \alpha$

$= \tan^{-1}\left(\frac{23.85}{0}\right) - 17.45^\circ$

$\theta_1 = 72.554$

$\theta_2 = \pi - \beta$

$= 180 - 145.108^\circ$

$\theta_2 = 34.892^\circ$

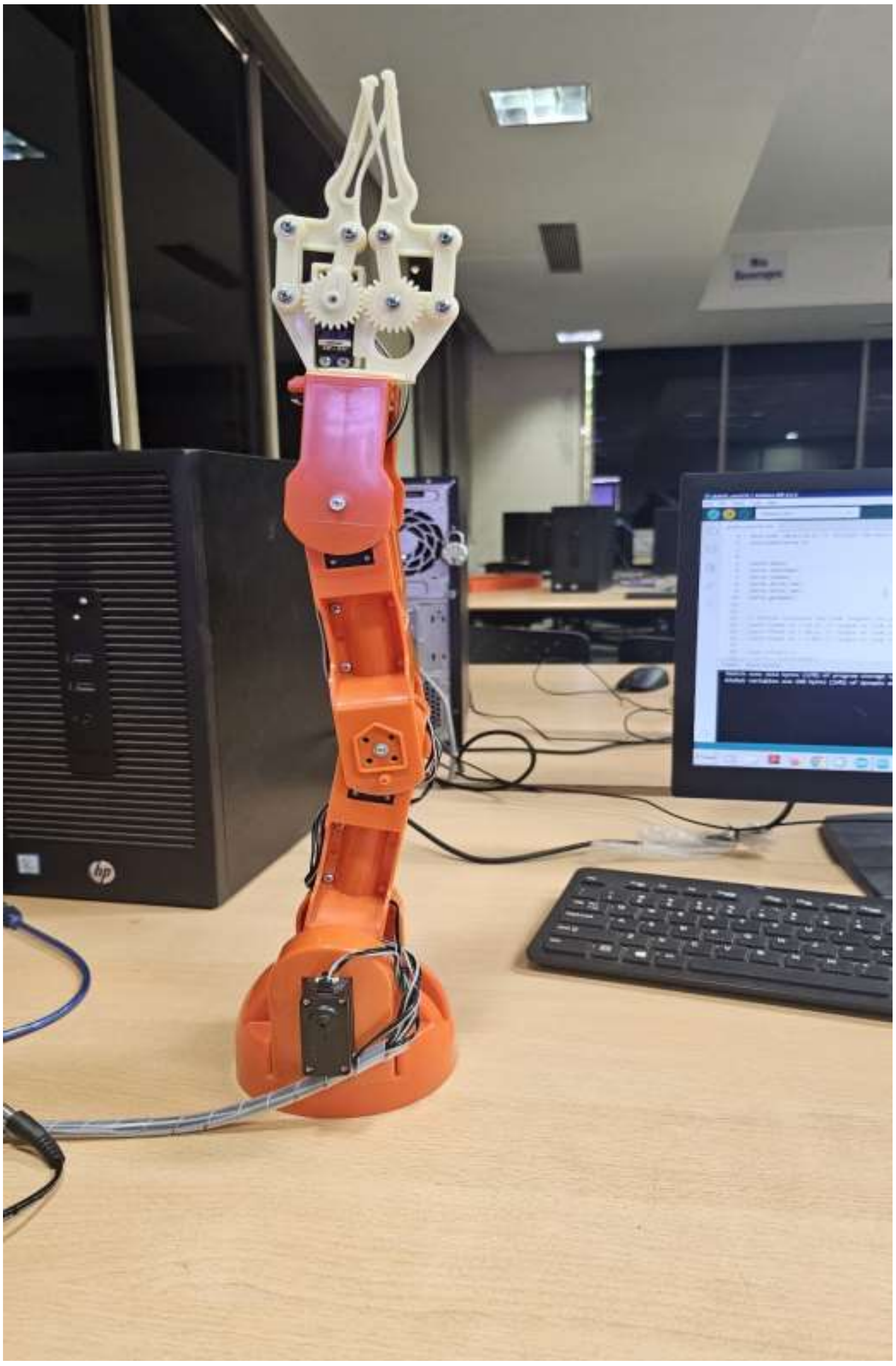
$\theta_3 = \gamma - \theta_1 - \theta_2$

$= 90 - 72.5 - 34.8$

$\theta_3 = -17.446^\circ$

- We can see that all three values from the three different method are same

- <u>Image of the final position of the braccio Robot Arm:</u>



7 . ***Observation:***

- ❖ From the above four experiments we have observed that using the inverse kinematics of the 3 degrees of freedom we can calculate the angles $\theta_1,\ \theta_2\ and\ \theta_3$ and move the Braccio Robot Arm to the desire position.
- ❖ We also created the one video for what we done so far (containing cpp code, Arduino ide code, the all experiment er have done in practically with Robot Arm, Arduino UNO, compatible shield and the adapter).

- ❖ **Link of the video containing Everything we have done so far:**

  - ❖ **GitHub Link;**
    - https://github.com/Divyesh-19/Inverse-Kinematics-of-3DOF
  - ❖ **You-Tube Video Link:**
    - https://youtu.be/a9EuVdCcAO0?si=HU1J5oCQkmUmMQ1N

8 . ***Result:***

- ❖ Using inverse kinematics of 3 degrees of freedom, we can determine $\theta_1,\ \theta_2\ and\ \theta_3$ which represent the angles of link1, link2 and link3 respectively.
- ❖ We then move the Braccio Robot arm accordingly.