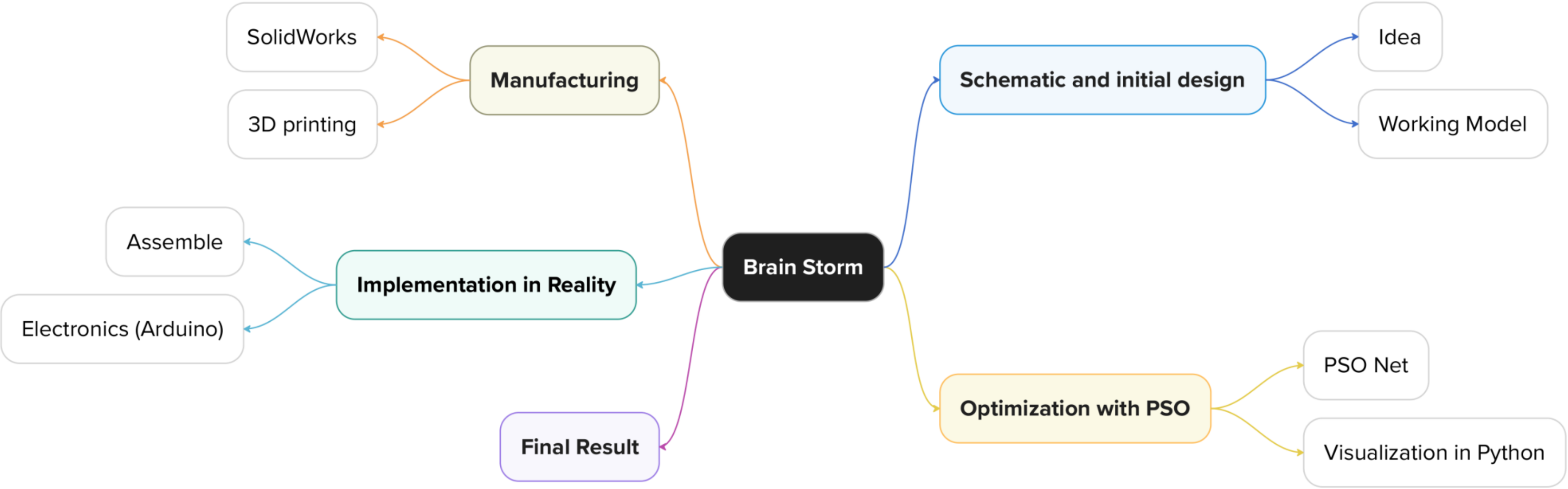
Design and Simulation of **Leg Mechanism**

Dr Zabihifar

Design of mechanisms Project



Brain Storm
Schematic and initial design
Idea
Working Model
Optimization with PSO
PSO Net
Visualization in Python
Manufacturing
SolidWorks
3D printing
Implementation in Reality
Assemble
Electronics (Arduino)
Final Result

01

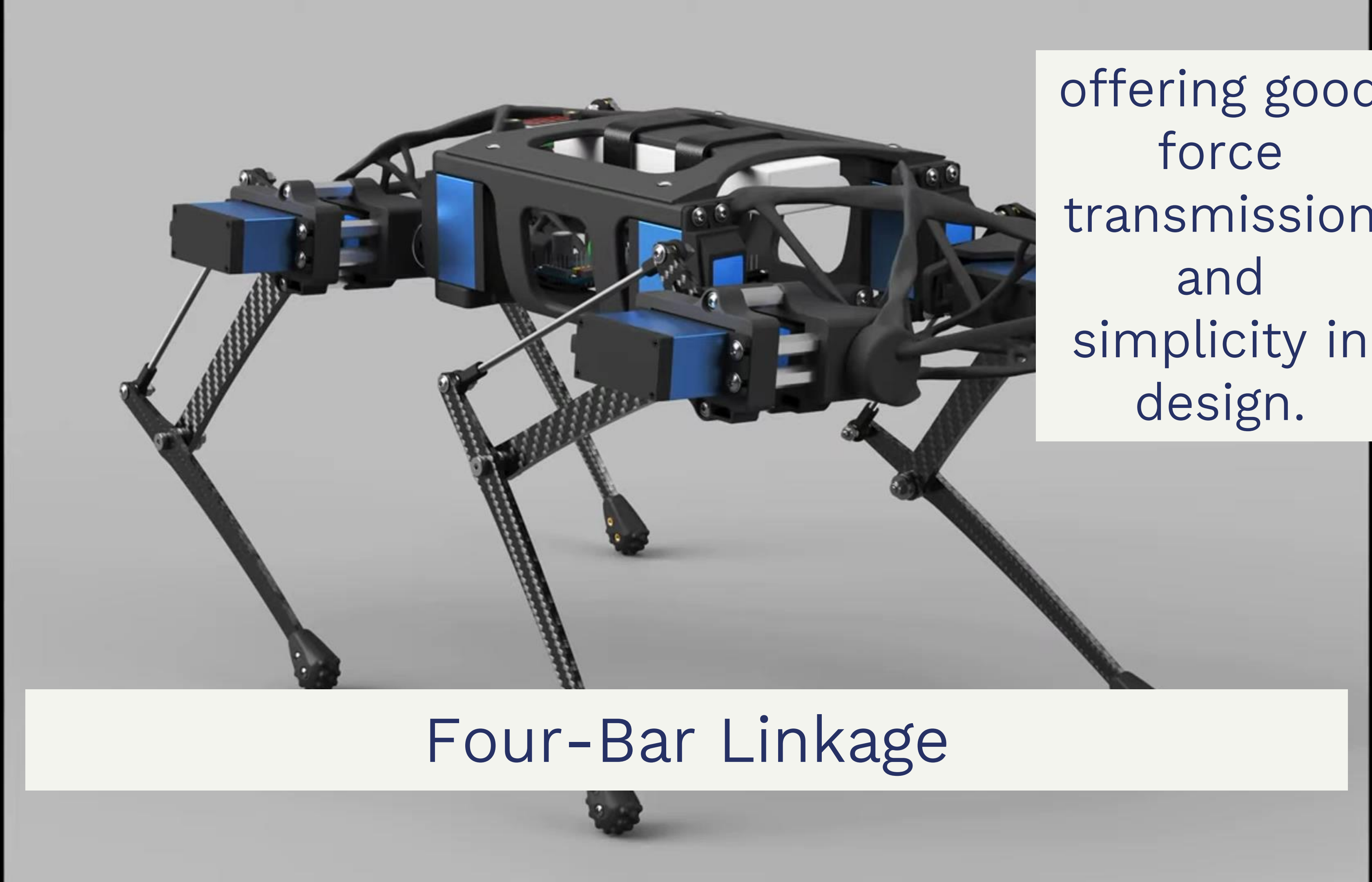
Schematic and initial design

- Idea
- Working Model

Quadruped legs
the concept

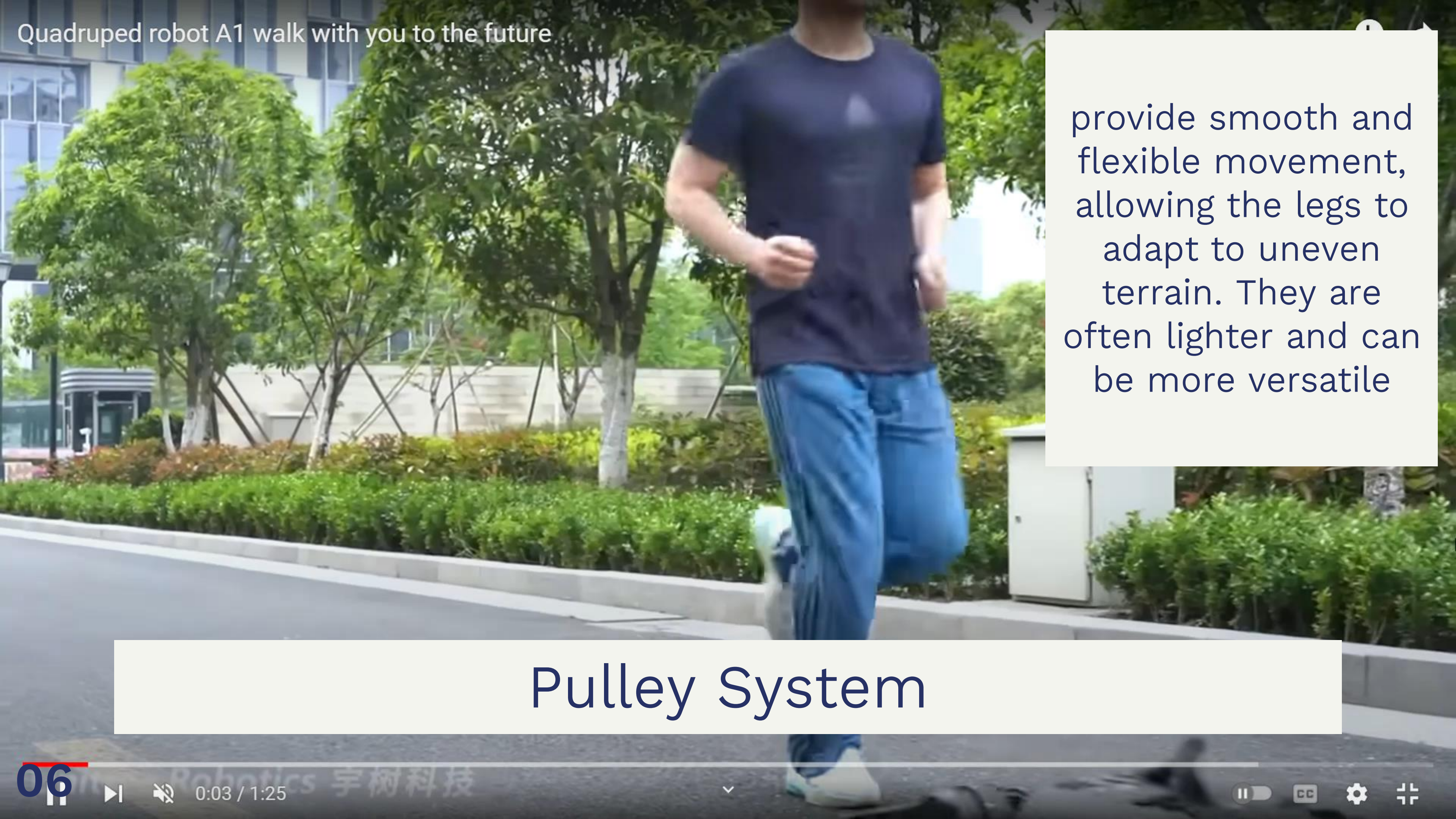
Quadruped robots can use different mechanisms for their legs, including:

1. four-bar linkages
2. pulley systems



Four-Bar Linkage

offering good force transmission and simplicity in design.



provide smooth and flexible movement, allowing the legs to adapt to uneven terrain. They are often lighter and can be more versatile

Pulley System

Agenda

Theoretical plan

Key Idea

- Using four-bar Linkage mechanism
- 2 DOF
- Jumping feature

Optimization

PSO Algorithm

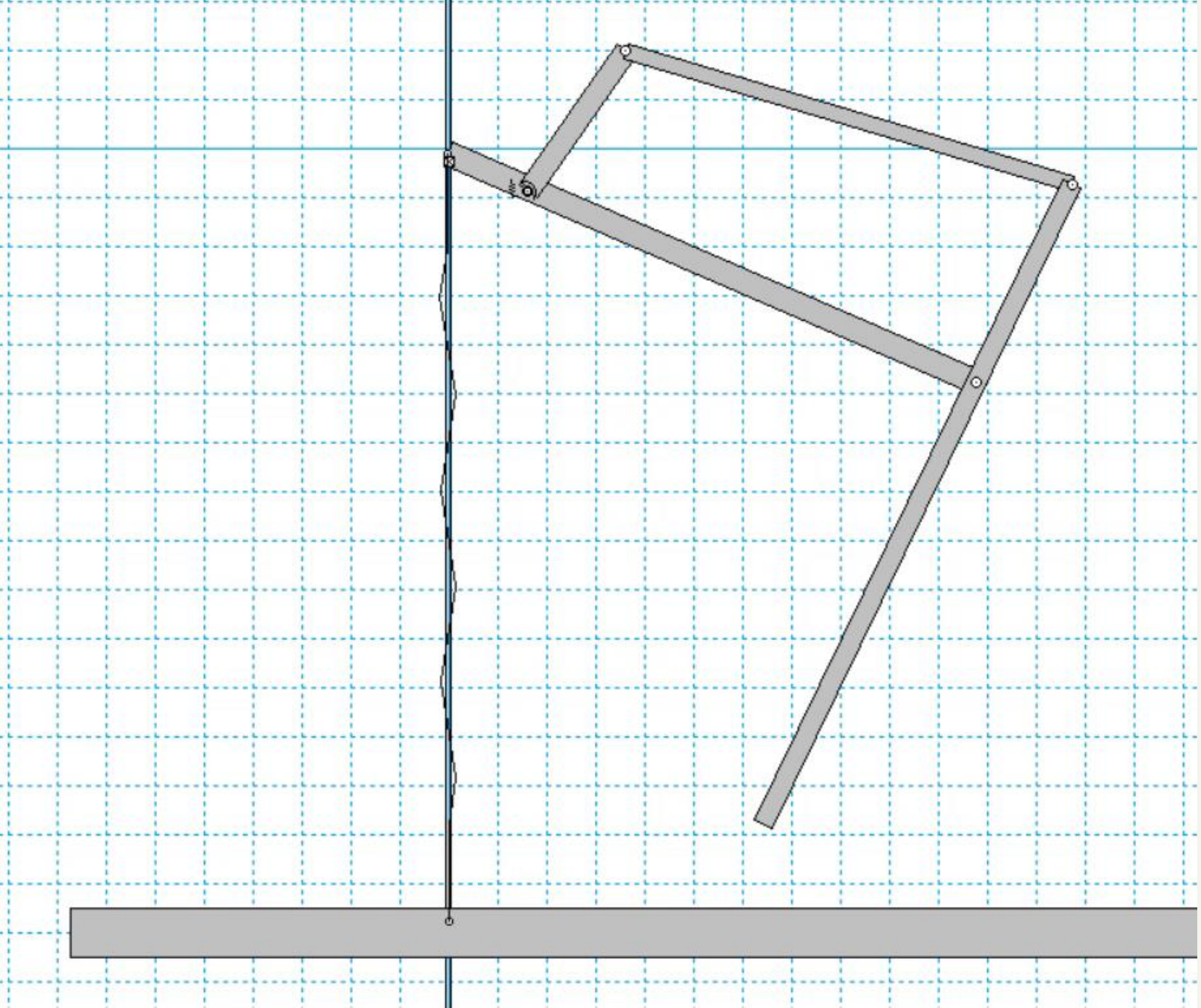
Using PSO to Optimize the functionality of our leg

Visualize

We developed a code which shows our four-bar leg in a close loop

Our Output is:

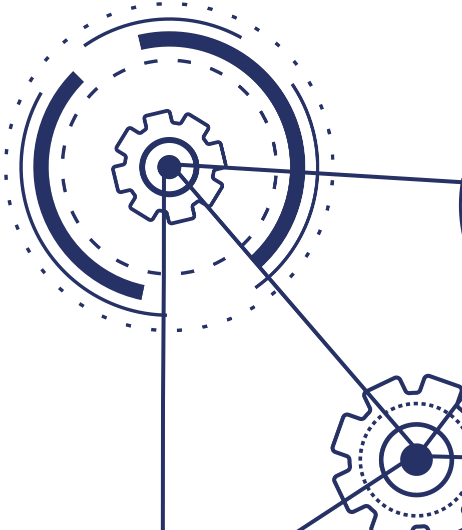
Four length Value for each of the links in our four-bar linkage
We visualize it in Python and WorkingModel



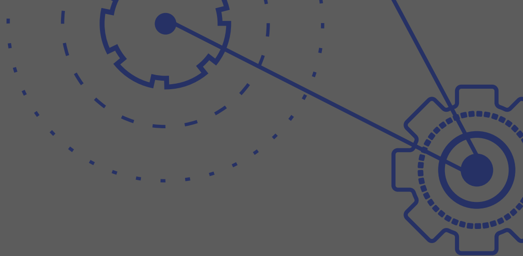
Visualization of the very first Idea

Optimization with PSO

- PSO Net
- Visualization in Python



02



Study **objectives**

Developing Our PSO Net

Getting Our Output from PSO

Visualize it in Python

1.
Import libraries

```
import numpy as np
from pyswarm import pso
from joblib import Parallel, delayed
```

2.
FourBarClass

```
class FourBarMechanism:
    def __init__(self, r1, r2, r3, r4, theta1):
        self.r1 = r1  # Ground link length
        self.r2 = r2  # Input link length
        self.r3 = r3  # Coupler link length
        self.r4 = r4  # Output link length
        self.theta1 = theta1  # Input angle
    def get_velocity(self, theta):
        r1, r2, r3, r4, theta1 = self.r1, self.r2, self.r3, self.r4, self.theta1
        t = np.arange(0, np.pi, 0.001)
        W = theta  # (rad/s)
        Theta2 = W * t
        A = 2 * r1 * r4 * np.cos(theta1) - 2 * r2 * r4 * np.cos(Theta2)
        B = 2 * r1 * r4 * np.sin(theta1) - 2 * r2 * r4 * np.sin(Theta2)
        C = r1 ** 2 + r2 ** 2 + r4 ** 2 - r3 ** 2 - 2 * r1 * r2 * (
            np.cos(theta1) * np.cos(Theta2) + np.sin(theta1) * np.sin(Theta2))
        delta = 4 * B ** 2 - 4 * (C - A) * (C + A)
        if np.any(delta < 0):
            return np.inf, np.inf
        theta4 = 2 * np.arctan2(-B + np.sqrt(delta), A + C)
        x_coupler = r2 * np.cos(Theta2) + r3 * np.cos(theta3)
        y_coupler = r2 * np.sin(Theta2) + r3 * np.sin(theta3)
        # Derivatives to get velocity
        dx_4_dt = np.gradient(x_coupler, t)
        dy_4_dt = np.gradient(y_coupler, t)
        x_4= r1 * np.sin(theta1) + r4 * np.sin(theta4)
        y_4= r1 * np.cos(theta1) + r4 * np.cos(theta4)
        dx_4_dt = np.gradient(x_4, t)
        dy_4_dt = np.gradient(y_4, t)
        velocity = np.sqrt(dx_4_dt**2 + dy_4_dt**2)

        return velocity[0]
```

3. Objective function

```python
def objective_function(params):
    r1, r2, r3, r4 = params
    mechanism = FourBarMechanism(r1, r2, r3, r4, theta1=0)

    max_deviation = 0
    for i in np.linspace(0, 2 * np.pi, 100):
        velocity = mechanism.get_velocity(i)
        if np.isinf(velocity).any():
            return np.inf
        deviation = np.abs(velocity - target_velocity(i))
        max_deviation = max(max_deviation, deviation)

    return max_deviation
```

4. Target function

```python
def target_velocity(theta):
    # Assume a simple target velocity function for demonstration
    return 12500  # constant target velocity
```

5. PSO Net

```python
# Define bounds for the parameters (adjust as needed for your specific mechanism)
bounds = [(5, 10), (2, 4), (8, 10.5), (2, 4.5)]
# Perform PSO optimization
best_params, best_score = pso(objective_function, lb=[b[0] for b in bounds], ub=[b[1] for b in bounds], swarmsize=100)
```

```
        velocity = np.sqrt(dx_4_dt**2 + dy_4_dt**2)

        return velocity[0]


# Define the objective function to optimize
def objective_function(params):
    r1, r2, r3, r4 = params
    mechanism = FourBarMechanism(r1, r2, r3, r4, theta1=0)

    max_deviation = 0
    for i in np.linspace(0, 2 * np.pi, 100):
        velocity = mechanism.get_velocity(i)
        if np.isinf(velocity).any():
            return np.inf
        deviation = np.abs(velocity - target_velocity(i))
        max_deviation = max(max_deviation, deviation)

    return max_deviation

def target_velocity(theta):
    # Assume a simple target velocity function for demonstration
    return 12500  # constant target velocity

# Define bounds for the parameters (adjust as needed for your specific mechanism)
bounds = [(
# Perform P
best_params

print(f'Best Parameters: {best_params}')
print(f'Best Score: {best_score:0.3} ')
```

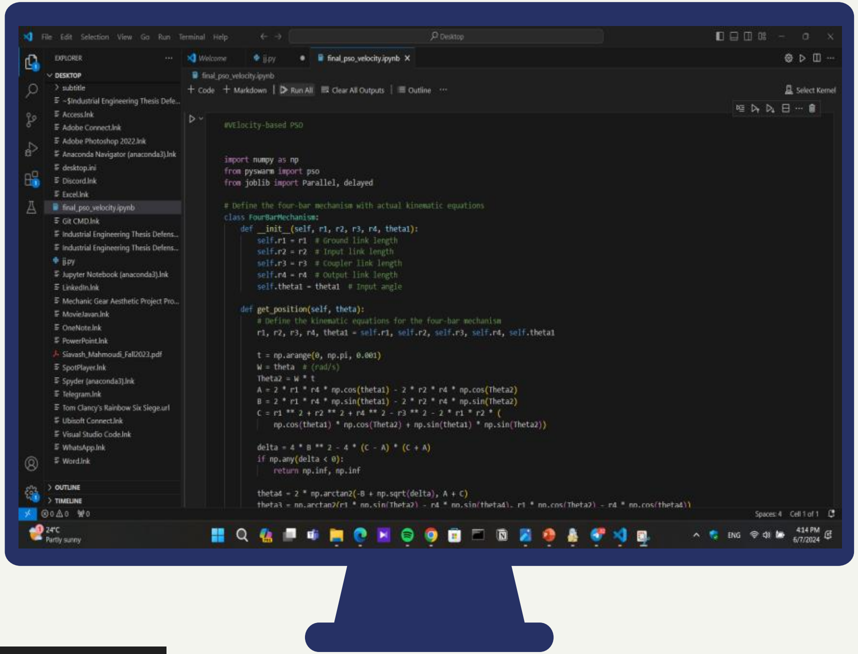
PSO Output

After we reached to the maximum iterations out output will be shown

Optimized length

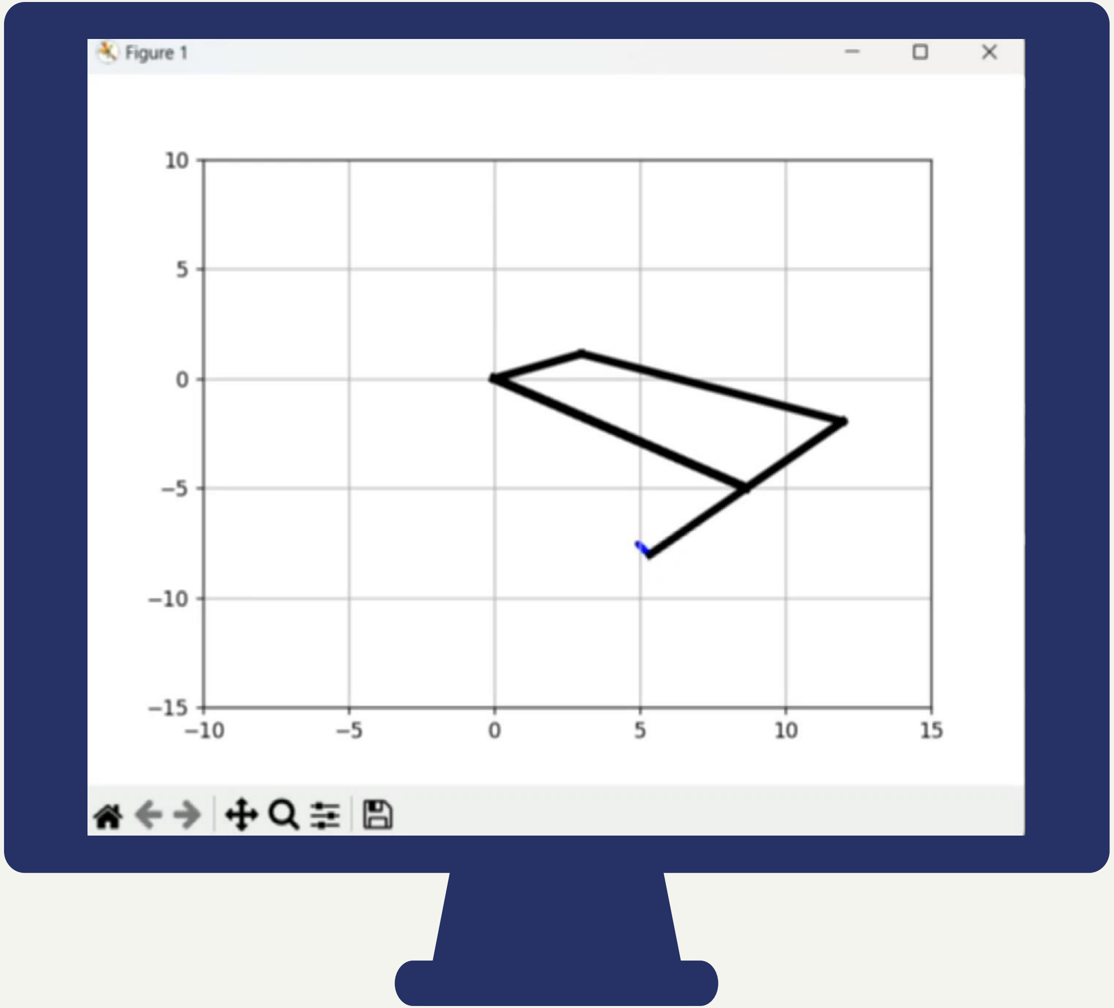
Simply Our output is four value for L1 , L2 , L3, L4 respectively.

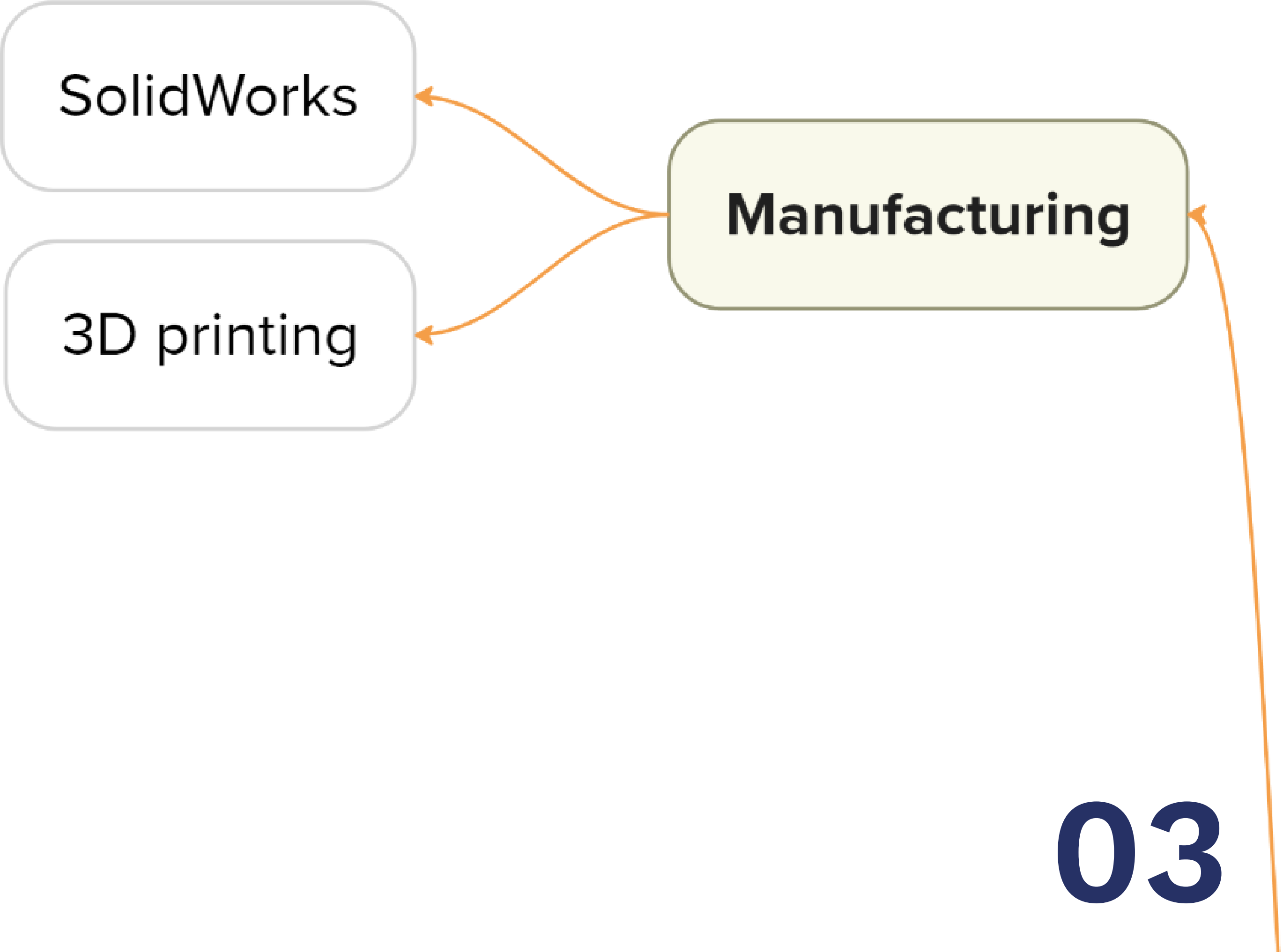
10 cm
3.2 cm
9.5 cm
4.5 cm




```
...   Stopping search: maximum iterations reached --> 100
      Best Parameters: [9.91902416 3.24943772 9.558697   4.48924005]
      Best Score: 1.25e+04
```

Visualize Optimized Linkage in Python



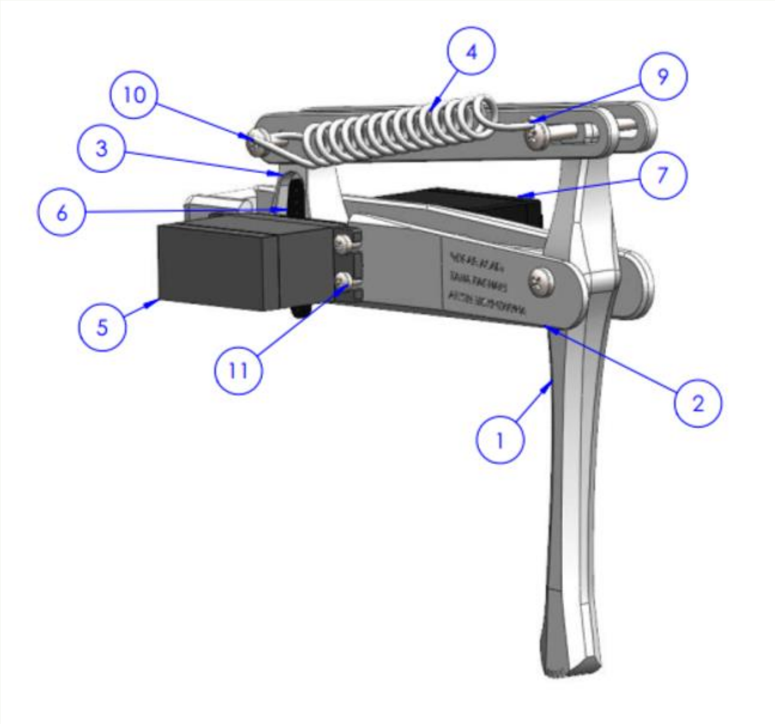


03

Designing in **SolidWorks**

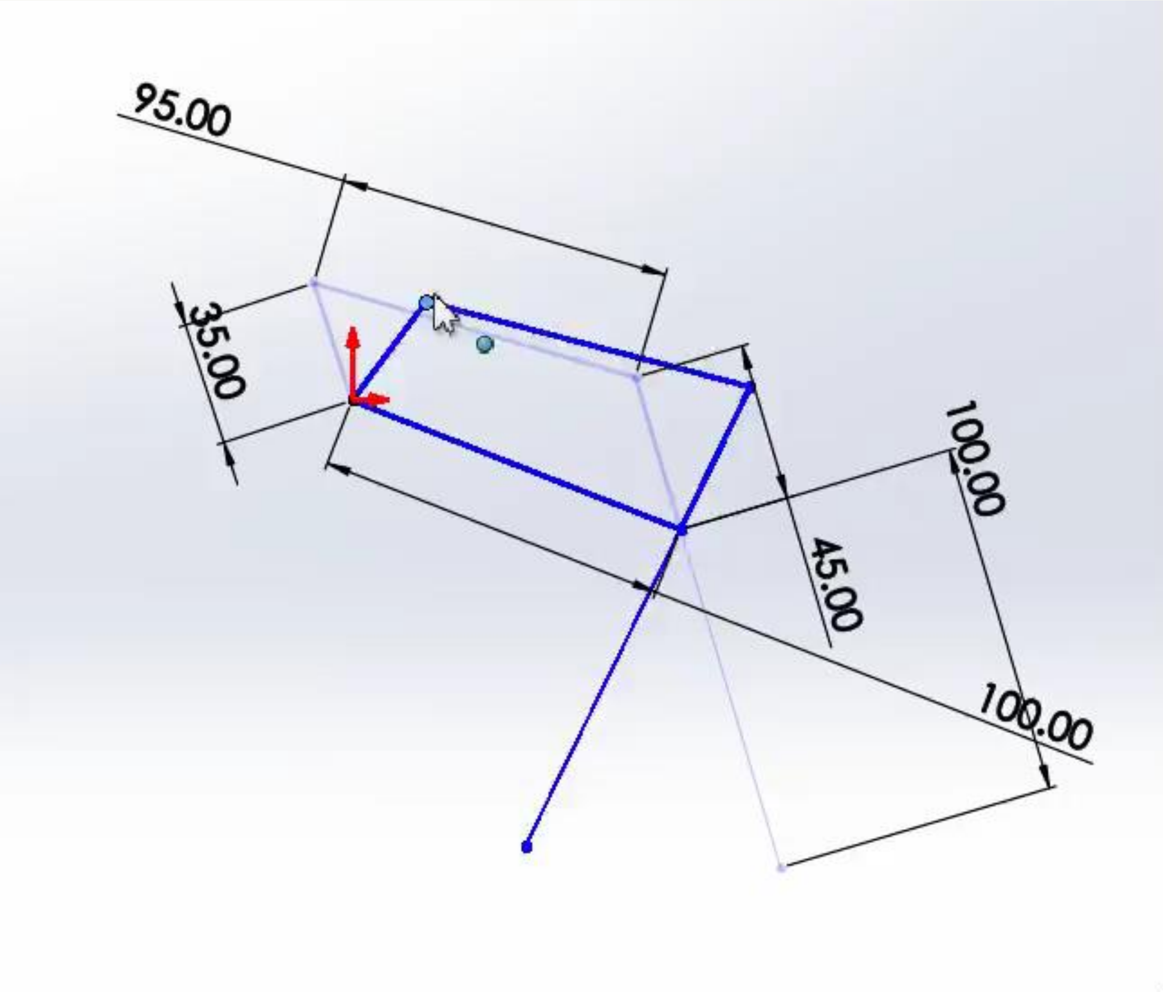
The design process in SolidWorks will be explained in the following steps.

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.
1	Knee		1
2	HIP		1
3	Crank		1
4	Coupler		2
5	Mg995		1
6	attachmentStraight		1
7	servoMotorMG996R		1
8	attachmentCircular		1
9	Spring	Corrosion-Resistant Extension Springs with Hook Ends	1
10	B18.6.7M - M4 x 0.7 x 35 Type I Cross Recessed PHMS -- 35C		3
11	B18.6.7M - M3 x 0.5 x 16 Type I Cross Recessed PHMS -- 16C		4



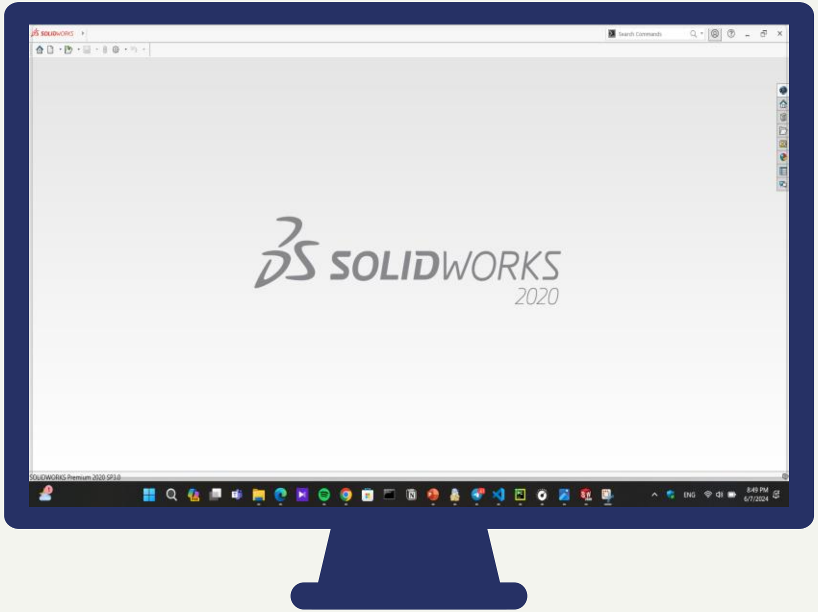
Step 1.
The simplified schematic of the mechanism in **sketch**

As the first step we drew an sketch in order to understand the mechanism and its movements

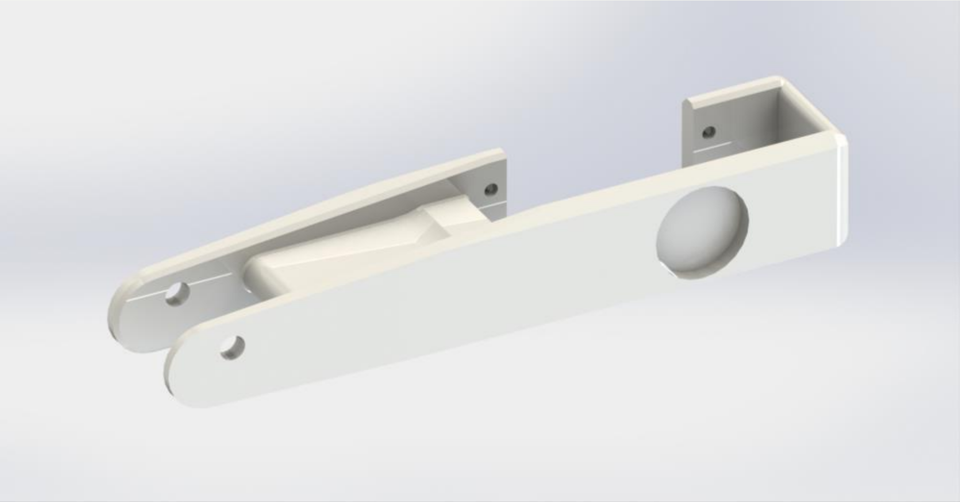


Step 2:
Designing the mechanism components with **optimized** dimensions

Including parts of our four-bar linkage and their components



Link 1
the HIP

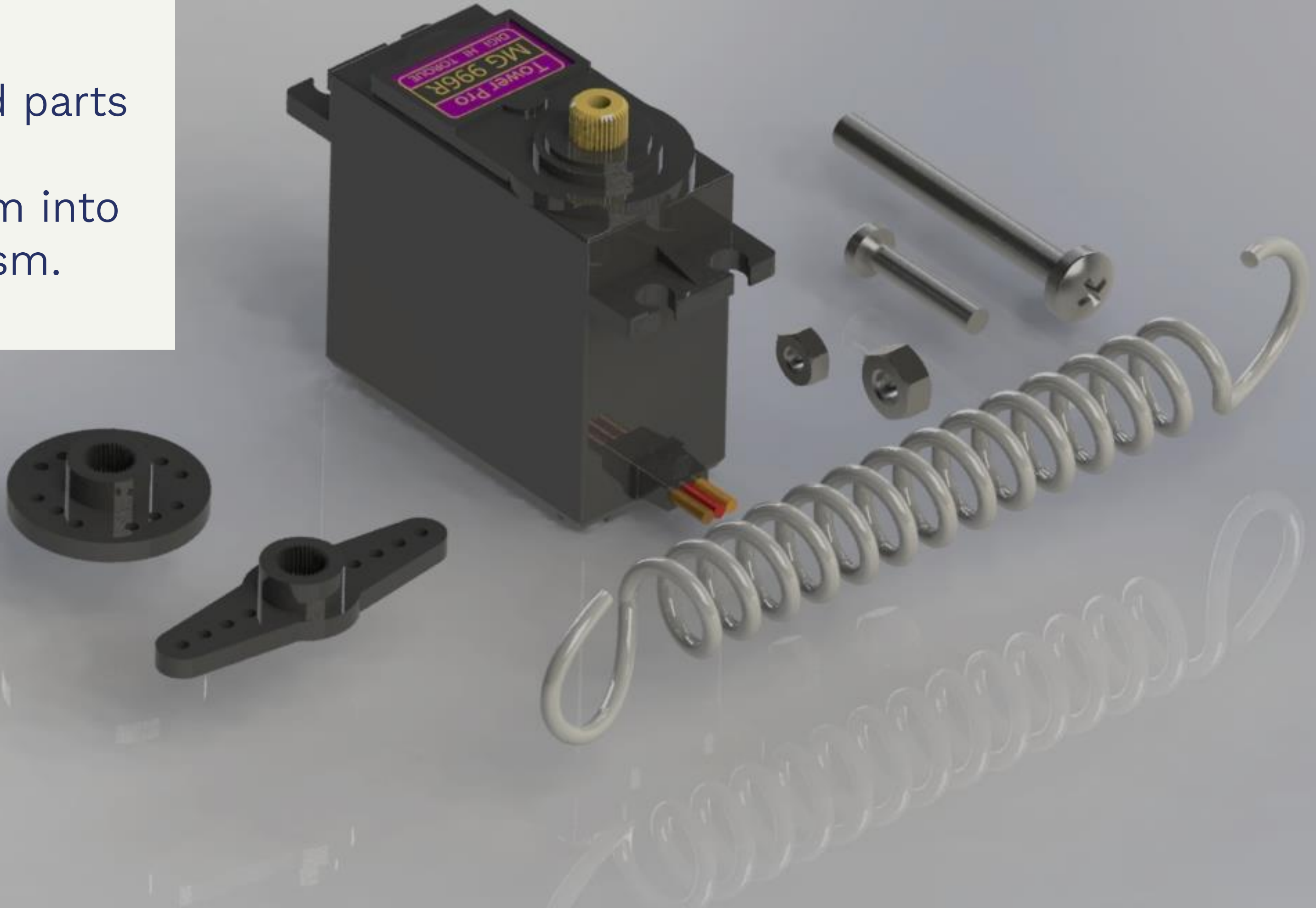


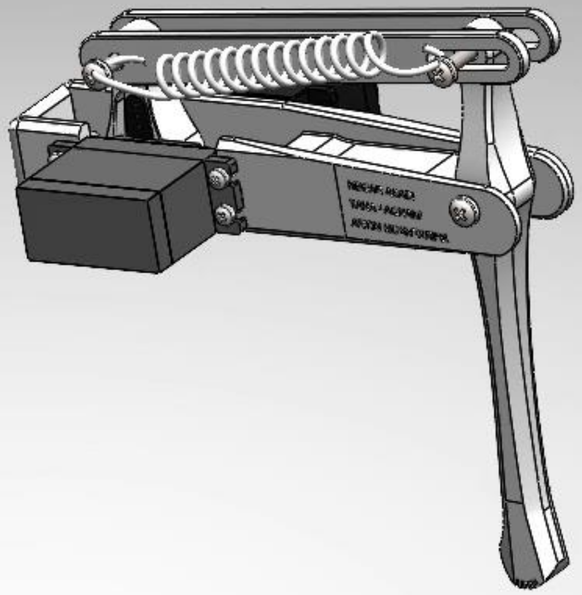
Link 2
the Crank

Link 3
the Coupler

Link 4
the Knee

Step 3
Finding standard parts
and
assembling them into
the mechanism.





Step 4: **Assembling** all component and checking for **alignments** and **connections**
.

Current situation & **problems statement**

Current situation

We faced two problems...

01

Impedance and admittance problem

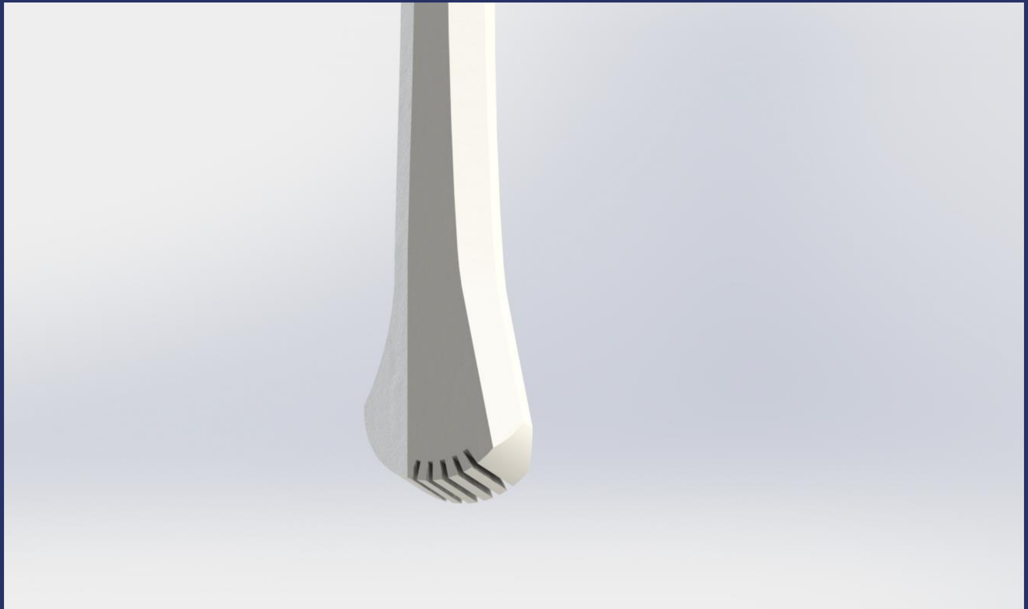
02

Spring problem

Problems

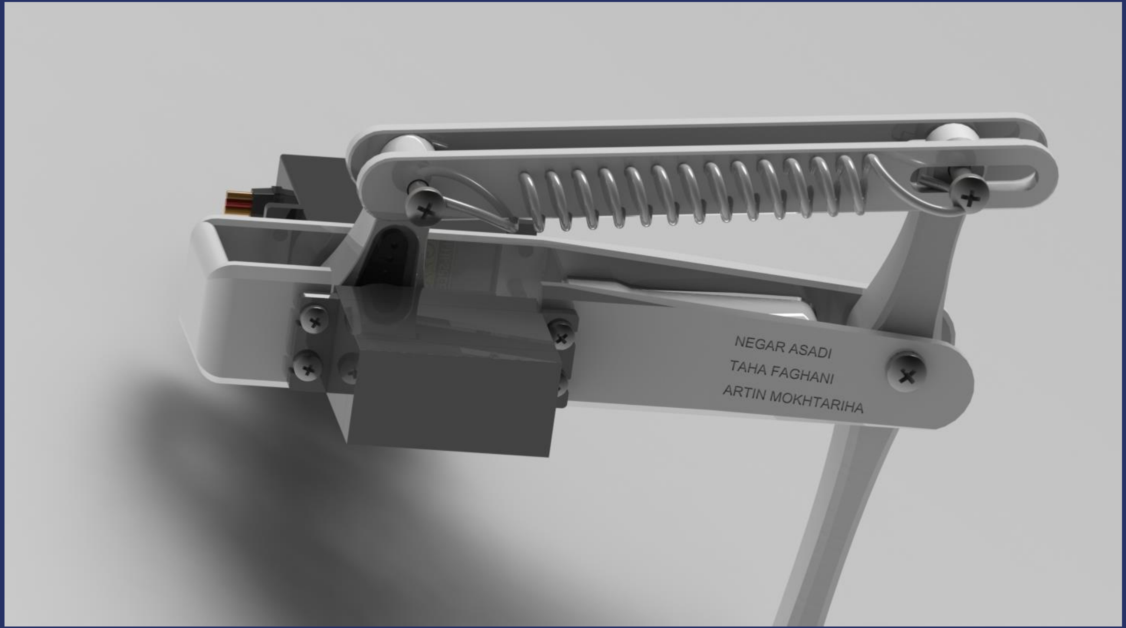
Impedance and admittance

- **80% density**
- Adding lines at the points of intersection.

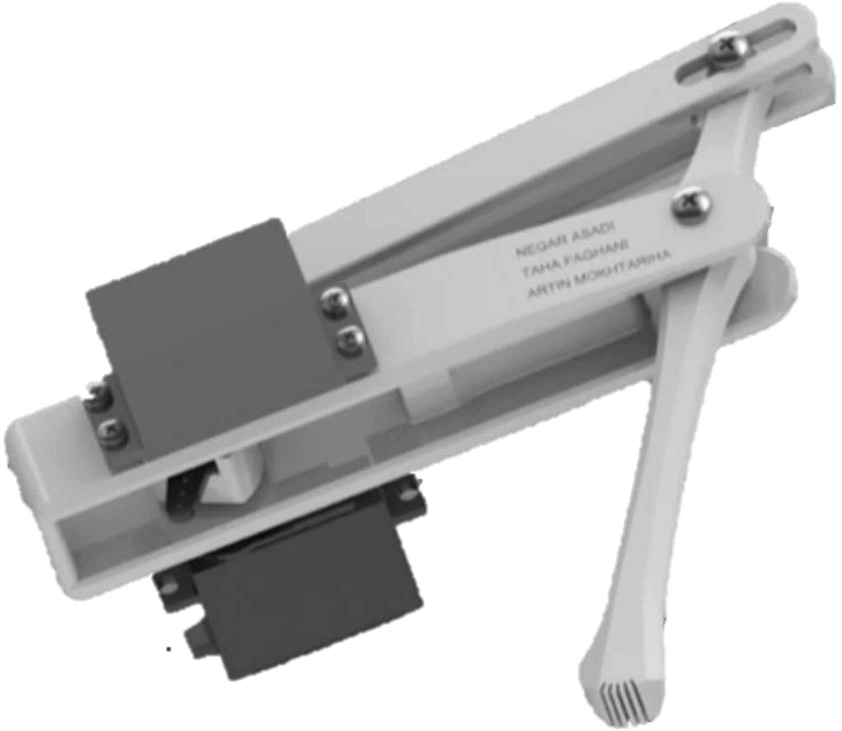


Spring

- Position
- Stiffness factor
- Length of spring



Final performance review and error correction upon observing the printed version



Final Design

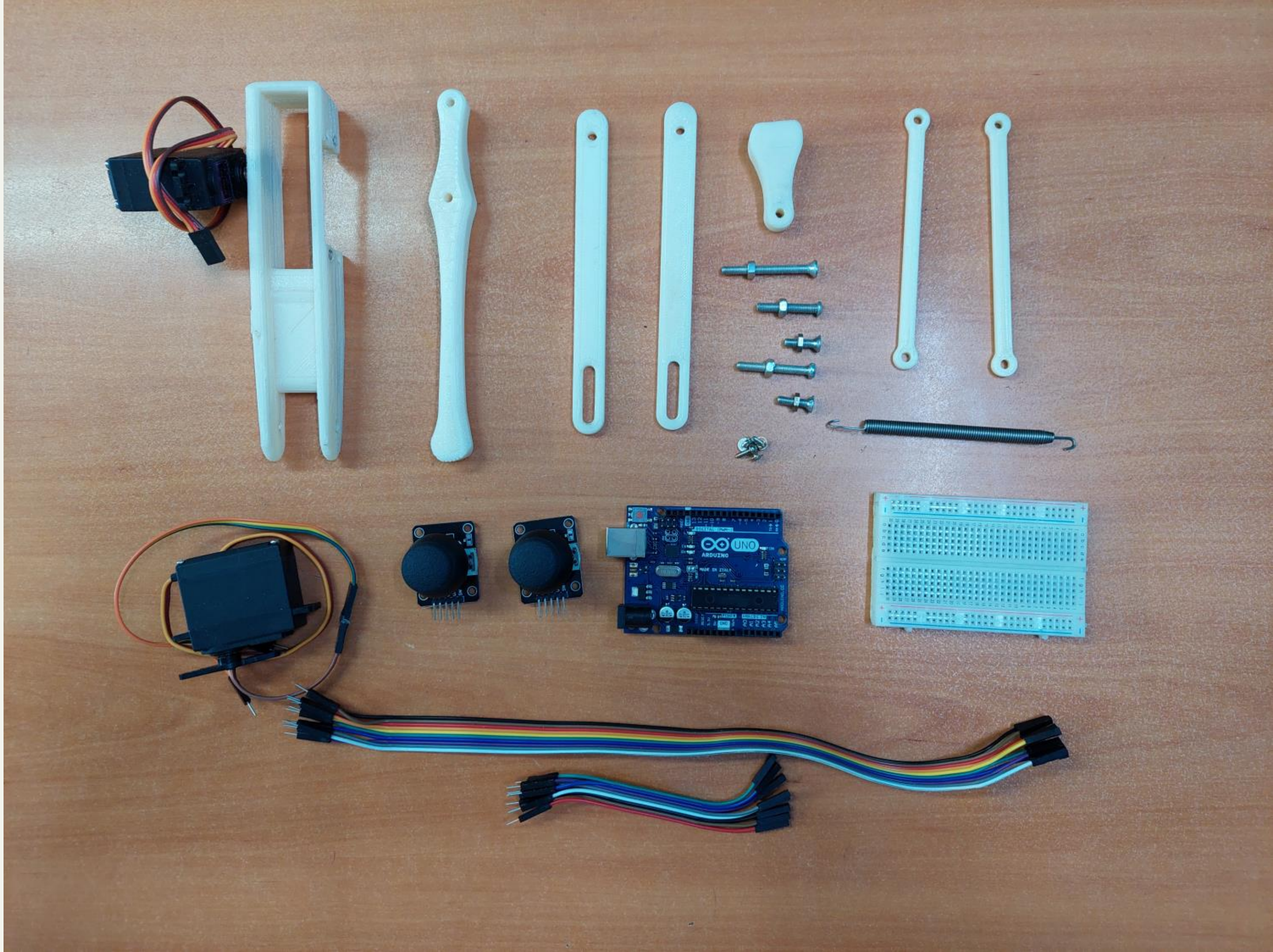
04

Implementation in Reality

- Assemble
- Electronics (Arduino)

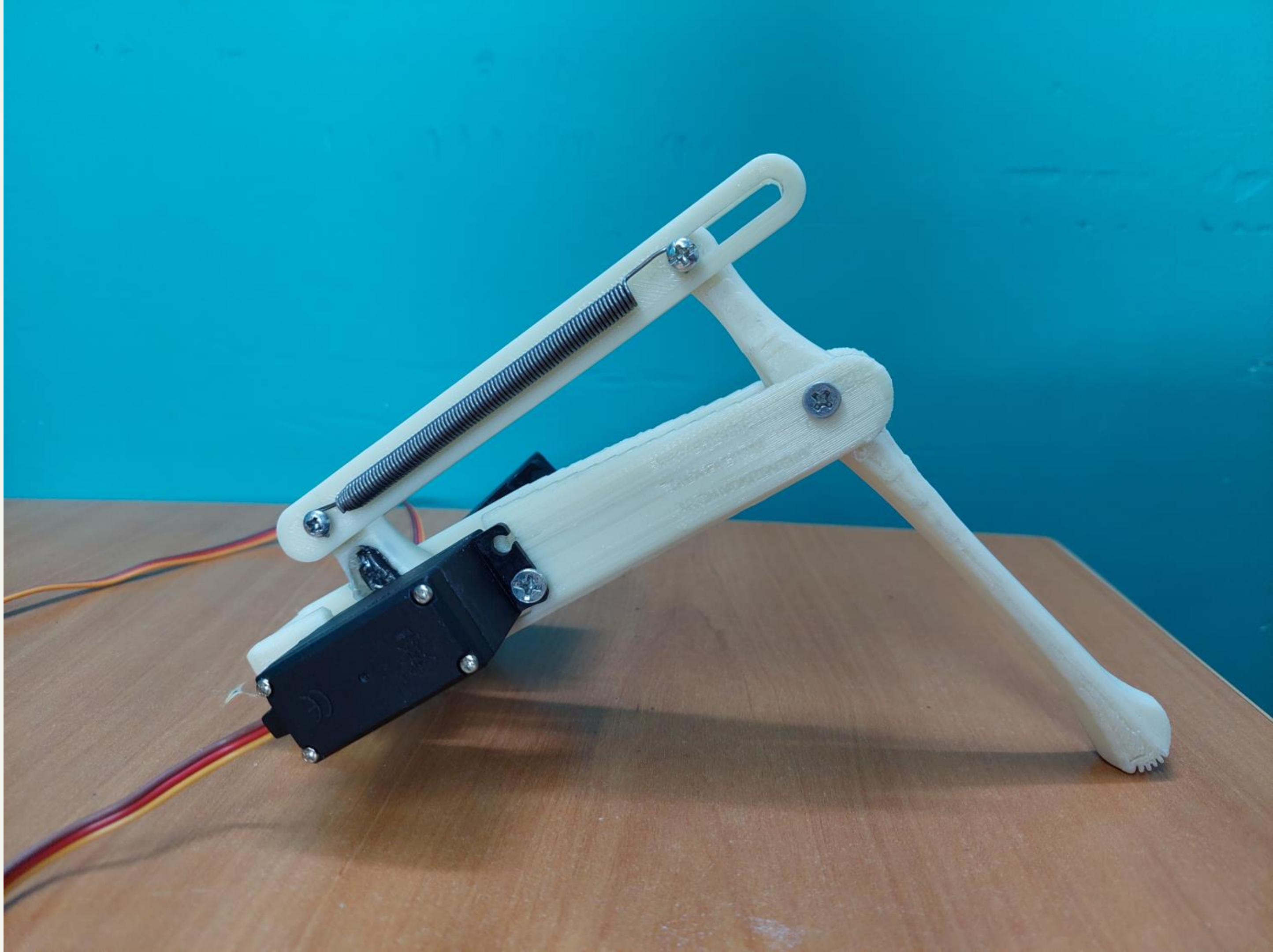
Printed parts

Ready to Assemble



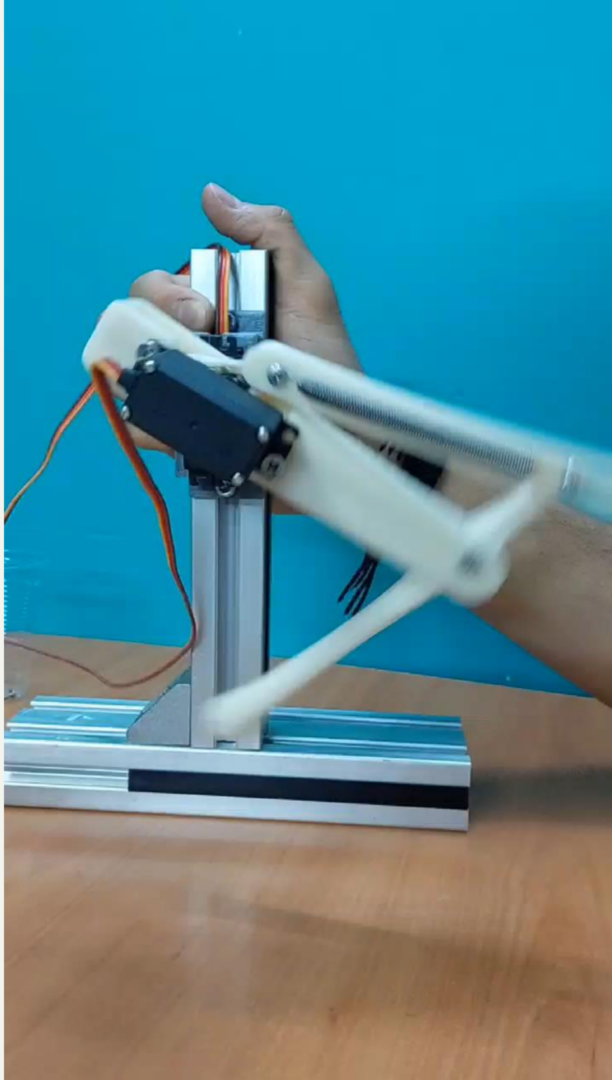
Assembled leg

How can we test it?



Stand

The leg will be placed on this

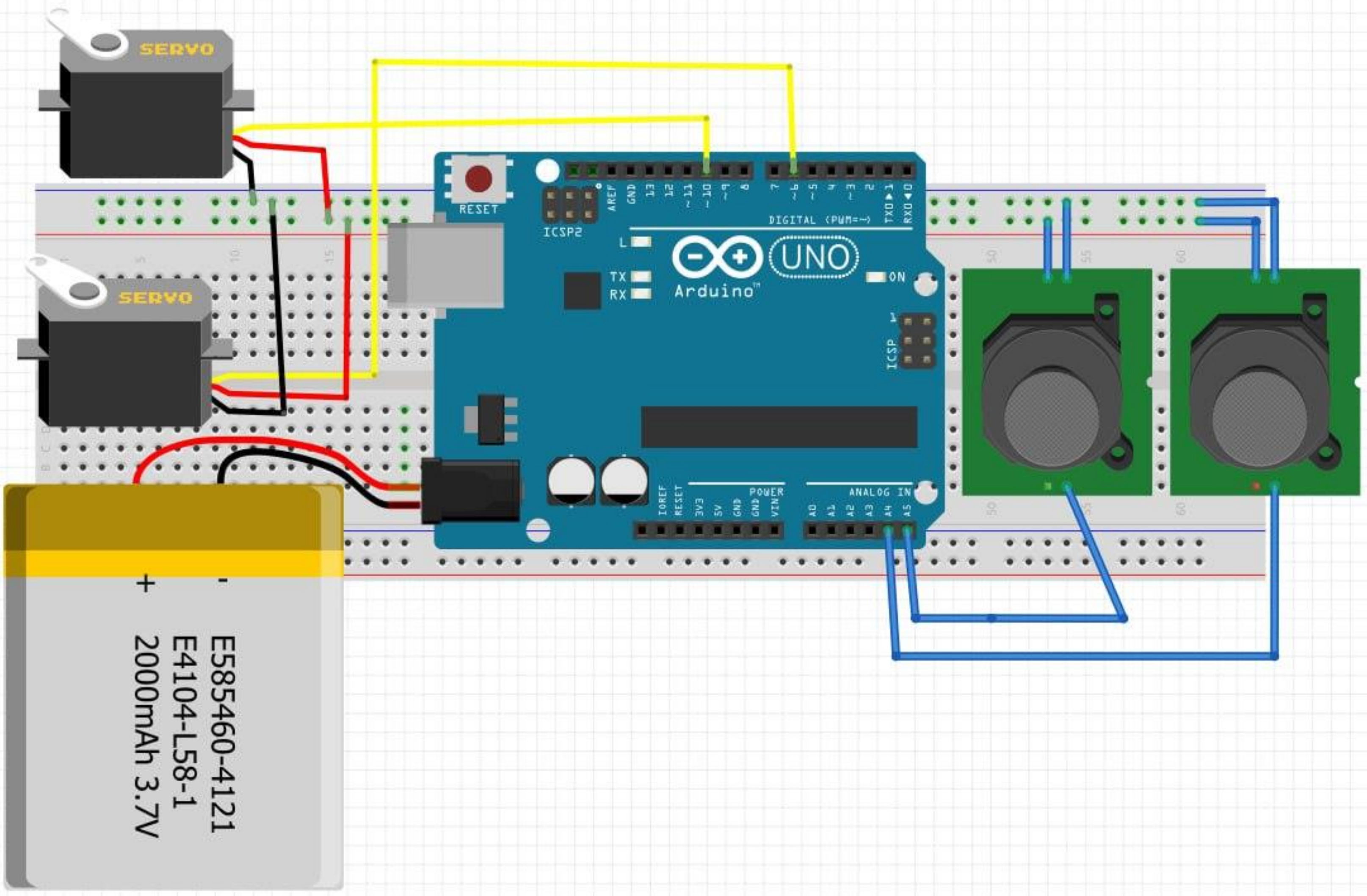


Electronics (Arduino)

Brain Out

Hardware and Elements	Servo , joystick , Breadboard
Programming in Arduino	A simple program to Connects servos to the Joysticks
Build up the circuit	Connecting pins on Arduino board
Final Result	Test it!

Joysticks And Arduino



1. Library

```
#include <Servo.h>
```

2. Objects of the Servo class and define Joysticks

```
Servo servo1;
Servo servo2;

int joy1Pin = A0;  // analog pin used to connect the joystick 1
int joy2Pin = A1;  // analog pin used to connect the joystick 2
```

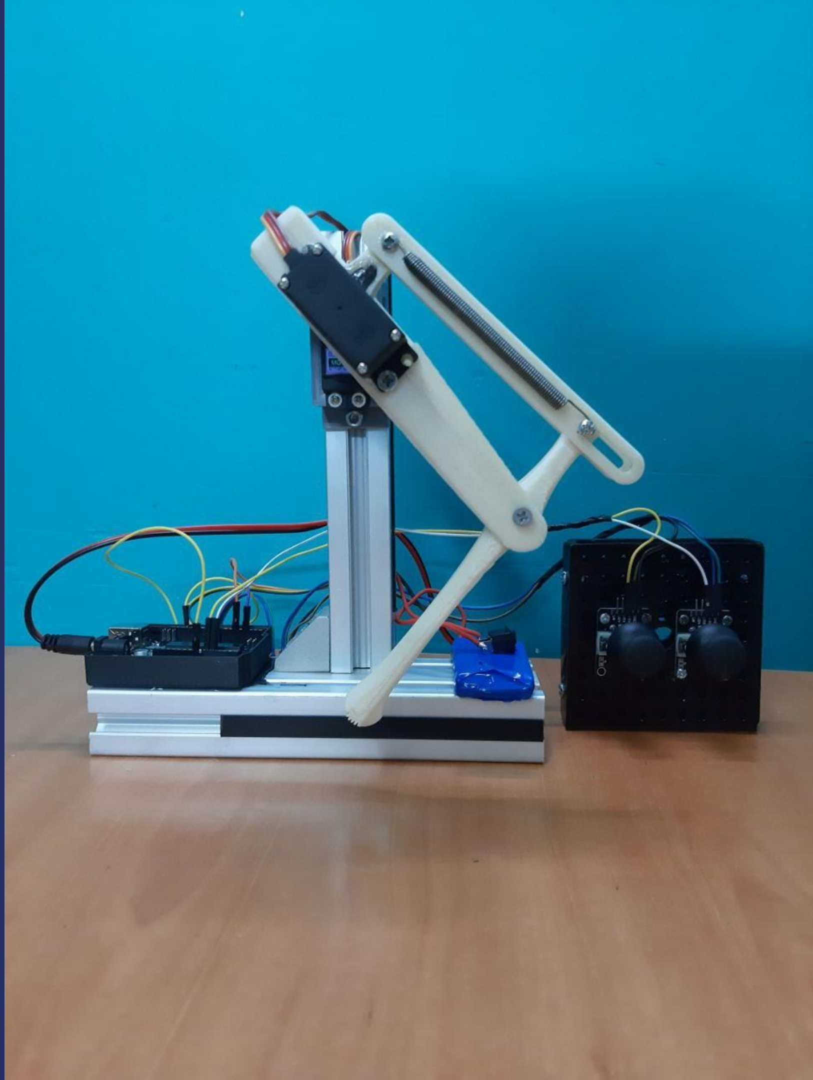
3. Setup function

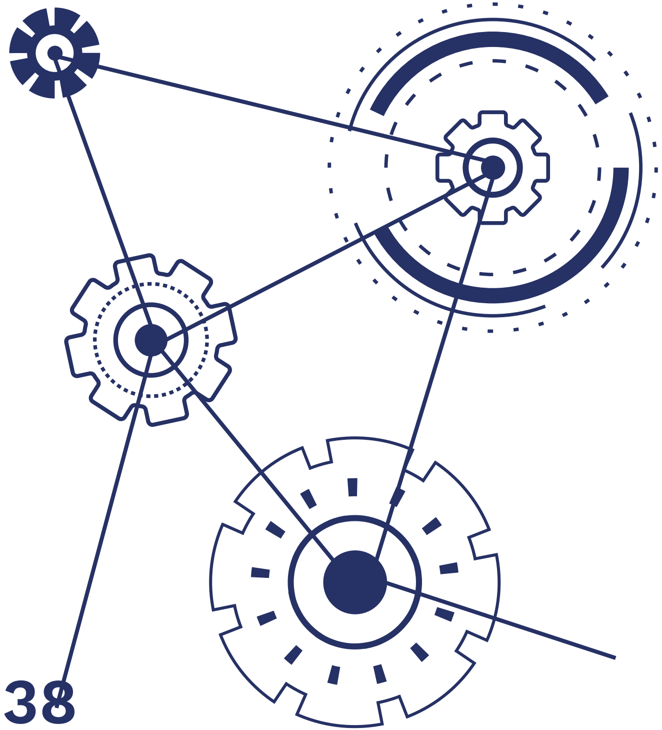
```
void setup()
{
  servo1.attach(3);  // attaches the servo on pin 9
  servo2.attach(5); // attaches the servo on pin 10
}
```

4. Loop function

```
void loop()
{
  int joy1Val = analogRead(joy1Pin);
  int joy2Val = analogRead(joy2Pin);

  // map the joystick values to servo values

  int servo1Val = map(joy1Val, 0, 1023, 0, 100);
  int servo2Val = map(joy2Val, 0, 1023, 0, 100);

  // move the servos
  servo1.write(servo1Val);
  servo2.write(servo2Val);

  delay(15);
}
```

All in one glance



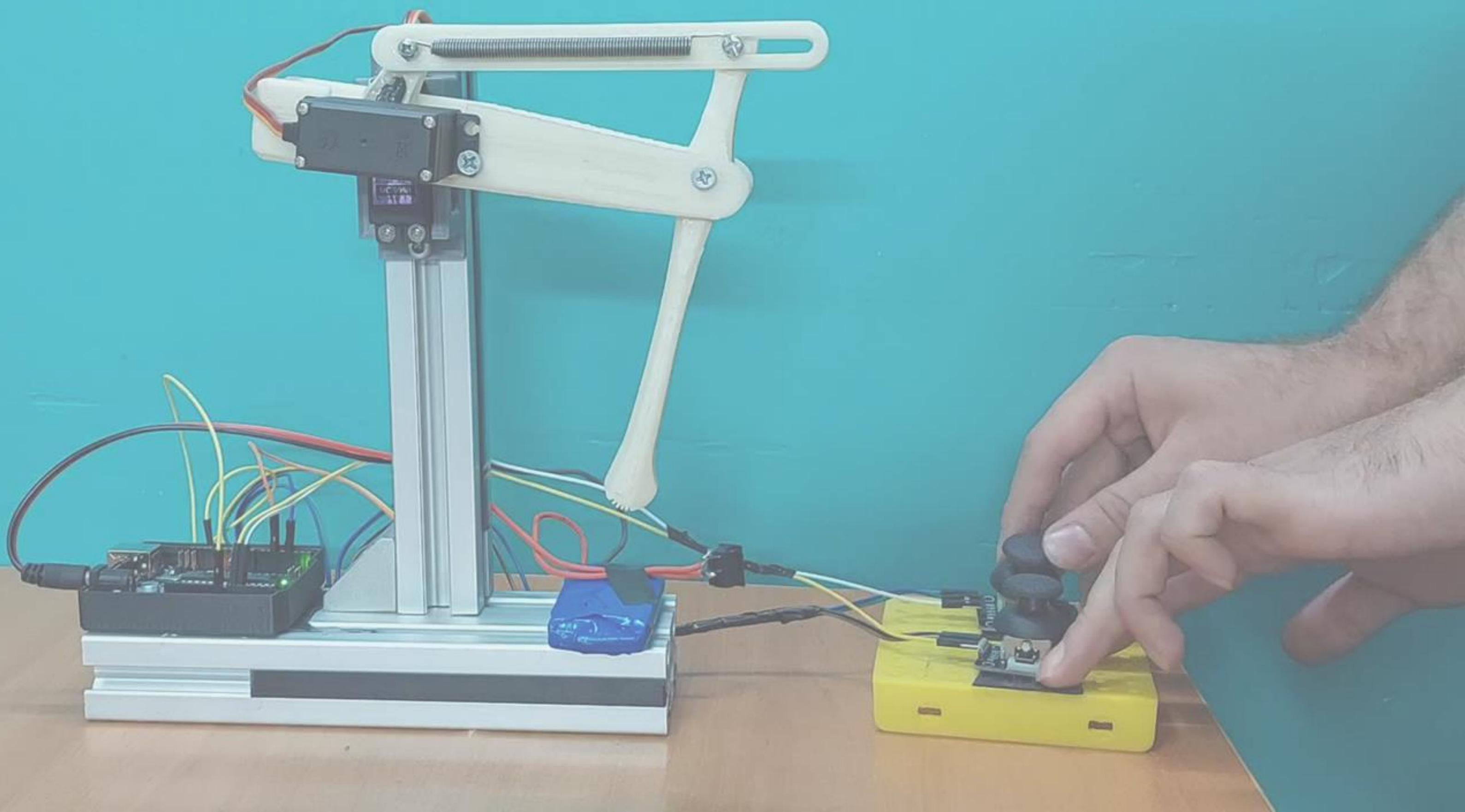


05

Final Result

Thanks!

Do you have any questions?



Sources

....

Literature

- TAHA FAGHANI. (2024). Publisher
  - Designer
- NEGAR ASADI. (2024). Publisher
  - Designer
- ARTIN MOKHTARIHA. (2024). Publisher
  - Designer