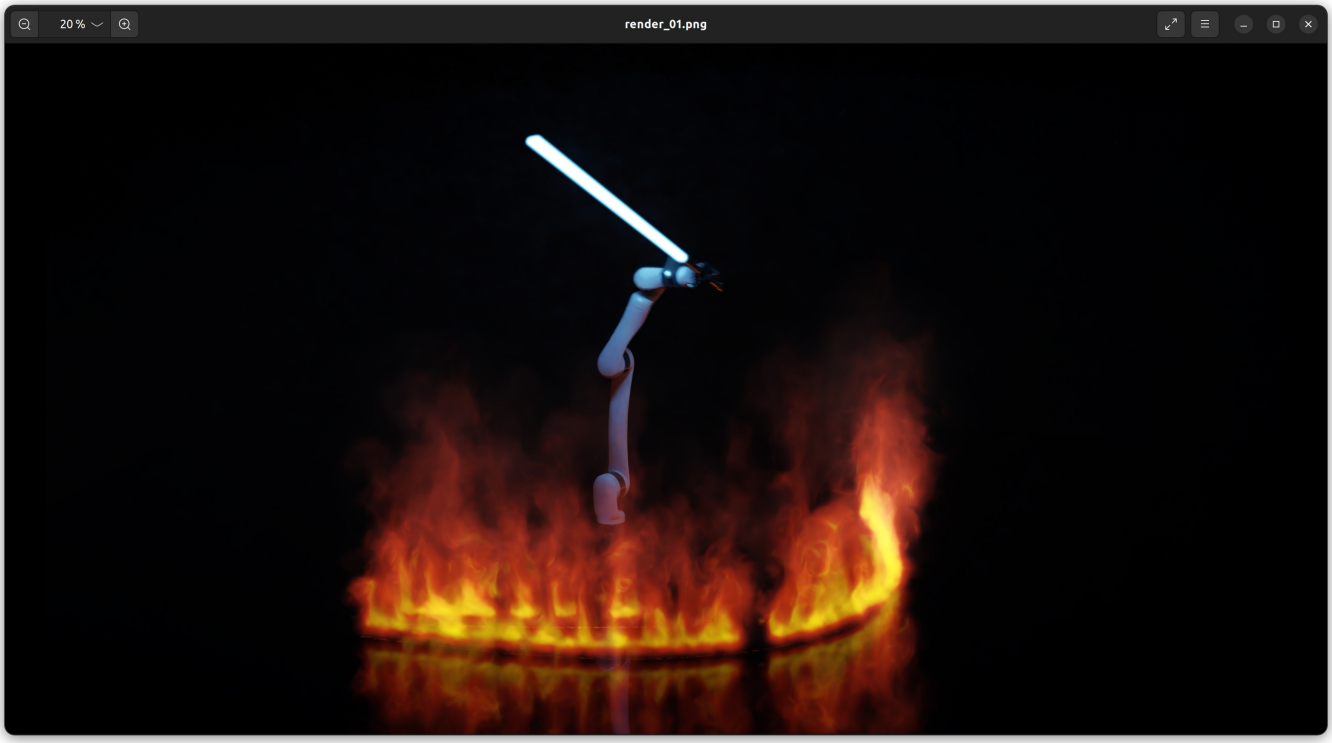
Project 2: C++ FK & IK

Overview

- **The goal of this project was to write the two methods `forward_kinematic` and `inverse_kinematic` to move a 6DOF Robot in blender according to our inputs.**
- **This project was completed by:**
  - CAZAUBON Lorenz *ROB4*
  - KOMARYAN Vahan *ROB4*
  - LEROY Loïc *ROB4*
  - MAUVOISIN Paul *ROB4*
  - PERDREAU Robin *ROB4*
- **Software Versions:**
  - **Blender: *4.3.0***
  - **GNU Make: *4.3***
  - **gcc: *11.4.0***
- **Table of Content**
  1. **Testing**
  2. **How it Works**
  3. **Renderings**
- **Sneak Peek of the Renderings ;)**



1. Testing

Forward Kinematic

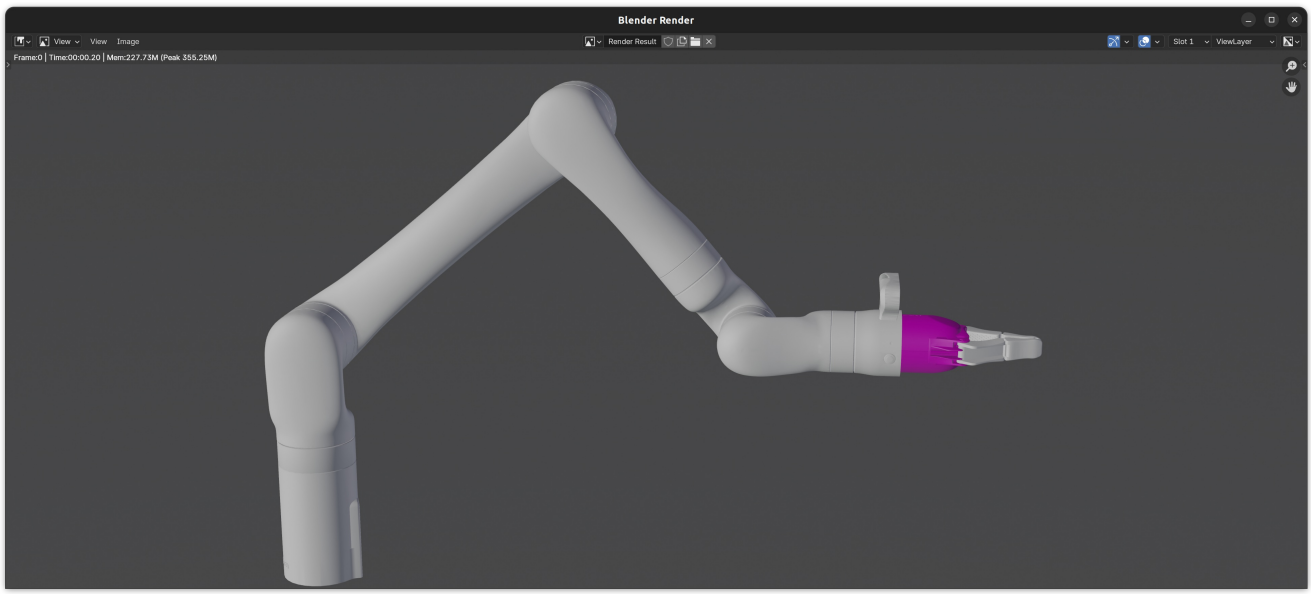
1. The command used to test the `forward_kinematic` method:

```
./robot_kinematic  --angles -70,-50,90,0,-50,-90
```

2. Log of the output

```
################################################################################
##                                                                            ##
##            Forward and Inverse Kinematic (Release version 1.0)             ##
##    Compiled on Sun Dec  8 19:28:48 2024 from source ID 129f65eb44a7+12     ##
##                                                                            ##
################################################################################
2024-12-08T21:51:47GMT [INFO]  Parsing config file config.toml
2024-12-08T21:51:47GMT [INFO]  Creating robot Kinova Gen3 6DF with vision
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../base_link.stl' with 209784 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L00.Base' loaded in 1 ms with 2,126 vertices and 4,194 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../shoulder_link.stl' with 476984 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L01.Shoulder' loaded in 3 ms with 4,793 vertices and 9,538 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../bicep_link.stl' with 783384 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L02.Bicep' loaded in 5 ms with 7,845 vertices and 15,666 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../forearm_link.stl' with 456284 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L03.Forearm' loaded in 3 ms with 4,593 vertices and 9,124 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../spherical_wrist_1_link.stl' with 546484 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L04.Wrist1' loaded in 3 ms with 5,496 vertices and 10,928 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../spherical_wrist_2_link.stl' with 516984 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L05.Wrist2' loaded in 3 ms with 5,173 vertices and 10,338 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../bracelet_with_vision_link.stl' with 2636684 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L06.Camera' loaded in 18 ms with 25,945 vertices and 52,732 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../gripper_base_link.stl' with 1277984 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L07.Effector' loaded in 9 ms with 12,707 vertices and 25,558 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../left_finger_prox_link.stl' with 865584 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L08.LeftFinger' loaded in 6 ms with 8,655 vertices and 17,310 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../left_finger_dist_link.stl' with 654484 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L09.LeftFingerEnd' loaded in 4 ms with 6,546 vertices and 13,088 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../right_finger_prox_link.stl' with 866584 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L10.RightFinger' loaded in 5 ms with 8,665 vertices and 17,330 facets
2024-12-08T21:51:47GMT [INFO]  Parsing binary STL file '.../right_finger_dist_link.stl' with 654484 bytes
2024-12-08T21:51:47GMT [INFO]  Mesh 'L11.RightFingerEnd' loaded in 4 ms with 6,546 vertices and 13,088 facets
2024-12-08T21:51:47GMT [INFO]  Robots creation time: 107 ms
2024-12-08T21:51:47GMT [INFO]  --------------------------------
2024-12-08T21:51:47GMT [INFO]  --   Start Forward Kinematic   --
2024-12-08T21:51:47GMT [INFO]  --------------------------------
2024-12-08T21:51:47GMT [INFO]  --------------------------------
2024-12-08T21:51:47GMT [INFO]  Compute forward kinematic with angles -70,-50,90,0,-50,-90
2024-12-08T21:51:47GMT [INFO]  Target position (x, y, z) = (-0.23507285, -0.64191204, 0.30754)
2024-12-08T21:51:47GMT [INFO]  Target rotation (r, p, y) = (0, 90, -109.99959)
Target Transform = {
{          0,      0.9397,    -0.34201,    -0.23507}
{          0,    -0.34201,     -0.9397,    -0.64191}
{         -1,           0,           0,     0.30754}
{          0,           0,           0,           1}
}
```

3. In Blender



Inverse Kinematic

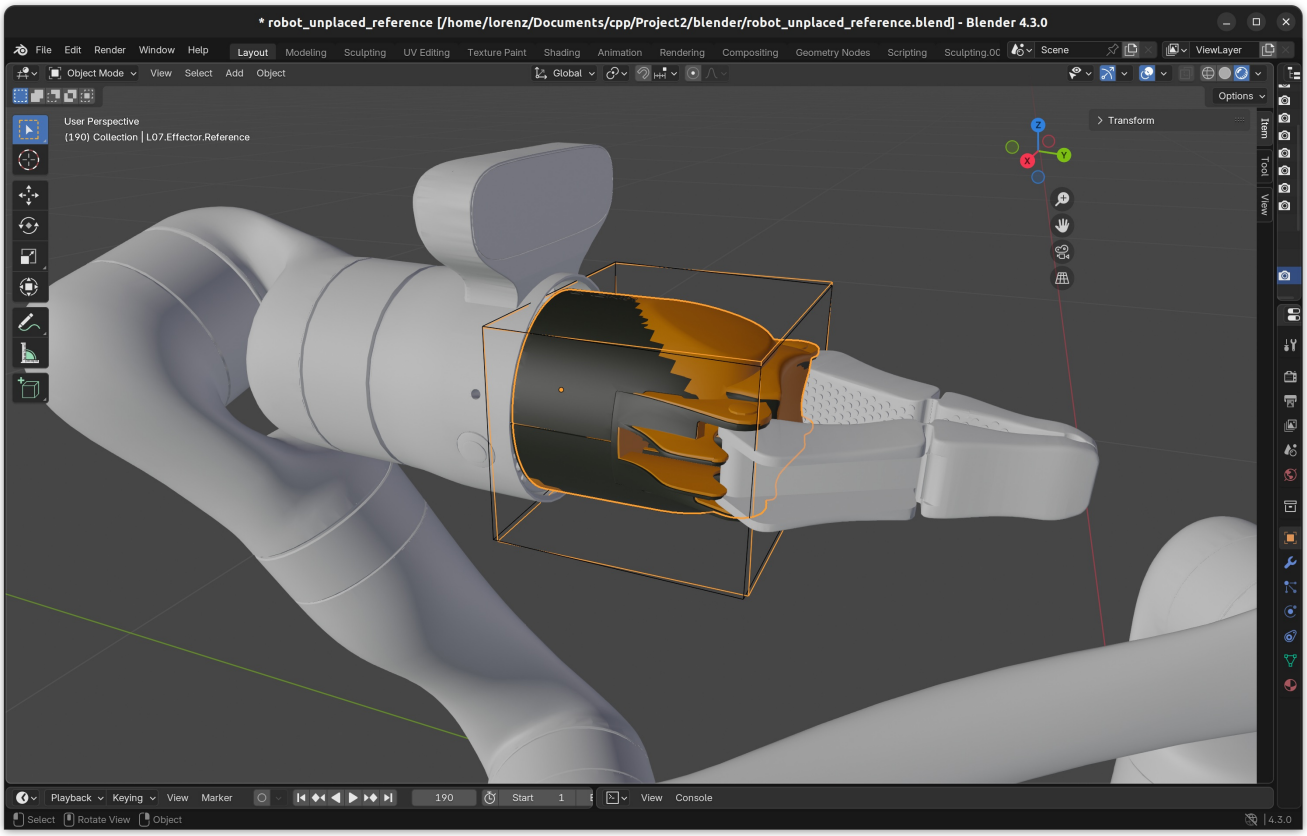
1. The command used to test the `inverse_kinematic` method:

```
./robot_kinematic  --angles -70,-50,90,0,-50,-90 --xyz 0.302,-0.294,0.486 --rpy 0,90,90
```

2. Log of the output

```
2024-12-08T21:04:03GMT [INFO]  Robots creation time: 107 ms
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  --   Start Forward Kinematic   --
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  Compute forward kinematic with angles -70,-50,90,0,-50,-90
2024-12-08T21:04:03GMT [INFO]  Target position (x, y, z) = (-0.23507285, -0.64191204, 0.30754)
2024-12-08T21:04:03GMT [INFO]  Target rotation (r, p, y) = (0, 90, -109.99959)
Target Transform = {
{          0,      0.9397,    -0.34201,    -0.23507}
{          0,    -0.34201,     -0.9397,    -0.64191}
{         -1,           0,           0,     0.30754}
{          0,           0,           0,           1}
}
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  --   Start Inverse Kinematic   --
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  Using seed 877825431
2024-12-08T21:04:03GMT [INFO]  Target position is (0.302,-0.294,0.486)
2024-12-08T21:04:03GMT [INFO]  Target rotation is (0,90,90)
2024-12-08T21:04:03GMT [INFO]  Distance threshold: 0.01
2024-12-08T21:04:03GMT [INFO]  Angle resolution bits: 16
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  --           Threads           --
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  Number of threads: 32
2024-12-08T21:04:03GMT [INFO]  Thread task nb: 0, current best: 0.1308731
2024-12-08T21:04:03GMT [INFO]  Thread  0 succeeded, distance: 0.009944662
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  --           Results           --
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  Reached Target: true
2024-12-08T21:04:03GMT [INFO]  Final distance: 0.009944662
2024-12-08T21:04:03GMT [INFO]  Elapsed time: 11 ms
2024-12-08T21:04:03GMT [INFO]  Joint angles are (-123.337,-100.095,-71.875,37.1317,-114.408,106.533)
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  --    Write Blender Script     --
2024-12-08T21:04:03GMT [INFO]  --------------------------------
2024-12-08T21:04:03GMT [INFO]  Generate blender script to file my_script.py
2024-12-08T21:04:03GMT [INFO]  Script file generated with 191 poses
```

3. In Blender



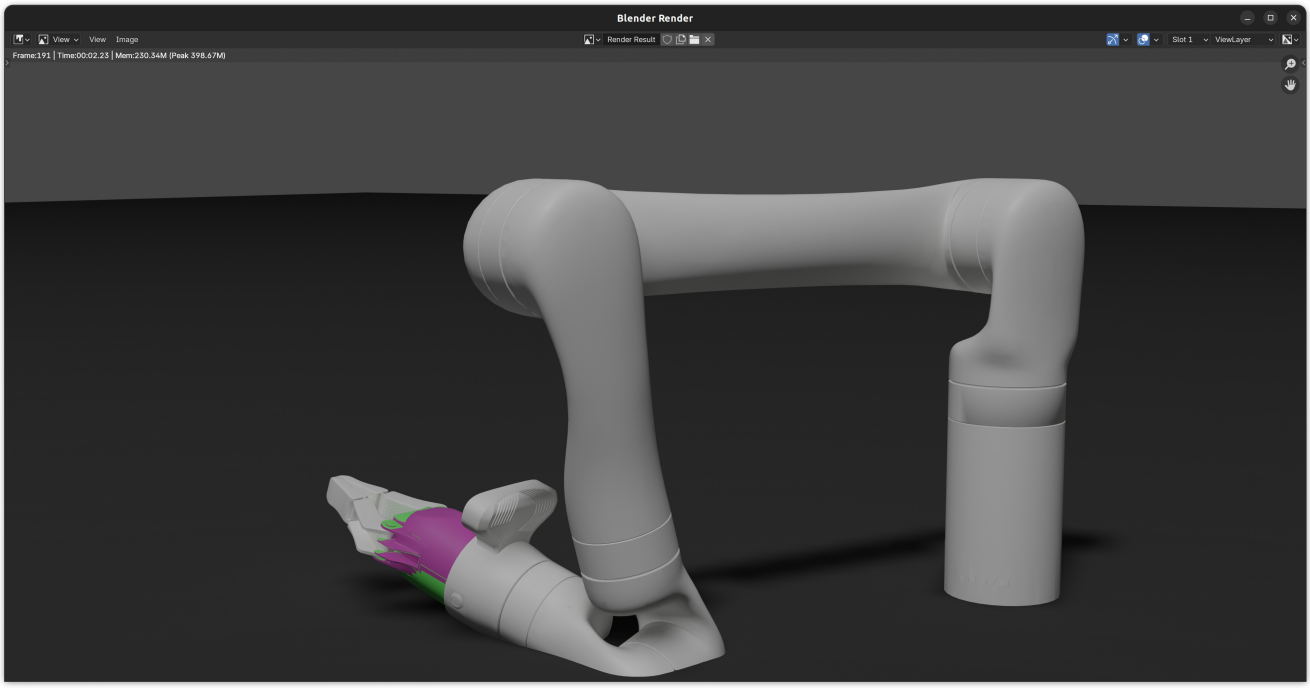
Collision Detection

To test the collision detection with the ground we ran the same IK with and without this feature enabled and compared the results: *(Self Collision Not Implemented Yet)*

- **Without**

```
./robot_kinematic  --angles -700,-50,90,0,-50,-90 --xyz 0.302,-0.294,0.036 --rpy 0,70,-90
```

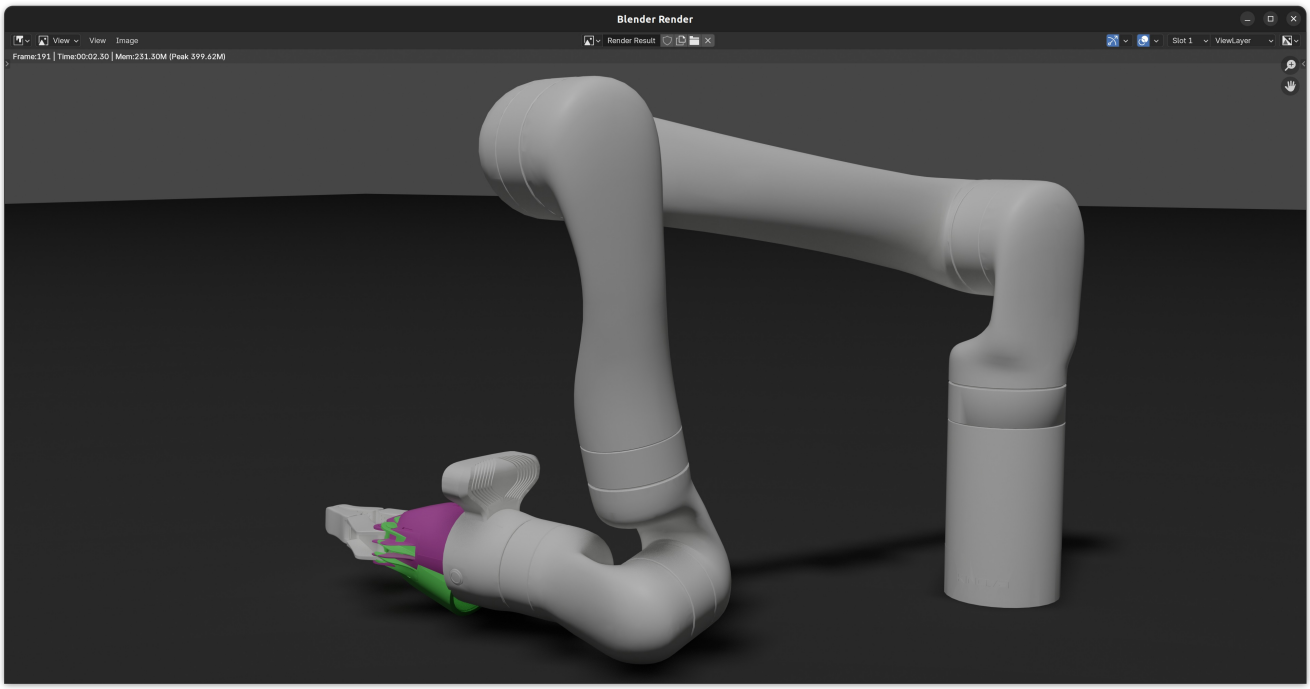
```
2024-12-08T21:46:50GMT [INFO]  Reached Target: true
2024-12-08T21:46:50GMT [INFO]  Final distance: 0.009937733
2024-12-08T21:46:50GMT [INFO]  Elapsed time: 21 ms
2024-12-08T21:46:50GMT [INFO]  Joint angles are (24.2631,89.5963,-105.149,-71.0075,114.124,75.3002)
```



- **With**

```
./robot_kinematic  --angles -700,-50,90,0,-50,-90 --xyz 0.302,-0.294,0.036 --rpy 0,70,-90  --method collisions
```

```
2024-12-08T21:43:34GMT [INFO]  Reached Target: false
2024-12-08T21:43:34GMT [INFO]  Final distance: 0.12970564
2024-12-08T21:43:34GMT [INFO]  Elapsed time: 1323 ms
2024-12-08T21:43:34GMT [INFO]  Joint angles are (22.2126,80.3433,-112.476,-70.9084,97.0316,77.2117)
```



2. How it Works

- **Multi Threading**
  - First of all, we create 1 Robot per Thread *(We duplicate the data, to prevent having to reparse and reload the meshes)*, this will allow us to compute on each thread without having issues with shared assets.
  - Each Thread will compute 1 task, which is an `inverse_kinematic` try. Until either the target or `max_threads_tasks` is reached.
  - Since each thread task has a different seed, we minimize the exposure to local minimas.
  - If one thread reaches the target we stop early, if not we will keep the "best" value/robot found by all the threads tasks.
- **Inverse Kinematic**
  - For `nb_iterations * degrees_of_freedom` we select 1 joint at random.
  - For this joint we try every angle between min_angle and max_angle with a step defined by the angle resolution.
  - To improve speed the min and max angles are reduced the more iterations the algorithm goes through.
  - We keep the best position, the one that reduces the distance between the EndEffector Bbox and the Target Bbox the most, and if the collisions detection is enabled we check this condition aswell.
  - If the target is reached, we stop early.
  - Return the `best_distance`.
- **Collision Detection**
  - We go through all the links in the robot.
  - Get the transformed Bbox for each of them and check if one of its vertices is below ground.
  - Return the `boolean`.
- **Forward Kinematic**
  - We go through all the joints and set them to the specified angle.
- **Interpolation**
  - This method is used for the animation, it generates poses between a start_pose and an end_pose, which allow us to do some smooth renderings in blender.
  - It works by moving each joint by a small step angle.
  - Save the pose of the robot.
  - and move by another step, and so on.

3. Renderings

Check the animated version here : Animation Render