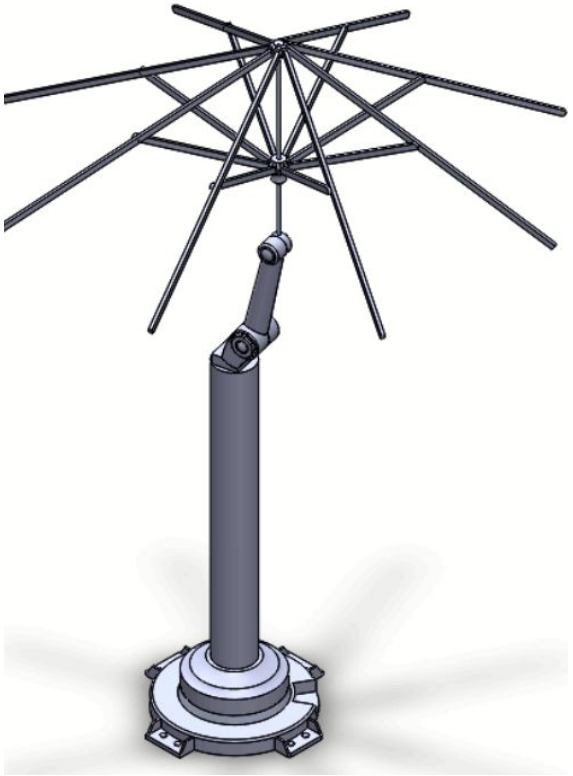
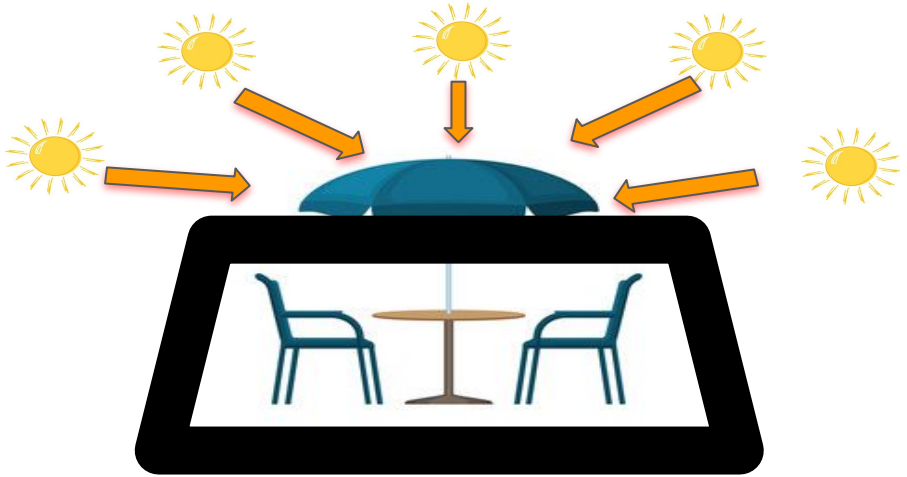
ALFA CENTAURI

Victor Daniel Alvarez Valencia
Diego Ivan Perea Montealegre
Juan Pablo Agudelo Barajas

ALFA CENTAURE



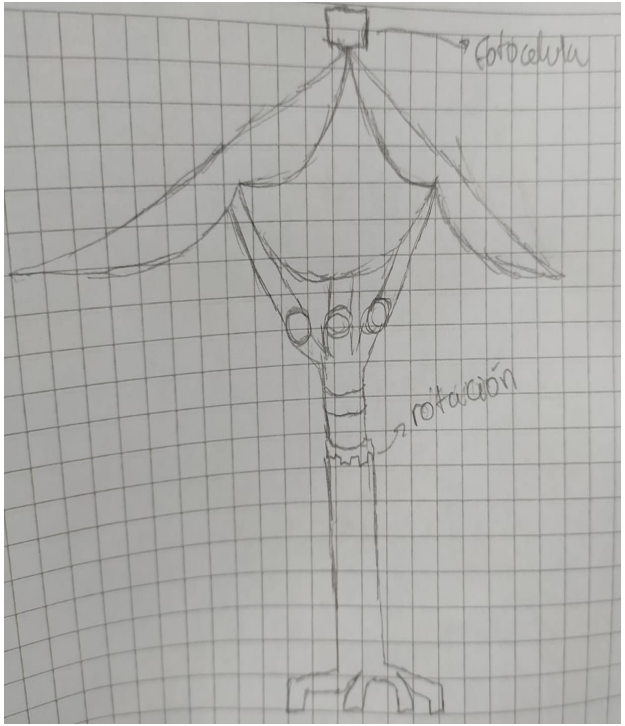
34% of the total ultraviolet radiation is filtered through the sunshade umbrellas also did well, **blocking** at least **77%** of UV light .

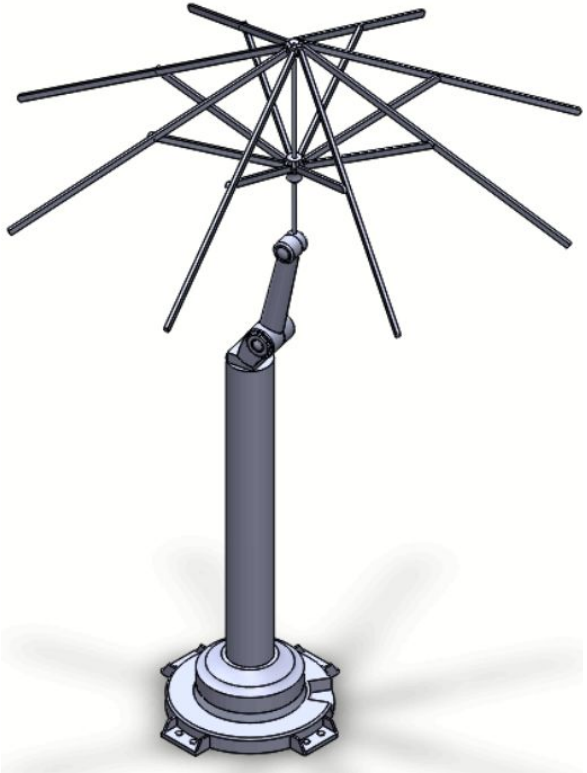


SHADOWN

ALFA CENTAURE

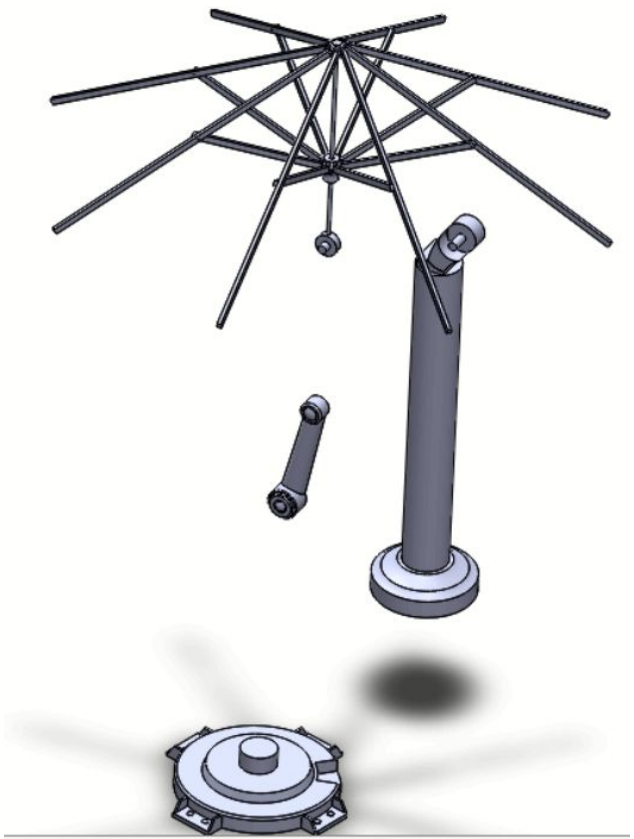
DESIGNED 3D





ALFA CENTAURI

EXPLODED VIEW

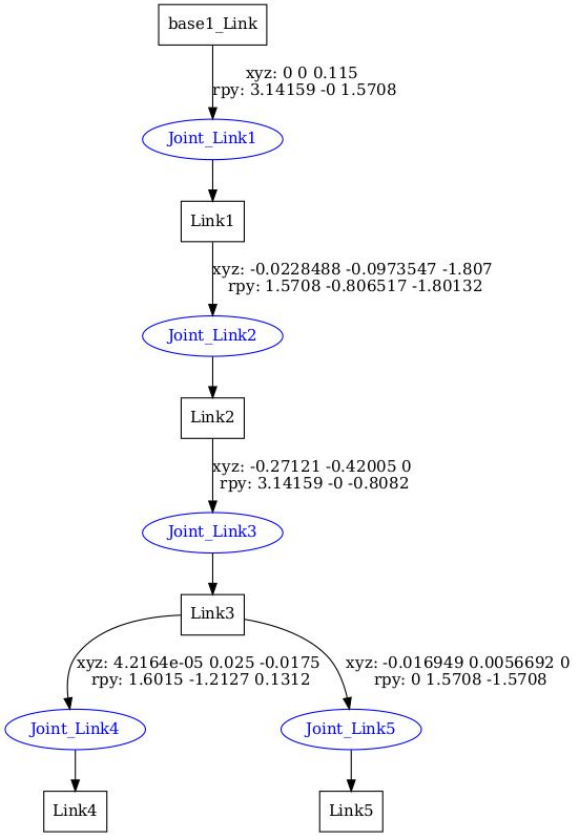


The morphology of the robot is based on the structure of a normal table umbrella, with the difference that this design consists of three parts:

- The base which is a rotational joint
- The robotic arm is rotational about the axis of its joint
- The upper part of the parasol behaves as a single extension

ALFA CENTAURE

URDF CODE :




```xml
<?xml version="1.0" encoding="utf-8"?>
<robot
  name="robotmejorado4">
  <link
    name="base1_Link">
    <inertial>
      <origin
        xyz="0.00530265169310419 0.00528029914101924 0.0513462352345183"
        rpy="0 0 0" />
      <mass
        value="37.3619805208222" />
      <inertia
        ixx="1.07878719190163"
        ixy="0.0421503007964398"
        ixz="0.00134509247170474"
        iyy="1.07844693370143"
        iyz="0.00131074568973874"
        izz="2.053204468573" />
    </inertial>
    <visual>
      <origin
        xyz="0 0 0"
        rpy="0 0 0" />
      <geometry>
        <mesh
          filename="package://robot3_description/meshes/base_Link.STL" />
      </geometry>
      <material
        name="">
        <color
          rgba="0.458 0.6 0.92564 1" />
      </material>
    </visual>
    <collision>
      <origin
        xyz="0 0 0"
        rpy="0 0 0" />
      <geometry>
        <mesh
          filename="package://robot3_description/meshes/base_Link.STL" />
      </geometry>
    </collision>
  </link>
```

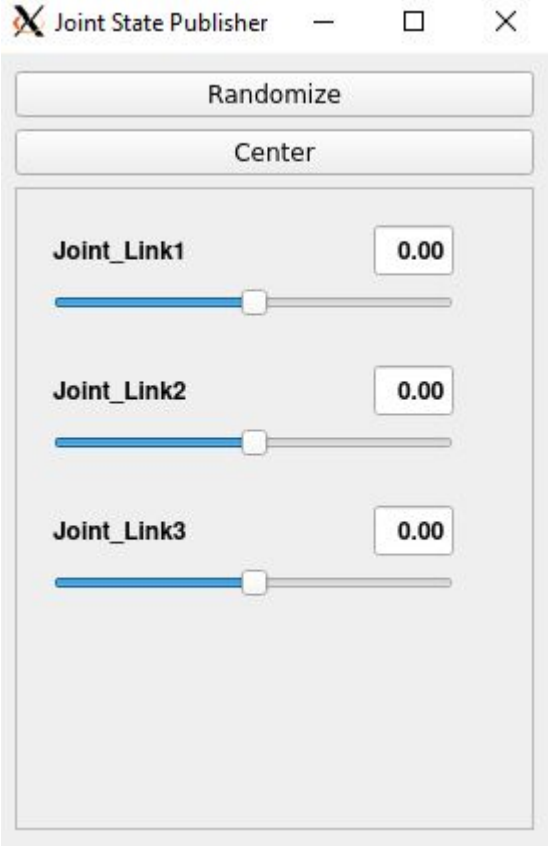
```xml
  <link
    name="Link1">
    <inertial>
      <origin
        xyz="-0.00150165357461152 -0.00252217720068532 -0.728144343309596"
        rpy="0 0 0" />
      <mass
        value="68.2835579500472" />
      <inertia
        ixx="22.0411080887108"
        ixy="-0.00744416311395271"
        ixz="-0.109354248005661"
        iyy="22.0362256160777"
        iyz="-0.182138134559127"
        izz="0.69058697507177" />
    </inertial>
    <visual>
      <origin
        xyz="0 0 0"
        rpy="0 0 0" />
      <geometry>
        <mesh
          filename="package://robot3_description/meshes/Link_Tronco.STL" />
      </geometry>
      <material
        name="">
        <color
          rgba="1 1 1 1" />
      </material>
    </visual>
    <collision>
      <origin
        xyz="0 0 0"
        rpy="0 0 0" />
      <geometry>
        <mesh
          filename="package://robot3_description/meshes/Link_Tronco.STL" />
      </geometry>
    </collision>
  </link>
  <joint
    name="Joint_Link1"
    type="continuous">
```

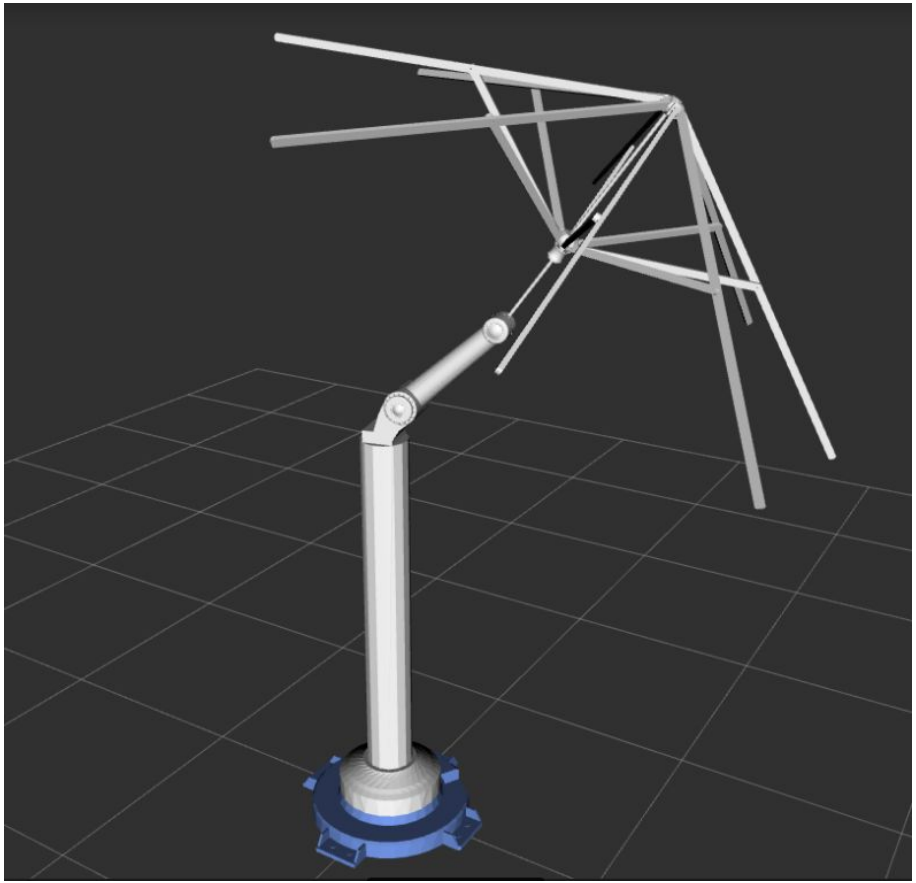
ALFA CENTAURI

launch

```python
import os
from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument
from launch.conditions import IfCondition, UnlessCondition
from launch.substitutions import Command, LaunchConfiguration
from launch_ros.actions import Node
from launch_ros.substitutions import FindPackageShare

def generate_launch_description():

  # Set the path to different files and folders.
  pkg_share = FindPackageShare(package='robot3_description').find('robot3_description')
  default_launch_dir = os.path.join(pkg_share, 'launch')
  default_model_path = os.path.join(pkg_share, 'models/robotmejorado4.urdf')
  robot_name_in_urdf = 'robotmejorado4'
  default_rviz_config_path = os.path.join(pkg_share, 'rviz/urdf_config.rviz')

  # Launch configuration variables specific to visualization

  gui = LaunchConfiguration('gui')
  model = LaunchConfiguration('model')
  rviz_config_file = LaunchConfiguration('rviz_config_file')
  use_robot_state_pub = LaunchConfiguration('use_robot_state_pub')
  use_rviz = LaunchConfiguration('use_rviz')
  use_sim_time = LaunchConfiguration('use_sim_time')

  # Declare the launch arguments
  declare_model_path_cmd = DeclareLaunchArgument(
    name='model',
    default_value=default_model_path,
    description='Absolute path to robot urdf file')

  declare_rviz_config_file_cmd = DeclareLaunchArgument(
    name='rviz_config_file',
    default_value=default_rviz_config_path,
    description='Full path to the RVIZ config file to use')

  declare_use_joint_state_publisher_cmd = DeclareLaunchArgument(
    name='gui',
    default_value='True',
    description='Flag to enable joint_state_publisher_gui')
```

After obtaining the urdf file in which all the description of the robot is contained, the launch file was created which will call the description of the robot, the union of the Joints, and the positions for the transformation in tf2

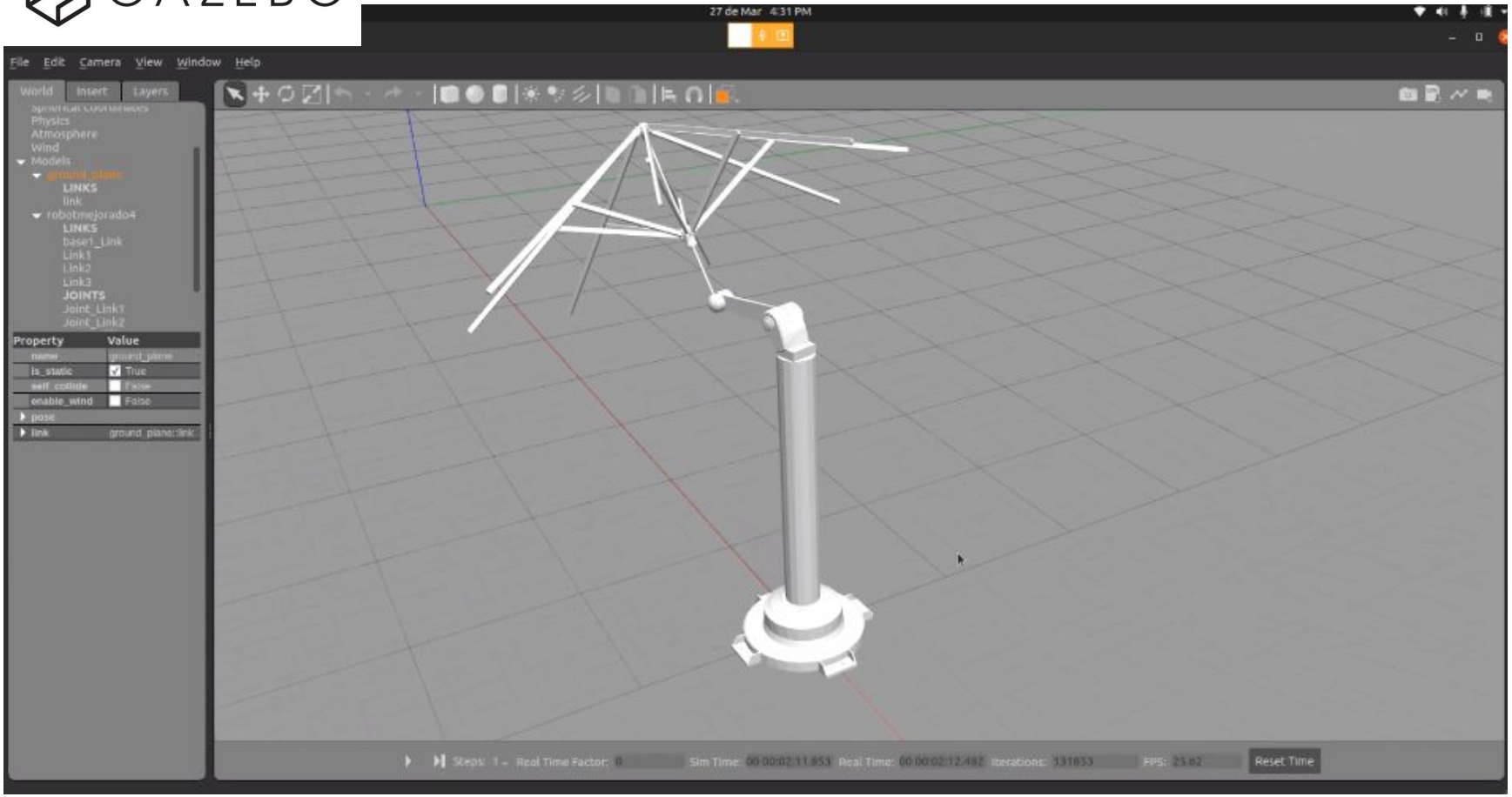




ALFA CENTAURE

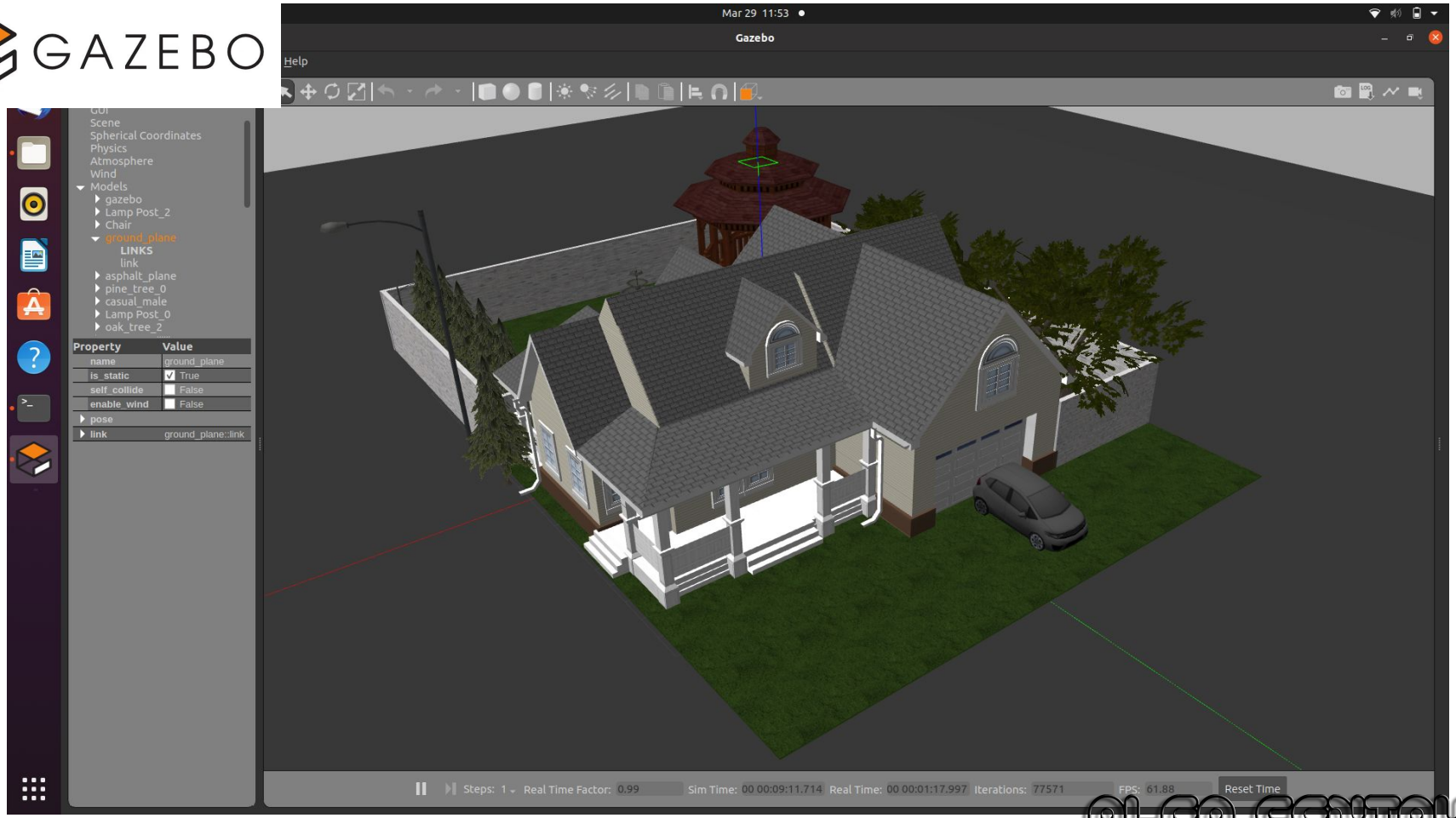
VISUALIZATION VIDEO

GAZEBO



ALFA CENTAURI

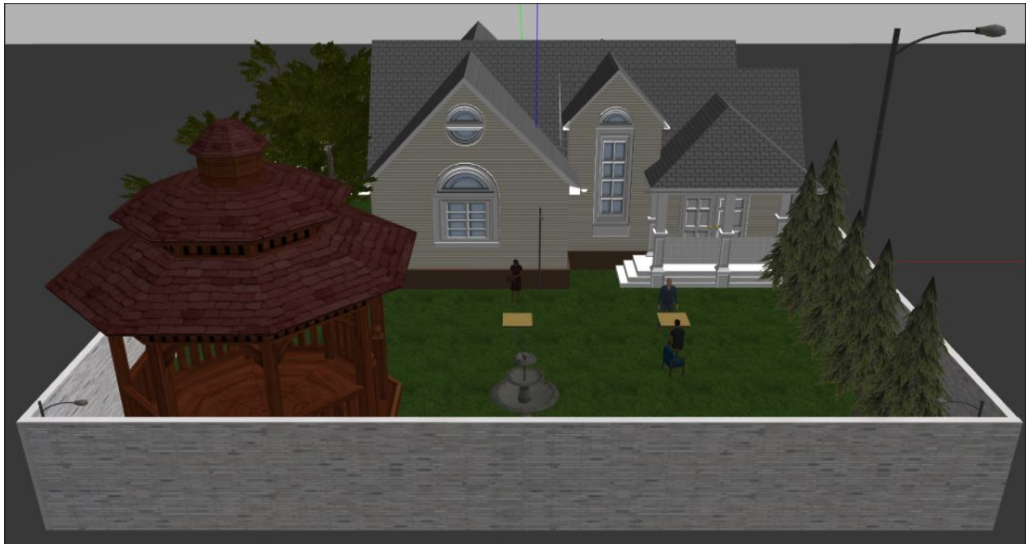
WORLD WHERE THE PROJECT WILL BE APPLIED



What does this World consist of and what is its relationship with our project?

The device is mainly focused on a commercial space where restaurants and shopping centers can easily implement these systems due to the energy consumption and the same cost that the proposed system can represent. However, in this simulation the system can be seen in a familiar atmosphere and the interaction with the environment.

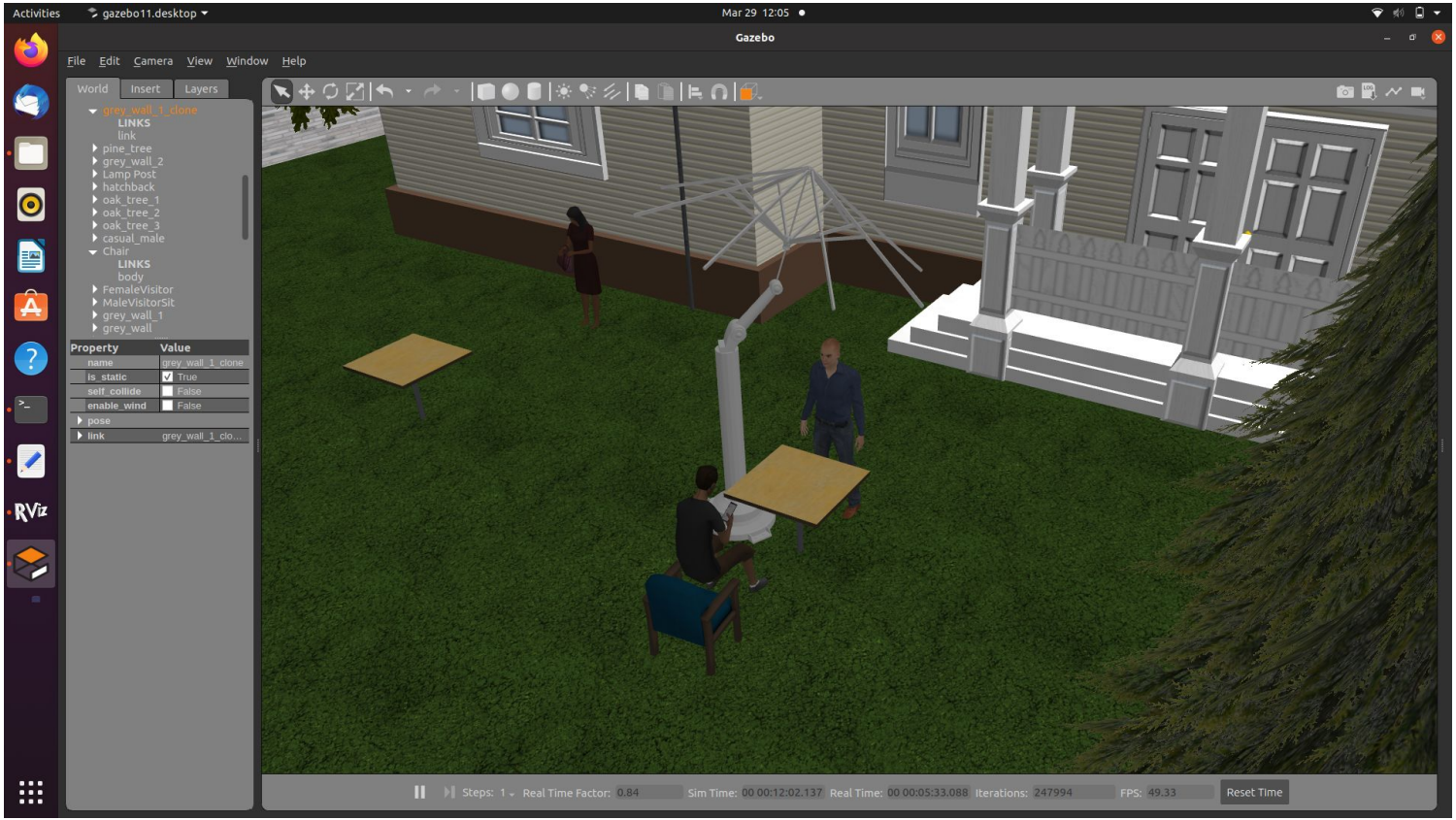
GAZEBO



ALFA CENTAURE

GAZEBO

First robot prototype in the world designed



ALFA CENTAURE

GAZEBO

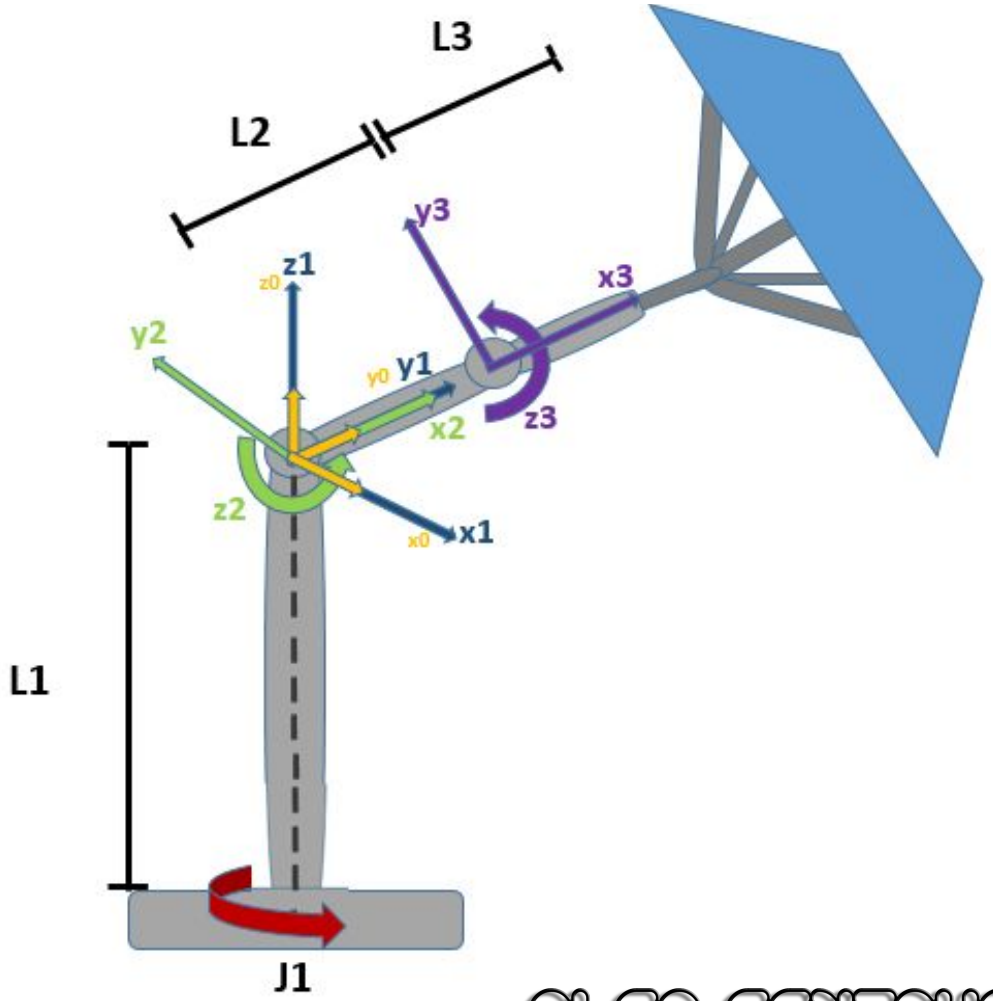
Other world

ALFA CENTAURE


GAZEBO
ALFA CENTAURE

Direct kinematic

i	αi-1	ai-1	di	Θi
1	0	0	0	Θ_1
2	90°	0	0	Θ_2
3	0	L_2	0	Θ_3



$$ {}^{0}_{1}T = \begin{bmatrix} c1 & -s1 & 0 & 0 \\ s1 & c1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{1}_{2}T = \begin{bmatrix} c2 & -s2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s2 & c2 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}_{2}T = \begin{bmatrix} c3 & -s3 & 0 & L_2 \\ s3 & c3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

i	αi-1	ai-1	di	Θi
1	0	0	0	Θ_1
2	90°	0	0	Θ_2
3	0	L_2	0	Θ_3

Inverse kinematic

$$ {}^{0}_{3}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T $$

$$ ({}^{0}_{1}T)^{-1}T = {}^{1}_{2}T\,{}^{2}_{3}T $$

$$ \begin{bmatrix} c1 & s1 & 0 & 0 \\ -s1 & c1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_x & O_x & a_x & P_x \\ n_y & O_y & a_y & P_y \\ n_z & O_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = {}^{1}_{2}T\,{}^{2}_{3}T $$

Choosing the element (3,4)

$$ -P_x s1 + P_y c1 = 0 $$

$$ \theta_1 = arctan2\left(\frac{P_y}{P_x}\right) $$

$$({}^{1}_{2}T)^{-1}({}^{0}_{1}T)^{-1}T = {}^{2}_{3}T$$

$$\begin{bmatrix} c2 & 0 & s2 & 0 \\ -s2 & 0 & c2 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c1 & s1 & 0 & 0 \\ -s1 & c1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} n_x & O_x & a_x & P_x \\ n_y & O_y & a_y & P_y \\ n_z & O_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = {}^{2}_{3}T$$

Choosing the element (2,4)

$$-P_x c1s2 - P_y s1s2 + P_z c2 = 0$$

$$s2\left(-P_x c1 - P_y s1\right) + P_z c2 = 0$$

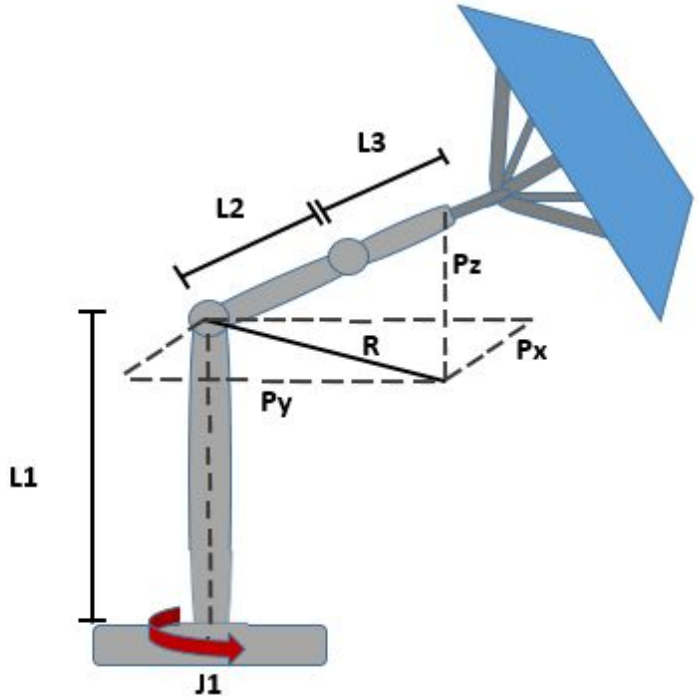
$$\theta_2 = arctan2 - \frac{\left(-P_x c1 - P_y s1\right)}{P_z}$$

Choosing the element (1,4)

$$P_x c1c2 + P_y c2s1 + P_z s2 = L_2$$

$$c2\left(P_x c1 + P_y s1\right) + P_z s2 = L_2$$

θ3 the geometric method was applied



$$R^2 = P_y{}^2 + P_x{}^2$$

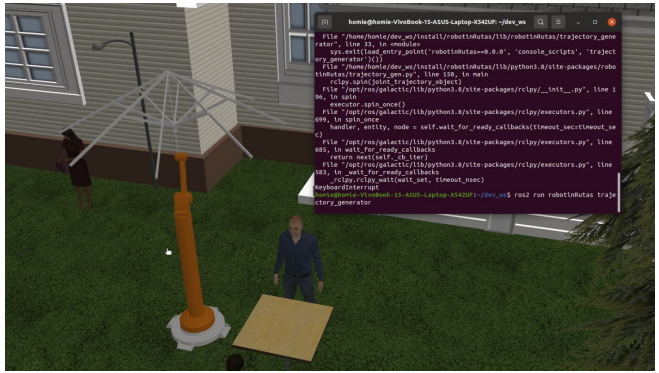
$$R^2 + P_z{}^2 = L_2^2 + L_3^2 + 2 * L_2^2 * L_3^2 cosq_3$$

$$cosq_3 = \frac{P_x{}^2 + P_y{}^2 + P_z{}^2 - L_2^2 - L_3^2}{2 * L_2^2 * L_3^2}$$

$$sinq_3 = \pm\sqrt{1 - cos^2 q_3}$$

$$\theta_3 = arctan2\frac{\pm\sqrt{1 - cos^2 q_3}}{cosq_3}$$

Video of trajectories and code



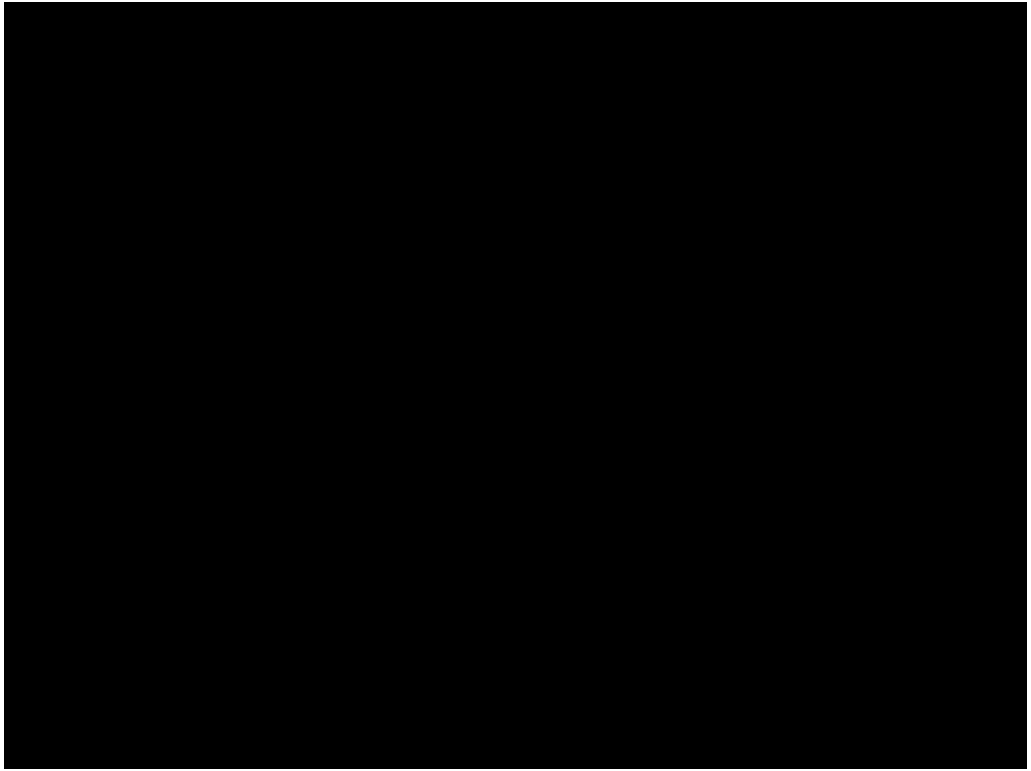

```python
class Trajectory_publisher(Node):

    def __init__(self):
        super().__init__('trajectory_publsiher_node')
        publish_topic = "/joint_trajectory_controller/joint_trajectory"
        self.trajectory_publihser = self.create_publisher(JointTrajectory,publish_topic, 10)
        timer_period = 7
        self.timer = self.create_timer(timer_period, self.timer_callback)
        CallAngulos = self.angulosinv()
        θ1= CallAngulos[0]
        θ2 = CallAngulos[1]
        θ3 = CallAngulos[2]
        θ11 = CallAngulos[3]
        θ21 = CallAngulos[4]
        θ31 = CallAngulos[5]
        θ12= CallAngulos[6]
        θ22 = CallAngulos[7]
        θ32 = CallAngulos[8]
        self.joints = ['joint1','joint2','joint3']
        self.goal_positions = [θ1,θ2,θ3]
        self.goal_positions1 = [θ11,θ21,θ31]
        self.goal_positions2 = [θ12,θ22,θ32]

    def angulosinv(self):
        Px=8
        Py=-1
        Pz=4
        l2=5
        l3=5
        θ1=arctan2(Py,Px)
        θ2=arctan2(Px*cos(θ1)+Py*sin(θ1),Pz)
        cosq3=(Px**2+Py**2+Pz**2-l2**2-l3**2)/(2*l2**2*l3**2)
        sinq3=-sqrt(1-cosq3**2)
        θ3=arctan2(sinq3,cosq3)

        Px1=-5
        Py1=-1
        Pz1=4
        l21=5
        l31=5
        θ11=arctan2(Py1,Px1)
        θ21=arctan2(Px1*cos(θ11)+Py*sin(θ11),Pz1)
        cosq3=(Px1**2+Py1**2+Pz1**2-l21**2-l31**2)/(2*l21**2*l31**2)
        sinq3=sqrt(1-cosq3**2)
        θ31=arctan2(sinq3,cosq3)
```

```python
        Px4=6
        Py4=-1
        Pz4=4
        l24=5
        l34=5
        θ14=arctan2(Py3,Px3)
        θ24=arctan2(Px3*cos(θ1)+Py3*sin(θ1),Pz3)
        cosq3=(Px4**2+Py4**2+Pz4**2-l24**2-l34-3**2)/(2*l24**2*l34**2)
        sinq3=-sqrt(1-cosq3**2)
        θ34=arctan2(sinq3,cosq3)

        return [θ1, θ2, θ3, θ11, θ21, θ31, θ12, θ22, θ32, θ13, θ23, θ33, θ14, θ24, θ34]


    def timer_callback(self):
        robotinRutas_trajectory_msg = JointTrajectory()
        robotinRutas_trajectory_msg.joint_names = self.joints
        ## creating a point
        point = JointTrajectoryPoint()
        point.positions = self.goal_positions
        point.time_from_start = Duration(sec=4)

        robotinRutas_trajectory_msg.points.append(point)
        ## adding newly created point into trajectory message
        point_1 = JointTrajectoryPoint()
        point_1.positions = self.goal_positions1
        point_1.time_from_start = Duration(sec=10)

        robotinRutas_trajectory_msg.points.append(point_1)

        point_2 = JointTrajectoryPoint()
        point_2.positions = self.goal_positions2
        point_2.time_from_start = Duration(sec=14)

        robotinRutas_trajectory_msg.points.append(point_2)

        point_3 = JointTrajectoryPoint()
        point_3.positions = self.goal_positions2
        point_3.time_from_start = Duration(sec=15)

        robotinRutas_trajectory_msg.points.append(point_3)

        point_4 = JointTrajectoryPoint()
        point_4.positions = self.goal_positions2
        point_4.time_from_start = Duration(sec=17)

        robotinRutas_trajectory_msg.points.append(point_4)

        self.trajectory_publihser.publish(robotinRutas_trajectory_msg)
```

ALFA CENTAURI

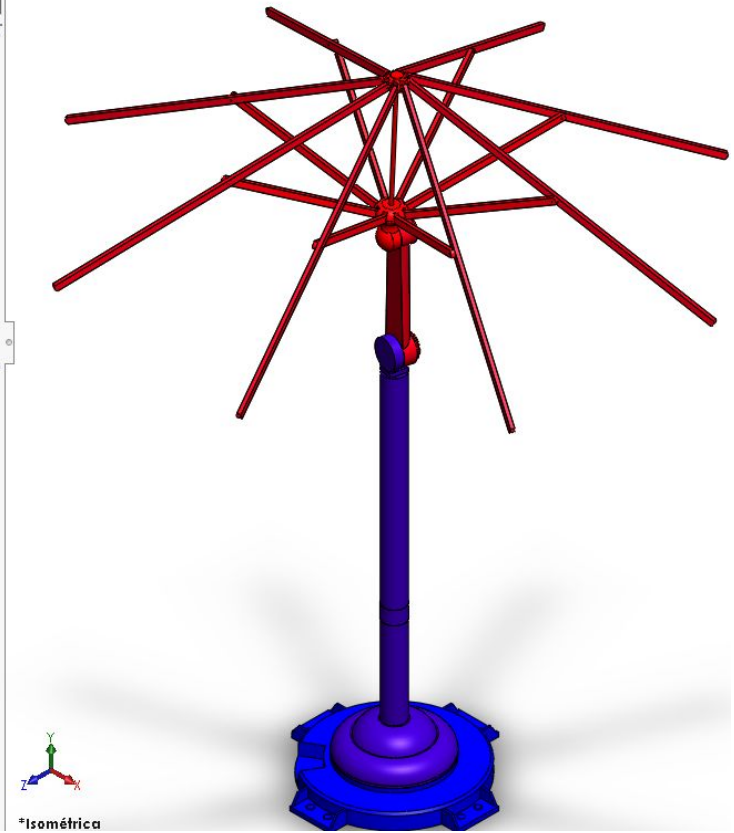
Video of trajectories

```
homie@homie-VivoBook-15-ASUS-Laptop-X542UF:~$ ros2 topic echo /joint_trajectory_controller/joint_trajectory
header:
  stamp:
    sec: 0
    nanosec: 0
  frame_id: ''
joint_names:
- joint1
- joint2
- joint3
points:
- positions:
  - -0.16514867741462683
  - 0.989099767628959
  - -1.5683963244908907
  velocities: []
  accelerations: []
  effort: []
  time_from_start:
    sec: 4
    nanosec: 0
- positions:
  - -2.9441970937399127
  - 0.905600271782078
  - 1.5771963704863685
  velocities: []
  accelerations: []
  effort: []
  time_from_start:
    sec: 10
    nanosec: 0
- positions:
  - -2.761086276477428
  - -0.9932097191700782
  - -1.5803964742570122
  velocities: []
  accelerations: []
  effort: []
  time_from_start:
    sec: 14
    nanosec: 0
- positions:
  - -2.761086276477428
  - -0.9932097191700782
  - -1.5803964742570122
  velocities: []
  accelerations: []
  effort: []
  time_from_start:
    sec: 15
    nanosec: 0
- positions:
  - -2.761086276477428
  - -0.9932097191700782
  - -1.5803964742570122
  velocities: []
  accelerations: []
  effort: []
  time_from_start:
    sec: 17
    nanosec: 0
```



Mass analysis and others

Nombre de archivo	Ca...	Peso total	SW-Masa	Peso total
Runner Plastic Sleeve	1	13.84	13.84	13.84
Umbrella Arm Connecting Ring	2	64.65	32.32	64.65
paloArriba	1	128.12	128.12	128.12
Top Hoist Body	1	213.73	213.73	213.73
ConexionPaloBrazo	1	444.66	444.66	444.66
Hoist Body	1	644.80	644.80	644.80
Umbrella Arm	8	1531.41	191.43	1531.41
base2Sombrilla1.1	1	3113.91	3113.91	3113.91
Umbrella Upper Arm	8	3162.33	395.29	3162.33
base1Sombrilla	1	25685.38	25685.38	25685.38
baseSombrilla	1	34036.64	34036.64	34036.64



Physical model

ALFA CENTAURI

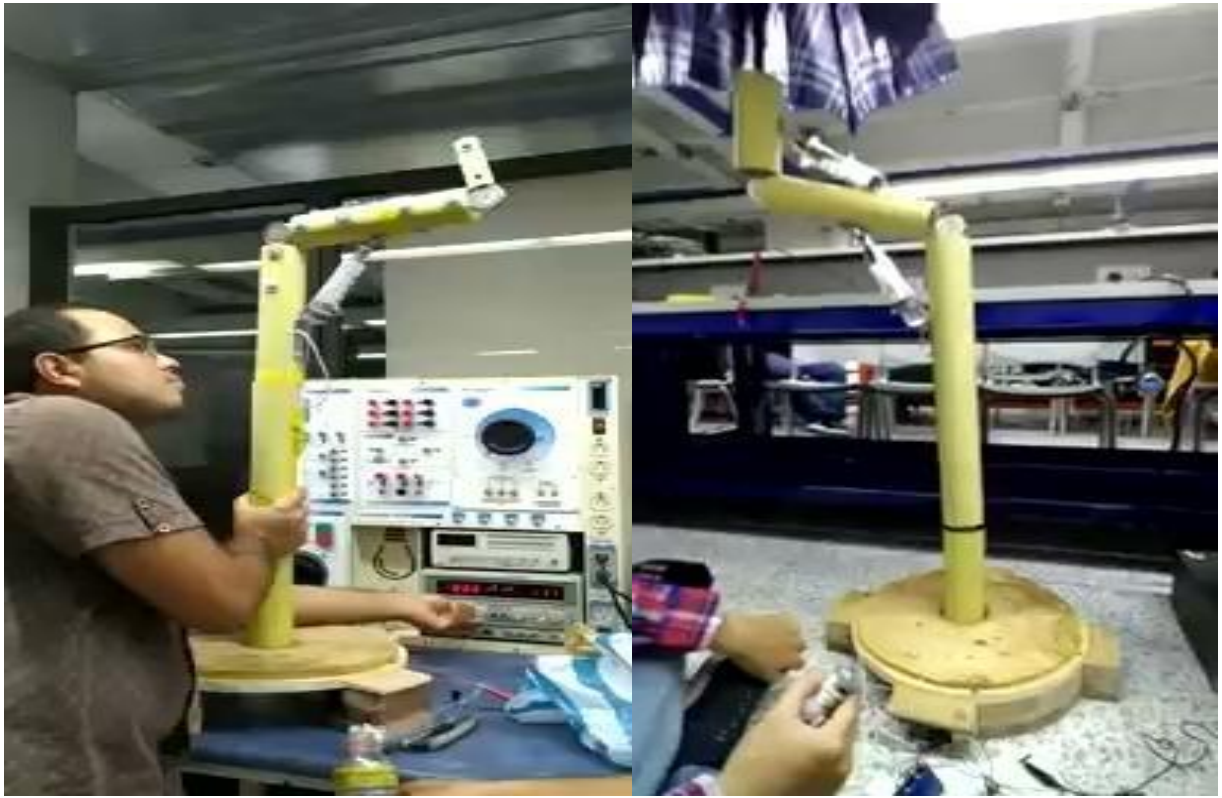
Physical model

ALFA CENTAURE

Physical model

ALFA CENTAURI

Video Physical model



GitHub CODE

ALFA CENTAURE

Relax and join us in this hot adventure