به نام خدا

FUM CARE
CENTER OF ADVANCED
REHABILITATION & ROBOTICS RESEARCH

# گزارش کارآموزی

## پیاده سازی و وریفای مدل ربات شش درجه آزادی FUM-6R در محیط شبیه سازی Gazebo به کمک ROS

---

دانشجو: نرگس رجبیون

تابستان ۱۴۰۱

## فهرست

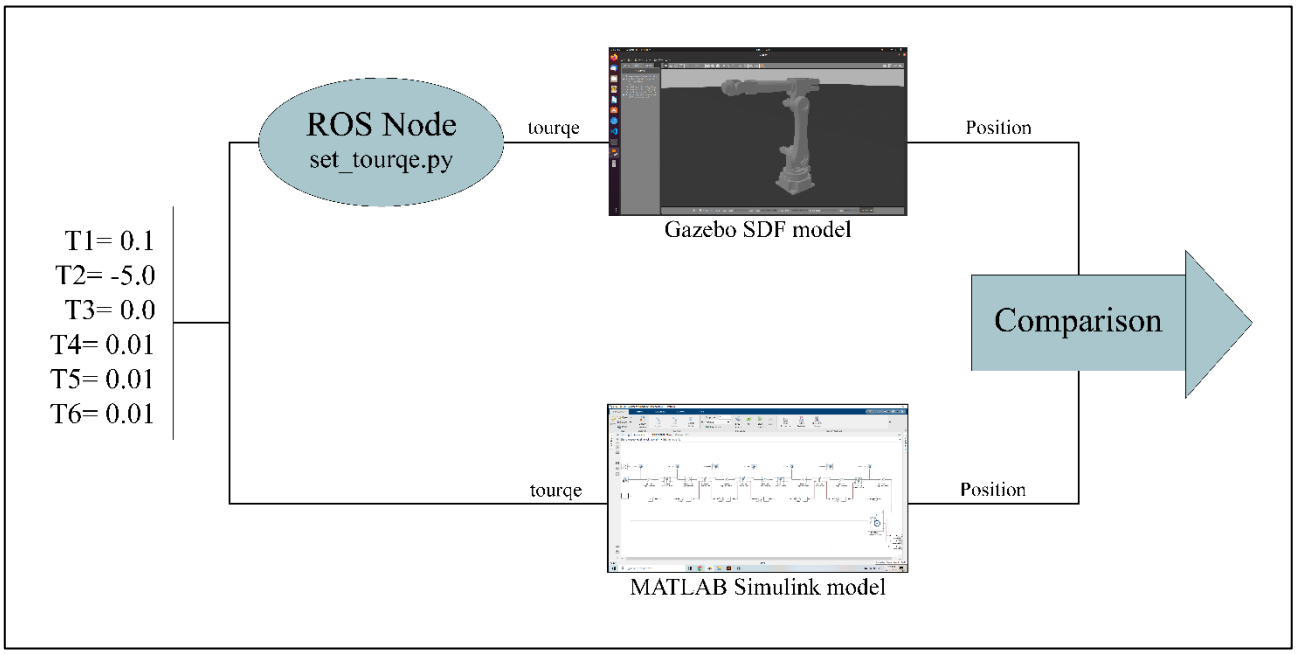
## چکیده

امروزه، ابزار های شبیه سازی نقش پررنگی را در رشد روز افزون تکنولوژی و دانش حوزه رباتیک ایفا می‌کنند. Gazebo یکی از ابزار های قدرتمند در زمینه شبیه سازی می‌باشد. من جمله مزیت های این ابزار امکان برقراری ارتباط آسان با محیط ROS[1] می‌باشد. این دو ابزار در کنار یکدیگر، مدیریت و شبیه سازی ربات ها را برای محققان راحت تر کرده است.

در این گزارش ابتدا به پیاده سازی ربات FUM-6R در Gazebo به کمک پکیج های Gazebo-ROS و فایل URDF[2] توصیف ربات پرداخته شده و سپس روابط سینماتیک و دینامیک ربات با نتایج متلب بررسی صحت سنجی می‌شود.

برای انجام این صحت سنجی مراحل زیر انجام شده است:

۱- اعمال گشتاور ثابت به مدل Gazebo به کمک ROS Node و ذخیره موقعیت مفاصل ربات
۲- اعمال همان گشتاور ها به مدل Simulink و ذخیره موقعیت مفاصل ربات
۳- مقایسه موقعیت های دریافتی از دو مدل برای سنجش یکسان بودن نتایج




شکل ۱: مسیر صحت سنجی ربات FUM-6R

[1] Robot Operating System
[2] Unified Robotics Description Format

# فصل اول: مقدمه ای بر ROS و Gazebo

## ۱-۱) معرفی ROS

در این بخش معرفی مختصری از سیستم عامل ربات یا ROS و نقشه راه یادگیری آن خواهیم داشت. ROS یک نرم افزار متن باز[1] است که ابتدا در سال ۲۰۰۷ در دانشگاه استنفورد توسعه پیدا کرد اما در سال ۲۰۱۴ به طور رسمی معرفی و گسترش یافت و تحولی بزرگ در رباتیک ایجاد کرد و از آن موقع روز به روز در حال گسترش است.


شکل ۲: لوگوی سیستم عامل ربات

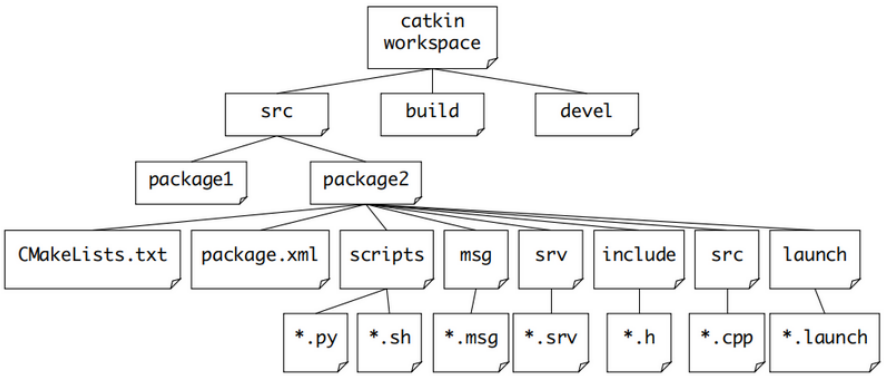
در ابتدا توسعه دهندگان اولیه بدلیل آنکه در ویندوز امکان دسترسی به فایل‌های سیستمی وجود نداشت، این سیستم عامل را بر روی لینوکس توسعه دادند ولی در سال ۲۰۲۰ نسخه ای از این نرم‌افزار ارائه شد که قابلیت اجرا در محیط ویندوز را نیز فراهم می‌کرد. با این حال همچنان توصیه می‌شود از نسخه‌هایی که تحت لینوکس کار می‌کنند استفاده شود چرا که رسیدن نسخه‌ی ویندوز به نسخه‌ای قابل اطمینان و رایج زمان بیشتری نیاز دارد.

برای استفاده از ROS ابتدا باید آن را مطابق آنچه در ادامه ارائه می‌شود بر روی سیستم عامل لینوکس نصب نمایید. سپس فضای کاری‌ای ایجاد نمایید تا بتوانید پکیج‌هایی را به آن اضافه کنید. به طور مختصر می‌توان مجموعه ای از کدها و اسکریپت‌هایی تعریف کرد که برای هدف خاصی در کنار یکدیگر قرار گرفته‌اند.

پکیج‌ها دو نوع هستند، پکیج‌هایی که خود ما می‌توانیم کدنویسی و ایجاد کنیم و پکیج‌هایی که به صورت آماده می‌توان برای اهداف مختلف اضافه کرد. در طی سالیان اخیر پکیج های زیادی در زمنیه‌های مختلف مانند بینایی‌ماشین، هوش‌مصنوعی، راه‌اندازی یک سنسور خاص و ... برای نسخه‌های مختلف ROS نوشته شده است که با جست و جویی ساده می‌توانید متناسب با نیاز خود آنها را تهیه کنید.

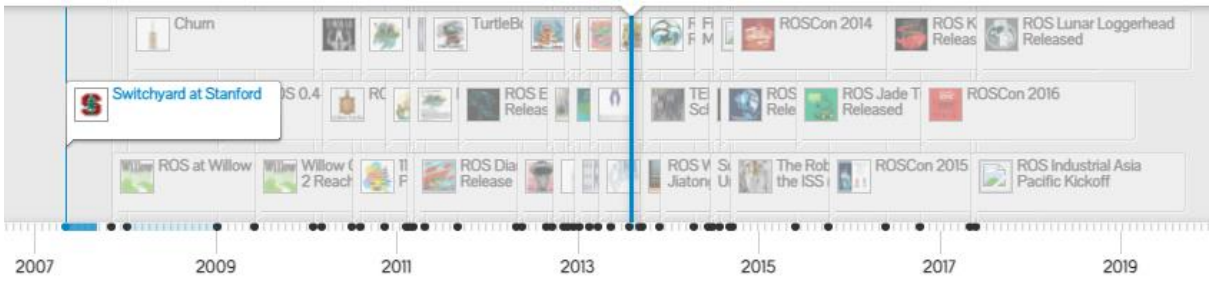
به کدهایی که پکیج‌ها را تشکیل می‌دهند Node گفته می‌شود و همانطور که اشاره شد هر پکیج شامل تعدادی از این Node ها می‌باشد که فارغ از زبان برنامه نویسی‌ای که هر یک از این Node ها نوشته شده است، میتوانیم توسط ROS به روش‌های مختلف بین آنها ارتباط برقرار کنیم و یک مجموعه ی کامل را برای هدفی مشخص ایجاد نماییم.

[1] Open Source




شکل ۳: ساختار فضای کاری در ROS

همانطور که اشاره شد، ROS نسخه‌های مختلفی ارائه کرده است که در این پروژه از ROS2 و نسخه‌ی Foxy استفاده شده است چرا که این نسخه نه آنقدر جدید است که پکیج‌های مورد نیاز برای آن پیدا نشود و هنوز ایرادات آن برطرف نشده باشد و نه آنقدر قدیمی است که پاسخگوی نیاز ما نباشد.




شکل ۴: سیر زمانی ROS و توزیع های مختلف آن

## ۱-۲) معرفی Gazebo

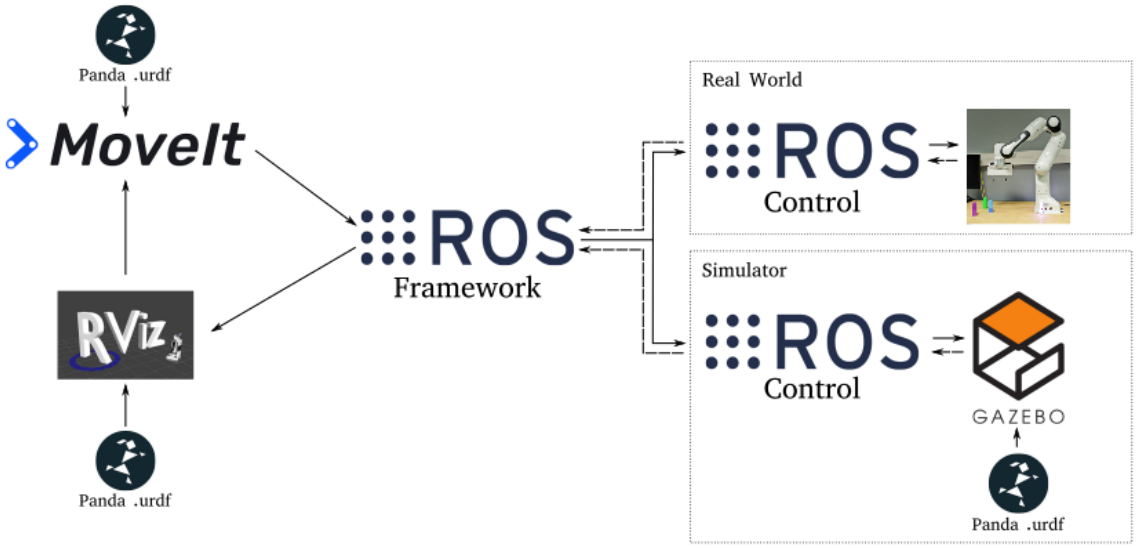
این نرم‌افزار یک شبیه‌ساز متن باز است که امکان شبیه‌سازی محیط های بسیار پیچیده را با گرافیک بالا و بسیار نزدیک به واقعیت فراهم می‌آورد. یک شبیه‌ساز خوب، امکان آزمایش سریع الگوریتم‌ها، طراحی ربات‌ها و آموزش هوش‌مصنوعی را با استفاده از سناریوهای واقع‌گرایانه فراهم می‌آورد. Gazebo با دراختیار داشتن این قابلیت‌ها، امکان شبیه‌سازی دقیق و کارآمد انواع ربات‌ها در محیط‌های پیچیده داخلی و خارجی را فراهم می‌کند.


شکل ۵: لوگوی شبیه ساز Gazebo

شبیه ساز گزبو Gazebo امکانات متنوعی برای رفتارهای فیزیکی گوناگون دارد. گزبو از دو بخش تشکیل شده است: یک سرور[1] (که با عنوان gzserver اجرا می‌شود) و یک کلاینت[2] (که با عنوان gzclient اجرا می‌شود). gzserver در واقع هسته مرکزی گزبو بوده و محاسبات فیزیکی و تولید داده‌های سنسورها را برعهده دارد. این قسمت در مواردی که نیاز به رابط کاربری نیست می‌تواند به صورت مستقل اجرا شود. عملکرد بخش کلاینت نیز به‌صورت ارائه یک نمایش گرافیکی رابط کاربری است که می‌تواند تجسم خوبی از شبیه‌سازی و کنترل راحت بر روی خواص مختلف شبیه سازی را فراهم کند.

یک قابلیت مؤثر و باارزش شبیه ساز Gazebo آن است که می‌تواند سنسورهایی نظیر نیرو، شتاب سنج، سونار، LIDAR، دوربین‌های رنگی و سنسورهای ابر نقاط را شبیه‌سازی کند. در نتیجه این امکان را فراهم می‌آورد که به راحتی سنسورهای مورد نیاز را به مدل اضافه کرد.

یکی از مهم‌ترین قابلیت‌های این شبیه‌ساز قابلیت آن در ارتباط با سایر نرم‌افزارها برای کنترل شبیه‌سازی است. از جمله نرم‌افزارهایی که ارتباط بسیار خوبی با این شبیه‌ساز دارد ROS است چرا که هر دوی این ترم‌افزارها خروجی شرکت OpenRobotics هستند. کما اینکه هر دوی این نرم‌افزارها مستقلا نیز قابل استفاده هستند. از مهم‌ترین مزایای استفاده از این مجموعه نرم‌افزاری آن است که ROS در این سیستم نقش مدیریت کدها و پکیج‌ها را برای اعمال در شبیه‌سازی ایفا می‌کند. حال آنکه همین نقش با همان ساختار قابلیت تعامل با محیط واقعی را نیز خواهد داشت و این به معنی جلوگیری از تکرار کارهای انجام شده در شبیه‌سازی برای پیاده‌سازی واقعی است.




شکل ۶: ساختار شبیه سازی و قابلیت جایگزینی آن با محیط واقعی

به صورت گسترده تر این مجموعه نرم‌افزاری را می‌توان مطابق تصویر ۷ در نظر گرفت که هر یک قابلیت‌های به خصوصی را در اختیار قرار می‌دهند و با یکدیگر در ارتباط هستند.

[1] Server
[2] Client


شکل ۷: مجموعه نرم افزاری شبیه سازی ROS-Gazebo

## ۱-۳) آشنایی با فایل های توصیفی ربات

برای تعریف یک مدل در محیط های شبیه سازی نظیر Gazebo نیازمند نگارش یک فایل برای توصیف ویژگی های آن مدل هستیم. این فایل توصیفی می‌تواند فرمت های متفاوتی داشته باشد. فرمت sdf و world فرمت هایی هستند که Gazebo به صورت مستقیم آن ها را می‌شناسد. فرمت world اغلب برای توصیف یک فضای متشکل از چند مدل مختلف استفاده می‌شود. فرمت دیگری نیز موجود است. فرمت URDF که این فرمت به صورت مستقیم برای Gazebo قابل شناسایی نیست اما به راحتی قابل تبدیل به فرمت sdf می‌باشد.

نگارش فایل توصیفی به فرمت sdf و URDF شباهت های بسیاری بهم دارند. در ادامه به بررسی فایل توصیفی با فرمت URDF خواهیم پرداخت.[۱] این فایل به زبان XML نوشته می‌شود و شامل تگ های مختلفی می‌باشد.

**تگ <robot>**: کلیه اطلاعات مربوط به مدل ربات در این تگ آورده می‌شود.

**تگ <link>** : در زیر مجموعه این تگ تمامی مشخصات مربوط به لینک های ربات آورده می‌شود. تگ های اصلی زیر مجموعه این تگ، تگ <inertial> ، تگ <visual> و تگ <collision> می‌باشند. که حاوی مشخصات مختلفی از لینک ها می‌باشند. لازم است تمام لینک های ربات به همین صورت معرفی شوند.

**تگ <joint>** : در این تگ معرفی مفاصل و اتصالات بین لینک ها معرفی می‌شوند. تگ های <parent>، <child> و <axis> زیر مجموعه این تگ هستند.

**تگ <gazebo>** : این تگ معرف مشخصات مربوط به محیط شبیه سازی می‌باشد. تگ <plugin> از جمله تگ های زیر مجموعه آن هستند.

[۱] معرفی فایل های توصیفی به تفسیر در ویدیو آموزش ROS و Gazebo جلسه سوم (Introduction to ROS and Gazebo- Part C) آورده شده است.

## ۱-۴) پیش نیاز ها

برای پیاده سازی ربات در محیط Gazebo مطابق این گزارش، در گام صفرم لازم است موارد زیر انجام شود.

۱- در این پروژه از نسخه ROS2 foxy استفاده شده است که این نسخه از ROS با ورژن ubuntu 20.04 مطابقت دارد. برای نصب ubuntu میتوان یکی از دو راه زیر را انتخاب کرد:

۱-۱) نصب ماشین مجازی VMware در ویندوز و نصب ubuntu روی آن؛ روش نصب و راه اندازی ubuntu در VMware در ویدیو آموزش ROS و Gazebo، جلسه اول[۱] آورده شده است.

لوگو نرم افزار VMware

لوگوی ubuntu

۱-۲) نصب ubuntu روی سیستم به صورت dual-Boot که به معنای نصب دو سیستم عامل در یک دستگاه می‌باشد؛ روش نصب ubuntu به صورت dual-Boot در لینک ارجاع شده[۲] آورده شده است.

۲- نصب ROS2 foxy ؛ روش نصب ROS در سایت رسمی ROS آورده شده است و کافی است تمامی مراحل را مطابق آن طی کرد. همچنین نصب ROS2 foxy در ویدیو آموزش ROS و Gazebo، جلسه اول آورده شده است.[۳] توجه شود که در نصب ROS از روش "Install ROS2 via Debian Packages" استفاده شود.

۳- اضافه کردن یک فضای کاری[۴] و پکیج؛ روش اضافه کردن یک فضای کاری و پکیج در ویدیو آموزشی ROS و Gazebo، جلسه اول آورده شده است.

۴- نصب Gazebo؛ برای نصب Gazebo دستورات زیر را به ترتیب اجرا کنید.

```
$ sudo apt update
$ sudo apt upgrade
$ sudo apt install gazebo9
```

۵- نصب پکیج ROS-Gazebo؛ برای نصب این پکیج که برای ایجاد ارتباط بین ROS و gazebo اهمیت دارد دستور زیر را اجرا کنید.

```
$ sudo apt install ros-foxy-gazebo-ros-pkgs
```

۶- نصب افزونه Solid Works URDF exporter؛ روش نصب این افزونه در لینک اراجاع شده[۵] آورده شده است. این افزونه باید روی نرم افزار Solid Works نصب شود.

[1] Introduction to ROS Video

۲ لینک آموزش نصب ubuntu به صورت dual-Boot : https://youtu.be/-iSAyiicyQY

[۳] لینک documentation مربوط به ROS2 foxy : https://docs.ros.org/en/foxy/Tutorials.html

[4] Workspace

[۵] لینک دانلود افزونه URDF exporter و راهنمای نصب آن : http://wiki.ros.org/sw_urdf_exporter

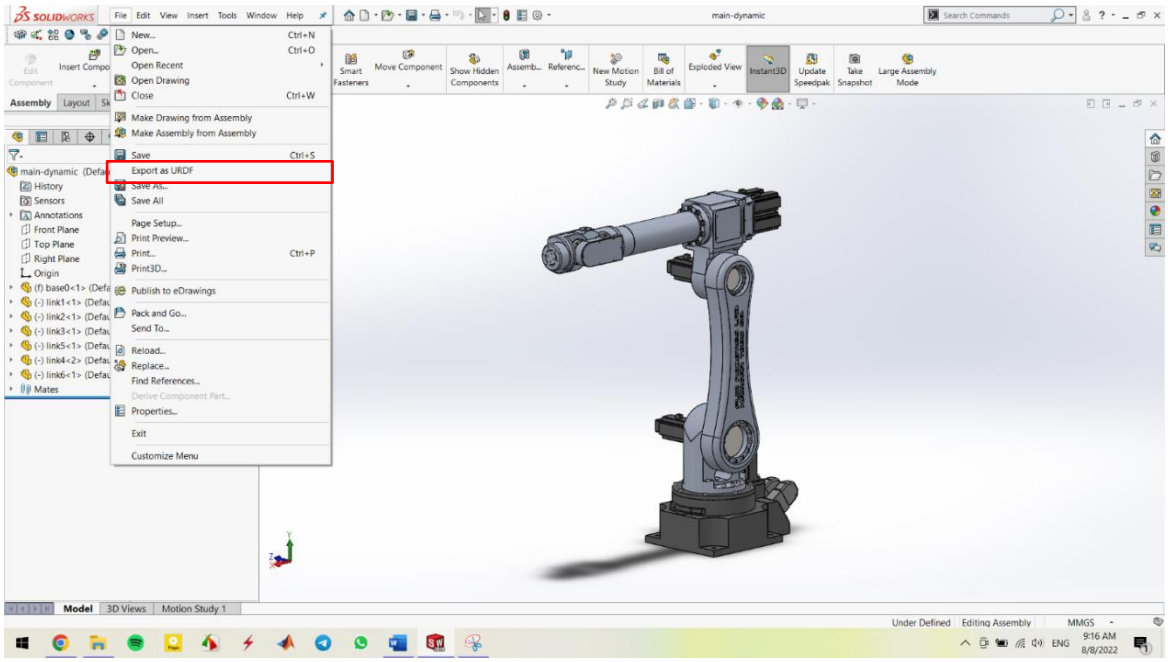
# فصل دوم – استخراج فایل توصیفی ربات از نرم افزار SolidWorks

## ۲-۱) استخراج فایل URDF ربات از نرم افزار SolidWorks به کمک افزونه URDF Exporter

همانطور که در بخش ۱-۳ اشاره شد، به منظور پیاده سازی ربات در محیط شبیه سازی Gazebo، لازم است فایلی توصیفی از ویژگی های مدل ربات به زبان xml، شامل شکل ظاهری، ویژگی های دینامیکی و غیره، تهیه شود.

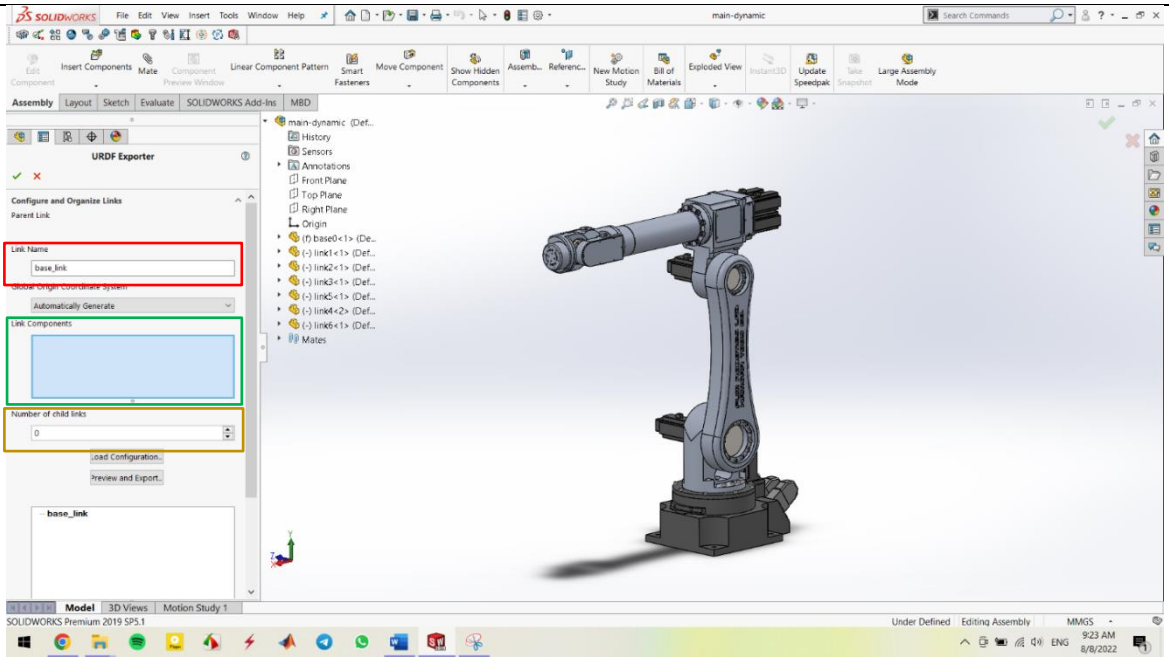
به جهت ساخت فایل URDF ربات های پیچیده ای مانند ربات 6R می‌توان از یک افزونه در نرم افزار SolidWorks به نام SolidWorks URDF exporter استفاده کرد. در ابتدا نیاز به ساده سازی مدل ساختی ربات می‌باشد؛ که به معنای حذف قطعات کوچک مانند پیچ ها و غیره و سپس یک پارچه سازی تمامی قطعات هر یک از لینک ها می‌باشد. در نهایت پس از ساده سازی در محیط اسمبلی SolidWorks شش Part از شش لینک ربات خواهیم داشت. حال با استفاده از افزونه SolidWorks URDF exporter لینک های ربات را یکی یکی انتخاب شده و در نهایت خروجی URDF مورد نظر دریافت می‌شود. در ادامه روش گرفتن این خروجی آورده شده است:

**گام اول)** از زیر مجموعه منوی File گزینه ی Export as URDF را انتخاب می‌کنیم



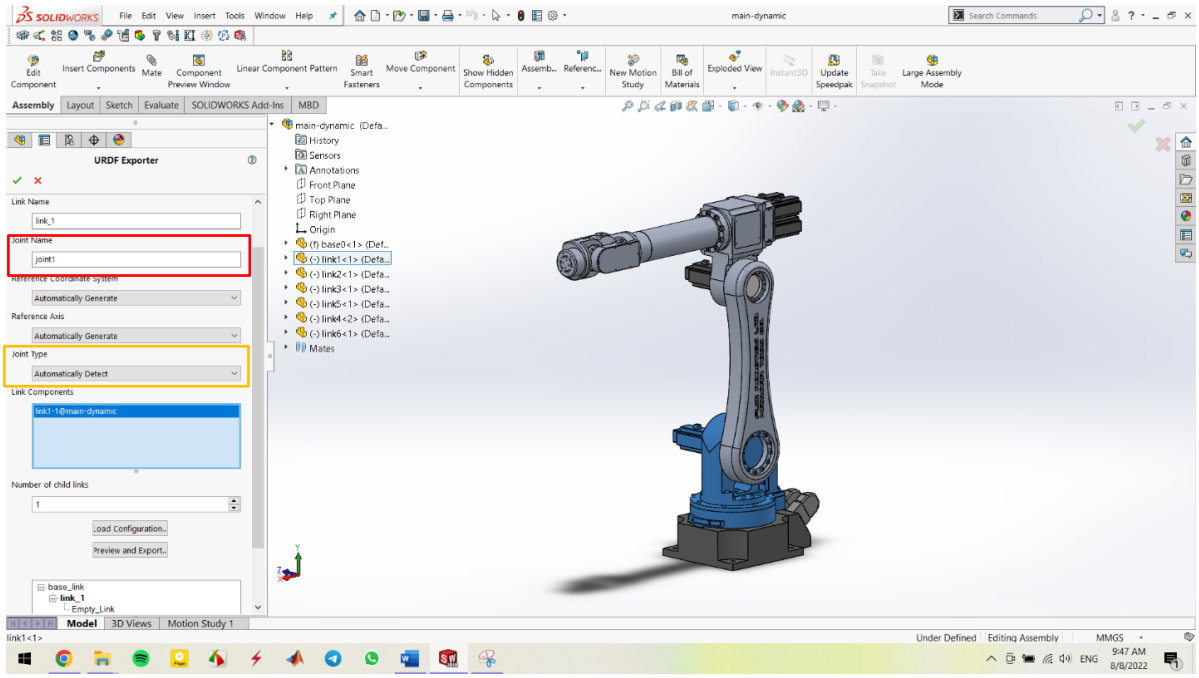

شکل ۸: گام اول – باز کردن پنجره دریافت خروجی URDF در سالیدورکس

**گام دوم)** با کلیک بر روی گزینه ی Export as URDF پنجره ی مربط به آن زیر باز می‌شود. در قسمت Link Name اسم اولین لینک مورد نظر را وارد می‌کنیم.(اولین لینک انتخابی همان لینک base ربات است) دقت شود که برای هر یک از لینک ها و مفصل ها باید یک اسم منحصر به فرد انتخاب شود. در قسمت Link Components از شاخه درختی پارت مربوط به لینک نام گذاری شده انتخاب می‌شود. در قسمت Number of child links تعداد لینک های متصل به لینک انتخاب شده را مشخص می‌شود که در این ربات سری هر یک از لینک ها به یک لینک بعد از خود متصل هستند.



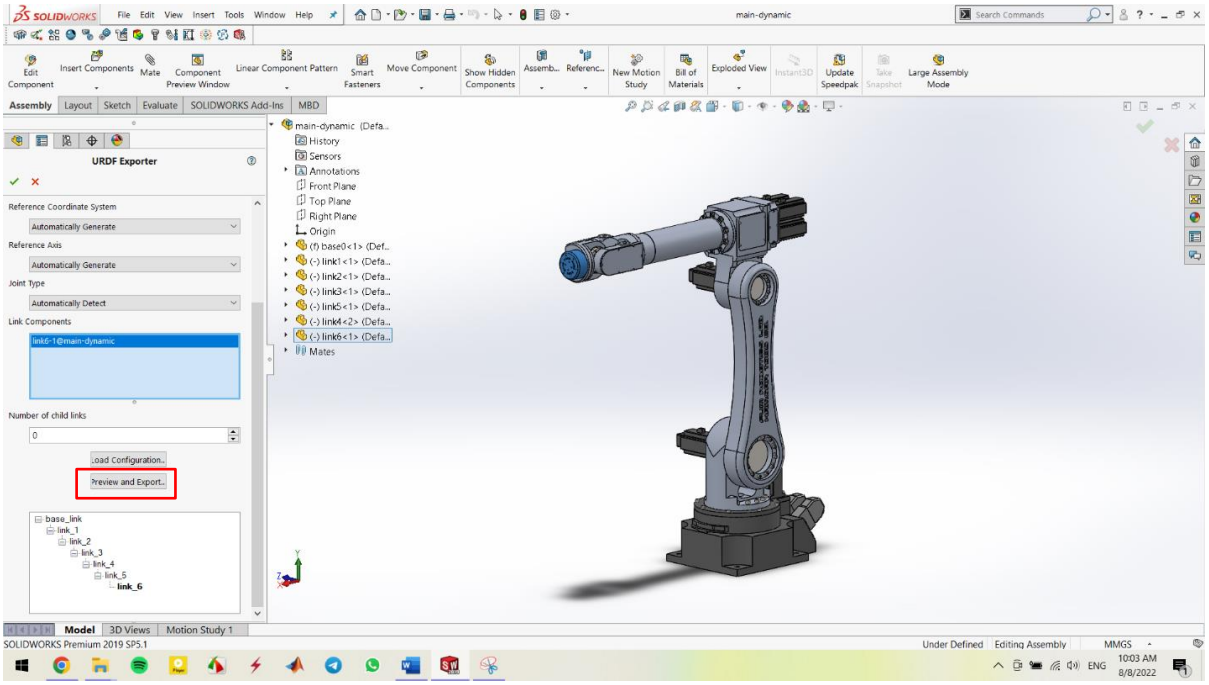

شکل ۹: گام دوم – تعریف لینک پایه

**گام سوم)** پس از انتخاب اولین لینک به عنوان base_link در شاخه درختی زیرمجموعه‌ی پنجره‌ی URDF یک زیرشاخه برای base_link ایجاد می‌شود. با کلیک بر روی Empty_Link پنجره ی جدیدی باز می‌شود که در این قسمت مانند لینک قبلی نام لینک، قطعه مربوط به لینک را انتخاب می‌شود. در قسمت Joint Name اسم مفصل بین لینک انتخاب شده و لینک قبلی را وارد می‌کنیم و همچنین در قسمت Joint Type نوع مفصل نیز انتخاب می‌شود. مجدد همین روند طی شده تا در نهایت به لینک آخر برسیم.

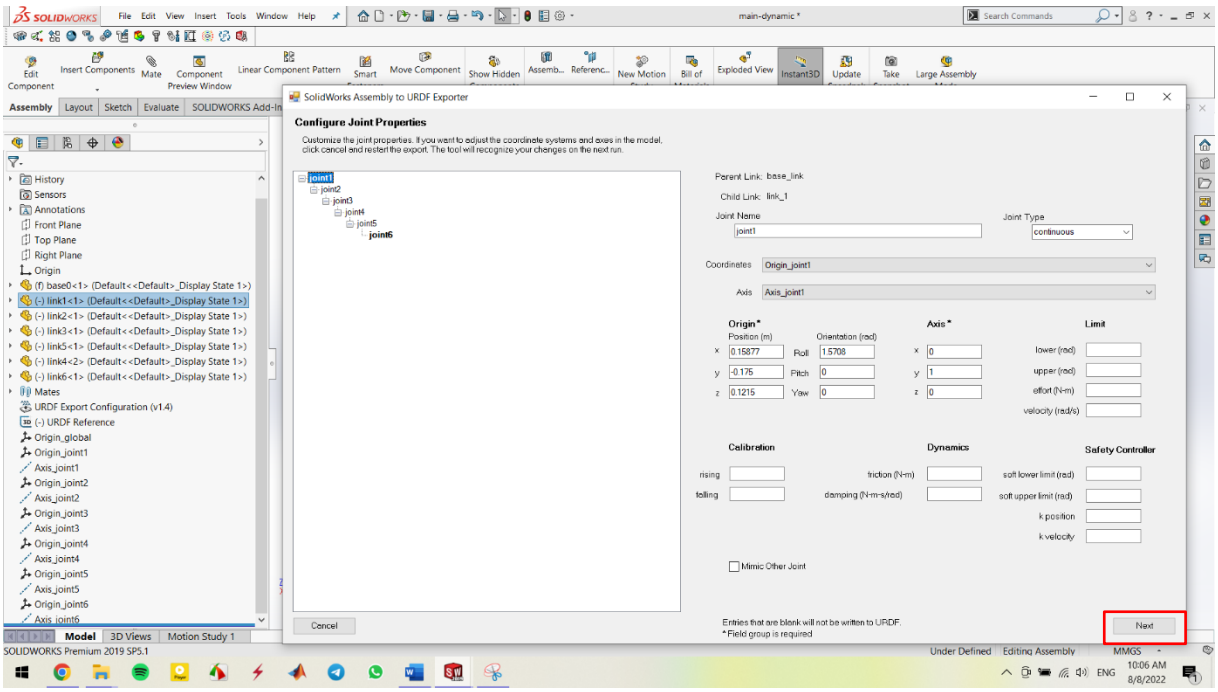



شکل ۱۰: گام سوم – تعریف لینک اول تا ششم ربات

**گام چهارم)** پس از به اتمام رسیدن تعریف تمامی لینک ها و مفصل ها بر روی گزینه preview and Export کلیک می‌کنیم. (شکل ۱۱) سپس پنجره‌ی شکل ۱۲ که شامل تمامی اطلاعات مربوط به مفصل ها است باز می‌شود که با کلیک بر روی Next می‌توان اطلاعات مربوط به لینک ها را نیز مشاهده کرد و در نهایت با انتخاب گزینه Export URDF and Meshes فایل توصیفی ربات در مسیر مطلوب ذخیره می‌شود.




شکل ۱۱: گام چهارم – پیش نمایش خروجی URDF در سالیدورکس

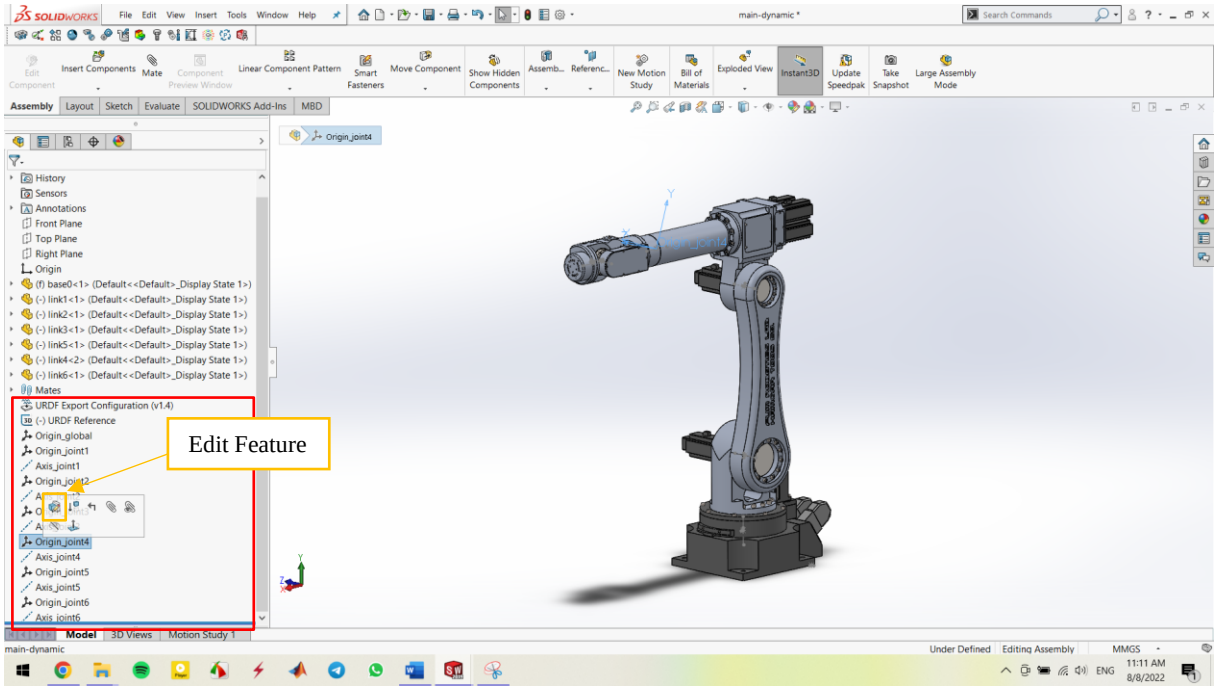



شکل ۱۲: گام چهارم – نمایش اطلاعات مربوط به مفاصل و لینک های ربات

### ۲-۲) اصلاح فایل URDF ربات

در بسیاری از موارد دستگاه های انتخاب شده توسط خود نرم افزار SolidWorks برای دریافت خروجی URDF دستگاه های مطلوب ما نیستند در این موارد می‌توان این دستگاه ها مطابق دستگاه گذاری مطلوب اصلاح کرد.

برای اصلاح دستگاه گذاری ها در مرحله شکل ۱۲ صفحه باز شده را بدون cancel کردن فقط و فقط minimize می‌کنیم. در شاخه درختی SolidWorks می‌توان دستگاه های مشخص شده را در زیر مجموعه URDF Reference مشاهده کرد (شکل ۱۳) با راست کلیک روی هر یک از محور ها و انتخاب گزینه Edit Feature می‌توان آن دستگاه را اصلاح کرد.




شکل ۱۳: اصلاح دستگاه های مختصات

در نهایت پس از اصلاح تمامی محور ها صفحه شکل ۱۲ را که minimize کرده بودیم مجدد باز می‌کنیم و گزینه cancel را انتخاب کرده و ذخیره‌ی تغییرات را تایید می‌کنیم. پس از آن دوباره مطابق شکل ۸ گزینه‌ی Export as URDF را انتخاب می‌کنیم؛ پنجره‌ی شکل ۱۲ مجدد با توجه به دستگاه گذاری جدید باز می‌شود این بار با انتخاب گزینه Export URDF and Meshes فایل توصیفی ربات در مسیر مطلوب در یک پوشه ذخیره می‌شود. در داخل پوشه ذخیره شده، یک پوشه با نام Meshes موجود است که فایل Mesh های مربوط به هر لینک آورده شده است. فایل توصیفی ربات یا همان فایل urdf. نیز داخل پوشه URDF آورده شده است.

با وجود اینکه فایل URDF به صورت مستقیم از SolidWorks خروجی گرفته شده است بازهم نیاز به بررسی دقیق تر این فایل و اصلاح آن وجود دارد.

برای استفاده از فایل URDF در محیط ROS نیاز است که ابتدا یک فضای کاری ایجاد کنیم و یک پکیج جدید در این فضای کاری آماده کنیم. در پکیج پوشه جدیدی به نام description ایجاد می‌کنیم و فایل URDF ربات با پسوند urdf. که در بالا به آن اشاره شد در آن پوشه قرار می‌دهیم. یک پوشه دیگر هم به نام launch تشکیل می‌دهیم؛ همچین پوشه meshes که در خروجی SolidWorks داشتیم را داخل این پکیج قرار می‌دهیم.

پس از ایجاد این پکیج که در اینجا به نام urdf_example اسـت. از داخل پوشـه description فایل URDF را باز می‌کنیم و اصـلاحات را بر روی آن انجام می‌دهیم. بخشی از این فایل در ادامه آورده شده است.[1]

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This URDF was automatically created by SolidWorks to URDF Exporter! Originally
created by Stephen Brawner (brawner@gmail.com)
     Commit Version: 1.6.0-1-g15f4949  Build Version: 1.6.7594.29634
     For more information, please see http://wiki.ros.org/sw_urdf_exporter -->
<robot name="FUMTI_1401FT_GAZEBOURDF.SLDASM">
  <link name="6RL0">
    <inertial>
      <origin xyz="0.003 0 -0.005" rpy="0 0 0" />
      <mass value="31.781" />
      <inertia
        ixx="0.376"
        ixy="0"
        ixz="-0.001"
        iyy="0.364"
        iyz="0"
        izz="0.65" />
    </inertial>
    <visual>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL0.STL" />
      </geometry>
      <material name="">
        <color rgba="0.89804 0.91765 0.92941 1" />
      </material>
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL0.STL" />
      </geometry>
    </collision>
  </link>
<joint name="6RJ1" type="revolute">
    <origin xyz="0 0 0.0635" rpy="0 0 0" />
    <parent link="6RL0" />
    <child link="6RL1" />
    <axis xyz="0 0 1" />
    <limit
      lower="0"
      upper="0"
      effort="0"
      velocity="0" />
  </joint>
```

۱

۴

۳

[1] در پیوست۱ به صورت کامل آورده شده است.

تغییرات لازم در فایل URDF ربات به شرح زیر است:

۱- به دلیل طولانی بودن و عدم انجام عملیات ریاضی فایل های urdf. ، می‌توان از فایل هایی با پسوند **urdf.xacro**. استفاده کرد که امکانات بیشتری را در اختیار ما قرار می‌دهند. از این رو با تغییر بخش ۱ در کد بالا به صورت زیر و همچنین تغییر اسم فایل ذخیره شده و اضافه کردن پسوند xacro به آن، این فایل را به فرمت xacro دربیاوریم.

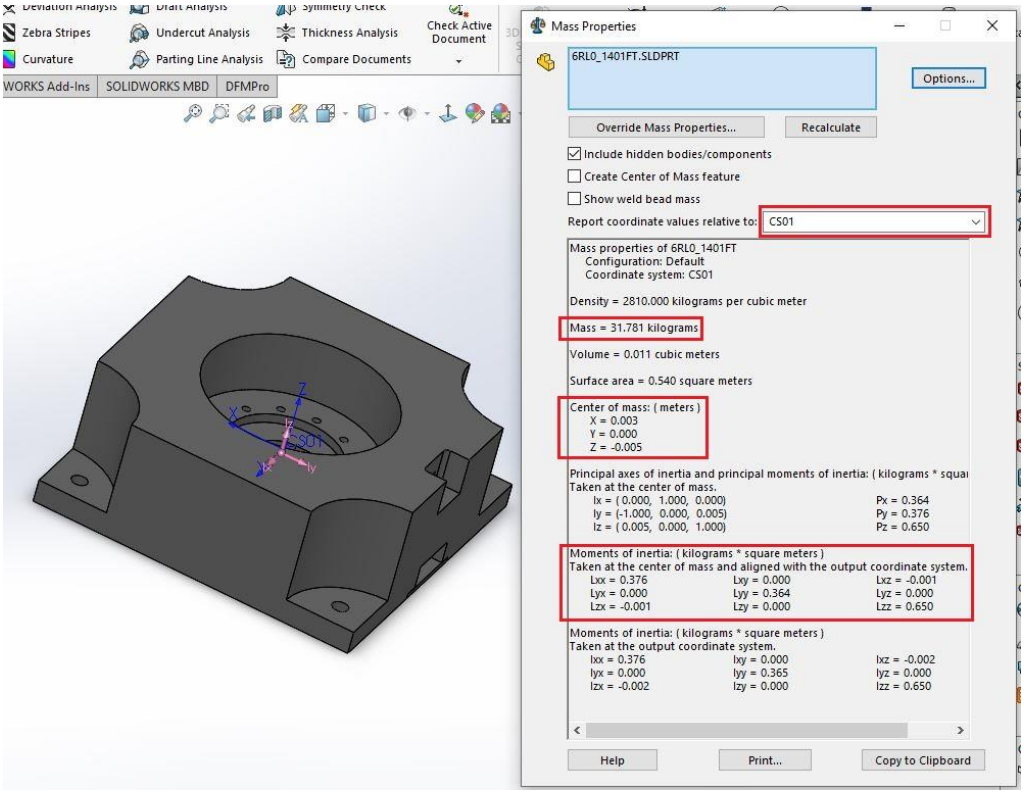
```
<robot xmlns:xacro="http://www.ros.org/wiki/xacro"
 name=" FUMTI_1401FT_GAZEBOURDF.SLDASM">
```

۲- همچنین برای قرار گرفتن پایه ربات روی زمین نیاز است که یک لینک بر روی world و یک مفصل بین این لینک و پایه ربات از نوع fixed اضافه شود.

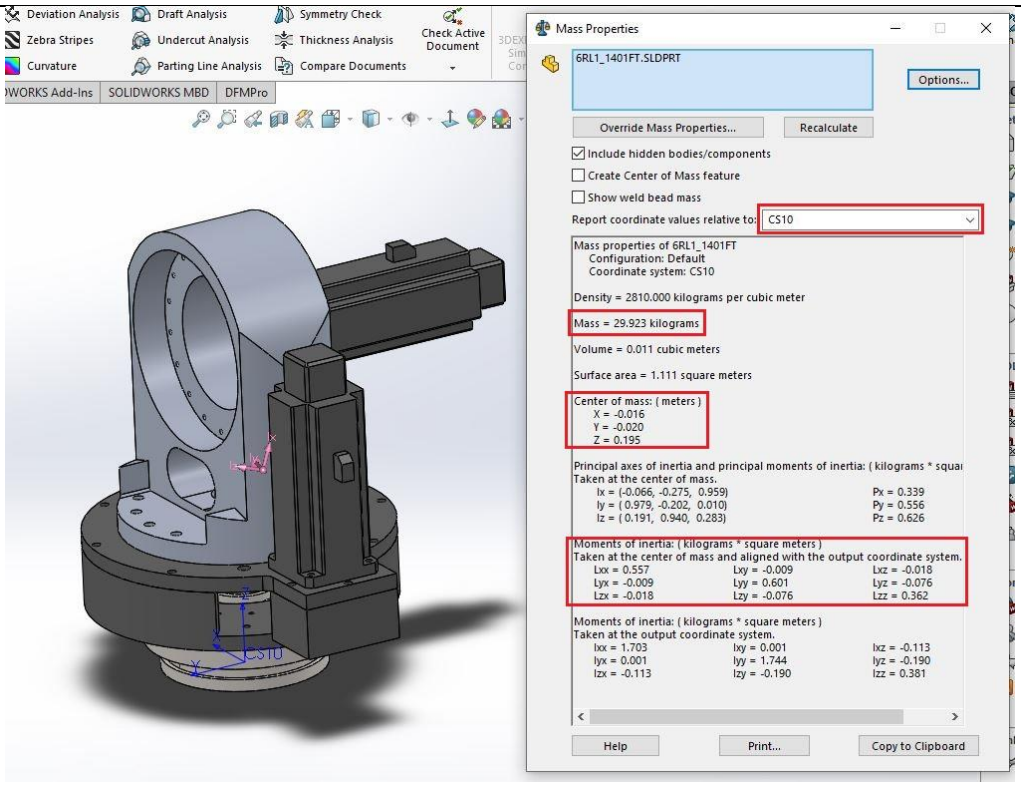
```
<link name="world">
    <origin xyz="0 0 0"rpy="0 0 0" />
  </link>
  <joint name="base-joint" type="fixed">
        <origin xyz="0 0 0" rpy="0 0 0"/>
        <parent link="world"/>
        <child link="6RL0"/>
  </joint>
```

۳- در تگ <mesh> باید آدرس mesh های هر لینک در سیستم خودمان را وارد کنیم. این کار باید برای تمامی لینک ها انجام شود. همانطور که قبلا نیز اشاره شد، فایل Mesh های ربات داخل پوشه Meshes در پکیج آورده شده اند.

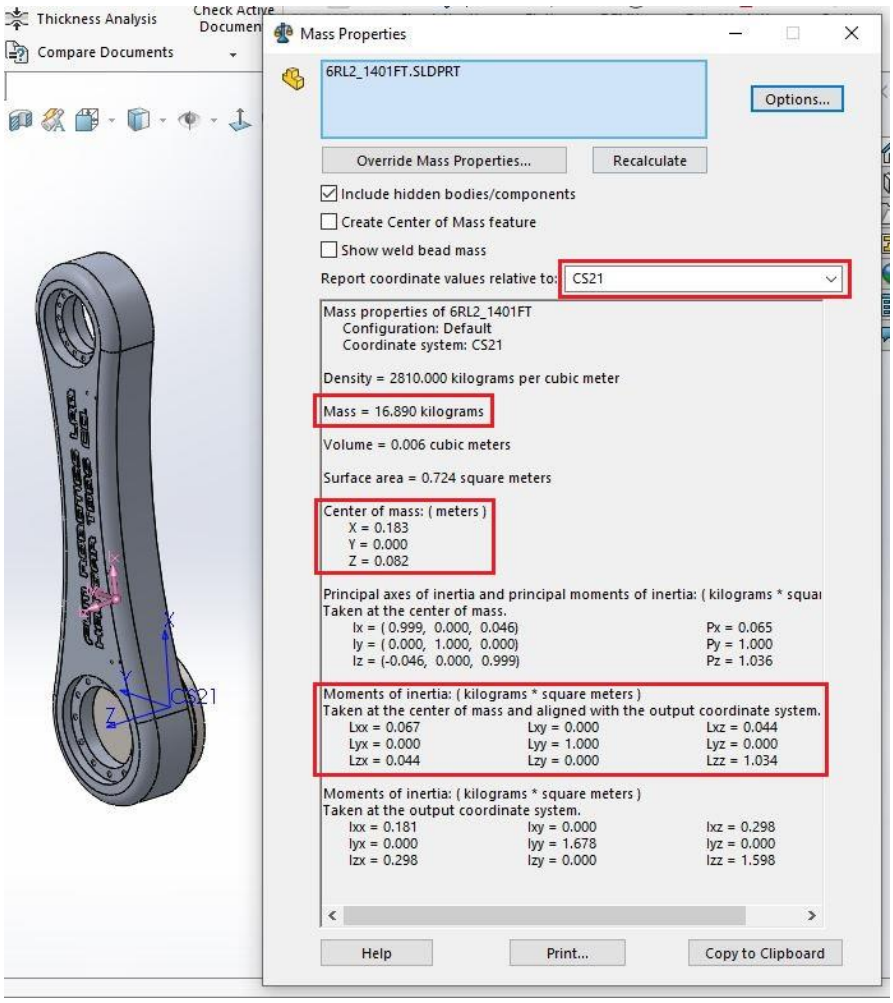
۴- مقادیر اینرسی و جرم لینک های ربات باید چک و اصلاح شود. مقادیر اینرسی لینک ها در تگ <inertia> و مقدار جرم لینک ها در تک <mass> ذخیره می‌شود. مقادیر صحیح اینرسی و جرم لینک ها به شرح زیر است:



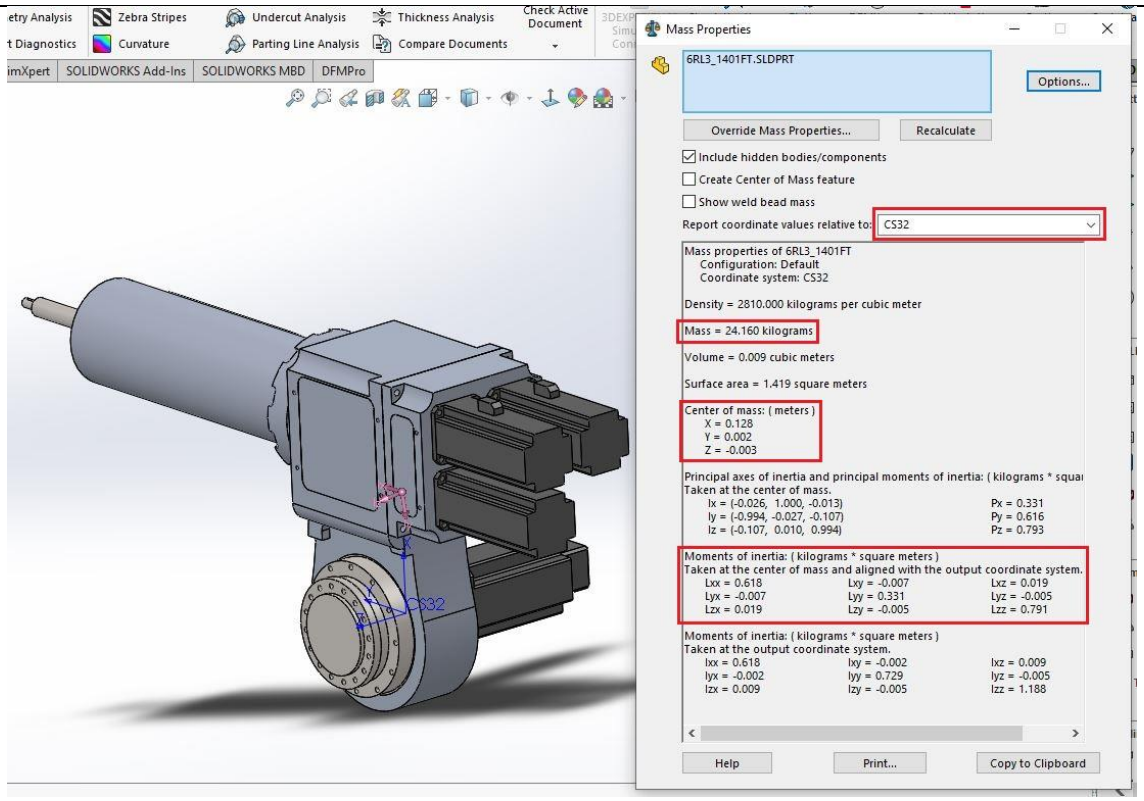

تصویر ۱: مشخصات دینامیکی لینک پایه ربات FUM-6R



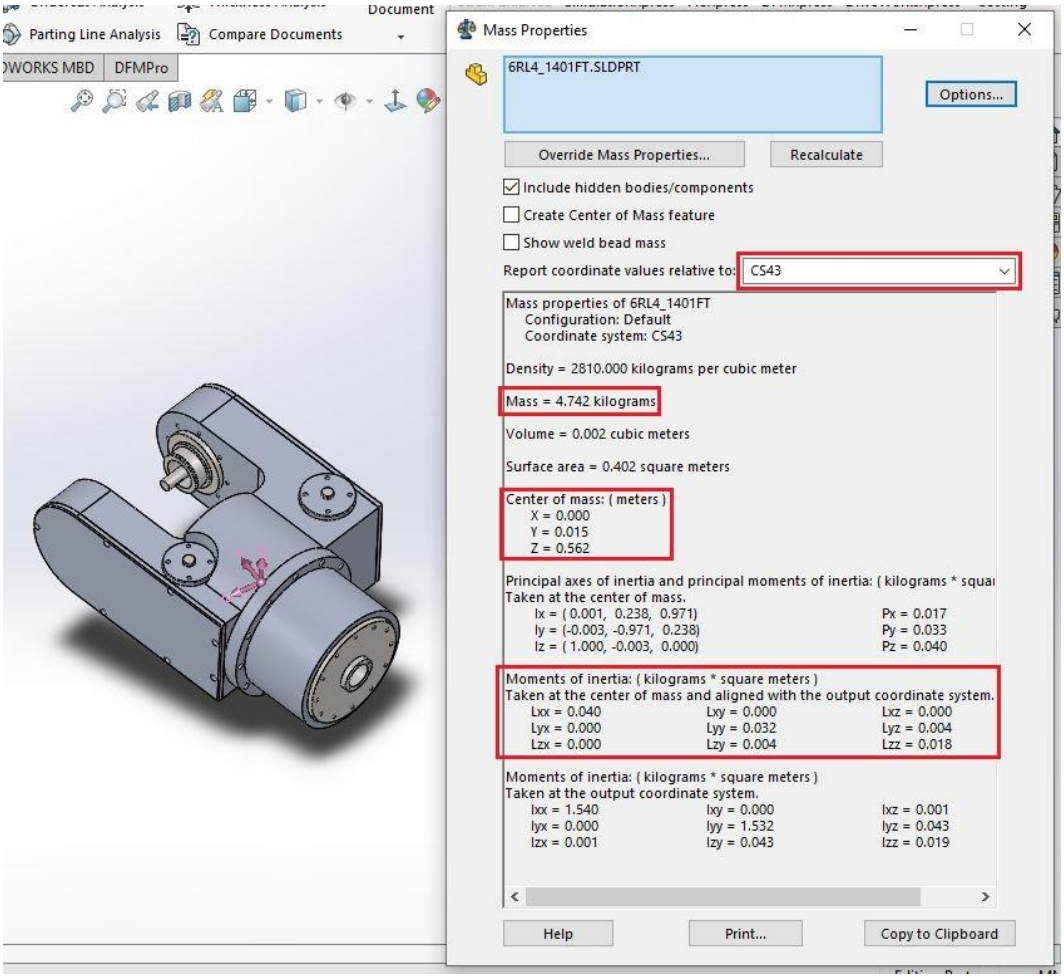

تصویر ۲: مشخصات دینامیکی لینک اول ربات FUM-6R



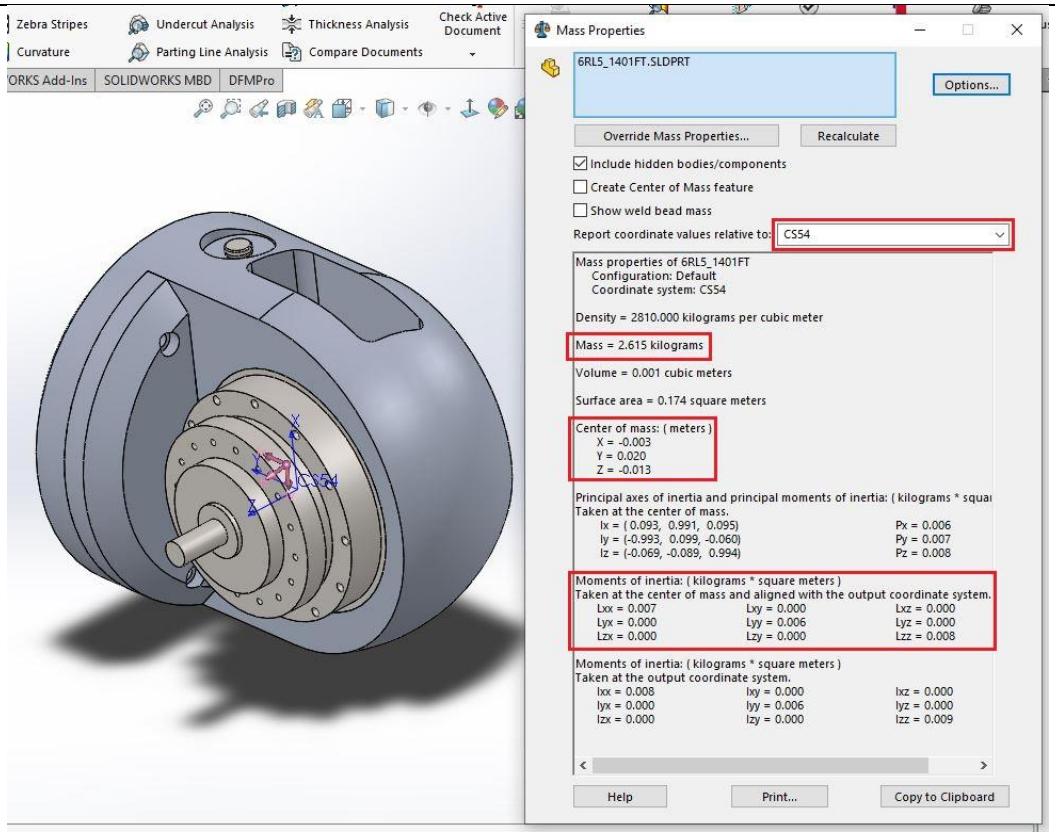

تصویر ۳: مشخصات دینامیکی لینک دوم ربات FUM-6R



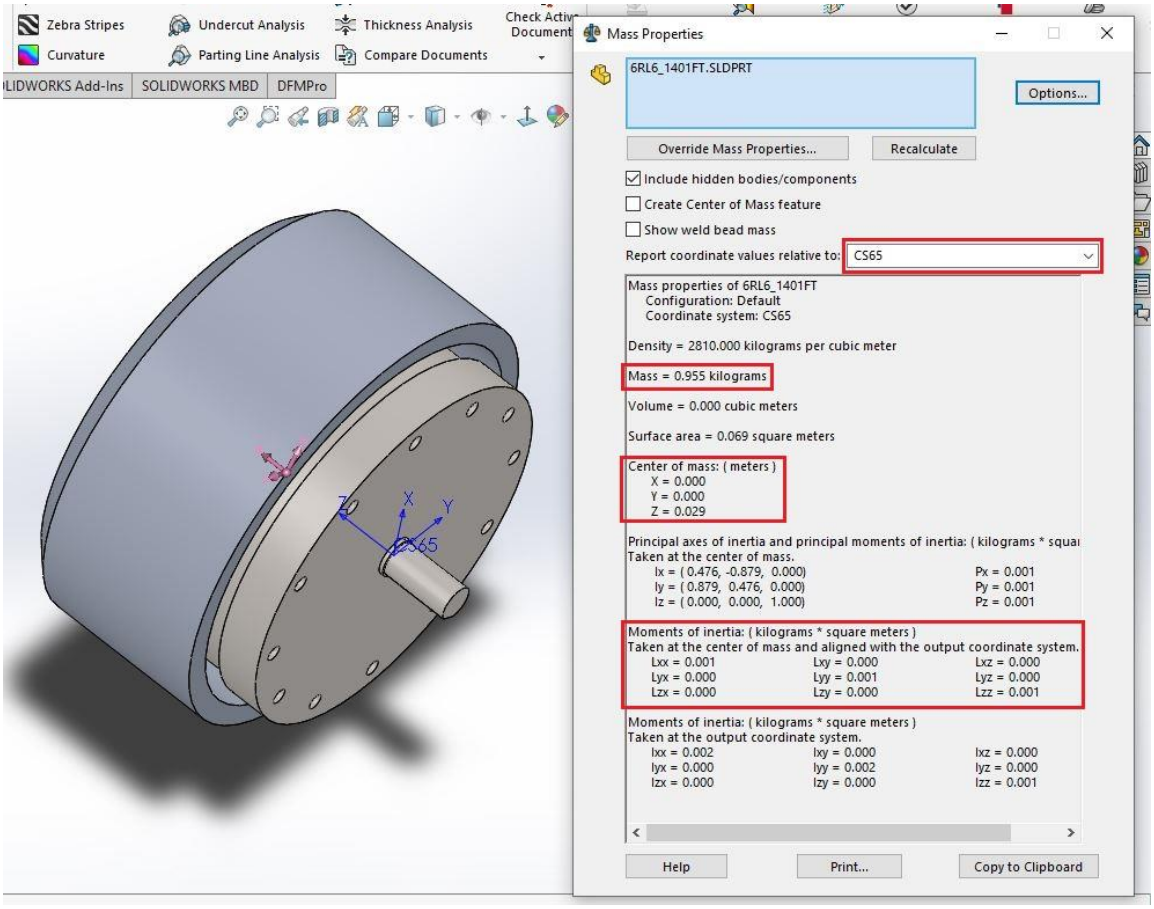

تصویر ۴: مشخصات دینامیکی لینک سوم ربات FUM-6R




تصویر ۵: مشخصات دینامیکی لینک چهارم ربات FUM-6R




تصویر ۶: مشخصات دینامیکی لینک پنجم ربات FUM-6R




تصویر ۷: مشخصات دینامیکی لینک ششم ربات FUM-6R

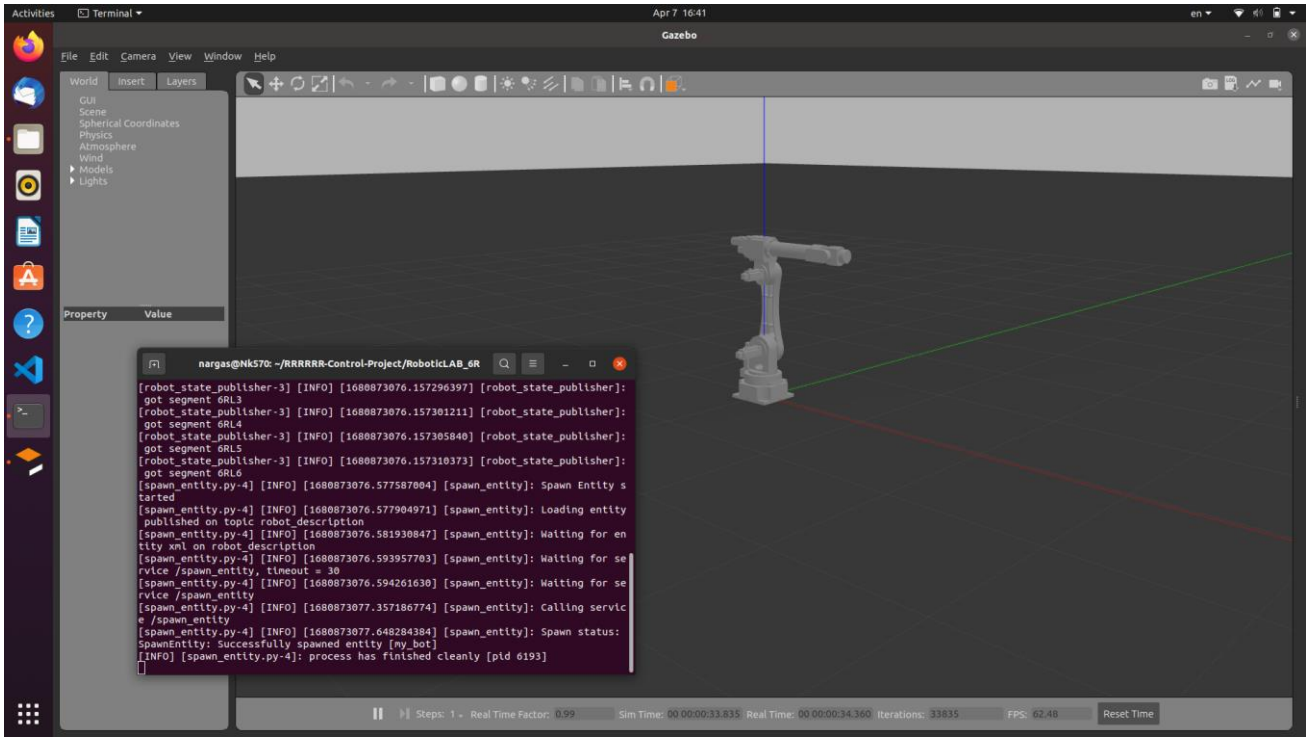
# فصل سوم – باز کردن فایل توصیفی ربات در Gazebo

## ۳-۱) باز کردن فایل URDF در Gazebo و گرفتن خروجی sdf

به جهت باز کردن URDF ربات در Gazebo یک برنامه راه اندازی[1] به زبان پایتون نوشـــته می‌شـــود.[2] این برنامه به فرمت launch.py. بوده و محل ذخیره سازی آن داخل یک پوشه به نام launch در پکیج ساخته شده می‌باشد. برای اجرای این برنامه از دستور زیر استفاده می‌کنیم.

```
$ ros2 launch “package _name”  “launch File _name”
```

پس از اجرای این دستور ربات مانند شکل ۱۴در محیط Gazebo باز می‌شود.




شکل ۱۴: نتیجه اجرای launch فایل نوشته شده

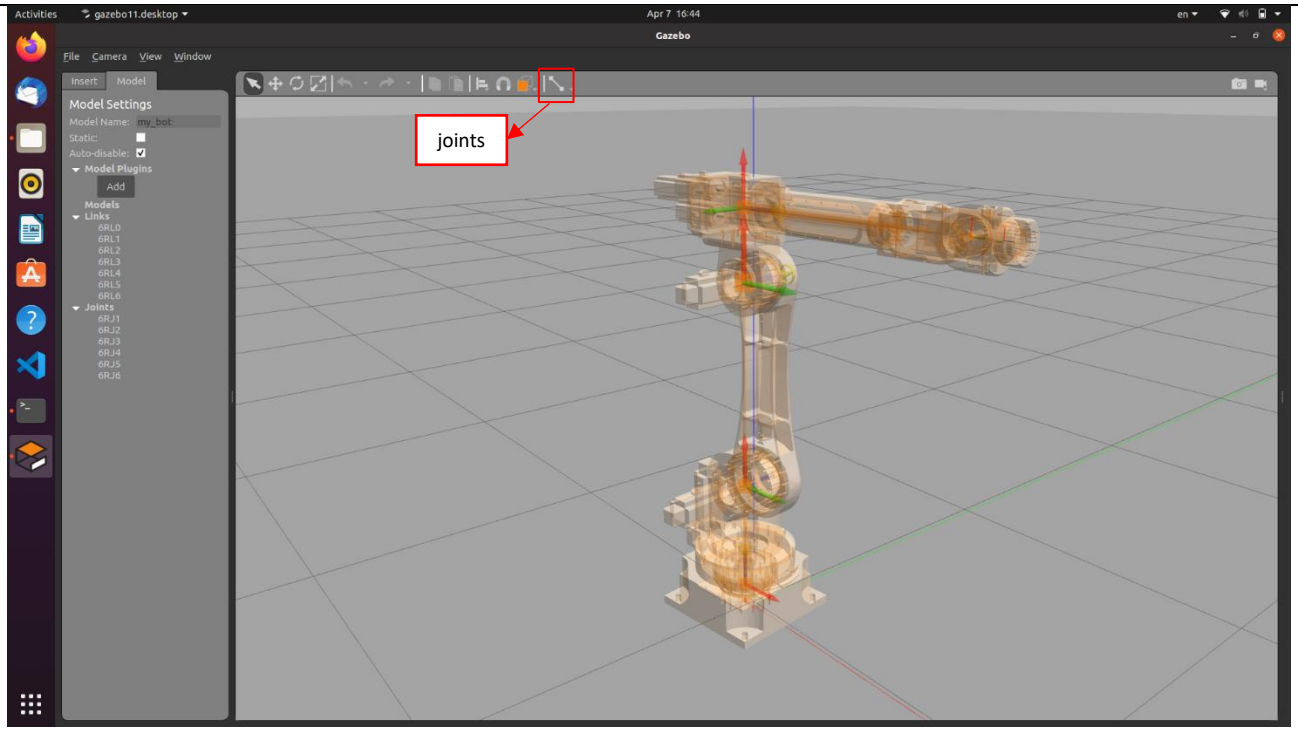
انتظار می‌رود که پس از وارد کردن مدل ربات، به دلیل اینکه هیچ گشتاور و نیرویی به ربات وارد نمی‌شود؛ ربات در اثر نیروی وزن خودش به زمین بیافتد. اما ممکن است پس از اضافه شدن urdf ربات به محیط Gazebo این اتفاق نیافتد. برای حل این مسئله باید مفاصل[3] تعریف شده را در Gazebo اصلاح کنیم.

به منظور انجام این‌کار در ســـربرگ world داخل Gazebo، زیر مجموعه models، مدل ربات (my_bot) را انتخاب و راســـت کلیک میکنیم سپس گزینه Edit model را انتخاب کرده و وارد محیط اصلاح مدل در Gazebo می‌شویم.

[1] Launch file

[2] برنامه نوشته شده در پیوست۲ آورده شده است

[3] Joints




شکل ۱۵: محیط اصلاح مدل در Gazebo

در این محیط سربرگ model را باز می‌کنیم. در این قسمت تمامی لینک ها و مفاصل آورده شده است. برای حل مشکل ذکر شده بهتر است مفاصل بین لینک ها را دوباره تعریف کنیم. برای این منظور ابتدا مفاصل را از زیر مجموعه joints انتخاب و پاک می‌کنیم. سپس گزینه joint را از نوار ابزار بالایی انتخاب می‌کنیم.

در پنجره create joint که در شکل ۱۶ آورده شده است بخش های زیر وجود دارد:

۱- بخش مربوط به تعریف نوع مفصل که در اینجا تمامی مفاصل از نوع revolute می‌باشند.
۲- بخش مربوط به تعریف لینک مادر[1]؛ در این بخش لینک قبل از اتصال را مشخص می‌کنیم.
۳- بخش مربوط به تعریف لینک فرزند[2]؛ در این بخش لینک بعد از اتصال را مشخص می‌کنیم.
۴- بخش مربوط به تعریف محور؛ در این بخش محوری که دوران مفصل حول آن می باشد مشخص می‌کنیم.

پس از تعریف تمامی این مشخصات بر روی گزینه Create کلیک کرده و مفصل را ایجاد می‌کنیم. این کار را برای تمامی مفاصل انجام می‌دهیم.

[1] Parent Link
[2] Child Link


شکل ۱۶: اضافه کرد مفصل به ربات در Gazebo

در نهایت برای ذخیره تغییرات ایجاد شده از سربرگ File گزینه Exit model Editor را انتخاب می‌کنیم و از پنجره باز شده نیز گزینه save and Exit را انتخاب می‌کنیم سپس در پنجره باز شده اسم و محل ذخیره سازی را مشخص کرده و آن را ذخیره می‌کنیم. فایل ذخیره شده به فرمت sdf خواهد بود.[۱]

برای باز کردن این فایل ابتدا به کمک دستور زیر محیط Gazebo را بالا می‌آوریم.

```
$ gazebo -u --verbose
```

پس از بالا آمدن محیط Gazebo سربرگ Insert را انتخاب کرده و گزینه Add path را انتخاب می‌کنیم. سپس مسیر ذخیره سازی فایل sdf را انتخاب کرده و ربات را وارد محیط Gazebo می‌کنیم. حال مدل ربات به درستی عمل خواهد کرد.

## ۳-۲) معرفی پلاگین 6r-gazebo-ros-plugin و اضافه کردن آن به sdf ربات

می‌توان از پلاگین ها به عنوان پیام رسان بین ساختار ROS و شبیه ساز Gazebo یادکرد. پلاگین های آماده بسیاری با کاربرد های متفاوت موجود هستند که می‌توان از آن‌ها کمک گرفت. همچنین می‌توان این پلاگین ها را به صورت شخصی سازی شده[۲] و برای اجرای یک کارکرد مشخص برنامه ریزی کرد. این پلاگین ها معمولا به زبان C نوشته می‌شوند.

پلاگین 6r-gazebo-ros-plugin با هدف دریافت مقادیر گشتاور مفاصل ها از نود[۳] نوشته شده در ROS و اعمال این گشتاورها به مفاصل در شبیه ساز Gazebo نوشته شده است. به جهت استفاده از این پلاگین از فایل so. استفاده می‌شود.[۴] مراحل اضافه کردن این پلاگین به ربات به صورت زیر است:

[۱] فایل توصیفی ربات به فرمت sdf در پیوست۳ آورده شده است.

[2] Customized

[3] Node

[۴] تمامی فایل های مورد نیاز و نام برده شده برای اضافه کردن این پلاگین، در ضمیمه گزارش آورده شده است.

**گام اول)** برای ارسال و دریافت پیغام ها بین ROS و Gazebo، پکیجی با نام tourqe_message ساخته شده است. به منظور استفاده از پلاگین، این پکیج باید حتما build و source شود. در مسیر پکیج tourqe_message دستور زیر را وارد می‌کنیم:

```
$ colcon build
```

سپس فایل setup.bash در پوشه install را source می‌کنیم:

```
$ source install/setup.bash
```

به کمک دستور زیر می‌توانیم تمامی پکیج های فعال را مشاهده کنیم و وجود پکیج tourqe_message را بررسی نماییم.

```
$ ros2 pkg list
```

**گام دوم)** برای اضافه کردن پلاگین به ربات، باید تگ زیر را به sdf ربات اضافه کنیم.

```
<plugin name=" any name ">
    filename= " plugin.so File path " >
</plugin>
```

نکته مورد توجه در استفاده از این پلاگین این است که حتما باید پکیج tourqe_message در تمامی صفحات ترمینالی که استفاده میکنیم source شود.

# فصل چهارم – ارسال گشتاور ثابت به joint ها و دریافت موقعیت مفاصل

### ۴-۱) نوشتن نود ارسال گشتاور ثابت به مفاصل ربات

برای ارسال گشتاور به موتور ها نیاز است ابتدا یک پکیج ایجاد کرده و یک نود به زبان پایتون به جهت ارسال گشتاورها بنویسیم.[۱] مقدار گشتاور ها از طریق یک پیغام به پلاگین نوشته شده ارسال می‌شود و پلاگین این گشتاور ها به مدل ربات در Gazebo اعمال می‌کند.

برای اجرای نود ها در ROS لازم است که کد نوشته شده را در فایل setup.py داخل پکیج اضافه کنیم. این فایل به صورت زیر است:

```
from setuptools import setup
package_name = 'controllerpkg'
setup(
    name=package_name,
    version='0.0.0',
    packages=[package_name],
    data_files=[
        ('share/ament_index/resource_index/packages',
            ['resource/' + package_name]),
        ('share/' + package_name, ['package.xml']),
    ],
    install_requires=['setuptools'],
    zip_safe=True,
    maintainer='narges',
    maintainer_email='nargesrajabiun@todo.todo',
    description='TODO: Package description',
    license='TODO: License declaration',
    tests_require=['pytest'],
    entry_points={
        'console_scripts': [
            'setTourqe = controllerpkg.set_Tourqe:main',
        ],
    },
)
```

با اضافه کردن کردن نود به setup.py نود نوشته شده در ROS شناخته می‌شود و برای اجرای آن از دستور زیر استفاده می‌کنیم:

```
$ ros2 run “package _name” “executable _name”
```

جمع بندی تمام کار های لازم برای ارسال گشتاور به مفاصل ربات به صورت زیر است:

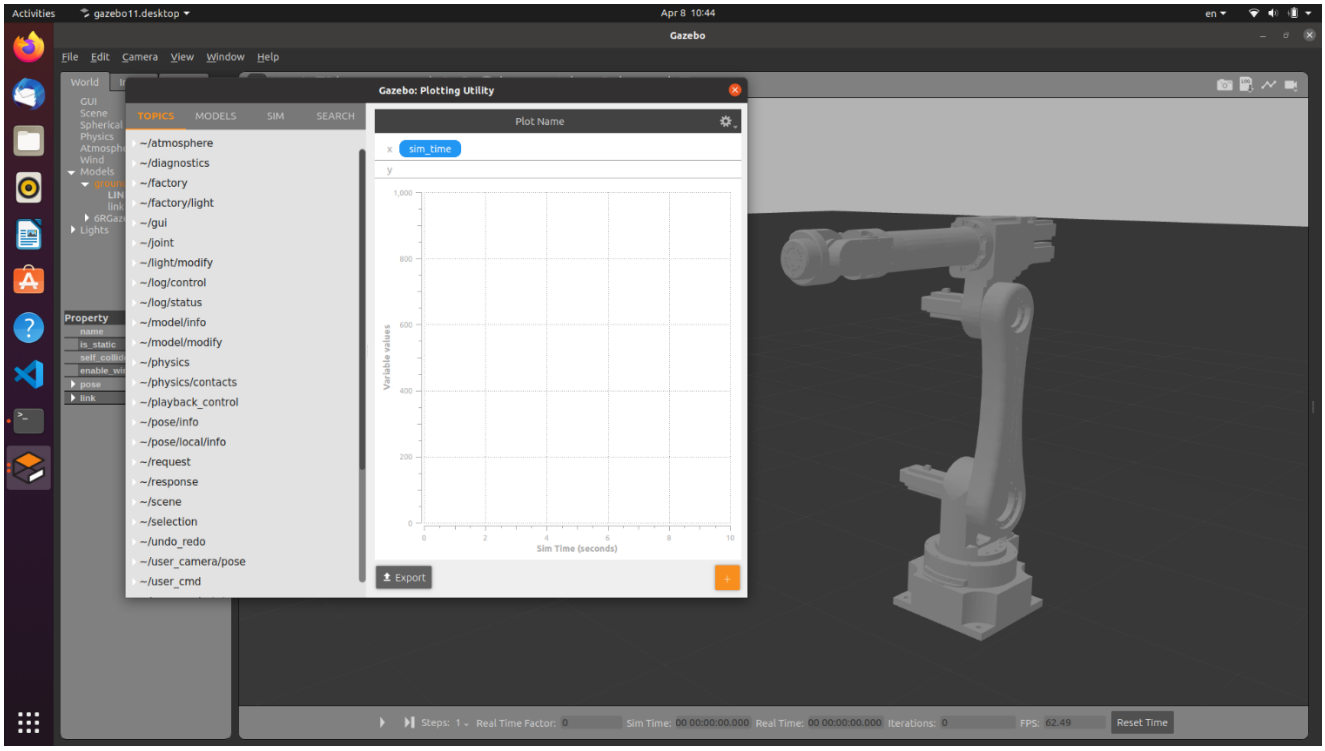
۱- در یک ترمینال پکیج tourqe_message را build و source می‌کنیم
۲- با دستور داده شده Gazebo را بالا می‌آوریم
۳- یک ترمینال جدید باز کرده و مجددا پکیج tourqe_message را source می‌کنیم
۴- به کمک دستورات cd و ls به مسیر فضای کاری می‌رویم و آن را build و source می‌کنیم
۵- نود نوشته شده را اجرا می‌کنیم

[۱] نود نوشته شده در پیوست۴آورده شده است.

## ۴-۲) دریافت و ذخیره موقعیت مفاصل در Gazebo

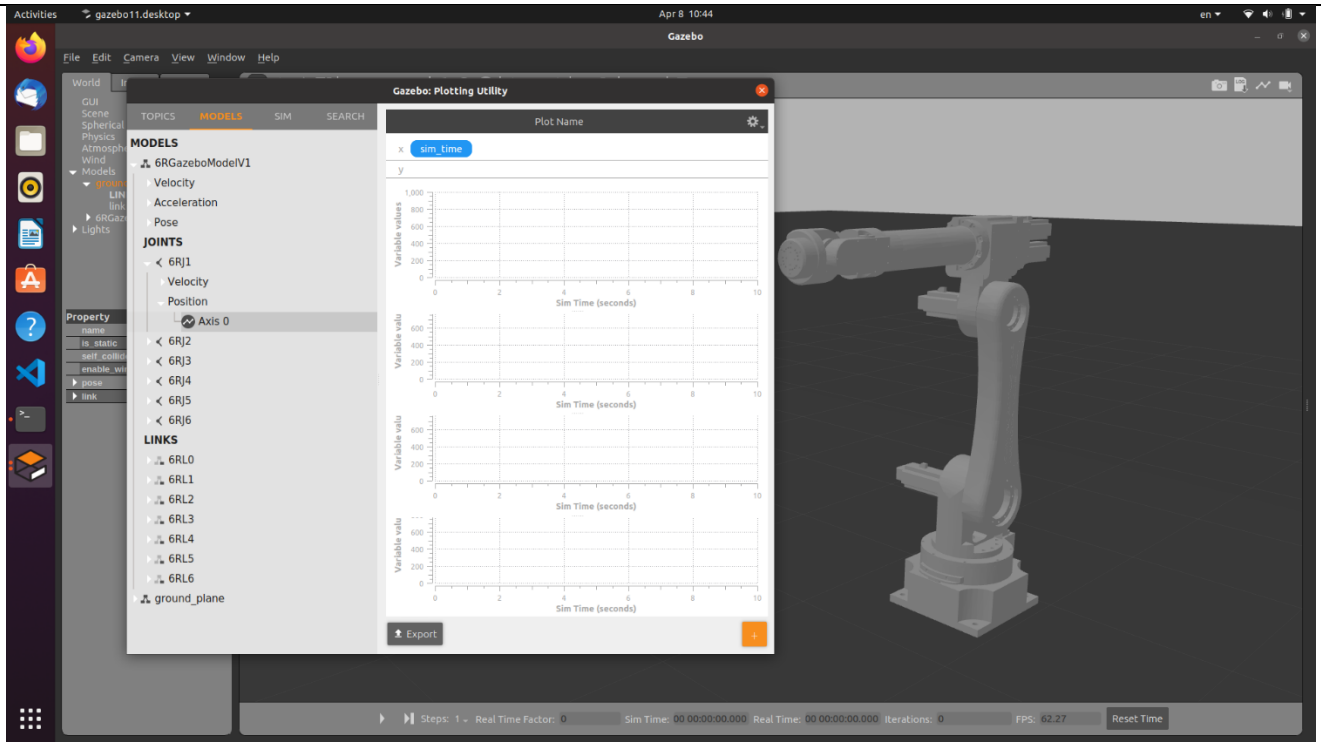
پس از پیاده سازی ربات در محیط Gazebo نوشتن یک نود در ROS برای ارسال گشتاور به مفاصل ربات، نیاز است تا موقعیت مفاصل را از Gazebo گرفته و ذخیره کنیم. برای انجام این کار هم می‌توان یک نود در ROS نوشت و هم می‌توان از خود نرم افزار Gazebo استفاده کرد. در اینجا از Gazebo برای استفاده می‌کنیم.

برای این منظور پس از آوردن ربات به محیط Gazebo، از سربرگ window، گزینه plot را انتخاب می‌کنیم و یا با استفاده از میانبر ctrl+P، پنجره ی plotting را باز می‌کنیم.




شکل ۱۷: پنجره plotting در Gazebo

از بخش سمت چپ این پنجره، سربرگ models، مدل ربات را انتخاب می‌کنیم. در اینجا لیستی از لینک ها و مفاصل خواهیم داشت که میتوانیم موقعیت و سرعت هر کدام را استخراج کنیم. در اینجا میخواهیم موقعیت مفاصل را استخراج و ذخیره کنیم. به این منظور بر روی هر یک از مفاصل کلیک کرده و گزینه Position را انتخاب کرده و Axis آن را انتخاب می‌کنیم. همچنین به کمک گزینه + در پایین سمت راست این پنجره می‌توانیم یک نمودار جدید نیز باز کنیم.



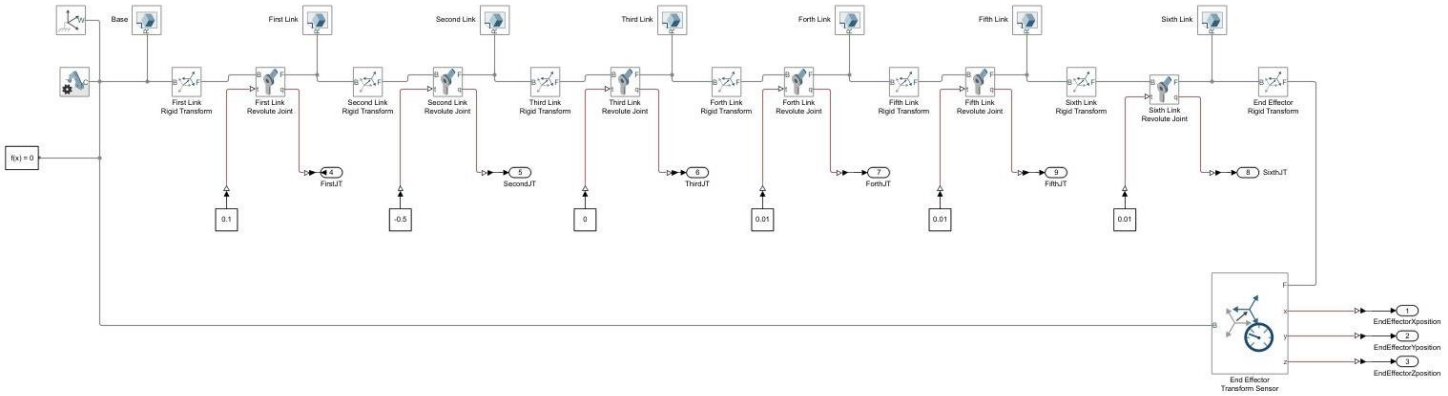

شکل ۱۸: ذخیره موقعیت مفاصل در Gazebo

پس از آماده کردن این نمودارها، نود ارسال گشتاور را اجرا می‌کنیم. پس از اتمام زمان شبیه سازی، خروجی Plot ها را به صورت csv ذخیره می‌کنیم.

# فصل پنجم – مقایسه نتایج Gazebo و مدل سیمولینک

## ۵-۱) نمایش مدل سیمولینک و ارسال همان گشتاور های ثابت به مفاصل و ذخیره نتایج

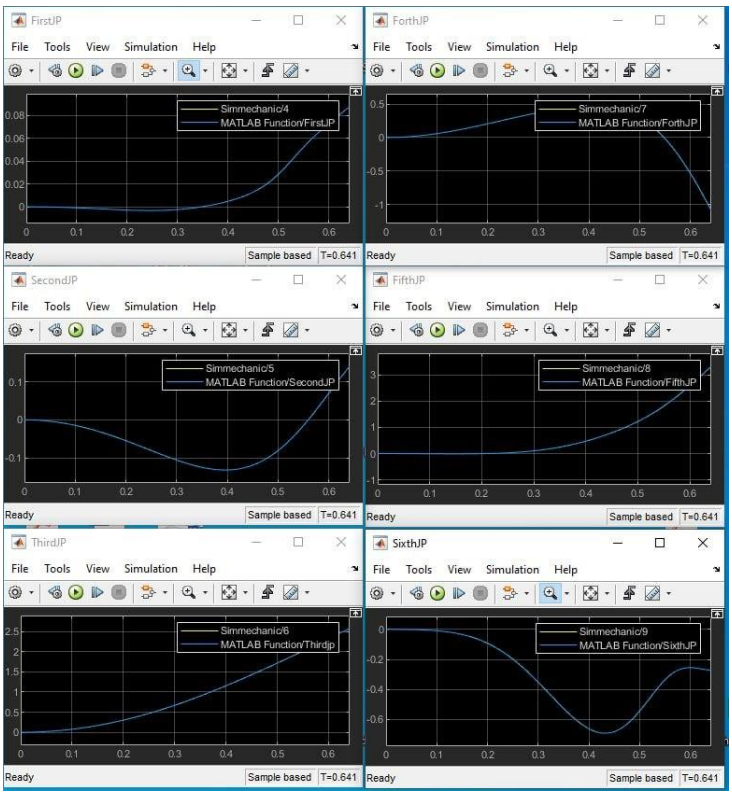
از مدل سیمولینک ربات برای صحت سنجی مدل استفاده می‌شود. در اینجا نیز گشتاور را ورودی مفاصل قرار داده و موقعیت را خروجی می‌گیریم و ذخیره می‌کنیم. مدل سیمولینک ربات به صورت زیر است:




شکل ۱۹: مدل Simulink ربات 6R در MATLAB

## ۵-۲) مقایسه نتایج موقعیت مفاصل بین دو مدل Gazebo و سیمولینک

نتایج گرفته شده از Gazebo و Simulink را که نشان دهنده موقعیت مفاصل ربات در هر شبیه ساز می‌باشد را در کنار یک دیگر رسم می‌کنیم.




شکل ۲۰: نتیجه مقایسه موقعیت مفاصل در Gazebo و simulink

همان طور که در شکل ۱۸ مشاهده می‌شود، نمودار ها کاملا منطبق می‌باشد. با توجه له این نتایج می‌توان گفت که مدل وارد شده کاملا صحیح بوده و قابل اعتماد می‌باشد.

# پیوست ها

**پیوست۱:**

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This URDF was automatically created by SolidWorks to URDF Exporter! Originally
created by Stephen Brawner (brawner@gmail.com)
     Commit Version: 1.6.0-1-g15f4949  Build Version: 1.6.7594.29634
     For more information, please see http://wiki.ros.org/sw_urdf_exporter -->
<robot xmlns:xacro="http://www.ros.org/wiki/xacro"
name="FUMTI_1401FT_GAZEBOURDF.SLDASM">
  <link name="6RL0">
    <inertial>
      <origin xyz="0.003 0 -0.005" rpy="0 0 0" />
      <mass value="31.781" />
      <inertia
        ixx="0.376"
        ixy="0"
        ixz="-0.001"
        iyy="0.364"
        iyz="0"
        izz="0.65" />
    </inertial>
    <visual>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL0.STL" />
      </geometry>
      <material name="">
        <color rgba="0.89804 0.91765 0.92941 1" />
      </material>
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL0.STL" />
      </geometry>
    </collision>
  </link>

<link name="world">
    <origin xyz="0 0 0"rpy="0 0 0" />
  </link>
  <joint name="base-joint" type="fixed">
        <origin xyz="0 0 0" rpy="0 0 0"/>
        <parent link="world"/>
        <child link="6RL0"/>
  </joint>
```

```
  <link name="6RL1">
    <inertial>
      <origin xyz="-0.016 -0.02 0.195" rpy="0 0 0" />
      <mass value="29.923" />
      <inertia
        ixx="0.557"
        ixy="-0.009"
        ixz="-0.018"
        iyy="0.601"
        iyz="-0.076"
        izz="0.362" />
    </inertial>
    <visual>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL1.STL" />
      </geometry>
      <material name="">
        <color rgba="0.89804 0.91765 0.92941 1" />
      </material>
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL1.STL" />
      </geometry>
    </collision>
  </link>

  <joint name="6RJ1" type="revolute">
    <origin xyz="0 0 0.0635" rpy="0 0 0" />
    <parent link="6RL0" />
    <child link="6RL1" />
    <axis xyz="0 0 1" />
    <limit
      lower="0"
      upper="0"
      effort="0"
      velocity="0" />
  </joint>

  <link name="6RL2">
    <inertial>
      <origin xyz="0.183 0 0.082" rpy="0 0 0" />
      <mass value="16.89" />
      <inertia
        ixx="0.067"
```

```
        ixy="0"
        ixz="0.044"
        iyy="1"
        iyz="0"
        izz="1.034" />
    </inertial>
    <visual>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL2.STL" />
      </geometry>
      <material name="">
        <color rgba="0.89804 0.91765 0.92941 1" />
      </material>
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL2.STL" />
      </geometry>
    </collision>
  </link>

  <joint name="6RJ2" type="revolute">
    <origin xyz="0 0 0.326" rpy="4.7124 -1.5708 0" />
    <parent link="6RL1" />
    <child link="6RL2" />
    <axis xyz="0 0 1" />
    <limit
      lower="0"
      upper="0"
      effort="0"
      velocity="0" />
  </joint>

  <link name="6RL3">
    <inertial>
      <origin xyz="0.128 0.002 -0.003" rpy="0 0 0" />
      <mass value="24.16" />
      <inertia
        ixx="0.618"
        ixy="-0.007"
        ixz="0.019"
        iyy="0.331"
        iyz="-0.005"
        izz="0.791" />
    </inertial>
    <visual>
```

```
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL3.STL" />
      </geometry>
      <material name="">
        <color rgba="0.89804 0.91765 0.92941 1" />
      </material>
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL3.STL" />
      </geometry>
    </collision>
  </link>

  <joint name="6RJ3" type="revolute">
    <origin xyz="0.6 0 0" rpy="0 0 0" />
    <parent link="6RL2" />
    <child link="6RL3" />
    <axis xyz="0 0 1" />
    <limit
      lower="0"
      upper="0"
      effort="0"
      velocity="0" />
  </joint>

  <link name="6RL4">
    <inertial>
      <origin xyz="0 0.015 0.562" rpy="0 0 0" />
      <mass value="4.742" />
      <inertia
        ixx="0.04"
        ixy="0"
        ixz="0"
        iyy="0.032"
        iyz="0.004"
        izz="0.018" />
    </inertial>
    <visual>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL4.STL" />
      </geometry>
      <material name="">
        <color rgba="0.89804 0.91765 0.92941 1" />
```

```
      </material>
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL4.STL" />
      </geometry>
    </collision>
  </link>

  <joint name="6RJ4" type="revolute">
    <origin xyz="0.2 0 0" rpy="-1.5708 0 0" />
    <parent link="6RL3" />
    <child link="6RL4" />
    <axis xyz="0 0 1" />
    <limit
      lower="0"
      upper="0"
      effort="0"
      velocity="0" />
  </joint>

  <link name="6RL5">
    <inertial>
      <origin xyz="-0.003 0.02 -0.013" rpy="0 0 0" />
      <mass value="2.615" />
      <inertia
        ixx="0.007"
        ixy="0"
        ixz="0"
        iyy="0.006"
        iyz="0"
        izz="0.008" />
    </inertial>
    <visual>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL5.STL" />
      </geometry>
      <material name="">
        <color rgba="0.89804 0.91765 0.92941 1" />
      </material>
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL5.STL" />
```

```
      </geometry>
    </collision>
  </link>

  <joint name="6RJ5" type="revolute">
    <origin xyz="0 0 0.6855" rpy="1.5708 0 0" />
    <parent link="6RL4" />
    <child link="6RL5" />
    <axis xyz="0 0 1" />
    <limit
      lower="0"
      upper="0"
      effort="0"
      velocity="0" />
  </joint>

  <link name="6RL6">
    <inertial>
      <origin xyz="0 0 0.029" rpy="0 0 0" />
      <mass value="0.955" />
      <inertia
        ixx="0.001"
        ixy="0"
        ixz="0"
        iyy="0.001"
        iyz="0"
        izz="0.001" />
    </inertial>
    <visual>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL6.STL" />
      </geometry>
      <material name="">
        <color rgba="0.89804 0.91765 0.92941 1" />
      </material>
    </visual>
    <collision>
      <origin xyz="0 0 0" rpy="0 0 0" />
      <geometry>
        <mesh
filename="/home/rak2218/RoboticLAB_6R/src/urdf_example/meshes/6RL6.STL" />
      </geometry>
    </collision>
  </link>

  <joint name="6RJ6" type="revolute">
    <origin xyz="0 0.079 0" rpy="-1.5708 0 0" />
    <parent link="6RL5" />
```

```
    <child link="6RL6" />
    <axis xyz="0 0 1" />
    <limit
      lower="0"
      upper="0"
      effort="0"
      velocity="0" />
  </joint>
</robot>
```

**پیوست ۲:**

```
import os
from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch.actions import IncludeLaunchDescription
from launch.launch_description_sources import PythonLaunchDescriptionSource
from launch_ros.actions import Node
import xacro
#.....................................................
def generate_launch_description():
    # Specify the name of the package and path to xacro file within the package
    pkg_name = 'gzsimulator'
    file_subpath = 'description/FUMTI_1401FT_GAZEBOURDF.urdf.xacro'
    # Use xacro to process the file
    xacro_file = os.path.join(get_package_share_directory(pkg_name),file_subpath)
    robot_description_raw = xacro.process_file(xacro_file).toxml()
    # Configure the node
    node_robot_state_publisher = Node(
        package='robot_state_publisher',
        executable='robot_state_publisher',
        output='screen',
        parameters=[{'robot_description': robot_description_raw,
        'use_sim_time': True}] # add other parameters here if required
    )
    gazebo = IncludeLaunchDescription(
        PythonLaunchDescriptionSource([os.path.join(
            get_package_share_directory('gazebo_ros'), 'launch'), '/gazebo.launch.py']),
        )
    spawn_entity = Node(package='gazebo_ros', executable='spawn_entity.py',
                    arguments=['-topic', 'robot_description',
                               '-entity', 'my_bot'],
                    output='screen')
    # Run the node
    return LaunchDescription([
        gazebo,
        node_robot_state_publisher,
        spawn_entity
    ])
```

**پیوست ۳:**

```
<?xml version='1.0'?>
<sdf version='1.7'>
  <model name='6RGazeboModelV1'>
    <pose>0 0 1 0 0 0</pose>

    <link name='6RL0'>
      <pose>0 0 0 0 -0 0</pose>
      <inertial>
        <pose>0.003 0 -0.005 0 -0 0</pose>
        <mass>31.781</mass>
        <inertia>
          <ixx>0.376</ixx>
          <ixy>0</ixy>
          <ixz>-0.001</ixz>
          <iyy>0.364</iyy>
          <iyz>0</iyz>
          <izz>0.65</izz>
        </inertia>
      </inertial>
      <self_collide>0</self_collide>
      <enable_wind>0</enable_wind>
      <kinematic>0</kinematic>
      <visual name='6RL0_visual'>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <scale>1 1 1</scale>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL0.STL</uri>
          </mesh>
        </geometry>
        <material>
          <lighting>1</lighting>
          <script>
            <uri>file://media/materials/scripts/gazebo.material</uri>
            <name>Gazebo/Black</name>
          </script>
          <shader type='pixel'>
            <normal_map>__default__</normal_map>
          </shader>
          <ambient>0 0 0 1</ambient>
          <diffuse>0.7 0.7 0.7 1</diffuse>
          <specular>0.01 0.01 0.01 1</specular>
          <emissive>0 0 0 1</emissive>
        </material>
        <transparency>0</transparency>
        <cast_shadows>1</cast_shadows>
      </visual>
```

```
      <collision name='6RL0_collision'>
        <laser_retro>0</laser_retro>
        <max_contacts>10</max_contacts>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL0.STL</uri>
            <scale>1 1 1</scale>
          </mesh>
        </geometry>
        <surface>
          <friction>
            <ode>
              <mu>1</mu>
              <mu2>1</mu2>
              <fdir1>0 0 0</fdir1>
              <slip1>0</slip1>
              <slip2>0</slip2>
            </ode>
            <torsional>
              <coefficient>1</coefficient>
              <patch_radius>0</patch_radius>
              <surface_radius>0</surface_radius>
              <use_patch_radius>1</use_patch_radius>
              <ode>
                <slip>0</slip>
              </ode>
            </torsional>
          </friction>
          <bounce>
            <restitution_coefficient>0</restitution_coefficient>
            <threshold>1e+06</threshold>
          </bounce>
          <contact>
            <collide_without_contact>0</collide_without_contact>
            <collide_without_contact_bitmask>1</collide_without_contact_bitmask>
            <collide_bitmask>1</collide_bitmask>
            <ode>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
              <max_vel>0.01</max_vel>
              <min_depth>0</min_depth>
            </ode>
            <bullet>
              <split_impulse>1</split_impulse>
              <split_impulse_penetration_threshold>-
0.01</split_impulse_penetration_threshold>
```

```
            <soft_cfm>0</soft_cfm>
            <soft_erp>0.2</soft_erp>
            <kp>1e+13</kp>
            <kd>1</kd>
          </bullet>
        </contact>
      </surface>
    </collision>
  </link>
  <link name='6RL1'>
    <pose>-0 -0 0.0635 0 -0 0</pose>
    <inertial>
      <pose>-0.016 -0.02 0.195 0 -0 0</pose>
      <mass>29.923</mass>
      <inertia>
        <ixx>0.557</ixx>
        <ixy>-0.009</ixy>
        <ixz>-0.018</ixz>
        <iyy>0.601</iyy>
        <iyz>-0.076</iyz>
        <izz>0.362</izz>
      </inertia>
    </inertial>
    <self_collide>0</self_collide>
    <enable_wind>0</enable_wind>
    <kinematic>0</kinematic>
    <visual name='6RL1_visual'>
      <pose>0 0 0 0 -0 0</pose>
      <geometry>
        <mesh>
          <scale>1 1 1</scale>
          <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL1.STL</uri>
        </mesh>
      </geometry>
      <material>
        <lighting>1</lighting>
        <script>
          <uri>file://media/materials/scripts/gazebo.material</uri>
          <name>Gazebo/Black</name>
        </script>
        <shader type='pixel'>
          <normal_map>__default__</normal_map>
        </shader>
        <ambient>0 0 0 1</ambient>
        <diffuse>0.7 0.7 0.7 1</diffuse>
        <specular>0.01 0.01 0.01 1</specular>
        <emissive>0 0 0 1</emissive>
      </material>
      <transparency>0</transparency>
```

```
        <cast_shadows>1</cast_shadows>
      </visual>
      <collision name='6RL1_collision'>
        <laser_retro>0</laser_retro>
        <max_contacts>10</max_contacts>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL1.STL</uri>
            <scale>1 1 1</scale>
          </mesh>
        </geometry>
        <surface>
          <friction>
            <ode>
              <mu>1</mu>
              <mu2>1</mu2>
              <fdir1>0 0 0</fdir1>
              <slip1>0</slip1>
              <slip2>0</slip2>
            </ode>
            <torsional>
              <coefficient>1</coefficient>
              <patch_radius>0</patch_radius>
              <surface_radius>0</surface_radius>
              <use_patch_radius>1</use_patch_radius>
              <ode>
                <slip>0</slip>
              </ode>
            </torsional>
          </friction>
          <bounce>
            <restitution_coefficient>0</restitution_coefficient>
            <threshold>1e+06</threshold>
          </bounce>
          <contact>
            <collide_without_contact>0</collide_without_contact>
            <collide_without_contact_bitmask>1</collide_without_contact_bitmask>
            <collide_bitmask>1</collide_bitmask>
            <ode>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
              <max_vel>0.01</max_vel>
              <min_depth>0</min_depth>
            </ode>
            <bullet>
              <split_impulse>1</split_impulse>
```

```
              <split_impulse_penetration_threshold>-
0.01</split_impulse_penetration_threshold>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
            </bullet>
          </contact>
        </surface>
      </collision>
    </link>
    <link name='6RL2'>
      <pose>0 -0 0.3895 1.5568 -1.57079 -3.12758</pose>
      <inertial>
        <pose>0.183 0 0.082 0 -0 0</pose>
        <mass>16.89</mass>
        <inertia>
          <ixx>0.067</ixx>
          <ixy>0</ixy>
          <ixz>0.044</ixz>
          <iyy>1</iyy>
          <iyz>0</iyz>
          <izz>1.034</izz>
        </inertia>
      </inertial>
      <self_collide>0</self_collide>
      <enable_wind>0</enable_wind>
      <kinematic>0</kinematic>
      <visual name='6RL2_visual'>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <scale>1 1 1</scale>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL2.STL</uri>
          </mesh>
        </geometry>
        <material>
          <lighting>1</lighting>
          <script>
            <uri>file://media/materials/scripts/gazebo.material</uri>
            <name>Gazebo/Yellow</name>
          </script>
          <shader type='pixel'>
            <normal_map>__default__</normal_map>
          </shader>
          <ambient>255 255 0 1</ambient>
          <diffuse>0.7 0.7 0.7 1</diffuse>
          <specular>0.01 0.01 0.01 1</specular>
          <emissive>0 0 0 1</emissive>
```

```
        </material>
        <transparency>0</transparency>
        <cast_shadows>1</cast_shadows>
      </visual>
      <collision name='6RL2_collision'>
        <laser_retro>0</laser_retro>
        <max_contacts>10</max_contacts>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL2.STL</uri>
            <scale>1 1 1</scale>
          </mesh>
        </geometry>
        <surface>
          <friction>
            <ode>
              <mu>1</mu>
              <mu2>1</mu2>
              <fdir1>0 0 0</fdir1>
              <slip1>0</slip1>
              <slip2>0</slip2>
            </ode>
            <torsional>
              <coefficient>1</coefficient>
              <patch_radius>0</patch_radius>
              <surface_radius>0</surface_radius>
              <use_patch_radius>1</use_patch_radius>
              <ode>
                <slip>0</slip>
              </ode>
            </torsional>
          </friction>
          <bounce>
            <restitution_coefficient>0</restitution_coefficient>
            <threshold>1e+06</threshold>
          </bounce>
          <contact>
            <collide_without_contact>0</collide_without_contact>
            <collide_without_contact_bitmask>1</collide_without_contact_bitmask>
            <collide_bitmask>1</collide_bitmask>
            <ode>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
              <max_vel>0.01</max_vel>
              <min_depth>0</min_depth>
            </ode>
```

```
            <bullet>
              <split_impulse>1</split_impulse>
              <split_impulse_penetration_threshold>-
0.01</split_impulse_penetration_threshold>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
            </bullet>
          </contact>
        </surface>
      </collision>
    </link>
    <link name='6RL3'>
      <pose>-4e-06 -0 0.9895 1.5568 -1.57079 -3.12758</pose>
      <inertial>
        <pose>0.128 0.002 -0.003 0 -0 0</pose>
        <mass>24.16</mass>
        <inertia>
          <ixx>0.618</ixx>
          <ixy>-0.007</ixy>
          <ixz>0.019</ixz>
          <iyy>0.331</iyy>
          <iyz>-0.005</iyz>
          <izz>0.791</izz>
        </inertia>
      </inertial>
      <self_collide>0</self_collide>
      <enable_wind>0</enable_wind>
      <kinematic>0</kinematic>
      <visual name='6RL3_visual'>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <scale>1 1 1</scale>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL3.STL</uri>
          </mesh>
        </geometry>
        <material>
          <lighting>1</lighting>
          <script>
            <uri>file://media/materials/scripts/gazebo.material</uri>
            <name>Gazebo/Yellow</name>
          </script>
          <shader type='pixel'>
            <normal_map>__default__</normal_map>
          </shader>
          <ambient>255 255 0 1</ambient>
          <diffuse>0.7 0.7 0.7 1</diffuse>
```

```
          <specular>0.01 0.01 0.01 1</specular>
          <emissive>0 0 0 1</emissive>
        </material>
        <transparency>0</transparency>
        <cast_shadows>1</cast_shadows>
      </visual>
      <collision name='6RL3_collision'>
        <laser_retro>0</laser_retro>
        <max_contacts>10</max_contacts>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL3.STL</uri>
            <scale>1 1 1</scale>
          </mesh>
        </geometry>
        <surface>
          <friction>
            <ode>
              <mu>1</mu>
              <mu2>1</mu2>
              <fdir1>0 0 0</fdir1>
              <slip1>0</slip1>
              <slip2>0</slip2>
            </ode>
            <torsional>
              <coefficient>1</coefficient>
              <patch_radius>0</patch_radius>
              <surface_radius>0</surface_radius>
              <use_patch_radius>1</use_patch_radius>
              <ode>
                <slip>0</slip>
              </ode>
            </torsional>
          </friction>
          <bounce>
            <restitution_coefficient>0</restitution_coefficient>
            <threshold>1e+06</threshold>
          </bounce>
          <contact>
            <collide_without_contact>0</collide_without_contact>
            <collide_without_contact_bitmask>1</collide_without_contact_bitmask>
            <collide_bitmask>1</collide_bitmask>
            <ode>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
              <max_vel>0.01</max_vel>
```

```
              <min_depth>0</min_depth>
            </ode>
            <bullet>
              <split_impulse>1</split_impulse>
              <split_impulse_penetration_threshold>-
0.01</split_impulse_penetration_threshold>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
            </bullet>
          </contact>
        </surface>
      </collision>
    </link>
    <link name='6RL4'>
      <pose>-5e-06 -0 1.1895 1e-05 -1.57079 -3.14159</pose>
      <inertial>
        <pose>0 0.015 0.562 0 -0 0</pose>
        <mass>4.742</mass>
        <inertia>
          <ixx>0.04</ixx>
          <ixy>0</ixy>
          <ixz>0</ixz>
          <iyy>0.032</iyy>
          <iyz>0.004</iyz>
          <izz>0.018</izz>
        </inertia>
      </inertial>
      <self_collide>0</self_collide>
      <enable_wind>0</enable_wind>
      <kinematic>0</kinematic>
      <visual name='6RL4_visual'>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <scale>1 1 1</scale>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL4.STL</uri>
          </mesh>
        </geometry>
        <material>
          <lighting>1</lighting>
          <script>
            <uri>file://media/materials/scripts/gazebo.material</uri>
            <name>Gazebo/Black</name>
          </script>
          <shader type='pixel'>
            <normal_map>__default__</normal_map>
          </shader>
```

```
          <ambient>0 0 0 1</ambient>
          <diffuse>0.7 0.7 0.7 1</diffuse>
          <specular>0.01 0.01 0.01 1</specular>
          <emissive>0 0 0 1</emissive>
        </material>
        <transparency>0</transparency>
        <cast_shadows>1</cast_shadows>
      </visual>
      <collision name='6RL4_collision'>
        <laser_retro>0</laser_retro>
        <max_contacts>10</max_contacts>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL4.STL</uri>
            <scale>1 1 1</scale>
          </mesh>
        </geometry>
        <surface>
          <friction>
            <ode>
              <mu>1</mu>
              <mu2>1</mu2>
              <fdir1>0 0 0</fdir1>
              <slip1>0</slip1>
              <slip2>0</slip2>
            </ode>
            <torsional>
              <coefficient>1</coefficient>
              <patch_radius>0</patch_radius>
              <surface_radius>0</surface_radius>
              <use_patch_radius>1</use_patch_radius>
              <ode>
                <slip>0</slip>
              </ode>
            </torsional>
          </friction>
          <bounce>
            <restitution_coefficient>0</restitution_coefficient>
            <threshold>1e+06</threshold>
          </bounce>
          <contact>
            <collide_without_contact>0</collide_without_contact>
            <collide_without_contact_bitmask>1</collide_without_contact_bitmask>
            <collide_bitmask>1</collide_bitmask>
            <ode>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
```

```
              <kd>1</kd>
              <max_vel>0.01</max_vel>
              <min_depth>0</min_depth>
            </ode>
            <bullet>
              <split_impulse>1</split_impulse>
              <split_impulse_penetration_threshold>-
0.01</split_impulse_penetration_threshold>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
            </bullet>
          </contact>
        </surface>
      </collision>
    </link>
    <link name='6RL5'>
      <pose>0.685495 9e-06 1.1895 1.5568 -1.57079 -3.12758</pose>
      <inertial>
        <pose>-0.003 0.02 -0.013 0 -0 0</pose>
        <mass>2.615</mass>
        <inertia>
          <ixx>0.007</ixx>
          <ixy>0</ixy>
          <ixz>0</ixz>
          <iyy>0.006</iyy>
          <iyz>0</iyz>
          <izz>0.008</izz>
        </inertia>
      </inertial>
      <self_collide>0</self_collide>
      <enable_wind>0</enable_wind>
      <kinematic>0</kinematic>
      <visual name='6RL5_visual'>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <scale>1 1 1</scale>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL5.STL</uri>
          </mesh>
        </geometry>
        <material>
          <lighting>1</lighting>
          <script>
            <uri>file://media/materials/scripts/gazebo.material</uri>
            <name>Gazebo/Black</name>
          </script>
          <shader type='pixel'>
```

```
            <normal_map>__default__</normal_map>
          </shader>
          <ambient>0 0 0 1</ambient>
          <diffuse>0.7 0.7 0.7 1</diffuse>
          <specular>0.01 0.01 0.01 1</specular>
          <emissive>0 0 0 1</emissive>
        </material>
        <transparency>0</transparency>
        <cast_shadows>1</cast_shadows>
      </visual>
      <collision name='6RL5_collision'>
        <laser_retro>0</laser_retro>
        <max_contacts>10</max_contacts>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL5.STL</uri>
            <scale>1 1 1</scale>
          </mesh>
        </geometry>
        <surface>
          <friction>
            <ode>
              <mu>1</mu>
              <mu2>1</mu2>
              <fdir1>0 0 0</fdir1>
              <slip1>0</slip1>
              <slip2>0</slip2>
            </ode>
            <torsional>
              <coefficient>1</coefficient>
              <patch_radius>0</patch_radius>
              <surface_radius>0</surface_radius>
              <use_patch_radius>1</use_patch_radius>
              <ode>
                <slip>0</slip>
              </ode>
            </torsional>
          </friction>
          <bounce>
            <restitution_coefficient>0</restitution_coefficient>
            <threshold>1e+06</threshold>
          </bounce>
          <contact>
            <collide_without_contact>0</collide_without_contact>
            <collide_without_contact_bitmask>1</collide_without_contact_bitmask>
            <collide_bitmask>1</collide_bitmask>
            <ode>
              <soft_cfm>0</soft_cfm>
```

```
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
              <max_vel>0.01</max_vel>
              <min_depth>0</min_depth>
            </ode>
            <bullet>
              <split_impulse>1</split_impulse>
              <split_impulse_penetration_threshold>-
0.01</split_impulse_penetration_threshold>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
            </bullet>
          </contact>
        </surface>
      </collision>
    </link>
    <link name='6RL6'>
      <pose>0.764495 1e-05 1.18951 1e-05 -1.57079 -3.14159</pose>
      <inertial>
        <pose>0 0 0.029 0 -0 0</pose>
        <mass>0.955</mass>
        <inertia>
          <ixx>0.001</ixx>
          <ixy>0</ixy>
          <ixz>0</ixz>
          <iyy>0.001</iyy>
          <iyz>0</iyz>
          <izz>0.001</izz>
        </inertia>
      </inertial>
      <self_collide>0</self_collide>
      <enable_wind>0</enable_wind>
      <kinematic>0</kinematic>
      <visual name='6RL6_visual'>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <scale>1 1 1</scale>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL6.STL</uri>
          </mesh>
        </geometry>
        <material>
          <lighting>1</lighting>
          <script>
            <uri>file://media/materials/scripts/gazebo.material</uri>
            <name>Gazebo/Black</name>
```

```
          </script>
          <shader type='pixel'>
            <normal_map>__default__</normal_map>
          </shader>
          <ambient>0 0 0 1</ambient>
          <diffuse>0.7 0.7 0.7 1</diffuse>
          <specular>0.01 0.01 0.01 1</specular>
          <emissive>0 0 0 1</emissive>
        </material>
        <transparency>0</transparency>
        <cast_shadows>1</cast_shadows>
      </visual>
      <collision name='6RL6_collision'>
        <laser_retro>0</laser_retro>
        <max_contacts>10</max_contacts>
        <pose>0 0 0 0 -0 0</pose>
        <geometry>
          <mesh>
            <uri>/home/nargas/RRRRRR-Control-
Project/RoboticLAB_6R/src/gzsimulator/meshes/6RL6.STL</uri>
            <scale>1 1 1</scale>
          </mesh>
        </geometry>
        <surface>
          <friction>
            <ode>
              <mu>1</mu>
              <mu2>1</mu2>
              <fdir1>0 0 0</fdir1>
              <slip1>0</slip1>
              <slip2>0</slip2>
            </ode>
            <torsional>
              <coefficient>1</coefficient>
              <patch_radius>0</patch_radius>
              <surface_radius>0</surface_radius>
              <use_patch_radius>1</use_patch_radius>
              <ode>
                <slip>0</slip>
              </ode>
            </torsional>
          </friction>
          <bounce>
            <restitution_coefficient>0</restitution_coefficient>
            <threshold>1e+06</threshold>
          </bounce>
          <contact>
            <collide_without_contact>0</collide_without_contact>
            <collide_without_contact_bitmask>1</collide_without_contact_bitmask>
            <collide_bitmask>1</collide_bitmask>
```

```
            <ode>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
              <max_vel>0.01</max_vel>
              <min_depth>0</min_depth>
            </ode>
            <bullet>
              <split_impulse>1</split_impulse>
              <split_impulse_penetration_threshold>-
0.01</split_impulse_penetration_threshold>
              <soft_cfm>0</soft_cfm>
              <soft_erp>0.2</soft_erp>
              <kp>1e+13</kp>
              <kd>1</kd>
            </bullet>
          </contact>
        </surface>
      </collision>
    </link>

    <!--__________________fixing base_link to world__________________-->
    <joint name ="worldJ0" type="fixed">

        <parent>world</parent>
        <child>6RL0</child>
        <pose>0 0 1 0 0 0</pose>
    </joint>
    <!--__________________fixing base_link to world__________________-->


    <joint name='6RJ1' type='revolute'>
      <parent>6RL0</parent>
      <child>6RL1</child>
      <pose>0 0 0 0 -0 0</pose>
      <axis>
        <xyz>0 0 1</xyz>
        <limit>
          <lower>-1.79769e+308</lower>
          <upper>1.79769e+308</upper>
          <effort>-1</effort>
          <velocity>-1</velocity>
        </limit>
        <dynamics>
          <spring_reference>0</spring_reference>
          <spring_stiffness>0</spring_stiffness>
          <damping>0</damping>
          <friction>0</friction>
        </dynamics>
```

```
      </axis>
      <physics>
        <ode>
          <limit>
            <cfm>0</cfm>
            <erp>0.2</erp>
          </limit>
          <suspension>
            <cfm>0</cfm>
            <erp>0.2</erp>
          </suspension>
        </ode>
      </physics>
    </joint>
    <joint name='6RJ2' type='revolute'>
      <parent>6RL1</parent>
      <child>6RL2</child>
      <pose>0 0 0 0 -0 0</pose>
      <axis>
        <xyz>0 0 1</xyz>
        <limit>
          <lower>-1.79769e+308</lower>
          <upper>1.79769e+308</upper>
          <effort>-1</effort>
          <velocity>-1</velocity>
        </limit>
        <dynamics>
          <spring_reference>0</spring_reference>
          <spring_stiffness>0</spring_stiffness>
          <damping>0</damping>
          <friction>0</friction>
        </dynamics>
      </axis>
      <physics>
        <ode>
          <limit>
            <cfm>0</cfm>
            <erp>0.2</erp>
          </limit>
          <suspension>
            <cfm>0</cfm>
            <erp>0.2</erp>
          </suspension>
        </ode>
      </physics>
    </joint>
    <joint name='6RJ3' type='revolute'>
      <parent>6RL2</parent>
      <child>6RL3</child>
      <pose>0 0 0 0 -0 0</pose>
```

```
      <axis>
        <xyz>0 0 1</xyz>
        <limit>
          <lower>-1.79769e+308</lower>
          <upper>1.79769e+308</upper>
          <effort>-1</effort>
          <velocity>-1</velocity>
        </limit>
        <dynamics>
          <spring_reference>0</spring_reference>
          <spring_stiffness>0</spring_stiffness>
          <damping>0</damping>
          <friction>0</friction>
        </dynamics>
      </axis>
      <physics>
        <ode>
          <limit>
            <cfm>0</cfm>
            <erp>0.2</erp>
          </limit>
          <suspension>
            <cfm>0</cfm>
            <erp>0.2</erp>
          </suspension>
        </ode>
      </physics>
    </joint>
    <joint name='6RJ4' type='revolute'>
      <parent>6RL3</parent>
      <child>6RL4</child>
      <pose>0 0 0 0 -0 0</pose>
      <axis>
        <xyz>0 0 1</xyz>
        <limit>
          <lower>-1.79769e+308</lower>
          <upper>1.79769e+308</upper>
          <effort>-1</effort>
          <velocity>-1</velocity>
        </limit>
        <dynamics>
          <spring_reference>0</spring_reference>
          <spring_stiffness>0</spring_stiffness>
          <damping>0</damping>
          <friction>0</friction>
        </dynamics>
      </axis>
      <physics>
        <ode>
          <limit>
```

```
          <cfm>0</cfm>
          <erp>0.2</erp>
        </limit>
        <suspension>
          <cfm>0</cfm>
          <erp>0.2</erp>
        </suspension>
      </ode>
    </physics>
  </joint>
  <joint name='6RJ5' type='revolute'>
    <parent>6RL4</parent>
    <child>6RL5</child>
    <pose>0 0 0 0 -0 0</pose>
    <axis>
      <xyz>0 0 1</xyz>
      <limit>
        <lower>-1.79769e+308</lower>
        <upper>1.79769e+308</upper>
        <effort>-1</effort>
        <velocity>-1</velocity>
      </limit>
      <dynamics>
        <spring_reference>0</spring_reference>
        <spring_stiffness>0</spring_stiffness>
        <damping>0</damping>
        <friction>0</friction>
      </dynamics>
    </axis>
    <physics>
      <ode>
        <limit>
          <cfm>0</cfm>
          <erp>0.2</erp>
        </limit>
        <suspension>
          <cfm>0</cfm>
          <erp>0.2</erp>
        </suspension>
      </ode>
    </physics>
  </joint>
  <joint name='6RJ6' type='revolute'>
    <parent>6RL5</parent>
    <child>6RL6</child>
    <pose>0 0 0 0 -0 0</pose>
    <axis>
      <xyz>0 0 1</xyz>
      <limit>
        <lower>-1.79769e+308</lower>
```

```
          <upper>1.79769e+308</upper>
          <effort>-1</effort>
          <velocity>-1</velocity>
        </limit>
        <dynamics>
          <spring_reference>0</spring_reference>
          <spring_stiffness>0</spring_stiffness>
          <damping>0</damping>
          <friction>0</friction>
        </dynamics>
      </axis>
      <physics>
        <ode>
          <limit>
            <cfm>0</cfm>
            <erp>0.2</erp>
          </limit>
          <suspension>
            <cfm>0</cfm>
            <erp>0.2</erp>
          </suspension>
        </ode>
      </physics>
    </joint>
    <static>0</static>
    <allow_auto_disable>1</allow_auto_disable>

    <!-- ___________________gazebo ROS 6R plugin___________________ -->
    <plugin name='gazebo_ROS_6R_plugin'
    filename='/home/nargas/RRRRRR-Control-Project/plugin/SO_Files/libgazebo_ros_6r.so'>
    </plugin>
    <!-- ___________________gazebo ROS 6R plugin___________________ -->

  </model>
</sdf>
```

**پیوست ۴:**

```
import sys
import signal
import time
import math
from tokenize import Double
from typing import Counter
from torque_message.srv import TRMsg
from std_srvs.srv import Empty
from nav_msgs.msg import Odometry
from sensor_msgs.msg import JointState
import rclpy
from rclpy.node import Node
from decimal import *
import csv
import numpy as np
#.....................................................
getcontext().prec = 8

class MinimalClientAsync(Node):

    def __init__(self):
        super().__init__('minimal_client_async')

        # create service and client to apply Torque
        print("in calling service ")
        self.step_call_cli = self.create_client(TRMsg, 'torque_step')
        self.req = TRMsg.Request()

        # time counter
        self.counter = Decimal(0.0000)

# send request to Service
    def send_request(self):

        #writer = csv.writer(self.file)
        while self.counter <= Decimal(1.000) :   # total time

            # Apply calculated Torque on Delta robot
            self.req.t1 = float(0.1)
            self.req.t2 = float(-5.0)
            self.req.t3 = float(0)
            self.req.t4 = float(0.01)
            self.req.t5 = float(0.01)
            self.req.t6 = float(0.01)
            print("calling Service ")

            self.future_step = self.step_call_cli.call_async(self.req)
            print("next_step"+str(self.counter))
```

```
            self.counter += Decimal("0.0010")

def main(args=None):
    rclpy.init(args=args)

    minimal_client = MinimalClientAsync()
    minimal_client.send_request()
    minimal_client.destroy_node()
    rclpy.shutdown()

if __name__ == '__main__':
    main()
```