

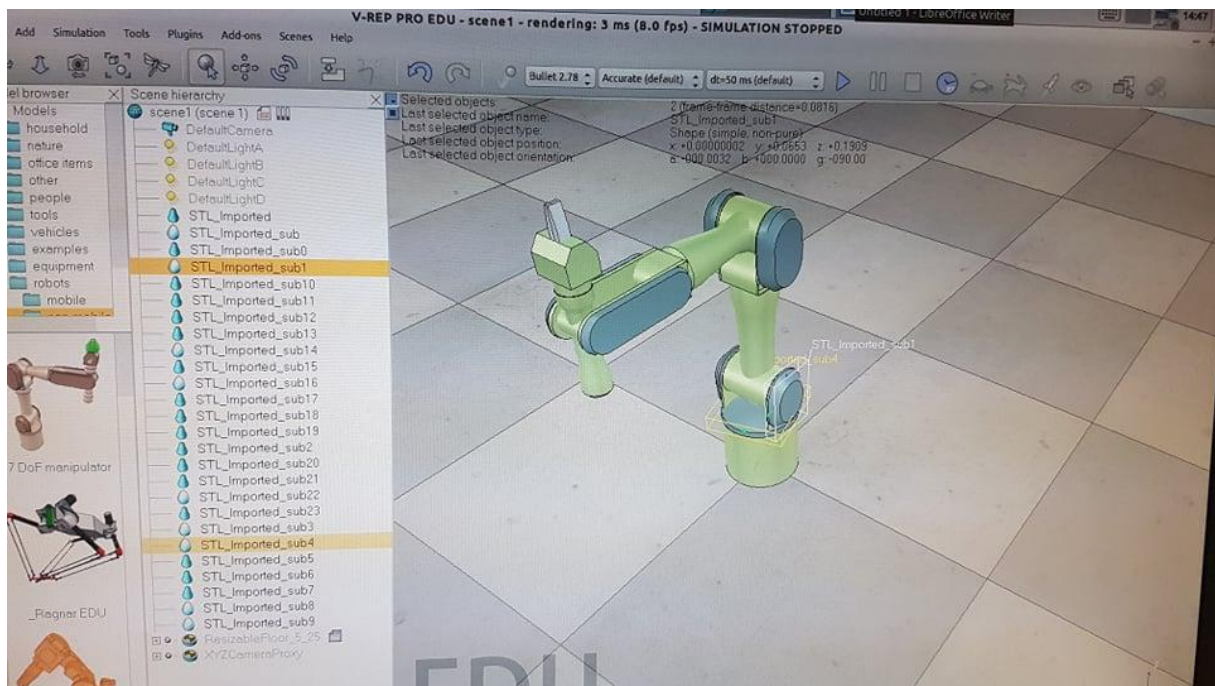
# Sprawozdanie Lab 2

Andrzej Żaba, gr\_lab 4 nr indeksu: 401490

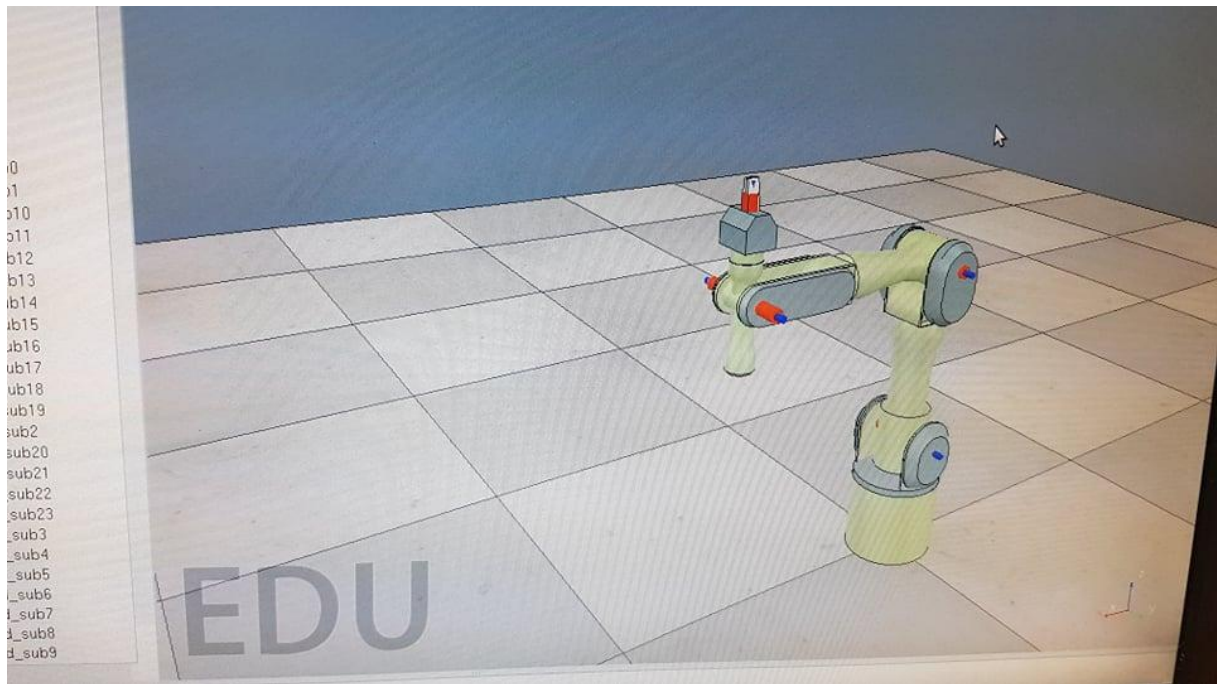
Do programu zaimportowano projekt robota z pliku redundantManipulator.stl.

Używając funkcji

„Edit -> grouping / Mering -> ungroup selected shapes” podzielono całość robota na poszczególne komponenty z których jest złożony. Następnie zmieniono kolory kilku z jego części.

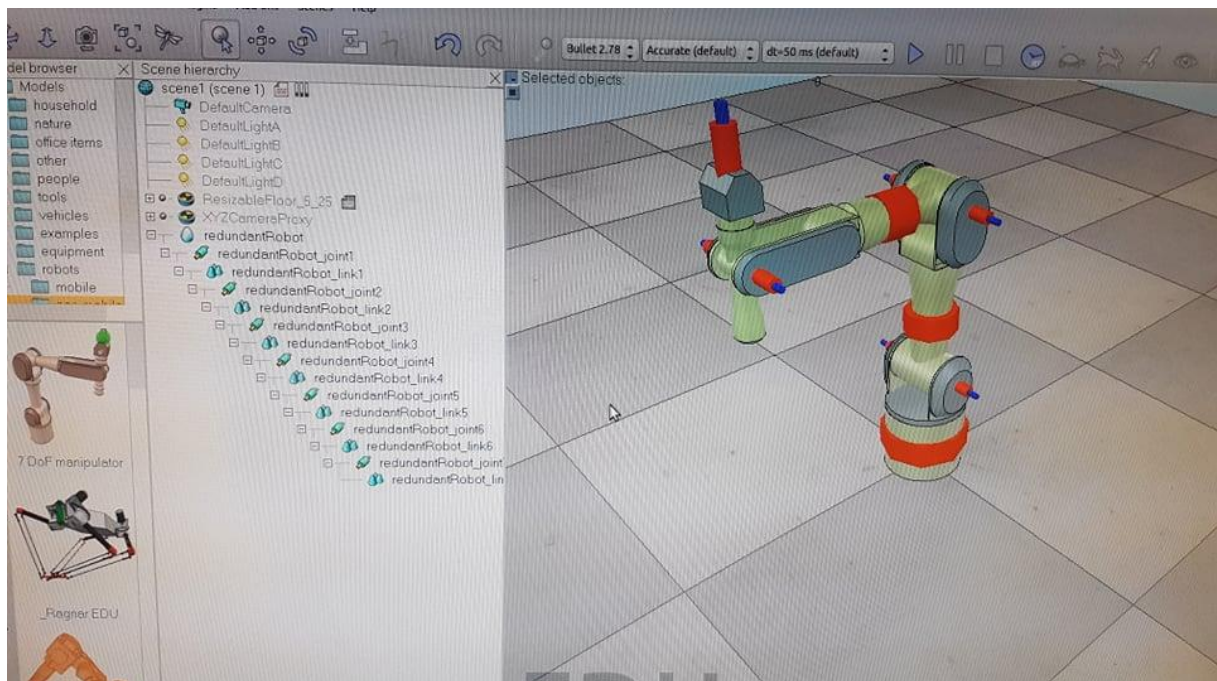


Dodano 7 złąceń - po jednym na każdy stopień swobody robota.



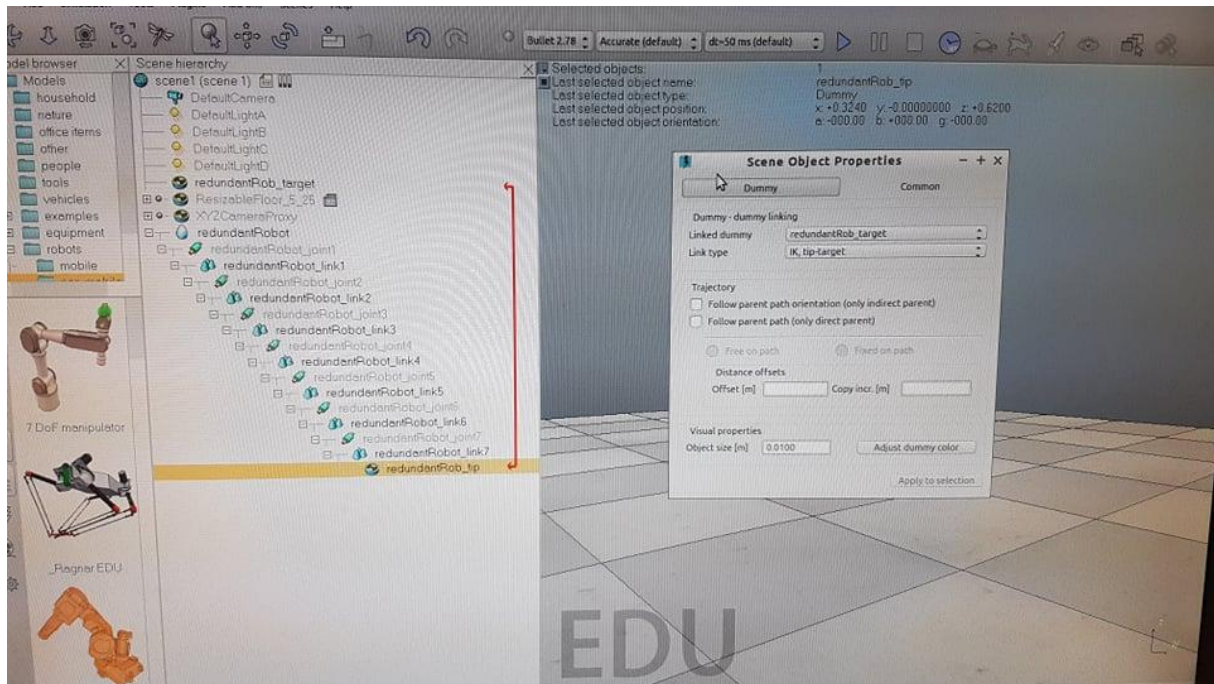
Następnie zmieniono ich wymiary oraz zgrupowano elementy należące do tego samego członu robota. Zastosowano odpowiednią hierarchię między grupami elementów a złączeniami. Złączenie jest rodzicem zgrupowanych członów.

Nadano również odpowiednie nazwy członom i złączom.



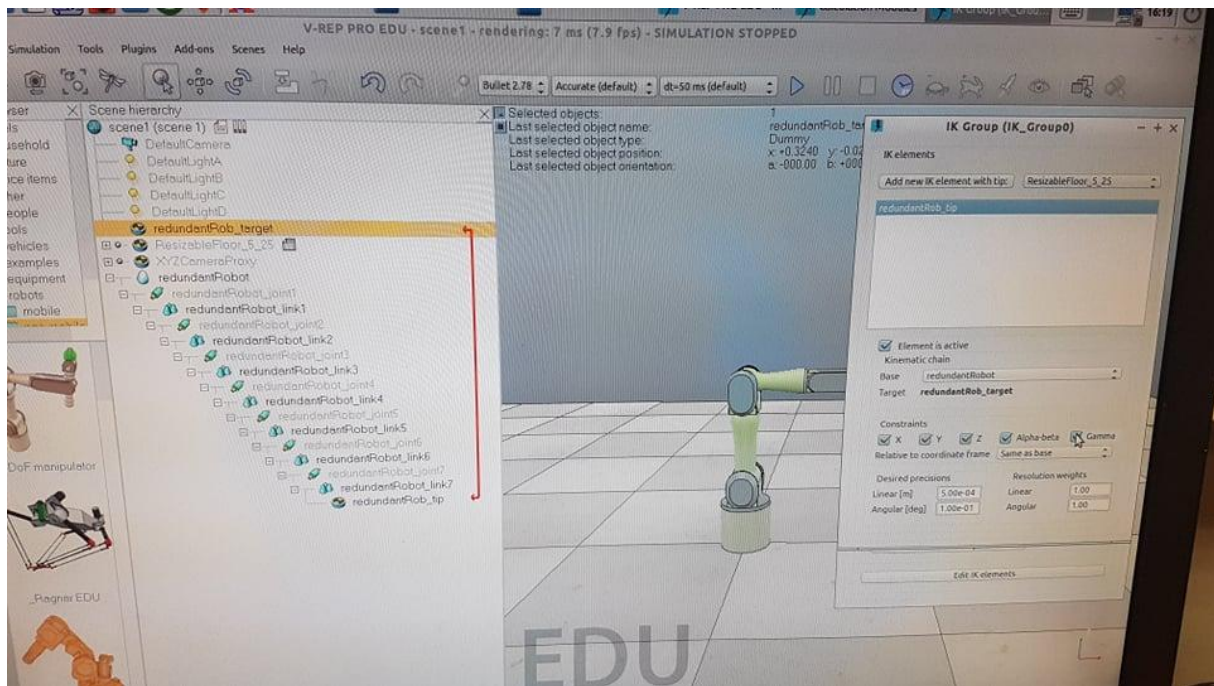
W oknie dialogowym połączenia dla wszystkich silników zaznaczono opcję: „mode: Inverse kinematics mode”. Następnie zmieniono warstwy widzialności z drugiej na dziesiątą, aby ukryć połączenia.

Zgodnie z instrukcją dodano dwie kukły (dummy) – tip oraz target. Ustalono im odpowiednią hierarchię oraz pozycję w układzie. Przypisano je do naszego robota oraz nadano im odpowiednie właściwości. Kukły te są naszą docelową parą kinematyki odwrotnej – dla tip ustawiono „redundantRob\_target” jako „Linked dummy”.



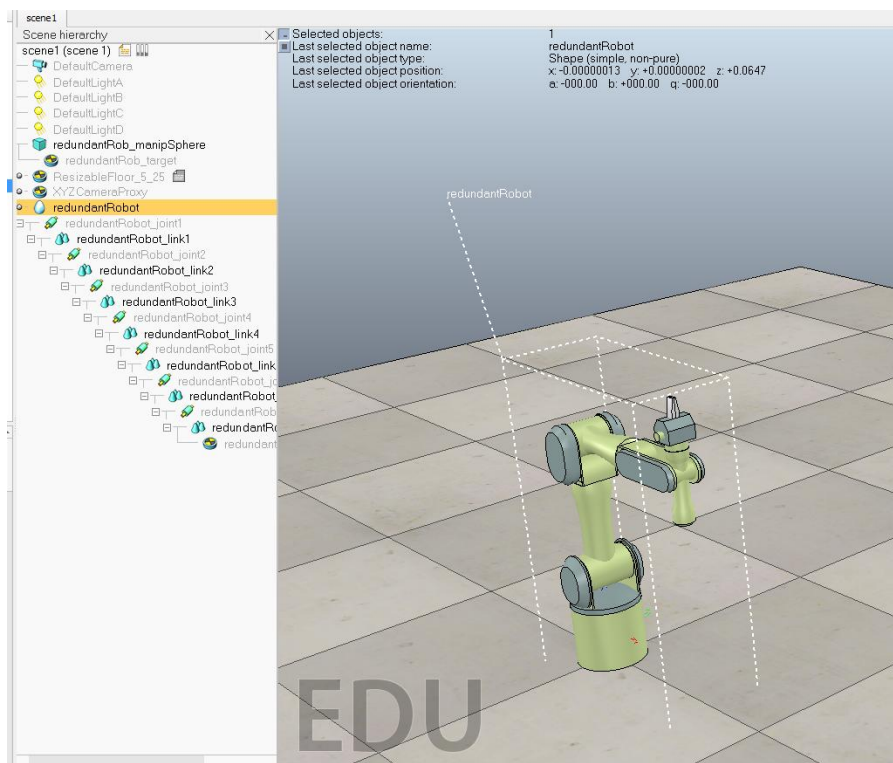
Kolejno stworzono grupę kinematyki odwrotnej. Dodano w niej tip jako nowy element oraz wskazano część redundantRob jako bazę. Zaznaczono wszystkie elementy w sekcji Constrains (włącznie z kątami alfa, beta i gamma) – oznacza to, że chcemy aby tip docelowo podążał za targetem.





Zauważono problemy z poruszaniem się manipulatora w czasie symulacji, gdy target znajdował się poza jego zasięgiem. Zmieniono wtedy właściwość Calc. method z pseudo inverse na DLS.

W oknie dialogowym redundantRob wybrano opcję „Select base of model instead” co spowodowało, że wszystkie elementy będące niżej w hierarchii zaczęły odnosić się do całości dużego mechanizmu.



[illegible]

The screenshot displays a ROS2 environment with a redundant robot arm simulation. The interface is divided into several panels:

- Scene Hierarchy (Left):** A tree view showing the scene structure. The selected object is `redundantRobot`, which contains `redundantRob_manipSphere` and `redundantRob_target`. The `redundantRob_target` is further detailed with joints and links.
- Top Status Bar:** Displays the selected object's name (`Dummy`), position (`x: +0.3240 y: +0.0250 z: +0.6200`), orientation (`a: -000.00 b: +000.00 q: -000.00`), and the collection it belongs to (`redundantRob (containing 18 objects)`).
- 3D View (Center):** A top-down perspective of the robot arm. The arm is yellow with red joints and is positioned on a light blue grid floor. A yellow dashed box highlights the target area.
- Right Panel:** A panel for managing collections and composing elements. It includes a **Collections** section with a `redundantRob` collection. Below it, the **Operation on selected collection** section shows options for what to include (All objects, Selected objects, Tree of selected object, Base included, Chain of selected object, Tip included). The **Composing elements and attributes** section shows a list of elements to be added to the collection, including joints and links.

The screenshot displays a ROS2 simulation environment. On the left, the 'Scene hierarchy' panel shows a tree structure of objects, with 'redundantRobot0' and 'redundantRobot1' selected. The 'Selected objects' panel at the top shows details for 'redundantRobot0', including its name, type, position, orientation, and simulation statistics. The 3D view shows two robots on a checkered floor: a red robot and a green robot. The red robot is a redundant robot with multiple joints, and the green robot is a standard robot with a single joint. The background is a light blue wall.

