2020

Le Robot Marcheur

PROJET ROBOTIQUE INDUSTRIELLE

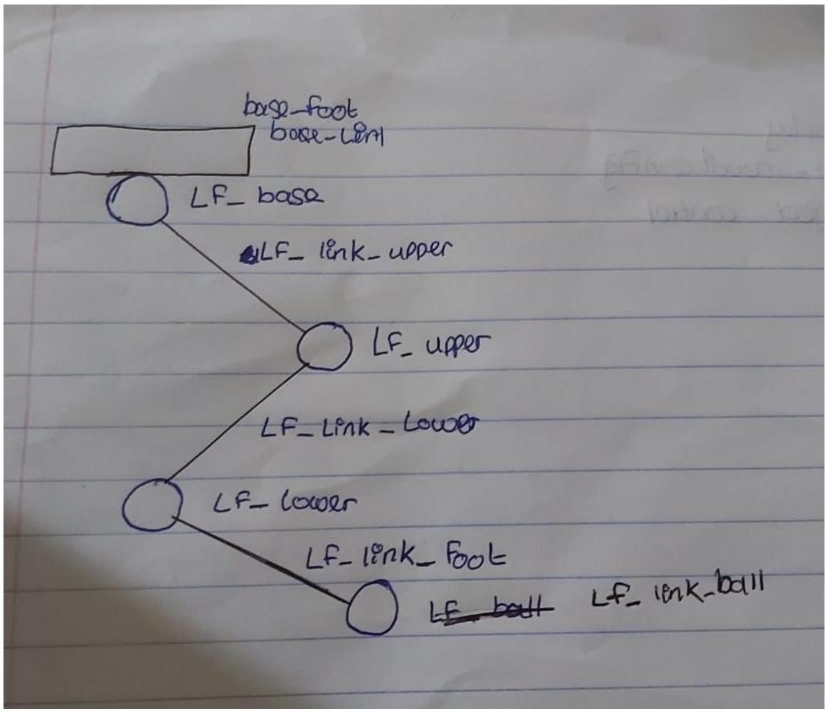
MUNISH DAWOONAH

Table of Contents

Partie 1 Conception d'une jambe

Please note that Miss Sungkur provide me with the vdi of her virtual box as I was having issues with mine.

1. Concevoir le schéma cinématique de la jambe de votre système.



2. **Indiquer les repères liés aux différents solides de votre système.**

Links: lf_link_upper, lf_link_lower, lf_link_foot

End effector: lf_link_ball

3. **Déterminer la matrice de Denavit-Hartenberg**

a) Determine the axes.

b) DH table

n	**θ**	**α**	**r**	**d**
1	180	90	0	0
2	θ2	0	a	0
3	90	180	0	0

4. Créer un package qui contiendra l'urdf de votre robot ainsi que les launchfile nécessaires à la visualisation des urdf.

Create package.

```
deeya@vunit:~$ cd catkin_ws/src
deeya@vunit:~/catkin_ws/src$ catkin_create_pkg urdf_oneleg rospy roscpp urdf std
_msgs geometry_msgs sensor_msgs
Created file urdf_oneleg/package.xml
Created file urdf_oneleg/CMakeLists.txt
Created folder urdf_oneleg/include/urdf_oneleg
Created folder urdf_oneleg/src
Successfully created files in /home/deeya/catkin_ws/src/urdf_oneleg. Please adju
st the values in package.xml.
deeya@vunit:~/catkin_ws/src$
```

Create launch file

```
deeya@vunit:~/catkin_ws/src/urdf_oneleg/urdf$ cd ..
deeya@vunit:~/catkin_ws/src/urdf_oneleg$ mkdir launch
deeya@vunit:~/catkin_ws/src/urdf_oneleg$ ls
CMakeLists.txt  include  launch  package.xml  src  urdf
deeya@vunit:~/catkin_ws/src/urdf_oneleg$ cd launch
deeya@vunit:~/catkin_ws/src/urdf_oneleg/launch$ touch launch.launch
deeya@vunit:~/catkin_ws/src/urdf_oneleg/launch$ ls
launch.launch
```

5. Créer l'urdf de votre robot avec une jambe et le base_link de votre robot.

Adding urdf

```
deeya@vunit:~/catkin_ws/src$ cd urdf_oneleg
deeya@vunit:~/catkin_ws/src/urdf_oneleg$ ls
CMakeLists.txt  include  package.xml  src
deeya@vunit:~/catkin_ws/src/urdf_oneleg$ mkdir urdf
deeya@vunit:~/catkin_ws/src/urdf_oneleg$ ls
CMakeLists.txt  include  package.xml  src  urdf
deeya@vunit:~/catkin_ws/src/urdf_oneleg$ cd urdf
deeya@vunit:~/catkin_ws/src/urdf_oneleg/urdf$ touch oneleg.urdf
deeya@vunit:~/catkin_ws/src/urdf_oneleg/urdf$ ls
oneleg.urdf
```

Using catkin build

```
deeya@vunit:~/catkin_ws$ catkin build
-----------------------------------------------------------
Profile:                     default
Extending:          [cached] /opt/ros/melodic
Workspace:                   /home/deeya/catkin_ws
-----------------------------------------------------------
Build Space:        [exists] /home/deeya/catkin_ws/build
Devel Space:        [exists] /home/deeya/catkin_ws/devel
Install Space:      [unused] /home/deeya/catkin_ws/install
Log Space:          [exists] /home/deeya/catkin_ws/logs
Source Space:       [exists] /home/deeya/catkin_ws/src
DESTDIR:            [unused] None
-----------------------------------------------------------
Devel Space Layout:          linked
Install Space Layout:        None
-----------------------------------------------------------
Additional CMake Args:       None
Additional Make Args:        None
Additional catkin Make Args: None
Internal Make Job Server:    True
Cache Job Environments:      False
-----------------------------------------------------------
Whitelisted Packages:        None
Blacklisted Packages:        None
```

6. Créer le package udm_project_moveitconfig avec le moveit assistant setup.

a) Create package.

```
deeya@vunit:~/catkin_ws/src$ mkdir udm_project_moveitconfig
```

b) Launch moveit.

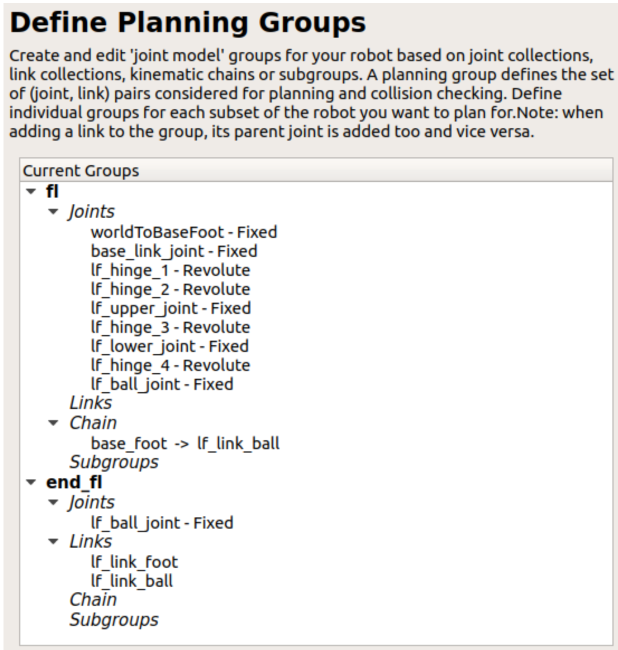
```
deeya@vunit:~/catkin_ws/src$ roslaunch moveit_setup_assistant setup_assistant.la
unch
```

c) Configure it.

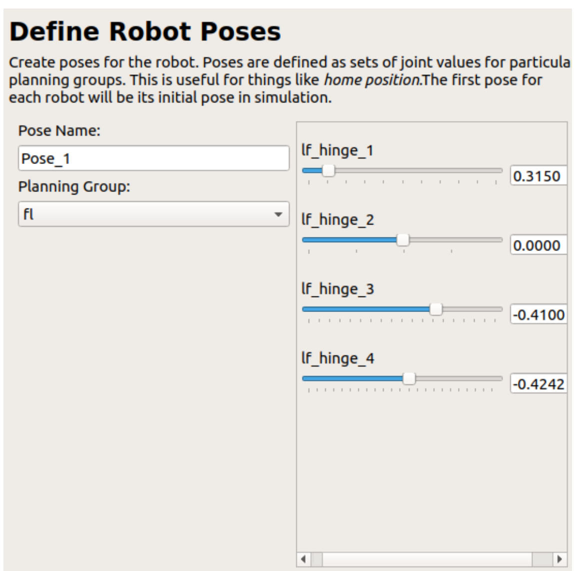
i. Generate Collision Matrix

ii. Create virtual joints

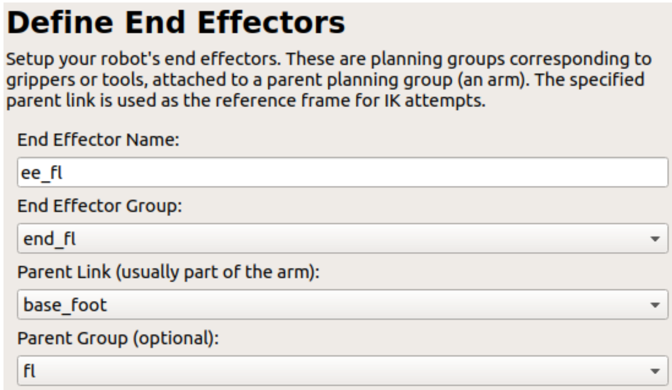
iii. Define planning groups



iv. Define Robot Poses



v. Define End Effectors



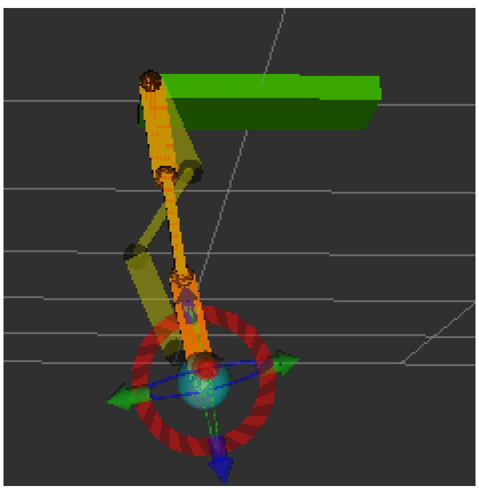
vi. Click on auto add_followJointsTrajectory….

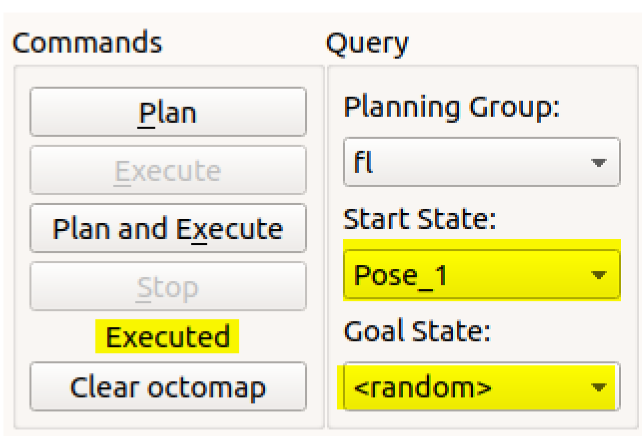
vii. Generate URDF (gazebo)

viii. Add author information.

ix. Generate configuration file in udm_project_moveitconfig folder.

x. Using rviz to plan and execute movement.





7. Créer un package udm_project_control avec un noeud permettant de contrôler la jambe de manière directe, puis un autre noeud contrôlant la jambe de manière indirecte

a) Create folder.

b) Launch demo.launch

```
deeya@vunit:~/catkin_ws$ roslaunch udm_project_moveitconfig demo.launch
```

c) Launch move_group.launch

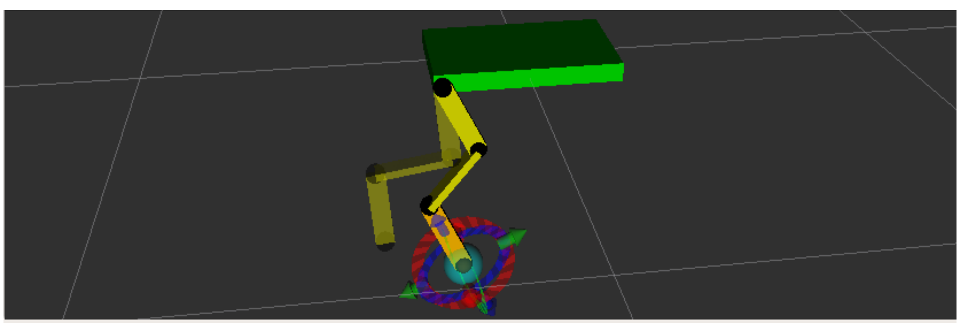
```
deeya@vunit:~/catkin_ws$ roslaunch udm_project_moveitconfig move_group.launch
```

d) Launch direct.launch

```
deeya@vunit:~/catkin_ws$ roslaunch udm_project_control direct.launch
```

e) Call rosservice

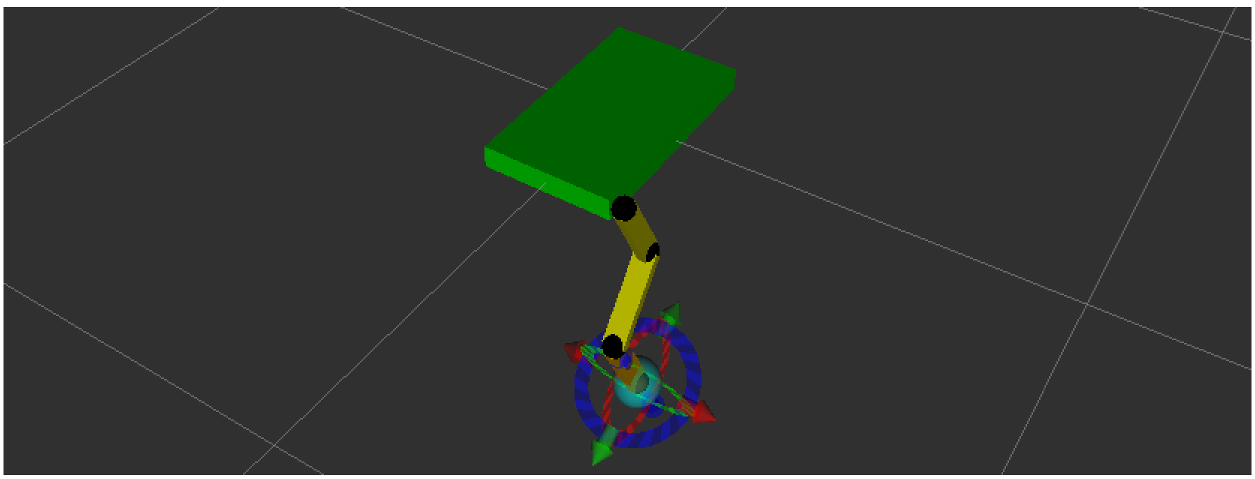
```
deeya@vunit:~/catkin_ws$ source devel/setup.bash
deeya@vunit:~/catkin_ws$ rosservice call /direct_kin_service_fl "joint1:
 data: 0.7100
joint2:
 data: 0.0
joint3:
 data: 0.0
joint4:
 data: 0.0"
res:
  data: True
message:
  data: "Success"
```



f) Launch indirect.launch

```
deeya@vunit:~/catkin_ws$ rosservice call /indirect_kin_service_fl "pose:
 orientation:
  w: 2
 position:
  x: 0.5
 position:
  y: 0.2
 position:
  z: -0.1"
res:
  data: True
message:
  data: "Success"
```

g) Call rosservice



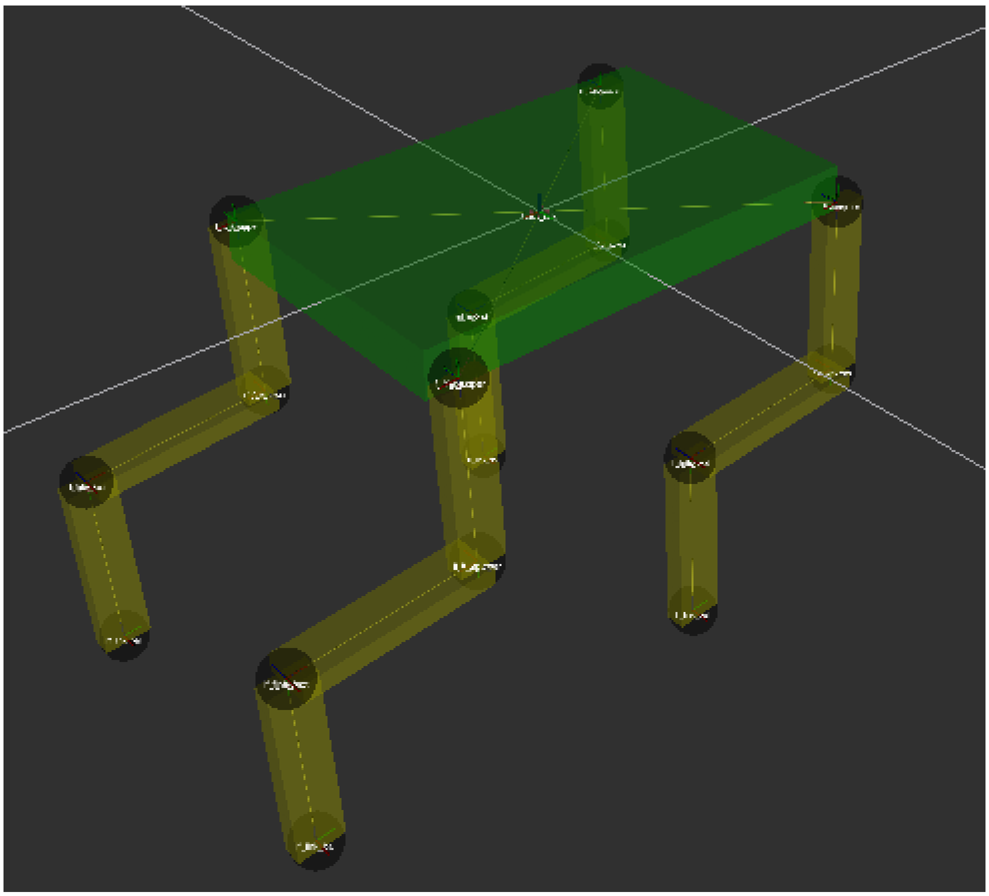
Partie 2 Ajout des autres jambes

1. A partir de l’urdf créé à la partie 1, créer un nouvelle urdf où vous ajouterez le reste des jambes de votre robot.

a) Launch RVIZ.

```
deeya@vunit:~/catkin_ws$ source devel/setup.bash
deeya@vunit:~/catkin_ws$ roslaunch urdf_oneleg launch.launch
```

b) Results



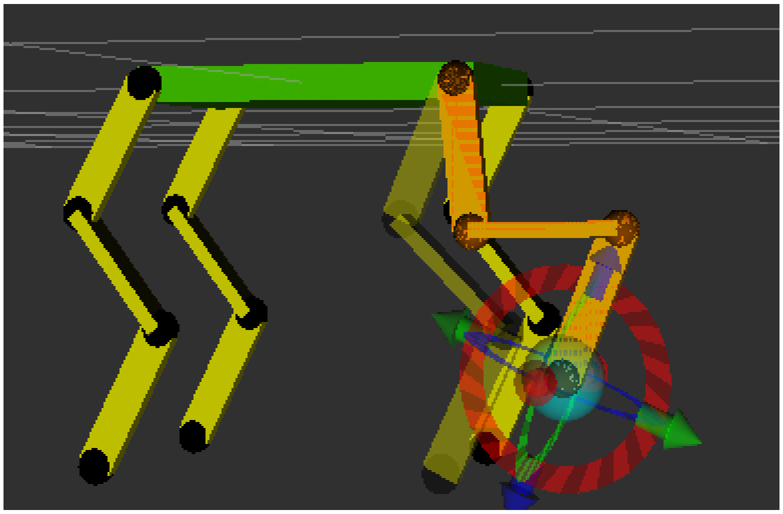
c) Create udm_project_moveitconfig

d) Open moveit_setup_assistant

```
deeya@vunit:~/catkin_ws$ roslaunch moveit_setup_assistant setup_assistant.launch
```

e) Launch rviz

```
deeya@vunit:~/catkin_ws$ roslaunch udm_project_moveitconfig demo.launch
```



2. Dans udm_project_control créer un service permettant de déplacer le robot en ligne droite, à gauche, à droite ou en arrière.

a) Launch demo.launch

```
deeya@vunit:~/catkin_ws$ roslaunch udm_project_moveitconfig demo.launch
```

b) Launch move_group.launch

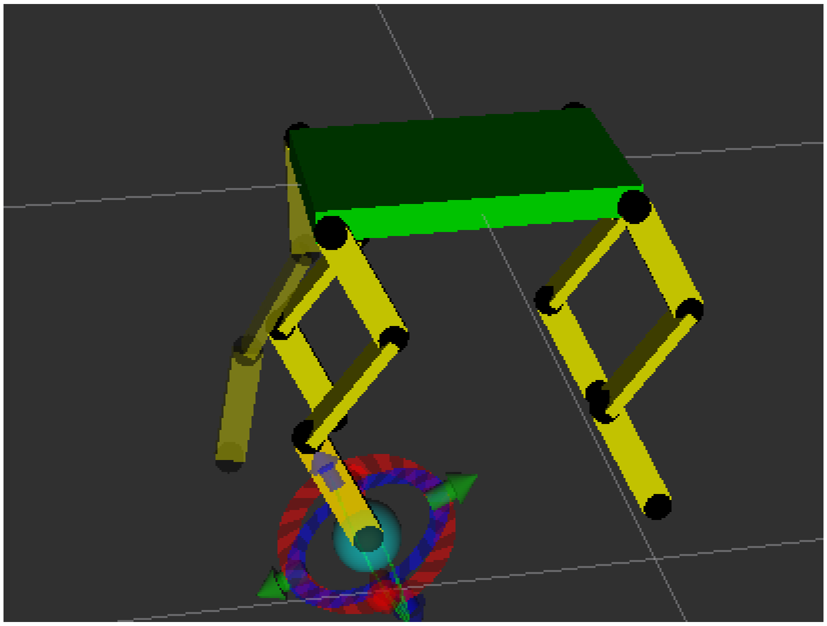
```
deeya@vunit:~/catkin_ws$ roslaunch udm_project_moveitconfig move_group.launch
```

c) Launch service.launch

```
deeya@vunit:~/catkin_ws$ roslaunch udm_project_control service.launch
```

d) Call rosservice

```
deeya@vunit:~/catkin_ws$ rosservice call /kin_service "movement:
 data: 'front'"
res:
  data: True
message:
  data: "Success"
```



```
deeya@vunit:~/catkin_ws$ rosservice call /kin_service "movement:
 data: 'back'"
res:
  data: True
message:
  data: "Success"
deeya@vunit:~/catkin_ws$
```

```
deeya@vunit:~/catkin_ws$ rosservice call /kin_service "movement:
 data: 'right'"
res:
  data: True
message:
  data: "Success"
```

Partie 3 Impact des configurations

En naviguant dans les fichiers de config du package créé avec moveit setup assistant, dans le dossier config se trouve des fichier de configuration.

Modifier ces fichiers config pour :

• Changer de solveur (en plus de ceux proposé, vous ajouterez aussi le IKFast solver)

• Changer les paramètres pour la planification de trajectoire

a) Change kinematics_solver

```
fl:
  kinematics_solver: lma_kinematics_plugin/LMAKinematicsPlugin
  kinematics_solver_search_resolution: 0.005
  kinematics_solver_timeout: 0.005
end_fl:
  kinematics_solver: srv_kinematics_plugin/SrvKinematicsPlugin
  kinematics_solver_search_resolution: 0.005
  kinematics_solver_timeout: 0.005
```

b) Change resolution and timeout

```
fl:
  kinematics_solver: lma_kinematics_plugin/LMAKinematicsPlugin
  kinematics_solver_search_resolution: 0.005
  kinematics_solver_timeout: 0.005
end_fl:
  kinematics_solver: srv_kinematics_plugin/SrvKinematicsPlugin
  kinematics_solver_search_resolution: 0.005
  kinematics_solver_timeout: 0.005
```

c) Interpretation

When changing solver, they both seem to obey joint limit found in the URDF file.

The kinematics_solver_search_resolution, state the resolution that a solver might use to search over the redundant space for inverse kinematics.

The kinematics_solver_timeout increase the time for each internal iteration that the solver performs.