Notes ROS - Vrac

Conception d'un robot amphibie à pattes bio-inspirées

A. Prince BENGUET

1 Notes sur les Ressources ROS et Simulateurs

Ressources Générales

- **Joints URDF** : Pour plus de détails sur les attributs des joints URDF, consultez la documentation officielle http://wiki.ros.org/urdf/XML/joint#Attributes.
- **Hydrodynamics Plugin (Gazebo)** : Tutoriel sur les plugins d'hydrodynamique dans Gazebo disponible ici https://classic.gazebosim.org/tutorials?tut=hydrodynamics&cat=plugins.
- **Tutoriels vidéo sur la flottabilité et l'hydrodynamique** :
  - Vidéo - Flottabilité -https://www.youtube.com/watch?v=NgmvhSEM5SQ
  - Tutoriel sur la flottabilité - The Construct : https://www.theconstruct.ai/ros-qa-buoyancy-neutral-objec
  - Partie 2 : https://www.theconstruct.ai/ros-qa-066-buoyancy-neutral-object-goes-hydrodynamics-pl

Simulateurs sous-marins

- **UUV Simulator** : Disponible sur GitHub https://github.com/uuvsimulator
- **Free Floating Gazebo world** : https://github.com/freefloating-gazebo
- **BlueROV2** : Modèle et fichiers pour le BlueROV2 https://github.com/fredvaz/bluerov2
- **DAVE Environment Simulator** :
  - GitHub : https://github.com/Field-Robotics-Lab/dave.git
  - Documentation du courant océanique (**Voir comment utiliser less services ROS pour avoir l'effet du courant**) : https://field-robotics-lab.github.io/dave.doc/contents/dave_env/Ocean-Current/
  - Tutoriel vidéo : https://www.youtube.com/watch?v=5HixvTjXzsg
- **UWSim** : Simulateur sous-marin pour la recherche en robotique marine
  - Site officiel : http://www.irs.uji.es/uwsim/
  - Installation d'UWSim : http://www.irs.uji.es/uwsim/wiki/index.php?title=Installing_UWSim

Quelques erreurs rencontrées : Python et ROS

- **Erreur ImportError dans ROS** : Solution possible disponible sur https://answers.ros.org/question/326226/importerror-dynamic-module-does-not-define-module-export-function-pyinit__tf2/
- **Utilisation de Python 3 avec ROS Melodic** : Voir la réponse ici https://answers.ros.org/question/295012/how-can-i-install-ros-melodic-with-python3/
- Pour définir la version de Python utilisée par ROS, ajoutez ceci dans votre fichier `.bashrc` :

```
echo "export ROS_PYTHON_VERSION=3" >> ~/.bashrc
source ~/.bashrc
```

Robots

- **Robot Bipède Cassie** : Documentation et tutoriels :
  - Documentation : https://github.com/jpreher/cassie_documentation
  - Tous les dépôts : https://github.com/jpreher
  - Tutoriel vidéo : https://www.theconstruct.ai/ros-qa-166-spawn-cassie-robot-gazebo/
- **Lancer plusieurs robots (exemples)** : Exécuter plusieurs robots avec des espaces de noms (définis par "ns" dans un fichier .launch) :
- `roslaunch my_package launch_multiple_robots.launch ns:=robot1 config:=robot1_rviz_config.rviz state_publisher:=True`
- `roslaunch my_package launch_multiple_robots.launch ns:=robot2 config:=robot2_rviz_config.rviz state_publisher:=True`

Commandes

- **Rendre un script exécutable** :

  ```
  chmod +x publish_world_models.py
  ```

  Vous pouvez également rendre tous les fichiers du répertoire exécutable avec :

  ```
  chmod -R +x .
  ```

Exemple de fichier URDF et de lancement

- Tutoriel vidéo pour créer un fichier de lancement URDF : https://www.theconstruct.ai/ros-qa-142-how-to-create-launch-file-for-urdf-and-open-in-gazebo/

MoveIt et bras robotique

- Tutoriel vidéo sur MoveIt : https://www.youtube.com/watch?v=DZB5_4JCS0A&list=PLeEzO_sX5H6TBD6EMGgV-qdhzxPY19m12&index=12
- Exemple de configuration MoveIt pour un bras robotique : https://github.com/ageofrobotics/import_your_custom_urdf_package_to_ROS-main/blob/2e713d1acf99981a315667f32bbb82ab184ffcfe/robot_arm_urdf/config/joint_trajectory_controller.yaml

2 Notes du 02/07/2024 au 19/07/2024

02/07/2024

Transformations TF

— `rosrun tf tf_echo /base_link /Link_Hiaw_D`
— `rosrun tf tf_echo /base_link /link1`
— `rosrun tf tf_monitor [source_frame] [target_frame]`
— `rosrun tf view_frames`
— `rosrun tf static_transform_publisher [x] [y] [z] [qx] [qy] [qz] [qw] [frame_id] [child_frame_id] [period_in_ms]`
  — Exemple : `rosrun tf static_transform_publisher 1 0 0 0 0 0 1 /world /base_link 100`
— `rosrun rqt_tf_tree rqt_tf_tree`
— `rosrun rqt_console rqt_console`
— `rosrun tf tf_remap [old_frame] [new_frame]`

Contrôleurs dans Gazebo

— **Référence** : *Mastering ROS for Robot Programming*, page 109
— Ajout des balises `<transmission>` pour activer le modèle.
— Ajout du plugin `gazebo_ros_control`.
— Utilisation des contrôleurs ROS pour déplacer les joints du robot dans Gazebo (page 115).
— **Note importante** (page 117) : Mention des paquets de navigation et MoveIt qui peuvent donner l'objectif aux contrôleurs de robots mobiles et de bras robotiques.
— Les interfaces matérielles représentent le robot et ses composants (capteurs, actionneurs, etc.).

04/07/2024

Contrôleurs pour bras robotique (ROS/Gazebo)

— Tutoriel vidéo sur YouTube : https://www.youtube.com/watch?v=nb1fz3ePkyM
— Suite de la vidéo : https://www.youtube.com/watch?v=x-Yqo4yNsFY

Services ROS

— Appeler un service : `rosservice call /Ros2_RobotBipede/set_volume_scaling "value: 1.2"`
— Lister les services disponibles : `rosservice list`
— Obtenir des informations sur un service : `rosservice info /Ros2_RobotBipede/set_use_global_current_velocity`

ROSParam

— Lister les paramètres disponibles : `rosparam list`

Commander les joints via les contrôleurs

— Identifier les topics avec : `rostopic list` pour repérer les topics du type `/Ros2_RobotBioint2_position_controller/command`.
— Publier une commande sur un topic :

```
rostopic pub /Ros2_RobotBioint2_position_controller/command std_msgs/Float64 1.0
```

08/07/2024

Modifications dans CMakeLists.txt pour Gazebo (Sauf évolution dans les fichiers)

```
find_package(catkin REQUIRED COMPONENTS
  gazebo_ros
  rospy
  roscpp
  std_msgs
  urdf
  controller_manager
  joint_state_controller
  robot_state_publisher)
```

Modifications dans package.xml pour Gazebo

```
<build_depend>controller_manager</build_depend>
<build_depend>joint_state_controller</build_depend>
<build_depend>robot_state_publisher</build_depend>
<build_depend>rospy</build_depend>
<build_depend>roscpp</build_depend>
<build_depend>std_msgs</build_depend>
<build_depend>urdf</build_depend>
```

ROS Control

— Tutoriel vidéo sur la publication des données des joints : https://www.youtube.com/watch?v=qPk9GaixLBo

— Étapes pour configurer des contrôleurs dans ROS : https://www.youtube.com/watch?v=ibUCPyMkPvU

— Dépôt GitHub pour le contrôleur : https://github.com/LearnRoboticsWROS/cobot

MoveIt

— Tutoriel vidéo sur l'utilisation de MoveIt avec Python : https://www.youtube.com/watch?v=DZB5_4JCS0A

— Exemple de configuration MoveIt : https://github.com/ageofrobotics/import_your_custom_urdf_package_to_ROS-main/blob/main/movit_robot_arm_sim/scripts/node_set_predefined_pose.py

— **Note** : Respecter l'indentation dans les fichiers YAML, utiliser deux espaces au lieu de TAB (référence : *Importing_URDF_Package_fromSolidworksInROS_and_MOVEIT de "Age of Robotics"* lien , page 23).

Assistant MoveIt

— Lancer l'assistant MoveIt :

```
roslaunch moveit_setup_assistant setup_assistant.launch
```

— **Erreur courante** : Ne pas définir les limites des "efforts" et "vitesses" à 0 dans le modèle URDF (au niveau des différents "Joints" : balises "effort" - "velocity"). Sinon, les articulations seront immobilisées.

Commandes pour le bras robotique avec Python

— Tutoriel vidéo sur la commande du bras avec un nœud Python : https://github.com/ageofrobotics/control_your_custom_robotic_arm_with_python/blob/main/movit_robot_arm_sim/scripts/node_set_predefined_pose.py

Attention lors des modifications du modèle après l'utilisation de MoveIt

— Le nom du modèle (cité à l'intérieur des documents) dans les fichiers URDF ou XACRO doit correspondre à celui du fichier SRDF fourni par MoveIt.

— Vérifier les fichiers de lancement MoveIt pour les noms de modèles utilisés.

15-19 Juillet 2024

Vérification de l'intégration ROS-Gazebo

— Vérifier que le service `/gazebo/set_physics_properties` est disponible pour confirmer la synchronisation entre ROS et Gazebo.

Référence : *Mastering ROS for Robot Programming*, chapitre 6, page 191.

— Si `MoveItSimpleControllerManager` se connecte correctement à Gazebo, la planification de mouvement sera réussie. Sinon, MoveIt ne transmettra pas la trajectoire à Gazebo.

Erreur Python avec ROS

— Erreur d'importation : `ImportError: dynamic module does not define module export function (PyInit_moveit_roscpp_initializer)`.

— SOLUTION UTILISÉE : Utiliser `!/usr/bin/env python2.7` comme shebang (sur la première ligne du fichier problématique) pour ROS Melodic qui utilise Python 2.7.

Changer la version de Python par défaut sur Ubuntu

— Utiliser l'utilitaire `update-alternatives` pour configurer Python 2.7 comme version par défaut :

```
sudo update-alternatives --install /usr/bin/python python /usr/bin/python2.7 1
sudo update-alternatives --install /usr/bin/python python /usr/bin/python3.8 2
sudo update-alternatives --config python
```