# 脚ロボット NMPC-WBC パッケージ「legged_control」の導入結果

報告者：宮本　指導：小長谷

1. build について

**catkin build 安易にしない→依存関係にない** ocs2がエラーでうまくビルドできない。

・基本的にはチュートリアルの通りに進めることでビルドは成功する。

・LCM を使うため、以下の apt が必要がある。

sudo apt install liblcm-dev

##Flow

# Clone legged_control
git clone git@github.com:qiayuanliao/legged_control.git

# Clone OCS2
git clone git@github.com:leggedrobotics/ocs2.git

# Clone pinocchio
git clone --recurse-submodules https://github.com/leggedrobotics/pinocchio.git

# Clone hpp-fcl
git clone --recurse-submodules https://github.com/leggedrobotics/hpp-fcl.git

# Clone ocs2_robotic_assets
git clone https://github.com/leggedrobotics/ocs2_robotic_assets.git

# Install dependencies
sudo apt install liburdfdom-dev liboctomap-dev libassimp-dev
catkin config -DCMAKE_BUILD_TYPE=RelWithDebInfo
catkin build ocs2_legged_robot_ros ocs2_self_collision_visualization

##Build
#変更の際にはこのパッケージ内をビルドする。

catkin build legged_controllers legged_unitree_description
catkin build legged_gazebo
catkin build legged_unitree_hw

##Quick Start
export ROBOT_TYPE=go1(使うロボットで変更「a1, aliengo, custom」)

#シミュレーション起動
roslaunch legged_unitree_description empty_world.launch

#実機を動かしたい時だけ以下を実行
roslaunch legged_unitree_hw legged_unitree_hw.launch

#load controller←初回のみ、時間がかかる。
roslaunch legged_controllers load_controller.launch cheater:=false

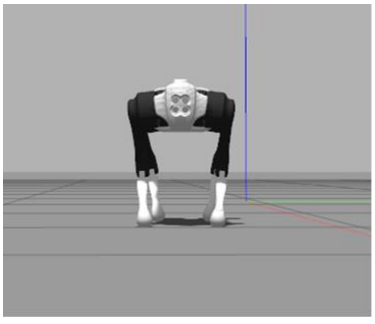
#サービス通信の開始
rosservice call /controller_manager/switch_controller "start_controllers:
['controllers/legged_controller']
stop_controllers: ['']
strictness: 0
start_asap: false
timeout: 0.0"

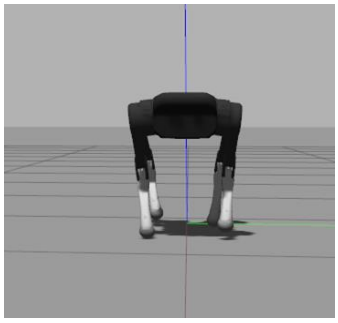
＃各自で速度指令を送るパッケージ~~を作成し、実行~~→Default のパッケージでも対応可能
roslaunch legged_controllers jjoy_teleop.launch

2. シミュレーション実験

・実行前注意
go1足初期角度違い↔issue より直せた
→スポーンの方法（落下）でおかしくなる。
→issue 内容
https://github.com/qiayuanl/legged_control/blob/master/legged_examples/legged_unitree/legged_unitree_description/urdf/common/leg.xacro#L189-L190 **from 0.6 to 0.2**
→この直し方は良くない（粘性係数をいじってる）。
→シミュレーションの初期姿勢は変でも動作した。

・結果
a1, go1　両機で速度指令動作ができた。
両機とも sdk に比べ、内股気味である。





3. 実機動作実験

・go1実機で動かした際に、傾いていってしまう。動画は後日参照先を設定

→初期エラー（WARN）を消してもX

→パラメータ調整の必要性（github のフォーク先の開発状況ではそれぞれ行っていた）

→パラメータ数、条件等から現状の適用は厳しい。

→MPC の学習を深めたあとでないと自作機に対する実用化は厳しいと考えられる。

4. その他注意

・速度指令,ポジション指令項

legged_controllers/include/TargetTrajectoriesPublisher.h#81-82

そのトピック名を組み合わせる ROS パッケージに合わせて「/cmd_vel , /move_base_simple/goal」を変更

・Docker 化

local で build して作成する。Docker 内だと初回 build で起動に時間がかかりすぎてしまう。Gazebo の動作が遅くなることも問題である。

**Dockerfile**

```
//このしたから
FROM osrf/ros:noetic-desktop-full
RUN apt-get update && apt-get install -y ¥
      git ¥
      libeigen3-dev ¥
      liburdfdom-dev ¥
      liboctomap-dev ¥
      libassimp-dev ¥
      liblcm-dev ¥
      python3-catkin-tools ¥
      python3-osrf-pycommon ¥
      ros-noetic-joy ¥
      ros-noetic-joy-teleop ¥
    && apt-get clean ¥
    && rm -rf /var/lib/apt/lists/*
RUN mkdir -p /root/catkin_ws/src
RUN cd /root/catkin_ws/src && ¥
    git clone https://github.com/qiayuanl/legged_control.git && ¥
    git clone https://github.com/leggedrobotics/ocs2.git && ¥
    git clone --recurse-submodules https://github.com/leggedrobotics/pinocchio.git && ¥
    git clone --recurse-submodules https://github.com/leggedrobotics/hpp-fcl.git && ¥
    git clone https://github.com/leggedrobotics/ocs2_robotic_assets.git
RUN . /opt/ros/noetic/setup.sh && ¥
    cd /root/catkin_ws && ¥
```

```
    catkin config -DCMAKE_BUILD_TYPE=RelWithDebInfo && ¥
    catkin build ocs2_legged_robot_ros ocs2_self_collision_visualization
RUN . /root/catkin_ws/devel/setup.sh && ¥
    cd /root/catkin_ws && ¥
    catkin build legged_controllers legged_unitree_description && ¥
    catkin build legged_gazebo && ¥
    catkin build legged_unitree_hw
RUN echo "source /opt/ros/noetic/setup.bash" >> /root/.bashrc && ¥
    echo "source /root/catkin_ws/devel/setup.bash" >> /root/.bashrc && ¥
    echo "export ROBOT_TYPE=go1" >> /root/.bashrc
ENV DEBIAN_FRONTEND=noninteractive
RUN apt-get update && apt-get install -y ¥
      vim-gtk3 ¥
      terminator ¥
    && apt-get clean ¥
    && rm -rf /var/lib/apt/lists/*
//ここまで
```

**docker-compose.yaml**

```
//このしたから
version: "3.7"

services:
  ros:
    build:
      context: .
      dockerfile: Dockerfile
    network_mode: host
    working_dir: /root
    tty: true
    environment:
      - DISPLAY=$DISPLAY
    privileged: true
    devices:
- /dev/input/js0:/dev/input/js0:mwr
//ここまで
```