



ROBOT SYSTEM DESIGN
LABORATORY

SEED-Noïd における 双腕作業のための RTC 群

名城大学

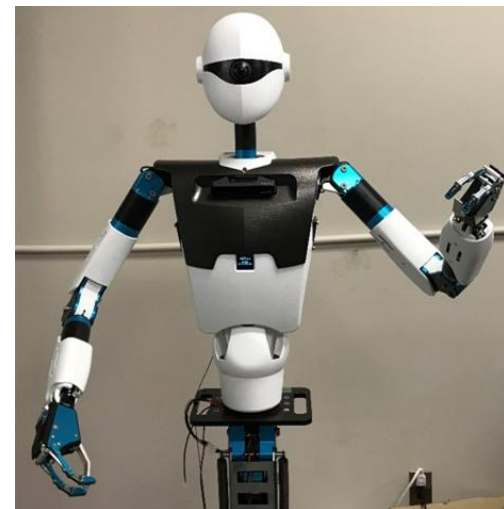
四位菜祐果 真崎聡士 大原賢一

作品概要

双腕ロボットに関する研究は広く行われている中、RTミドルウェアには、**双腕ロボット制御機能共通インターフェース**(以下、**双腕共通I/F**)というリソースがあるが、適用事例が少ないのが現状



双腕共通I/FをTHK(株)のSEED-Noidに適用した



THK(株) SEED-Noid

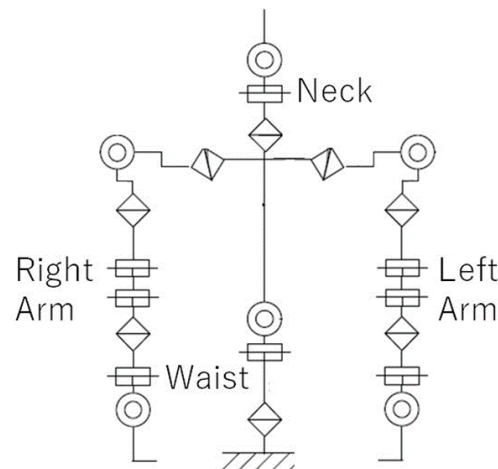
- ① 双腕共通I/Fを改良した双腕共通I/F2.0を提案を行う
- ② 双腕共通I/F2.0をTHK(株)のSEED-Noidに適用した

利用するハードウェア

THK(株)が開発した双腕ロボット SEED-Noid



上半身の自由度



腰	: 3自由度
右腕	: 7自由度
左腕	: 7自由度
頭	: 3自由度

双腕共通I/Fとは

NEDO 次世代ロボット知能化技術開発プロジェクトによって作成
 双腕を持つロボットアームの制御に関わるインタフェースの共通仕様を定義

5.2 は MotionCommands

アプリケーションは、この「双腕ロボット制御モジュール」に共通の MotionCommands を提供することになる。

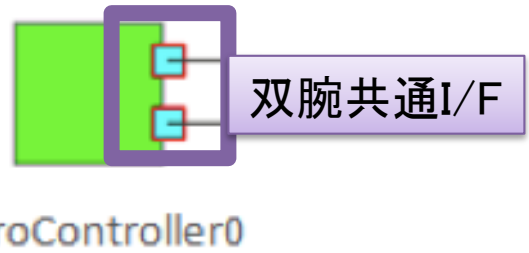
図 5.2 MotionCommands インタフェース

表 5.2 MotionCommands

パラメータ	データ型	説明
RobotSpeed	uint16_t	ロボットの速度を設定する。
RobotAngle	double	ロボットの関節角度を設定する。
RobotForce	double	ロボットの関節力矩を設定する。
RobotPose	double	ロボットの関節位置を設定する。
RobotSpeed	uint16_t	ロボットの速度を設定する。
RobotAngle	double	ロボットの関節角度を設定する。
RobotForce	double	ロボットの関節力矩を設定する。
RobotPose	double	ロボットの関節位置を設定する。

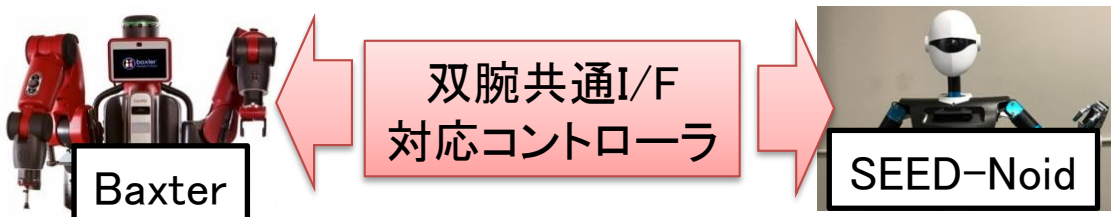
仕様書URL:
https://www.sec.co.jp/robot/_downloads/interface_doublearm_1.0.pdf

適用例：HIRO



出典:「頭部ステレオカメラを用いた双腕ロボットによるマニピュレーション作業」
http://openrtc.org/Task_Vision/systems/Hiro.html

メリット



ハードに依存せず再利用可能

双腕共通I/F2.0の提案

①関数の追加/修正

もともとあった関数

- ・各部サーボON/OFF
- ・双腕のグリッパ開閉/開度設定
- ・双腕の関節角度指定
- ・双腕の手先目標位置姿勢に直交空間で直線補間で動作 など

+

追加した関数

- ・双腕の相対手先位置姿勢関係の取得(未実装)
- ・双腕の手先目標位置姿勢に関節空間で直線補間で動作 など

②アーム共通I/Fの包括

双腕共通I/Fだけで、単腕も扱うためにアーム共通I/Fをインクルードした

双腕共通I/F2.0

双腕共通
I/F1.0

+

アーム
共通I/F
(右腕)

+

アーム
共通I/F
(左腕)



DualManipulatorCommonInterface_Common
DualManipulatorCommonInterface_Middle

URL : https://github.com/Mayuka-Shii/SEED-Noid_Dual-Arm_pkg/blob/master/interface_doublearm_2.0.pdf

仕様書を公開中

開発RTC紹介

双腕共通I/FでSEED-Noid実機を制御するRTC群の開発を行った

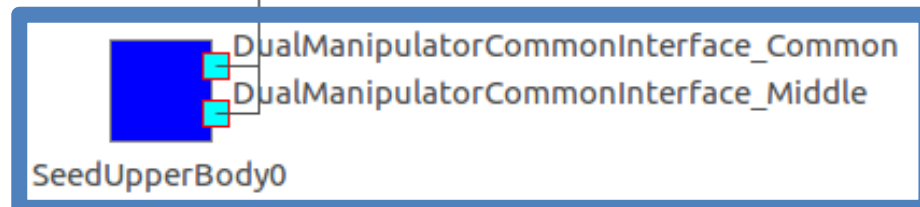


DualArmController

- ・双腕共通I/F2.0を用いたサンプルコントローラ

SeedUpperBody

- ・今回のメインRTC
- ・双腕共通I/F2.0によってSEED-Noid実機 of 双腕を制御することができる
- ・今後, コントローラ側を充実させ, 再利用予定



このRTC群でできること

- ・各部サーボON/OFF
- ・グリッパ開閉/開度設定
- ・関節角度指定
- ・手先目標位置姿勢に直交/関節空間で直線補間で動作

開発RTC紹介

RTC群デモ動画①

関節角度指定モードでSEED-Noi実機が動く様子

端末

```
sdlab@rsdlab: ~/workspace
etManipInfo
manufactur : Hardware: (株)THK Software: Meijo University robot systems design
laboratory
type       : Seed_RightArm
axisNum    : 7
cmdCycle   : 20
isGripper  : 1
uccess

etManipInfo
manufactur : Hardware: (株)THK Software: Meijo University robot systems design
laboratory
type       : Seed_LeftArm
axisNum    : 7
cmdCycle   : 20
isGripper  : 1
uccess

ct
pen COM Port:/dev/serial/by-id/usb-FTDI_TTL232R-3V3_FT98HKZC-if00-port0
nit position

rsdlab@rsdlab: ~/workspace/DualArmController/build/src
sdlab@rsdlab: ~/workspace/DualArmController/build/src$ ./DualArmControllerComp
lease Select Mode :)
-----
: movePTPJointAbs
: moveLinearCartesianAbs
: Gripper Open/Close
-----

```

SeedupperBody実行画面

RT System Editor - Eclipse SDK

クイック・アクセス

Java RT System Editor RTC Builder

*System Diagram

▼ rt localhost

▼ rsdlab|host_c

▼ DualArmCo

▼ SeedUpperE

DualManipulatorCommonInterface_Common

DualManipulatorCommonInterface_Middle

DualArmController0

DualManipulatorCommonInterface_Common

DualManipulatorCommonInterface_Middle

SeedUpperBody0

2018/12/17

DualArmController実行画面
関節角度指定モード

ROBOT SYSTEM DESIGN
LABORATORY

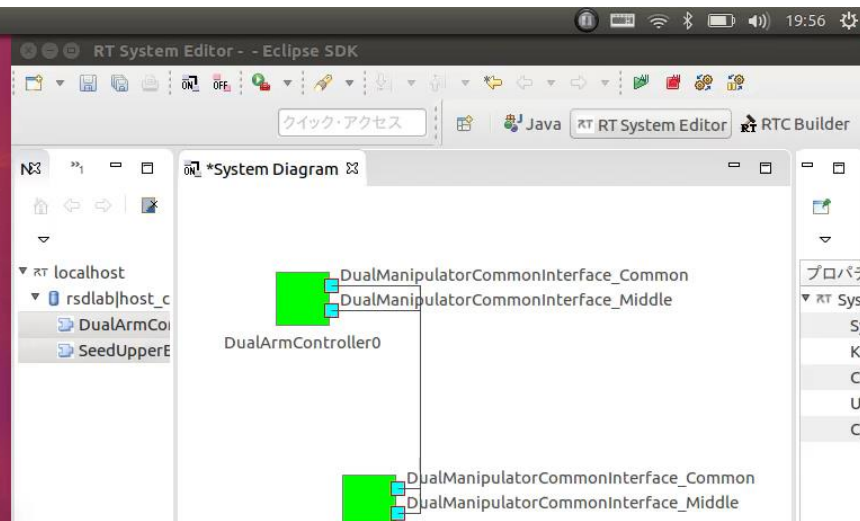
開発RTC紹介

RTC群デモ動画②

目標手先位置指定モードでSEED-Noi実機が動く様子

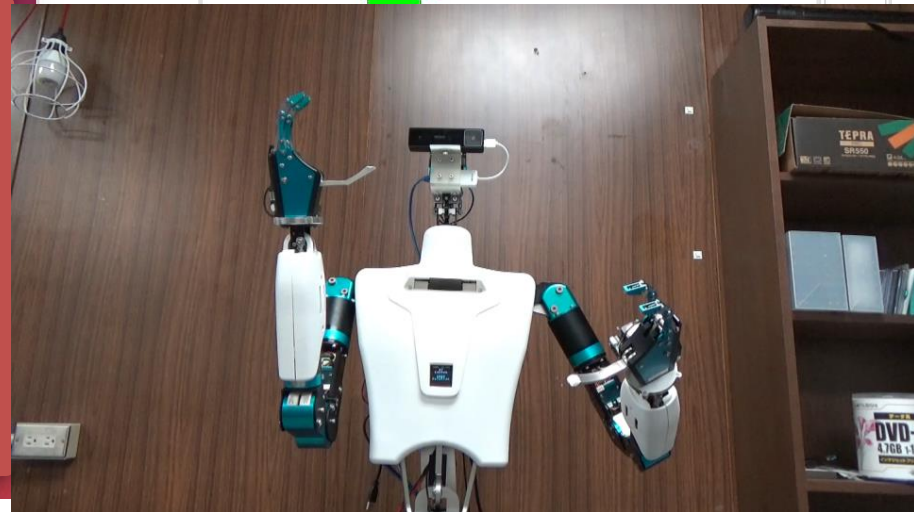
```
端末
Open COM Port:/dev/serial/b
init position
movePTPJointAbs
RightArmJointPos[0] = 0.0949927[°]
RightArmJointPos[1] = 0.0200412[°]
RightArmJointPos[2] = 0[°]
RightArmJointPos[3] = 179.752[°]
RightArmJointPos[4] = 0[°]
RightArmJointPos[5] = -0.155637[°]
RightArmJointPos[6] = 0[°]
LeftArmJointPos[0] = 0.0949927[°]
LeftArmJointPos[1] = 0.0209004[°]
LeftArmJointPos[2] = 0[°]
LeftArmJointPos[3] = 179.752[°]
LeftArmJointPos[4] = 0[°]
LeftArmJointPos[5] = -0.155637[°]
LeftArmJointPos[6] = 0[°]
1
2
3
4
Success
```

SeedupperBody実行画面



```
Left Arm
Joint 1
0
Joint 2
30
Joint 3
0
Joint 4
120
Joint 5
0
Joint 6
0
Joint 7
0
-----
Please Select Mode :)
-----
1 : movePTPJointAbs
2 : moveLinearCartesianAbs
3 : Gripper Open/Close
-----
```

DualArmController実行画面
目標手先位置姿勢指定モード



開発システム紹介

実機が無くてもロボットを動かせるようなシミュレーション環境の構築を行った

シミュレータの種類には
Choreonoid/V-REP/Gazeboなどたくさんあるが…



今回、SEED-NoidのモデルがROSパッケージにあったためROSとの連携が容易なGazeboを採用



RTMとROSを連携したシステムの構築を行った



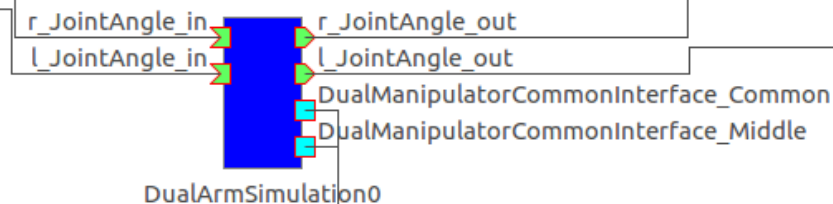
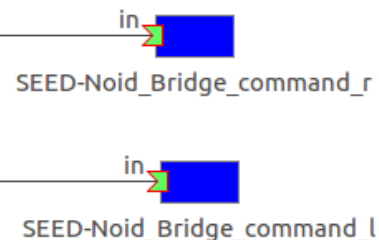
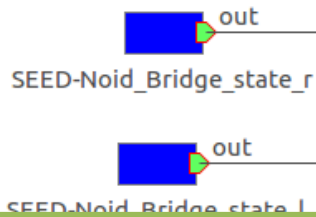
開発システム紹介

RTM

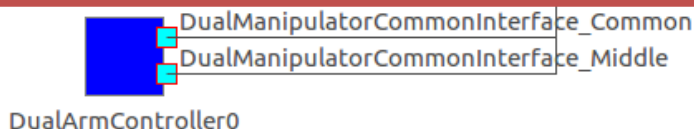
DualArmSimulation

- ・双腕共通I/F2.0を使用
- ・関節角度の読み取り/送信を行う

ROS側に目標
関節角度を出力する
ブリッジ群



ROS側から現在
関節角度を入力する
ブリッジ群



DualArmController

- ・双腕共通I/F2.0を用いた
サンプルコントローラ

現在使用できる機能

- ・関節角度指定
- ・手先目標位置姿勢に直交/関節空間で直線補間で動作

開発システム紹介

ROS

Gazebo

コントローラード

RTM側からの目標
関節角度を入力する
ブリッジ群

RTM側に現在
関節角度を出力する
ブリッジ群

in
SEED-Noid_Bridge_command_r

in
SEED-Noid_Bridge_command_l

arm_controller

larm_controller

out
SEED-Noid_Bridge_state_r

out
SEED-Noid_Bridge_state_l

既存システム

RTMとROSのブリッジ

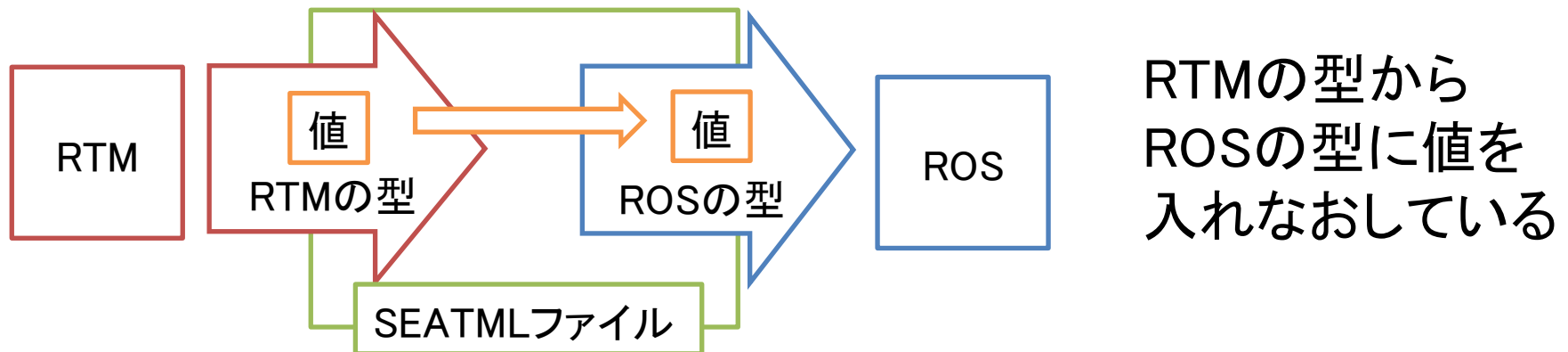
今回、eSEATによって動作するSEATMLファイルによって、RTMとROSのブリッジを作成した

eSEATとは

産総研の原氏が開発したソフトウェア

OpenRTM-aistのRTC、ROSノード、GUIパネル、Webサーバーとして動作

ブリッジの動作原理



eSEATを用いるとROSとRTMの連携が容易にできる

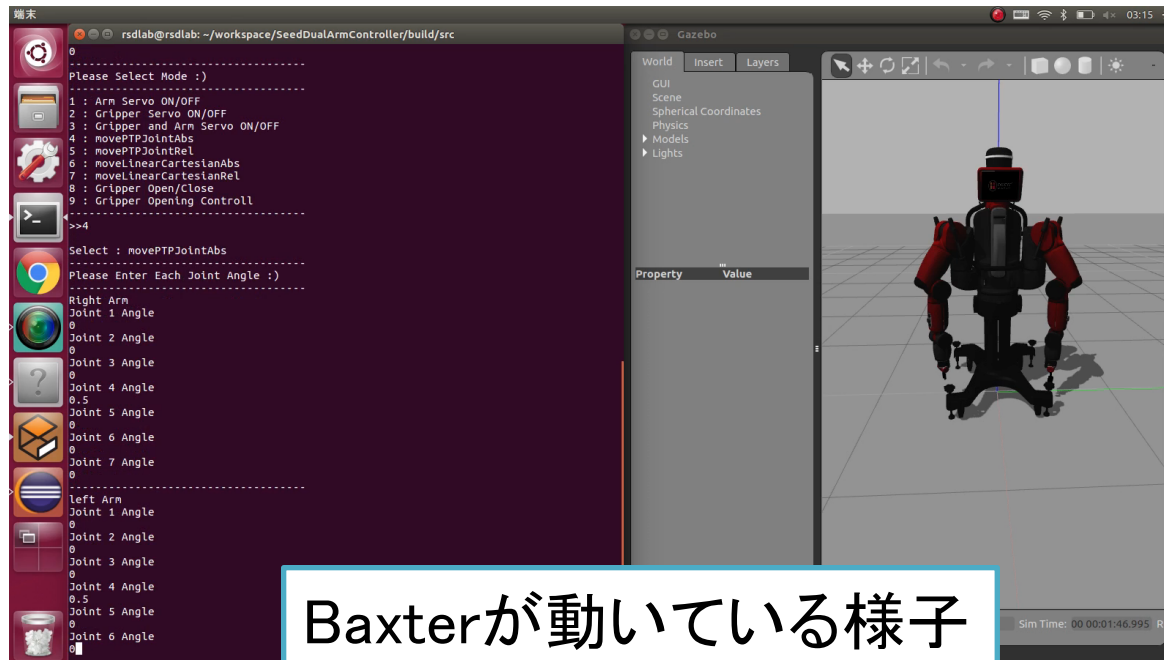
RTMとROSのブリッジ

URL : https://github.com/Mayuka-Shii/SEED-Noid_Dual-Arm_pkg/

公開中のSEATMLファイル

- ・SEED-Noidの関節角度
- ・Baxterの関節角度
- ・NEXTAGEの関節角度

RTMとROSのブリッジ作成
マニュアルを公開予定

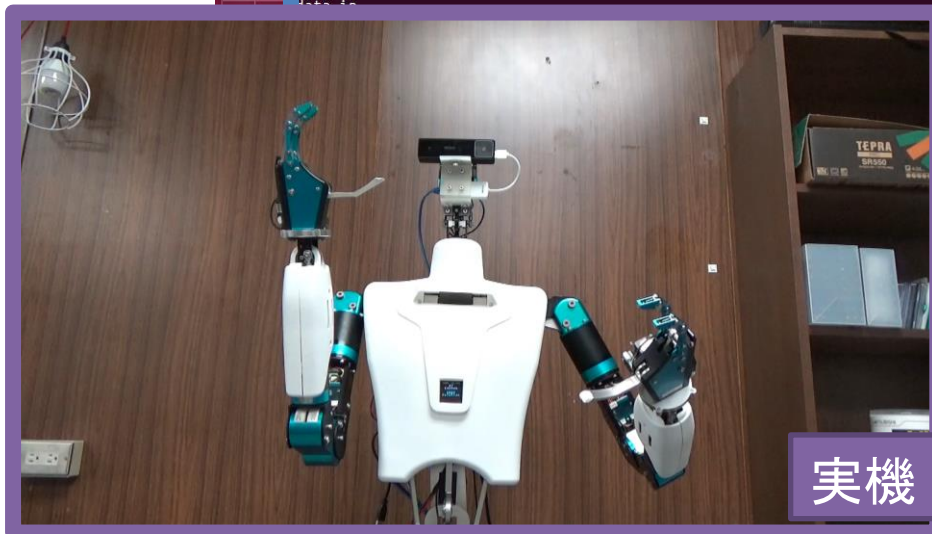
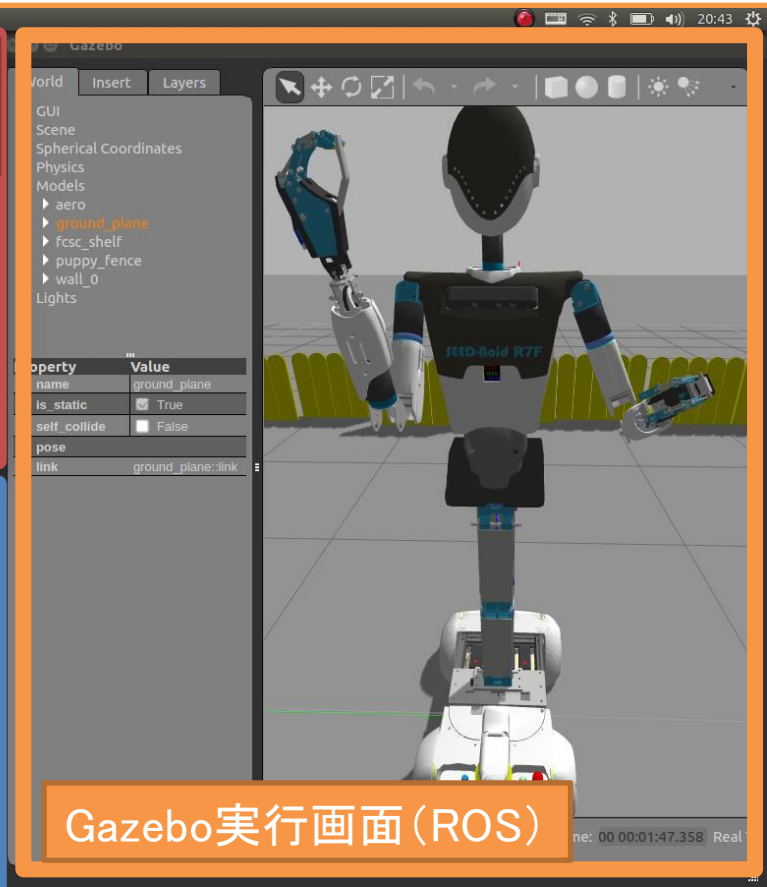
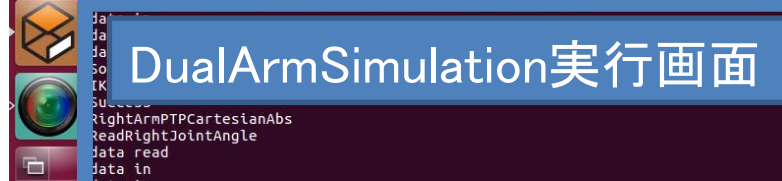


Baxterが動いている様子

開発システム紹介

RTC群デモ動画③

目標手先位置指定モードでシミュレーション上のSEED-Noïdが動く様子



DualArmControllerから
同じ値を指定した
場合の実機の動き

まとめ

- 双腕共通I/F2.0をTHK(株)のSEED-NoIdに適用するとともに, 実機と同様に動作するシミュレーション環境を構築した
- 双腕共通I/Fに関数を追加するなどした双腕共通I/F2.0を提案を行った
- RTMとROSのブリッジとしてeSEATが有用であるという紹介を行った





ご清聴ありがとうございました

eSEATを開発されました原様に御礼申し上げます

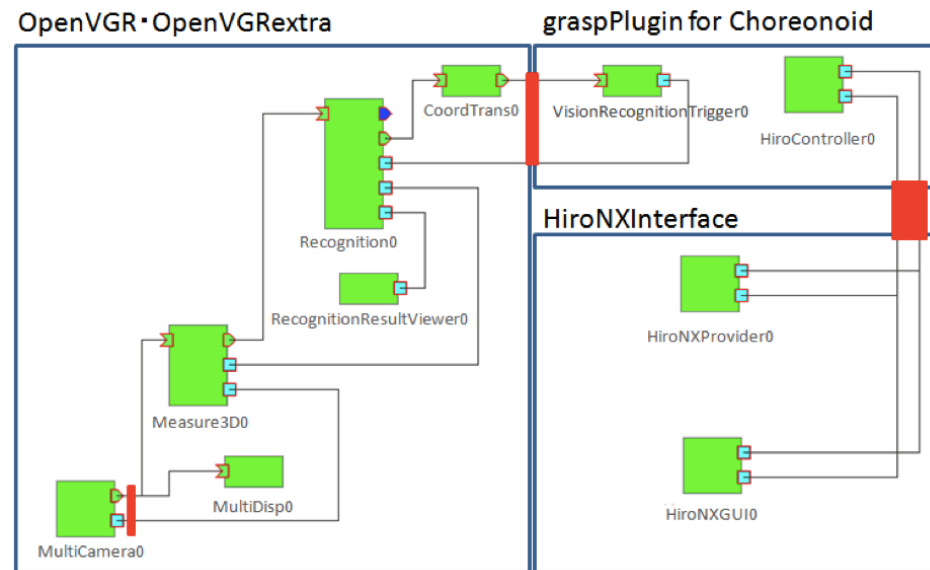
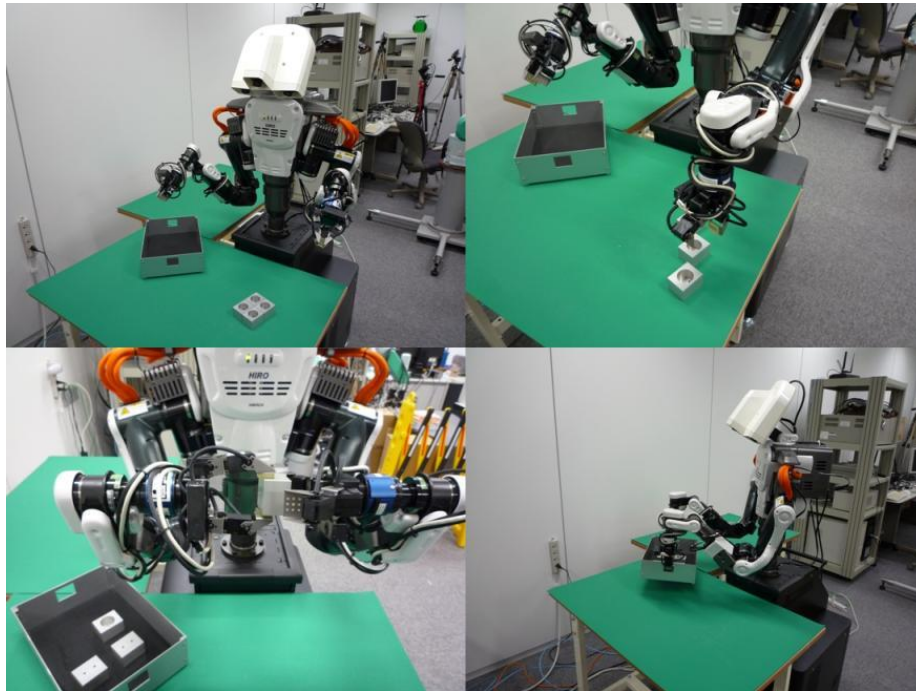


予備スライド



頭部ステレオカメラを用いた双腕ロボットによるマニピュレーション作業

作業台に置かれた部品を状況に応じて再配置し、個々の部品を検出して 双腕を活かして箱に整理して入れ、双腕で別の場所に運ぶPick & Place作業を行う



出典:「頭部ステレオカメラを用いた双腕ロボットによるマニピュレーション作業」

URL: http://openrtc.org/Task_Vision/systems/Hiro.html

双腕共通I/F1.0

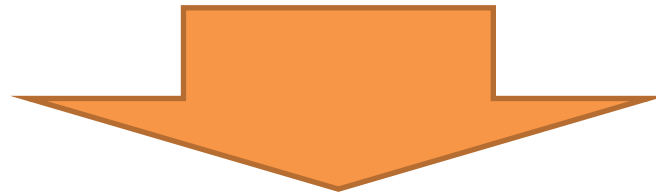
2.0で大きく修正した点

closeGripper()	グリッパを閉じる
moveGripper(r_angle,l_angle)	グリッパを指定開度を開く
moveLinearCartesianAbs(rArm,lArm)	ロボット座標系の絶対値で指定された目標位置に対し、直交空間における直線補間で動作
moveLinearCartesianRel(rArm,lArm)	ロボット座標系の相対値で指定された目標位置に対し、直交空間における直線補間で動作
movePTPJointAbs(JointPoints)	絶対関節座標で指定された目標位置に対し、関節空間における直線補間で動作
movePTPJointRel(JointPoints)	絶対関節座標で指定された目標位置に対し、関節空間における関節補間で動作
movePTPJointAbsSeq (JointPointsSeq)	両方の腕に対して、絶対関節座標で指定された目標位置に対し、関節空間における直線補間で動作
openGripper()	グリッパを開く
setSpeedCartesian(spdratio)	直交空間における動作時の速度を%指定
setSpeedJoint(spdratio)	直交空間における動作時の速度を%指定



双腕共通I/F2.0

movePTPJointAbs(JointPoints)	絶対関節座標で指定された目標位置に対し、関節空間における直線補間で動作
movePTPJointRel(JointPoints)	絶対関節座標で指定された目標位置に対し、関節空間における関節補間で動作
movePTPJointAbsSeq (JointPointsSeq)	両方の腕に対して、絶対関節座標で指定された目標位置に対し、関節空間における直線補間で動作



movePTPJointAbs(rArm,lArm)	絶対関節座標で指定された目標位置に対し、関節空間における直線補間で動作
movePTPJointRel(rArm,lArm)	絶対関節座標で指定された目標位置に対し、関節空間における関節補間で動作

双腕共通I/F2.0

最低限必要だと思われる機能

<code>movePTPCartesianAbs(rArm,lArm)</code>	ロボット座標系の絶対値で指定された目標位置に対し、関節空間における直線補間で動作
<code>movePTPCartesianRel(rArm,lArm)</code>	ロボット座標系の絶対値で指定された目標位置に対し、関節空間における直線補間で動作



直交空間における直線補間で動作だけでは不十分である

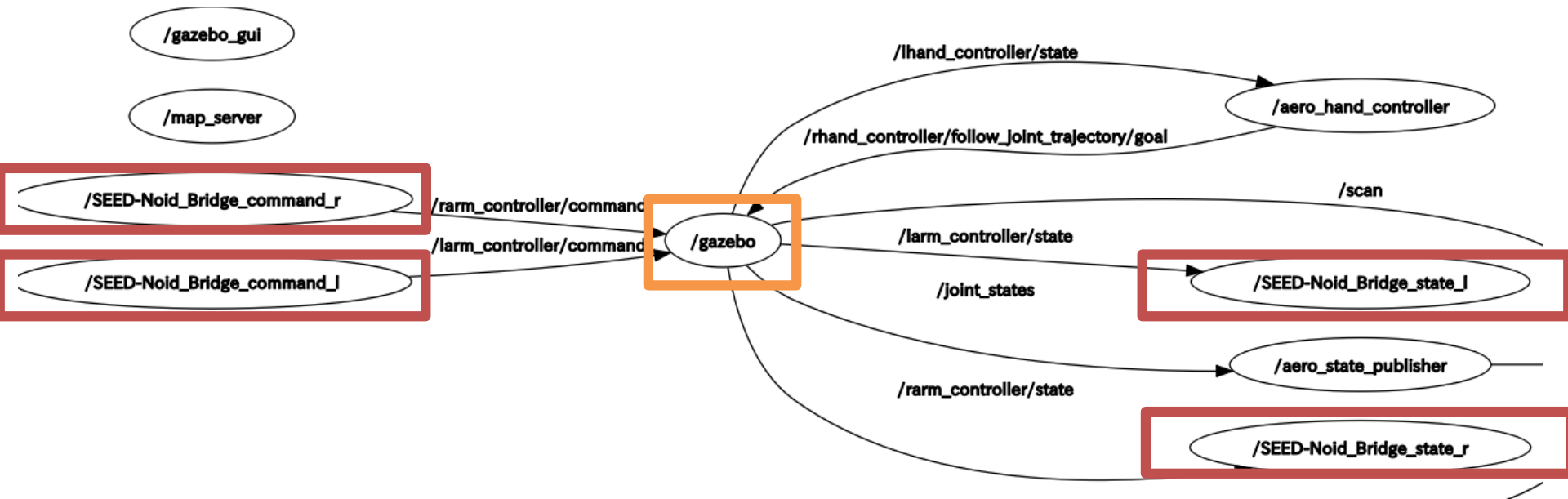
双腕を制御する際にあると便利だと思われる機能

<code>getRelativePosition(rArm,lArm)</code>	双腕の相対位置姿勢関係の取得
---	----------------



決まった位置姿勢関係で固定したい場合などに有用

ROS側 rqt_graph



アダプター

RTM側のアダプター

```
<seaml>
  <general name="SEED-NoId_Bridge_command_r">
    <adaptor name="in" type="rtcin" datatype="TimedFloatSeq" />
    <adaptor name="/rarm_controller/command" type="ros_pub" datatype="trajectory_msgs/JointTrajectory" size="10"/>
  </general>
```

ポート名 inかoutか? データ型

トピック名 pubかsubか? メッセージ

ROS側のアダプター