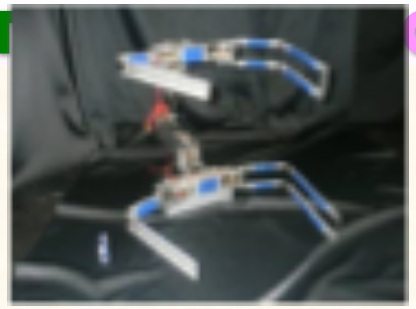
有用なRTCの紹介

菅　佑樹（早大，SSR）

自己紹介

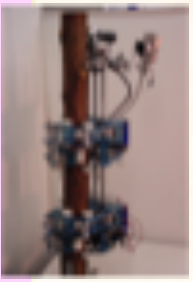
- 菅　佑樹 @ysuga
  - 2012～ 株式会社SUGAR SWEET ROBOTICS代表取締役
  - 2010～2012 株式会社リバスト
  - 2007～2010 早稲田大学総合機械工学科助手 (菅野研)

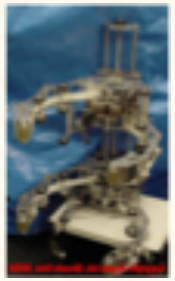
http://ysuga.net
http://revast.co.jp
http://sugarsweetrobotics.com

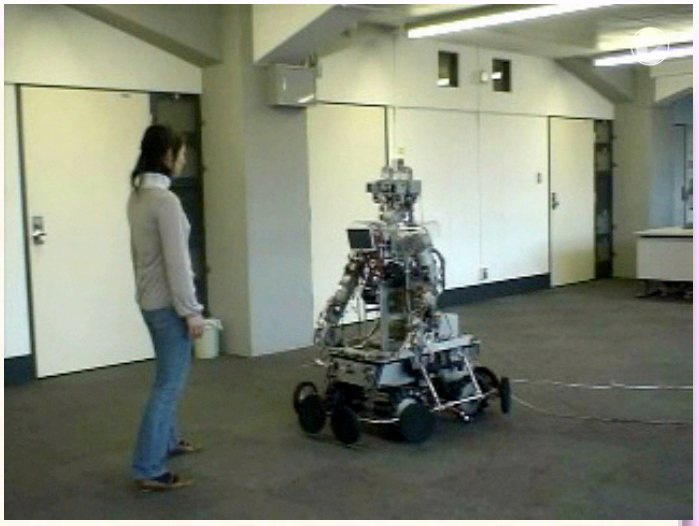


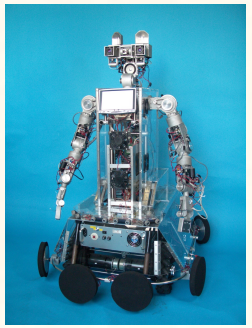
林業機械支援システム

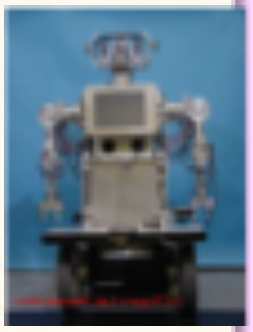
岐阜県・早稲田大学WABOT-HOUSE研究所





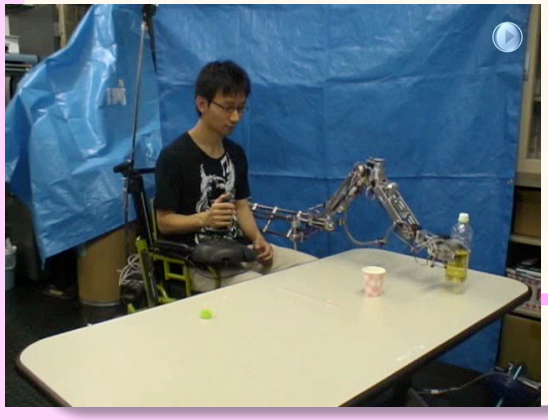






学習適応するコミュニケーションロボット

車いす搭載型ロボットアーム



2007～2010 早稲田大学総合機械工学科助手

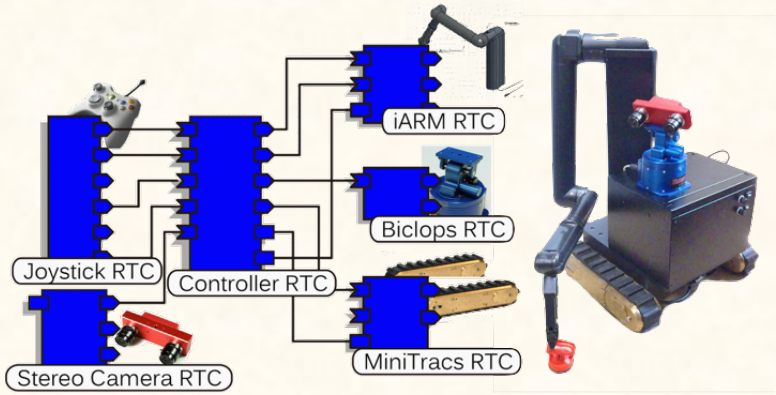
http://www.sugano.mech.waseda.ac.jp

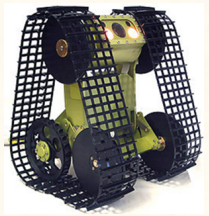
アメリカMobile Robot社
移動台車Pioneerシリーズ

- 2010～2012 株式会社リバスト
  - 海外製ロボットの輸入・販売
  - 研究・開発用ロボットの受託開発
  - ロボットの組み合わせ（システムインテグレーション

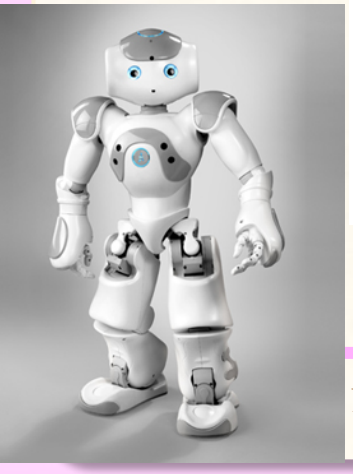
http://revast.co.jp

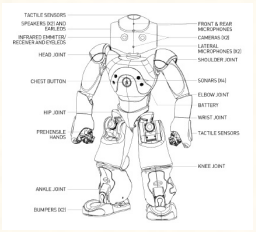
スイス Neuronics社
Katana ロボットアーム





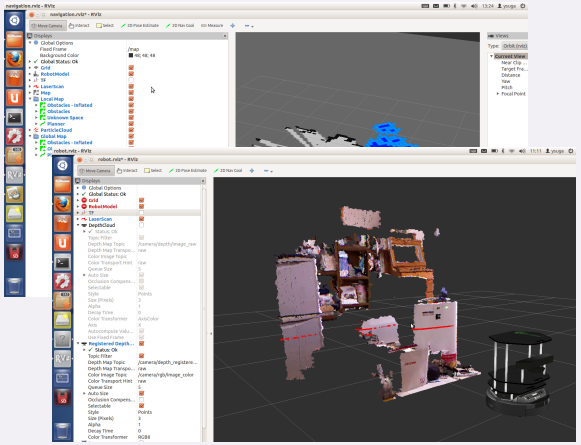
カナダ Inuktun社
探索ロボット



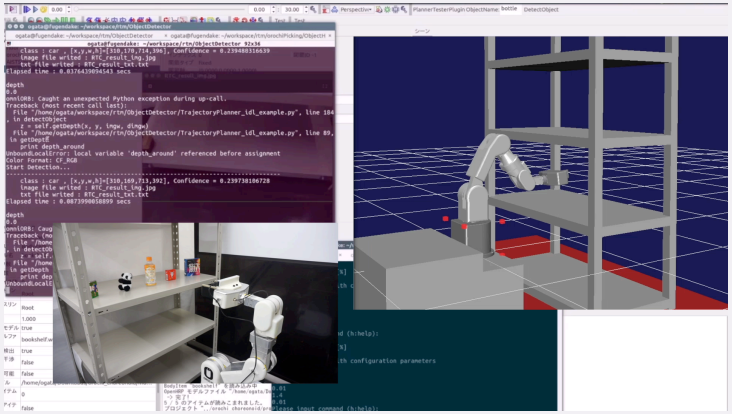


フランス Aldebaran Robotics社
NAO

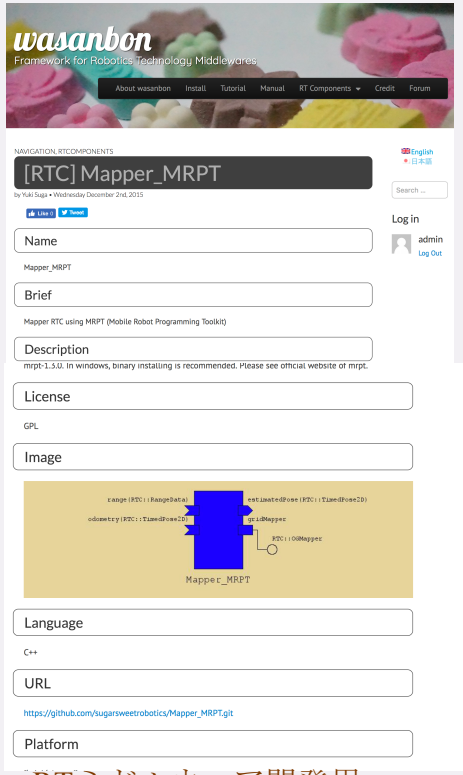
http://sugarsweetrobotics.com (株式会社SUGAR SWEET ROBOTICS)



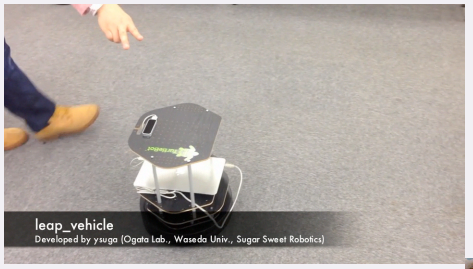
ROSを用いたプロトタイピング



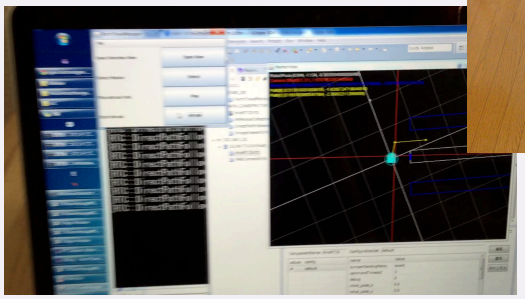
RTミドルウェアを用いたマニュピレーション用フレームワークの開発



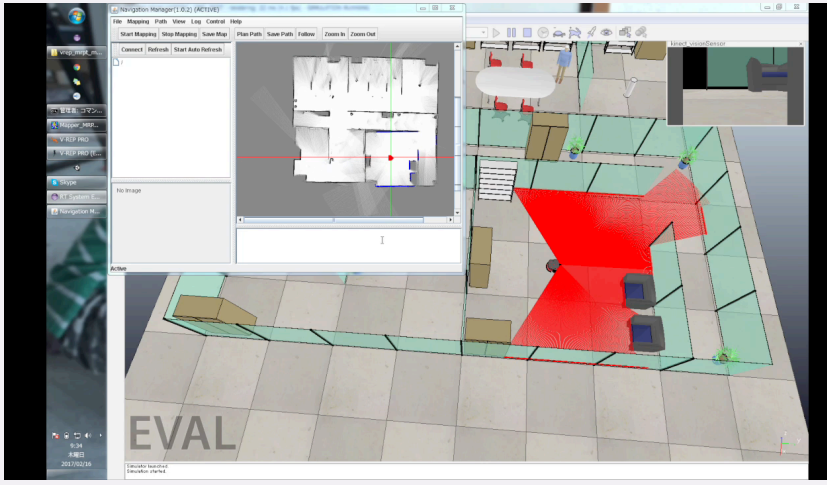
RTミドルウェア開発用フレームワーク「wasanbon」



ロボットおよびロボット用ミドルウェアに関する講義・講演



RTミドルウェアを用いたプロトタイピング

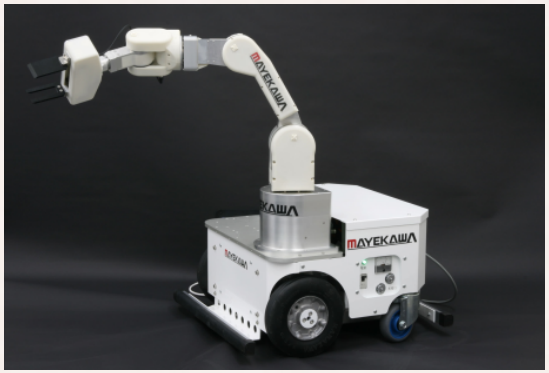


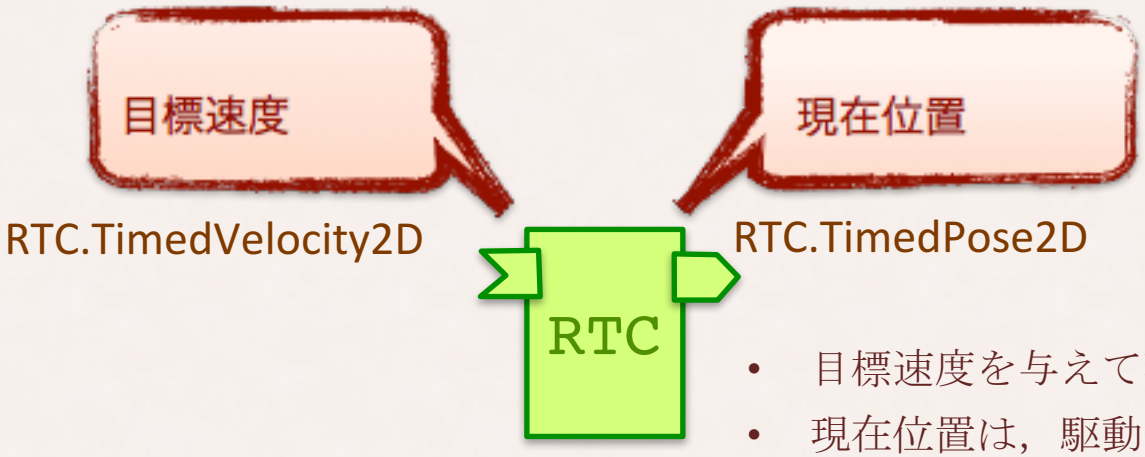
RTミドルウェアを用いた移動ロボットナビゲーション

移動ロボット

サマーキャンプで使える移動ロボットハードウェア

- Kabuki
- Pioneer 3DX
- OROCHI台車
- RaspberryPiマウス
- Roomba



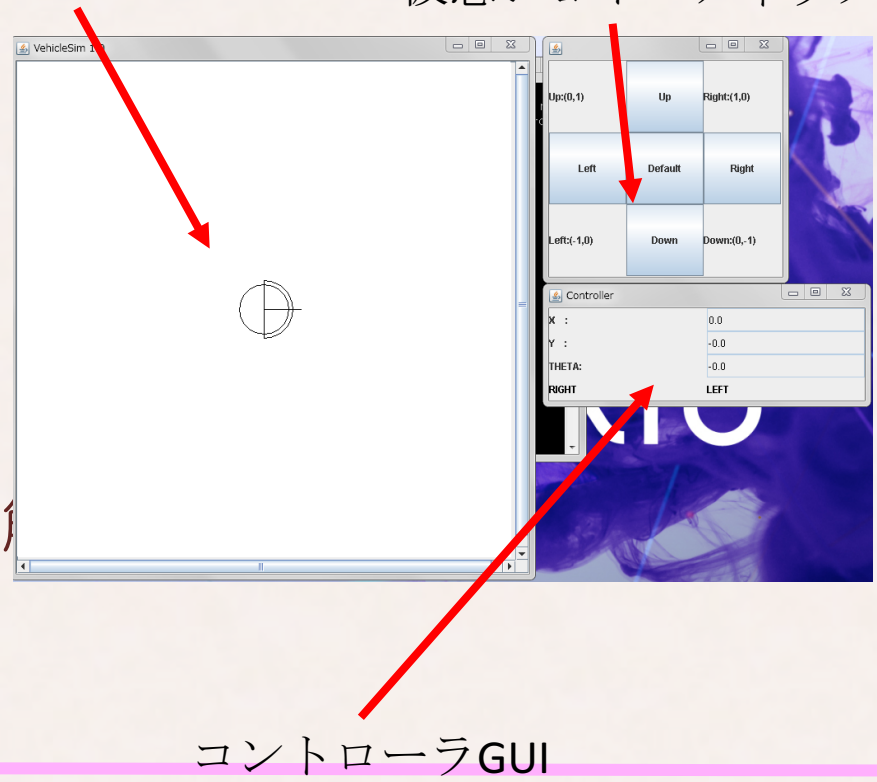


- 目標速度を与えて現在位置を返す.
- 現在位置は，駆動モータに着いたエンコーダの値の積算なので，スリップなどの影響で狂いやすい

OpenRTM-aist学習用台車シミュレータ

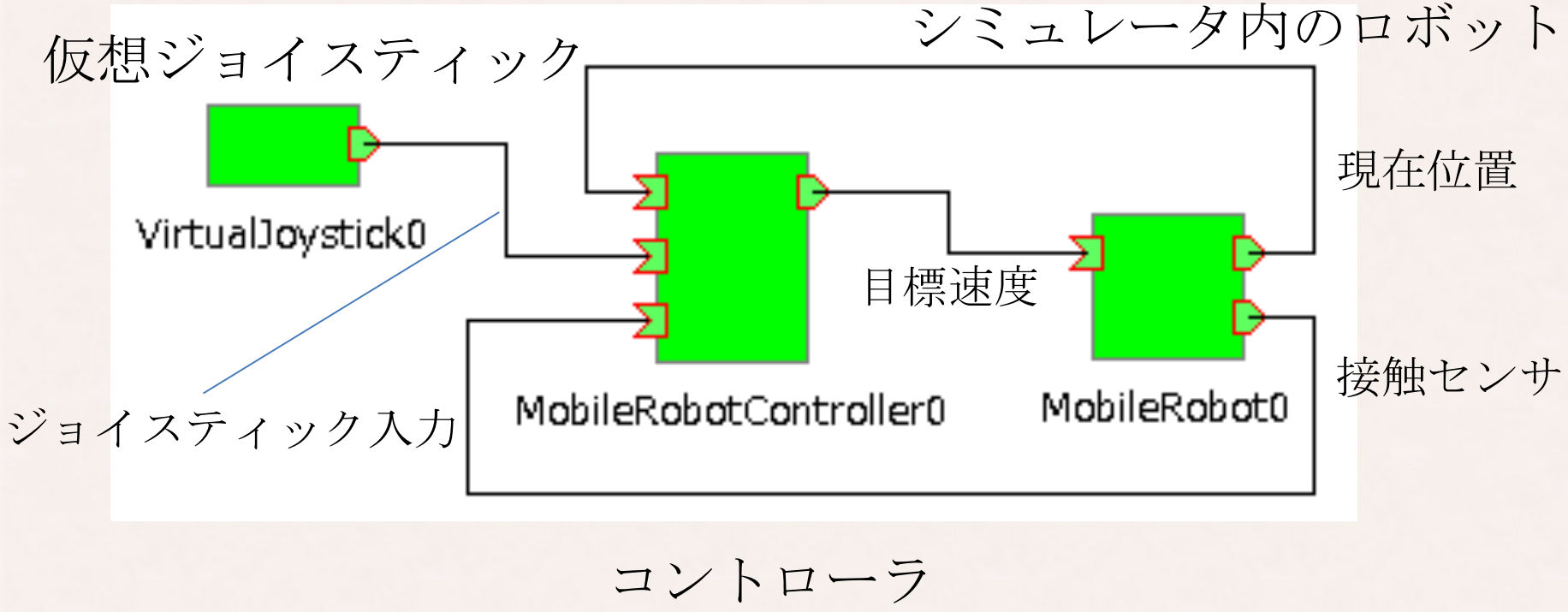
http://ysuga.net/?p=133

- Loader.batを実行
  - シミュレータ
  - 仮想ジョイスティック
  - コントローラ
- すべて接続してACTIVATE
- ジョイスティックで操作
- コントローラGUIで位置や接



台車シミュレータの中身

- RT System Editorで表示

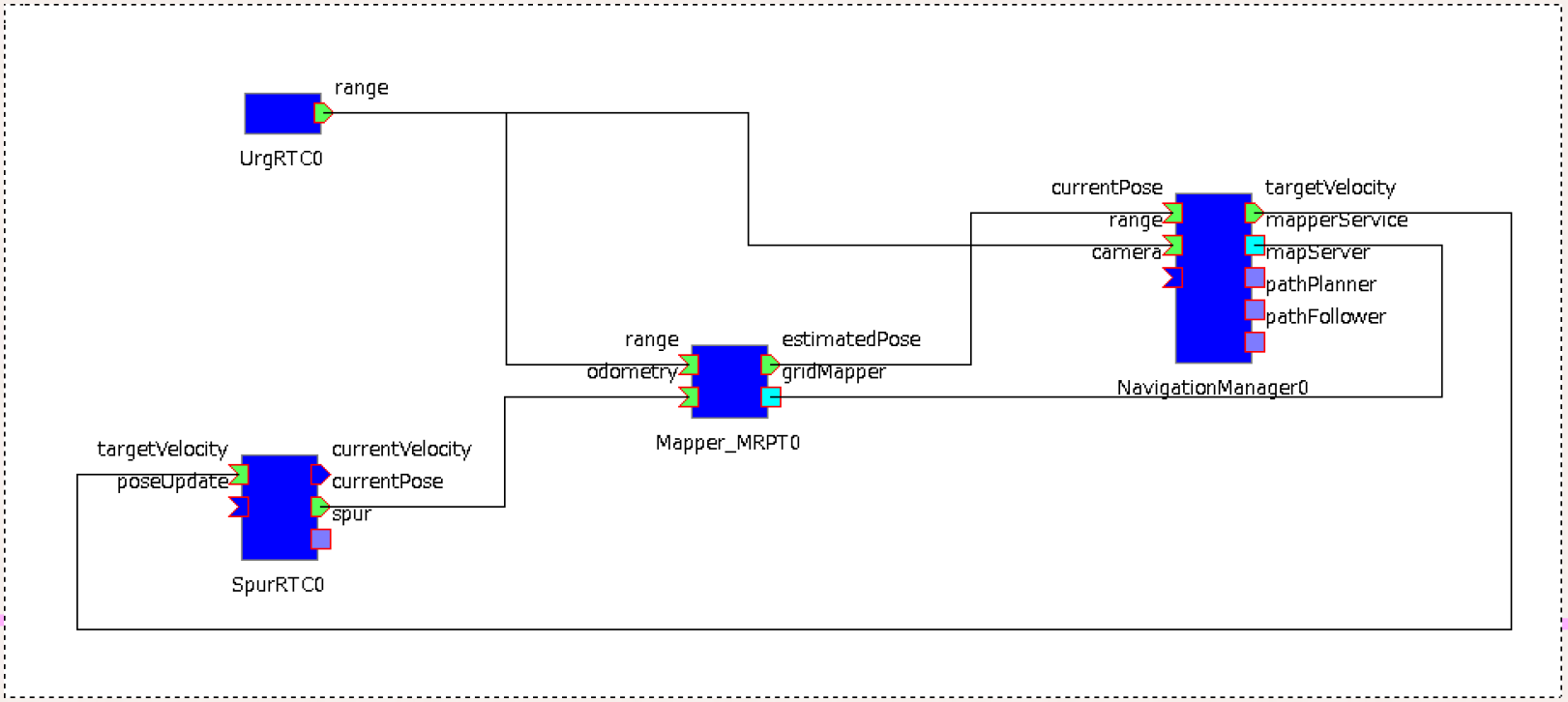


移動ロボットナビゲーション

- 台車型移動ロボットのナビゲーションが目的
  - すべてオープンソース
- マップ作成
  - オンラインSLAMのみ
- 地図を用いたナビゲーション
  - レーザーレンジセンサーを用いた位置推定
  - 軌道計画
  - 軌道追従
- 機能ごとにモジュール化しており，インターフェースを共通化して，入替えが可能
  - 例：独自の移動ロボットを適用
  - 例：独自のプランニング・パスフォロワーを適用
- 基本操作が可能なGUIを準備

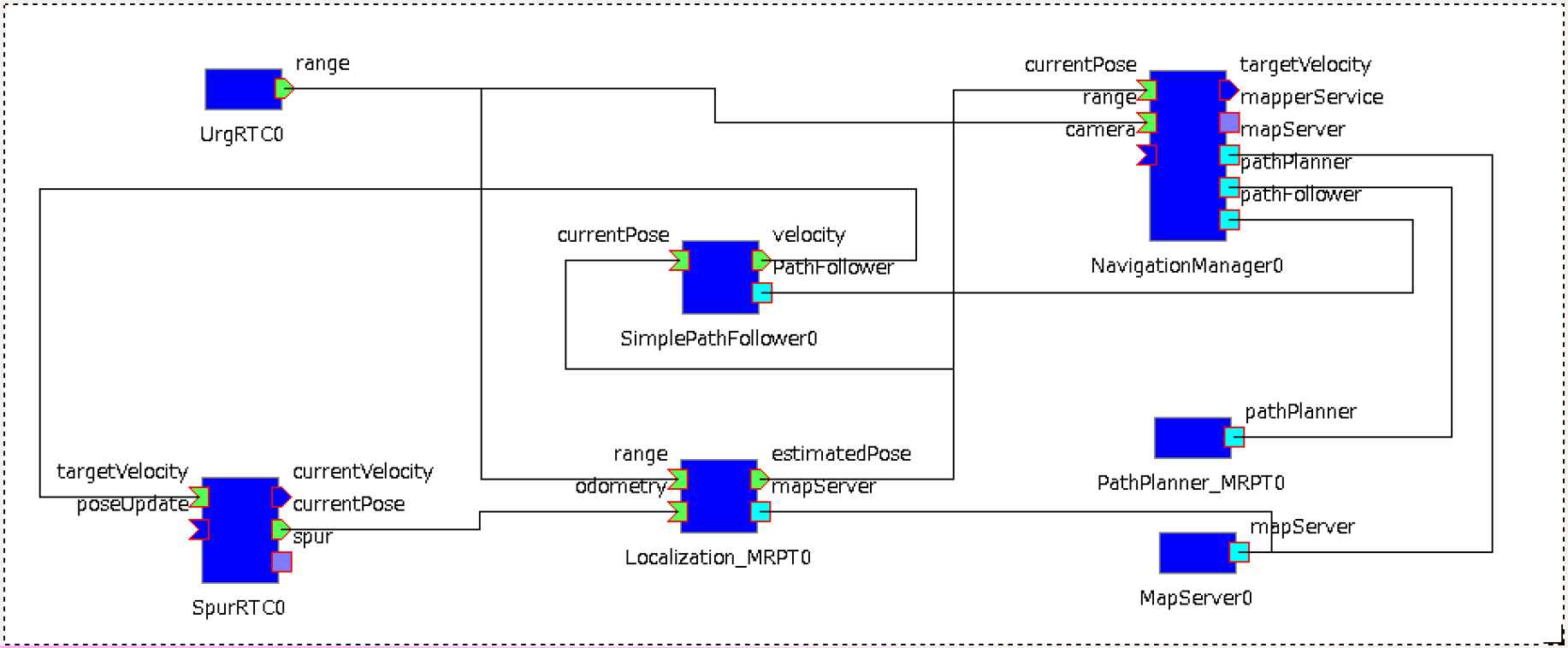
マップ作成

- 北陽電機URGセンサを使ってマップ作成
- MRPTを利用



ナビゲーション

- 出来上がったマップをMapServerで配信
- パーティクルフィルタでの自己位置推定
- Pathプランニングと簡易なパス追従が可能



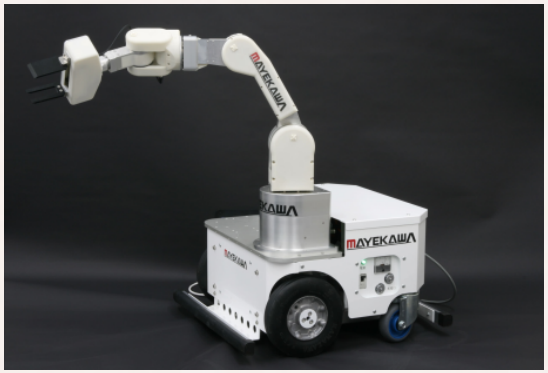
フレームワークの特徴

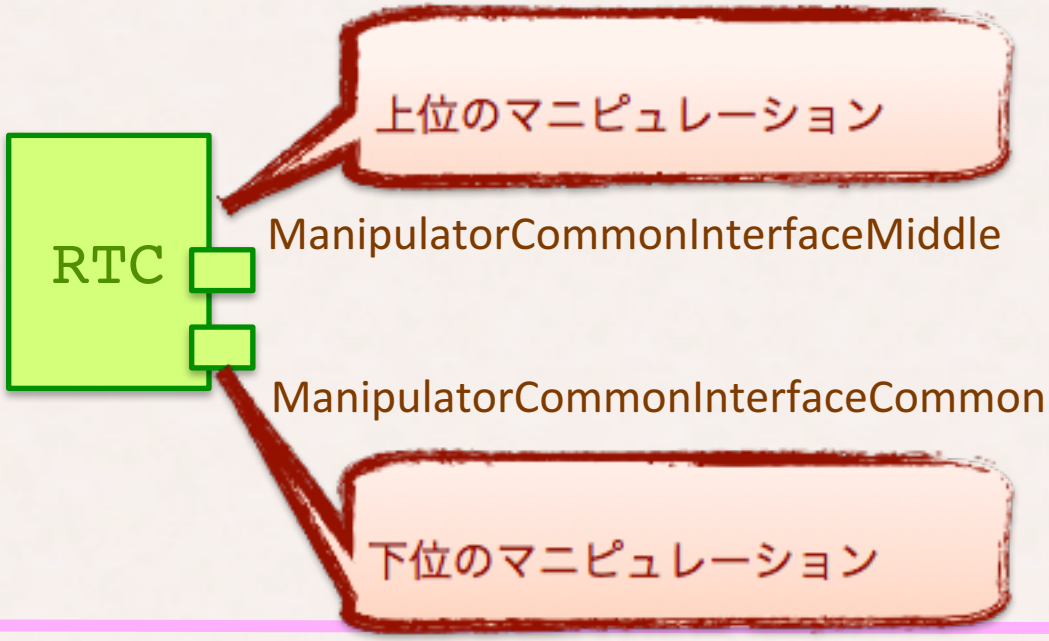
- 上位のインターフェースはすべてサービスポート
- 処理の成功・失敗を返り値で受け取ることができる
- サービスロボットの上位処理はデータフロー的ではなく，イベント的
- MobileRobot.idlでインターフェースを定義済み
  - これをインポートすればコンシューマ側は簡単に作成できる
  - ロボット側はTimedVelocity2D, TimedPose2Dのデータポート

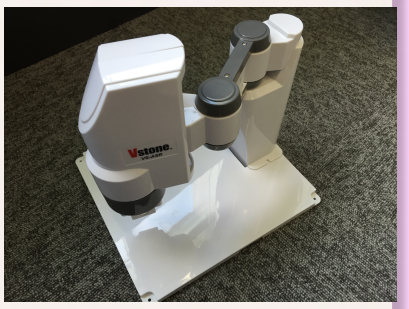
アーム系

サマーキャンプで使える アーム系ハードウェア

- OROCHIアーム
- Vistonアカデミックスカラロボット







- サービスポートで動作
- 手先の目標位置を指定するとそちらに直線で動く

マニピュレーター
共通インターフェース

- 2013年に改定 (埼玉大学 琴坂研究室による)
- ２つのレベルに分割
  - 共通レベル (ManipulatorCommonInterface_Common)
    - 各関節レベルでの動作，位置獲得
    - エラー情報取得・復帰
  - 中位レベル (ManipulatorCommonInterface_Middle)
    - 各関節レベルでの動作
    - 手先位置での動作
    - 手先移動範囲の指定や，ツールおよび台座のオフセット
    - ハンドの開閉
  - 共通するデータ型をManipulatorCommonInterface_DataTypesに規定
- http://github.com/sugarsweetrobotics/VS_ASR_RTC/
- https://github.com/sugarsweetrobotics/ManipulatorCommonTest

```
/*
Manipulator Common Interface (Data type defenition)
- This IDL is used as service port on RTC
- This command specification is provided by Intelligent RT Software
Project of JARA.
rev. 20140120
*/
#ifndef MANIPULATORCOMMONINTERFACE_DATATYPES_IDL
#define MANIPULATORCOMMONINTERFACE_DATATYPES_IDL
#include "BasicDataType.idl"
module JARA_ARM{
	typedef sequence<double> DoubleSeq;
	typedef sequence<double> JointPos;

	struct LimitValue {
		double upper;
		double lower;
	};
	struct RETURN_ID {
		long id;
		string comment;
	};
	const long OK = 0;
	const long NG = -1;
	const long STATUS_ERR = -2;
	const long VALUE_ERR = -3;
	const long NOT_SV_ON_ERR = -4;
	const long FULL_MOTION_QUEUE_ERR = -5;
	const long NOT_IMPLEMENTED = -6;
	struct TimedJointPos {
		RTC::Time tm;
		JointPos pos;
	};
	typedef unsigned long ULONG;
};
#endif // MANIPULATORCOMMONINTERFACE_DATATYPES_IDL
```

```
/*
Manipulator Common Interface (Common Commands)
- This IDL is used as service port on RTC
- This command specification is provided by Intelligent RT Software
Project of JARA.
rev. 20140120
*/
#ifndef MANIPULATORCOMMONINTERFACE_COMMON_IDL
#define MANIPULATORCOMMONINTERFACE_COMMON_IDL
#include "ManipulatorCommonInterface_DataTypes.idl"
module JARA_ARM{
	enum AlarmType {
		FAULT,
		WARNING,
		UNKNOWN
	};
	struct Alarm {
		unsigned long code;
		AlarmType type;
		string description;
	};
	typedef sequence<Alarm> AlarmSeq;
	typedef sequence<LimitValue> LimitSeq;
	struct ManipInfo {
		string manufactur;
		string type;
		ULONG axisNum;
		ULONG cmdCycle;
		boolean isGripper;
	};
```

```
        const ULONG CONST_BINARY_00000001 = 0x01; //isServoOn
        const ULONG CONST_BINARY_00000010 = 0x02; //isMoving
        const ULONG CONST_BINARY_00000100 = 0x04; //isAlarmed
        const ULONG CONST_BINARY_00001000 = 0x08; //isBufferFull

        interface ManipulatorCommonInterface_Common {
            RETURN_ID clearAlarms();
            RETURN_ID getActiveAlarm(out AlarmSeq alarms);
            RETURN_ID getFeedbackPosJoint(out JointPos pos);
            RETURN_ID getManipInfo(out ManipInfo mInfo);
            RETURN_ID getSoftLimitJoint(out LimitSeq softLimit);
            RETURN_ID getState(out ULONG state);
            RETURN_ID servoOFF();
            RETURN_ID servoON();
            RETURN_ID setSoftLimitJoint(in LimitSeq softLimit);
        };
};
#endif // MANIPULATORCOMMONINTERFACE_COMMON_IDL
```

```
/*
Manipulator Common Interface (Middle Level Commands)
- This IDL is used as service port on RTC
- This command specification is provided by Intelligent RT Software
Project of JARA.
rev. 20140120
*/
#ifndef MANIPULATORCOMMONINTERFACE_MIDDLE_IDL
#define MANIPULATORCOMMONINTERFACE_MIDDLE_IDL

#include "ManipulatorCommonInterface_DataTypes.idl"

module JARA_ARM{

	typedef double HgMatrix [3][4];

	struct CarPosWithElbow {
		HgMatrix carPos;
		double elbow;
		ULONG structFlag;
	};

	struct CartesianSpeed {
		double translation;
		double rotation;
	};

	interface ManipulatorCommonInterface_Middle {
		RETURN_ID closeGripper();
		RETURN_ID getBaseOffset(out HgMatrix offset);
		RETURN_ID getFeedbackPosCartesian(out CarPosWithElbow pos);
		RETURN_ID getMaxSpeedCartesian(out CartesianSpeed speed);
		RETURN_ID getMaxSpeedJoint(out DoubleSeq speed);
		RETURN_ID getMinAccelTimeCartesian(out double aclTime);
		RETURN_ID getMinAccelTimeJoint(out double aclTime);
		RETURN_ID getSoftLimitCartesian(out LimitValue xLimit,
			out LimitValue yLimit, out LimitValue zLimit );
		RETURN_ID moveGripper(in ULONG angleRatio);
		RETURN_ID moveLinearCartesianAbs(in CarPosWithElbow carPoint);
		RETURN_ID moveLinearCartesianRel(in CarPosWithElbow carPoint);
		RETURN_ID movePTPCartesianAbs(in CarPosWithElbow carPoint);
		RETURN_ID movePTPCartesianRel(in CarPosWithElbow carPoint);
		RETURN_ID movePTPJointAbs(in JointPos jointPoints);
		RETURN_ID movePTPJointRel(in JointPos jointPoints);
```

```
            RETURN_ID openGripper();
            RETURN_ID pause();
            RETURN_ID resume();
            RETURN_ID stop();
            RETURN_ID setAccelTimeCartesian(in double aclTime);
            RETURN_ID setAccelTimeJoint(in double aclTime);
            RETURN_ID setBaseOffset(in HgMatrix offset);
            RETURN_ID setControlPointOffset(in HgMatrix offset);
            RETURN_ID setMaxSpeedCartesian(in CartesianSpeed speed);
            RETURN_ID setMaxSpeedJoint(in DoubleSeq speed);
            RETURN_ID setMinAccelTimeCartesian(in double aclTime);
            RETURN_ID setMinAccelTimeJoint(in double aclTime);
            RETURN_ID setSoftLimitCartesian(in LimitValue xLimit,
                  in LimitValue yLimit, in LimitValue zLimit);
            RETURN_ID setSpeedCartesian(in ULONG spdRatio);
            RETURN_ID setSpeedJoint(in ULONG spdRatio);
            RETURN_ID moveCircularCartesianAbs(in CarPosWithElbow carPointR,
                  in CarPosWithElbow carPointT);
            RETURN_ID moveCircularCartesianRel(in CarPosWithElbow carPointR,
                  in CarPosWithElbow carPointT);
            RETURN_ID setHome(in JointPos jointPoint);
            RETURN_ID getHome(out JointPos jointPoint);
            RETURN_ID goHome();
      };
};
#endif // MANIPULATORCOMMONINTERFACE_MIDDLE_IDL
```

組み込み

RTno (あーるてぃーの)

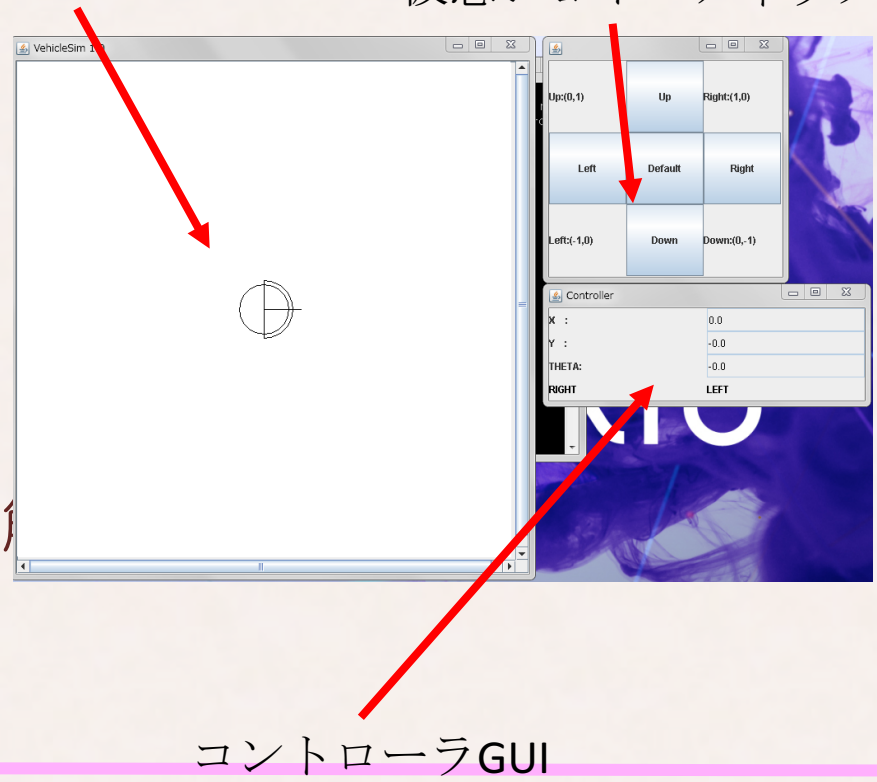
- Arduino + RT = RTno
  - Arduinoを簡単にRTコンポーネントにするライブラリ
  - 自作の回路やデバイスをRTコンポーネントにできる
  - http://ysuga.net/?p=124

シミュレーション

OpenRTM-aist学習用台車シミュレータ

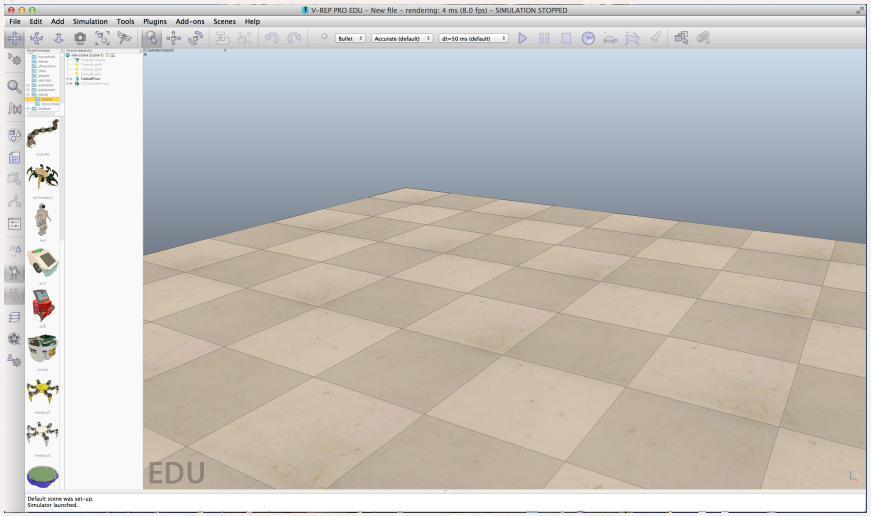
http://ysuga.net/?p=133

- Loader.batを実行
  - シミュレータ
  - 仮想ジョイスティック
  - コントローラ
- すべて接続してACTIVATE
- ジョイスティックで操作
- コントローラGUIで位置や接



V-REPシミュレータ

- RTコンポーネントとの通信に対応した動力学シミュレータ V-REP
  - プラグインをインストールして使う
  - 使い方はこちら
    - http://ogata-lab.jp/?p=1264



ysuga関連のRTCの収集

ysuga関連のウェブサイト

- ysuga.net: http://ysuga.net
  - RTミドルウェア入門記事などがある
  - ちょっと古い情報だがわかりやすいと思います.
- 会社のウェブ: http://sugarsweetrobotics.com
  - 開発してるツールへのリンクがあります.
- 尾形研究室のウェブ: http://ogata-lab.jp
  - C言語ラッパー, C#ラッパー
  - UnityでのRTMの使い方
  - V-REPシミュレータ＋OpenRTM-aist
  - などなど
- wasanbon: http://wasanbon.org
  - ysugaの作ったRTCのソースコード収集と配布
  - カテゴリに分かれており探しやすいサイトだと思います.

- 移動台車
  - 速度司令して位置を受け取る
  - SLAMとNavigation（ゴールを司令）
- アーム
  - 関節角度を直接司令
  - 逆運動学
- シミュレータ
  - V-REP
- 組み込み
  - Arduino（RTno）
- ysuga関連のRTCの収集
  - wasanbon.org
  - ogata-lab.jp

サマーキャンプのコツ

- せっかくおじさんたちがいるので，早めに聞きに来ること
  - おじさんたちは色々な事を知ってます
  - 具体的でない相談もここでは受け付けています
    - なんか良いRTCない？
    - このシステムどう？
  - 悩む時間はない！！開発しろ！！！

ご清聴ありがとうございました

菅　佑樹　(ysuga@ysuga.net)
株式会社SUGAR SWEET ROBOTICS代表取締役
早稲田大学表現工学科　尾形哲也研究室　次席研究員
http://ysuga.net
http://sugarsweetrobotics.com