

.env

ドットエンヴ

移動ロボット向け環境シミュレータ

version 1.0

セグウェイジャパン株式会社

目次

ソフトウェア使用許諾契約書.....	4
1. 機能説明.....	6
1.1. 概要.....	6
1.2. 動作環境.....	6
1.3. 対応要素.....	6
2. 構成.....	8
2.1. 構成詳細.....	9
3. 使い方.....	10
3.1. インストール.....	10
3.2. アンインストール.....	10
3.3. 起動方法.....	10
3.4. 終了方法.....	10
3.5. 設定ファイル.....	11
3.6. 壁データ.....	15
3.7. GPS誤差傾向確認用ソフト (GPSnoise.exe).....	16
3.8. RoboCarに関して.....	17
4. 画面・操作説明.....	20
5. チュートリアル.....	24
5.1. サンプルインターフェイスの動作.....	24
5.2. RT-Middlewareによる人追従機能.....	26
5.3. SRI KARTO SDKの動作.....	32
6. お問い合わせ.....	36

ソフトウェア使用許諾契約書

移動ロボット環境シミュレータ「.env(ドットエンヴ)」(以下本ソフトウェア)は、以下の条件全てに同意いただけた方のみご利用頂けます。

本ソフトウェアは、本ソフトウェアの著者(以下著作者)の著作物であり、著作者によってのみライセンスされます。

本ソフトウェアをインストールした時点(インストーラー形式でない場合は内容を読み取れる形式に変換した時点)で以下の条件に同意頂けたものとみなしますので、インストール前に各条件を十分にお読み下さい。同意できない場合は、たとえ試用目的であっても、本ソフトウェアを使用することはできません。使用を中止し、速やかに全てのインストールされたファイルを削除してください。

■「本ソフトウェア」について

- (1) 「本ソフトウェア」とは、「.env」の原本及びその複製物(部分的複製物及び他のプログラムに結合された複製物を含みます。)を意味します。「本ソフトウェア」には、機械で読み取りうる形の命令、その構成物、データ及びその他の関連するライセンス資料が含まれます。
- (2) 本ソフトウェアには、スパイウェア等の有害なプログラムは、一切含まれていません。
- (3) 本ソフトウェアは、本ソフトウェアの動作不良が死亡、けが又は重大な物理的又は環境的損害につながる恐れのある場所(例えば、原子力施設の運用、航空機の運行や航空管制、医療活動等のシステム)において使用されることを想定したものではありません。

■使用権

- (1) 本ソフトウェアをダウンロードその他の手段で提供し、使用する者(以下使用者)に対して、自ら使用する権利を与えます。この権利は、独占的に有する権利ではないものとします。
- (2) 使用者は非営利、商用目的に関わらず本ソフトウェアを使用することができます。ただし、本ソフトウェアから直接利益を得るような使用はできません。
- (3) 本使用許諾書は使用者が本ソフトウェアをインストールされた時点から発効します。
- (4) 使用者は、本ソフトウェアを複数のコンピュータにインストールすることができます。但し、一度に本ソフトウェアを使用するのは1つのみとします。
- (5) 使用者が、本使用許諾書により許諾される許諾プログラムの使用権を終了させる場合、記憶メディアからの削除をもって終了するものとします。この場合、複製物を含めた全ての本ソフトウェアに関するデータを破棄するものとします。
- (6) 本ソフトウェアの使用権は、本使用許諾書の使用条件の規定に基づき終了するまで有効に存続します。

■禁止事項

- (1) 使用者は、本ソフトウェアの全部または一部を、改変、リバースエンジニアリング、逆コンパイル又は逆アセンブルなどの解析作業や改変行為を一切行ってはいけません。
- (2) 使用者は、上記(1)の方法又はそれ以外の方法で、本ソフトウェアのソースコードの抽出又は派生物の作成を試みてはいけません。
- (3) 使用者は、本ソフトウェア上に表示され、または本ソフトウェア中に含まれている、所有権、商標又は著作権の表示を、除去又は破棄してはいけません。

■著作者の免責

- (1) 著作者は本ソフトウェアを特定物として現存するままの状態を提供します。本ソフトウェアが、直接又は間接的に損害を生じさせても、著作者は一切の責任を負いません。また、機器や媒体が原因の損害につきましても、著作者は一切の責任を負いません。よって、著作者は本ソフトウェアに関するいかなる保証も行いません。
さらに本ソフトウェアを使用した結果の影響に関しましても、一切責任を負わないものとします。
- (2) 著作者は、バージョンアップ、不具合修正の義務を負いません。

■管轄裁判所

- (1) 著作者の所在地を管轄する裁判所を第一審の管轄裁判所とします。

■その他

- (1) 本ソフトウェアの仕様及びマニュアル等の内容は将来予告無く変更されることがあります。
- (2) 本ソフトウェアの一部のソースコードはBSDライセンスの下で使用することができます。
詳細については/src/LICENSE-BSD.TXT を参照してください。
- (3) src/robocar/以下に収録されているライブラリ・サンプルプログラム・マニュアル等

ReferenceManual

zmprc_lib

zmprc_samples

RczHostApp

は株式会社ゼットエムピーの著作物になります。

株式会社ゼットエムピーはこれらのライブラリ・サンプルプログラムの使用時のサポートは行いません。

またこれらの再配布を行ってはいけません。

以上

1. 機能説明

1.1. 概要

- ・ロボットの動作を PC 上でシミュレート・可視化し、ロボットアプリケーション開発時の環境を PC 上で提供するとともに、可視化による直感的理解を助ける媒体として利用できるようにします。
- ・実現不可能・困難な環境下でのテスト環境を提供することができます。広大な敷地、多数の人が移動している環境などを作成することが可能です。
- ・ロボットアプリケーション開発時のトライ&エラー時間、主にロボット動作環境(場所)のセッティング、ロボットのスタート位置へのリセット等、現実環境でのトライ&エラーは非本質的な作業時間が長時間必要になります。シミュレータを使用することによって、このトライ&エラー時間を大幅に削減できます。

1.2. 動作環境


動作環境	Microsoft® Windows® XP / 7 / VISTA 32bit 版 Linux® 32bit 版 (Ubuntu® 10.04 動作確認済) OpenGL®対応グラフィックス環境
推奨環境	CPU 2.4GHz 以上 メモリ 2GB 以上 nvidia® GeForce9800®以上

1.3. 対応要素

■対応ミドルウェア


- ・RT-Middleware (産業技術総合研究所) 
<http://openrtm.org/openrtm/ja/content/openrtm-aist%E3%81%A8%E3%81%AF%EF%BC%9F>
- ・ROS (WillowGarage) 
<http://jp.willowgarage.com/drupal2/ja/pages/software/ros-platform>
- ・KARTO™ SDK 2.0 (SRI International) ※動作確認済 
<http://www.kartorobotics.com/>

■移動台車

- ・Segway® RMP (Robotic Mobility Platform) シリーズ
<http://www.segway-japan.co.jp/robot-hard/rmp.html>
- ・Blackship® シリーズ
<http://www.segway-japan.co.jp/robot-hard/blackship.html>
- ・RoboCar® (株式会社ゼットエムピー) 
<http://www.zmp.co.jp/e-nuvo/jp/robocar-110.html>
- ・自動車

■センサ

- ・慣性センサ(加速度センサ・ジャイロセンサ・地磁気センサ)

e-nuvo IMU-Z (株式会社ゼットエムピー) 

<http://www.zmp.co.jp/e-nuvo/jp/imu-z.html>

- ・測域センサ (SCIP2 形式採用)

SCIP2 対応センサ: 北陽電機株式会社 Top-URG (UTM-30LX) , ClassicURG (URG-04LX) 等

- ・GPS (NMEA 形式採用)

- ・赤外線センサ

- ・カメラ

■その他

- ・歩行者(コントローラでの操作, 指定位置への移動, ランダム歩行, 歩行者視点での移動)
- ・Segway® PT i2(歩行者が乗った場合にコントローラでの操作)

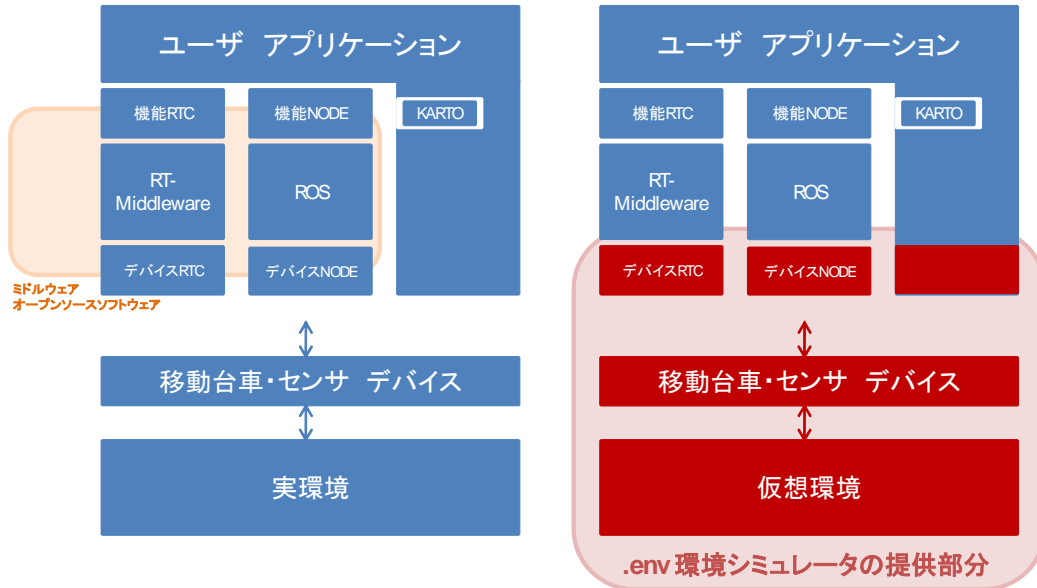
【諸注意】

※シミュレータの動作と同様のハードウェアの動作を保証するものではありません。

※対応ハードウェアと同等の機能を有するわけではありません。

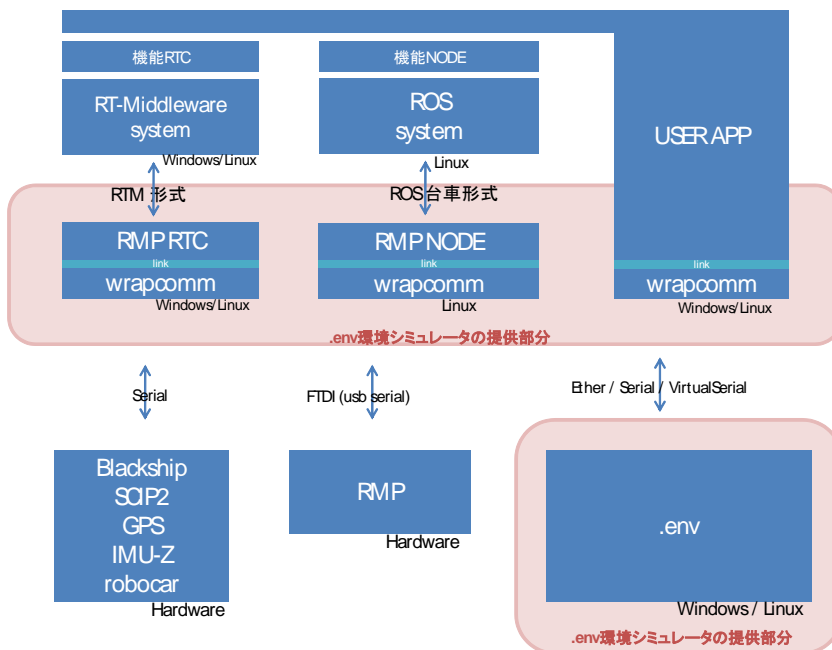
※対応要素は今後追加されていく予定です。

2. 構成

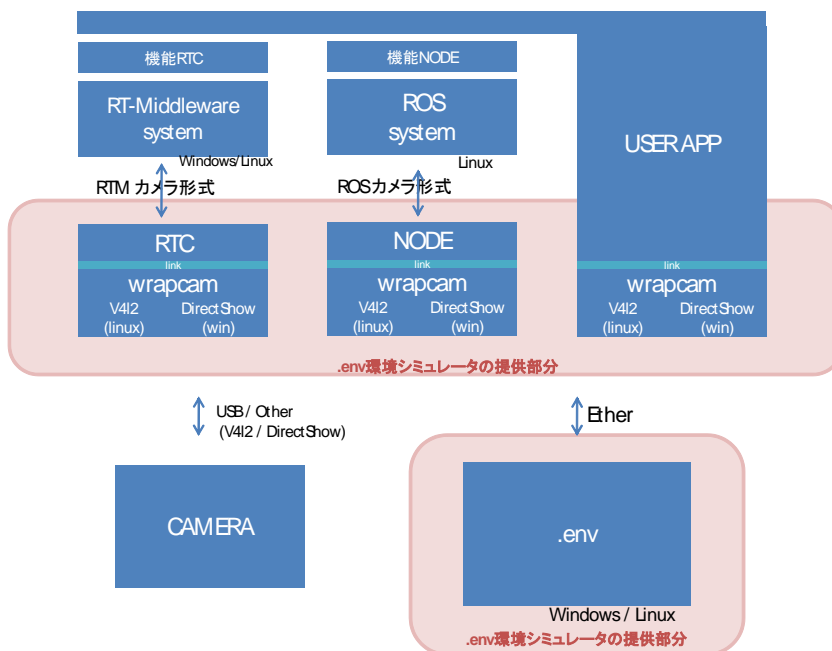


2.1. 構成詳細

RMP / Blackship / SCIP2 / GPS / IMU-Z / Robocar



Camera



3. 使い方

3.1. インストール

- Windows 1.インストーラをダブルクリックしてください。
2.表示される画面に従いインストールを行ってください。
- Linux 1.ターミナルウィンドウを開き、src/HASP/Redistribute/Runtime/script を参照します。
2.「./dinst .」と入力してインストールを行います。
3.実行環境がインストールされていることを確認します。
- ```
/usr/sbin/winehasp
/usr/sbin/aksusbd
/usr/sbin/hasplmd
/etc/init.d/aksusbd
```
- 4.bin フォルダに収録されている実行ファイル(robosim)を実行可能にします。  
bin フォルダを参照し、以下を実行します。
- ```
chmod 755 robosim
```
- ※インストールを行うには管理者権限が必要です。

3.2. アンインストール

- Windows 1.「コントロールパネル」から「プログラムの追加と削除」を開きます。
2.RobotSimulator を削除してください。
- Linux 1.ターミナルウィンドウを開き、src/HASP/Redistribute/Runtime/script を参照します。
2.「./dunst .」と入力してアンインストールを行います。
※アンインストールを行うには管理者権限が必要です。

3.3. 起動方法

※起動する前にUSB dongleキーをPCに挿してください。

- Windows
- robosim.exe をダブルクリックします。
※デフォルトの壁データ・設定ファイルで起動します。
 - コマンドプロンプトから実行ファイルが存在するディレクトリで
「robosim 壁データ 設定ファイル」と入力してシミュレータを起動します。
例)robosim walldata settingfile
- Linux
- ターミナルウィンドウから実行ファイルが存在するディレクトリで
「robosim 壁データ 設定ファイル」と入力してシミュレータを起動します。
例) ./robosim walldata settingfile

3.4. 終了方法

メインウィンドウ右上の[×]をクリックします。

3.5. 設定ファイル

移動台車の種類やその数、その台車に搭載するセンサ類やその方向、人の数や人の動作の種類などを指定することができます。

※/bin/sample 内にサンプルがありますので参考にしてください。

■台車の設定

設定方法: **name port x y z angle**

name: RMP50、RMP100、RMP200、RMP200AVT、RMP400、
BLACKSHIP、BS2W、ROBOCAR、CAR

port: 通信ポート COM○○: windows で通信を行う場合
/dev/○○: Linux で通信を行う場合
IP:port: ネットワーク通信を行う場合

x、y、z: 初期位置

angle: 台車の向き(ラジアンではなく度)

※初期位置を指定しない場合、壁データより初期位置を設定します。

壁データに初期位置が設定されていない場合は原点に配置されます。

■センサの設定

設定方法: **name port x y z pan tilt**

name: SCIP2CU、SCIP2TU、IMU-Z、IR

pan: 左右の角度(ラジアンではなく度)

tilt: 上下の角度(ラジアンではなく度)

■カメラの設定

設定方法: **CAMERA port x y z pan tilt fovy quality**

pan: 左右の角度(ラジアンではなく度)

tilt: 上下の角度(ラジアンではなく度)

fovy: 画角(上下方向。アスペクト比 4:3 固定。デフォルト 48 度)

quality: 送信 Jpeg 画像の画質(デフォルト 40)

■SegwayPT・歩行者の設定

設定方法: **name dummy x y z angle**

name: SEGWAY、SEGRIDE、HUMAN、WOMAN、BOY、GIRL

dummy: ダミーの値を設定してください。

■パッドの設定

設定方法: **PAD port**

※linux のみ有効 (port を指定しない場合、/dev/input/js0 が使用されます)

■遅延時間の設定

設定方法: **LATENCY latency**

latency: 遅延時間(マイクロ秒単位)

■ROBOCAR のオプション設定

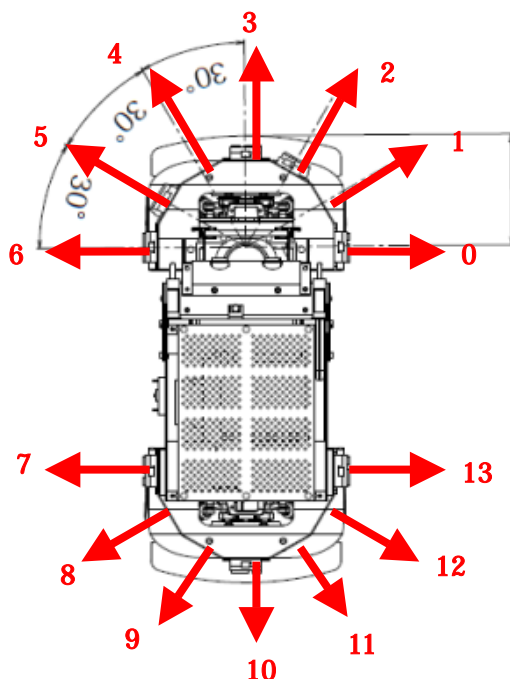
設定方法: **ROBOCARSERVO port**

: **ROBOCARIR ir**

ir には赤外線センサーの方向を指示する 16 進数の数値を指定しますが、赤外線センサーの方向の順番(台車前方右から順番に左回り)で下位 bit から bit を立てた方向の8個までが有効になります。

※ir の指定がない場合、0x155D が設定されます。

例: 0x155D の場合、2 進数では 1010101011101 となりまして、下図のうち、0,2,3,4,6,8,10,12 が有効になります。



■経緯度の設定

設定方法: **GEOCOORD kei latitude longitude**

kei: 1系~19系の原点 latitude: 緯度 longitude: 経度

※設定値の指定がない場合、原点=9、緯度=0、経度=0 が設定されます。

■GPS の設定

設定方法:

GPS port posx posy posz pan tilt DilutionLength HDOPlen Frequency Persistence Octave

※パラメータの詳細は、[3.7.GPS誤差傾向確認用ソフト \(GPSnoise.exe\)](#)の説明を参照ください。

※GEOCOORD を先に設定してください。

■マップファイルの設定

設定方法: **MAPIMAGE map**

map: マップファイル(name_○○x□□.ppm) ○○: マップの横幅 □□: マップの縦幅

※マップファイルに対応している拡張子は ppm です。

例: /bin/sample 内の mappark_626x616.ppm を設定すると下図のように表示されます。



■表示色の設定

設定方法: **COLOR red green blue**

red: 赤 green: 緑 blue: 青

※設定する値の範囲は 0.0~1.0 です。

※複数に設定したい場合は各台車、SegwayPT・歩行者の前に設定してください。

※台車、SegwayPT・歩行者の設定より前に設定してください。

■移動軌跡の設定

設定方法: **MARKING**

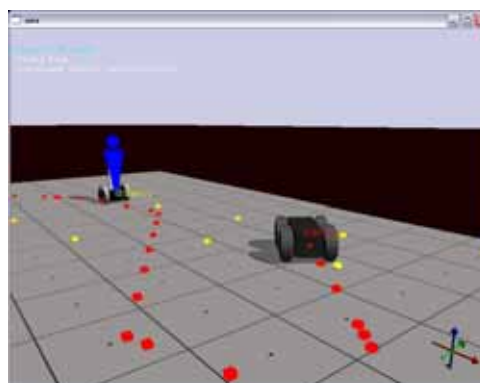
※複数に設定したい場合は各台車、SegwayPT・歩行者の前に設定してください。

※台車、SegwayPT・歩行者の設定より前に設定してください。

例: 以下の設定でシミュレータを起動すると、下図のように表示されます。

```
-----  
COLOR    1.0 0.0 0.0  
MARKING  
RMP400   127.0.0.1:55550
```

```
COLOR    1.0 1.0 0.0  
MARKING  
SEGWAY   a 2.0 0 0  
-----
```



■segwayPT・歩行者のランダム移動の設定

設定方法: **WALKRANDOM**

※複数に設定したい場合は各 SegwayPT・歩行者の前に設定してください。

※SegwayPT・歩行者の設定より前に設定してください。

3.6. 壁データ

壁データには 2 種類のフォーマットを採用しています。

※/bin/sample 内にサンプルがありますので参考にしてください。

■svg ファイルフォーマット

推奨ツール Inkscape / Illustrator

・対応要素

fill:透明(none)が推奨 ※黒の rect はスタート位置になります

rect、circle、ellipse、line、polyline、path (直線)

・黒 fill の rect はその真ん中がスタート位置 (面積の大きい順)

fill:#000000 もしくは指定なし(デフォルト)の場合

・スケールは、1.0 = 1cm

■dxf ファイルフォーマット(3次元壁データの定義)

推奨ツール Blender / Metasequoia / Maya

・対応要素

POLYLINE (3D)

3DFACE

※座標単位 m(メートル)

※BLOCK には対応していません。

※出力形式によっては読み込めないものがあります。

・LAYER 名に pos_NO_ANG のフォーマットを使用すると、その DXF オブジェクトの位置がロボット位置と方向になります。

例: pos_00_180, pos_01_90

特に指示のない場合には、設定ファイルに記載した位置が採用されます。

・Blender での推奨エクスポート(Autodesk DXF :標準で Blender に同梱)

<http://wiki.blender.org/index.php/Extensions:2.4/Py/Scripts/Export/DXF>

・Metasequoia での推奨エクスポート設定

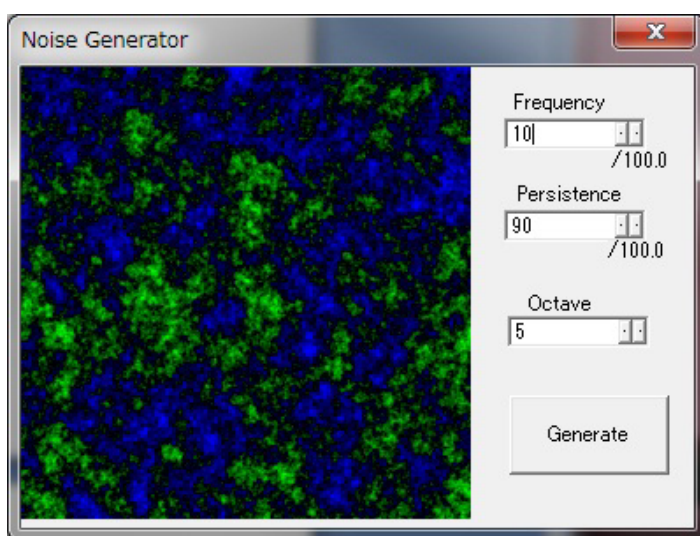
0.01 倍スケール

オブジェクト名をレイヤー名で出力

出力時座標変換:YZ 交換

<http://www.metaseq.net/>

3.7. GPS 誤差傾向確認用ソフト (GPSnoise.exe)



GPS センサ設定に使用する PerlinNoise のパラメータの確認ソフトです。

■ 設定内容

Frequency ノイズ周波数の初期値。重ねる数分倍になります。

Persistence Amplitude(強度)乗数。重ねる数分乗算されます。

Octave 重ねるノイズ数

※詳しくは一般の Perlin Noise の解説を参照ください。

■ 表示される画像について

緑: プラス方向

青: マイナス方向

※表示される画像のスケールは 1pixel=1m となり、上下 256m 左右 256m の誤差傾向が表示されます。

※GPS センサ設定の DilutionLength に設定される大きさが最大誤差となり、画像での一番明るい緑、青に相当します。

GPS センサの出力する緯度経度、HDOP と衛星数が誤差距離に応じて変化します。

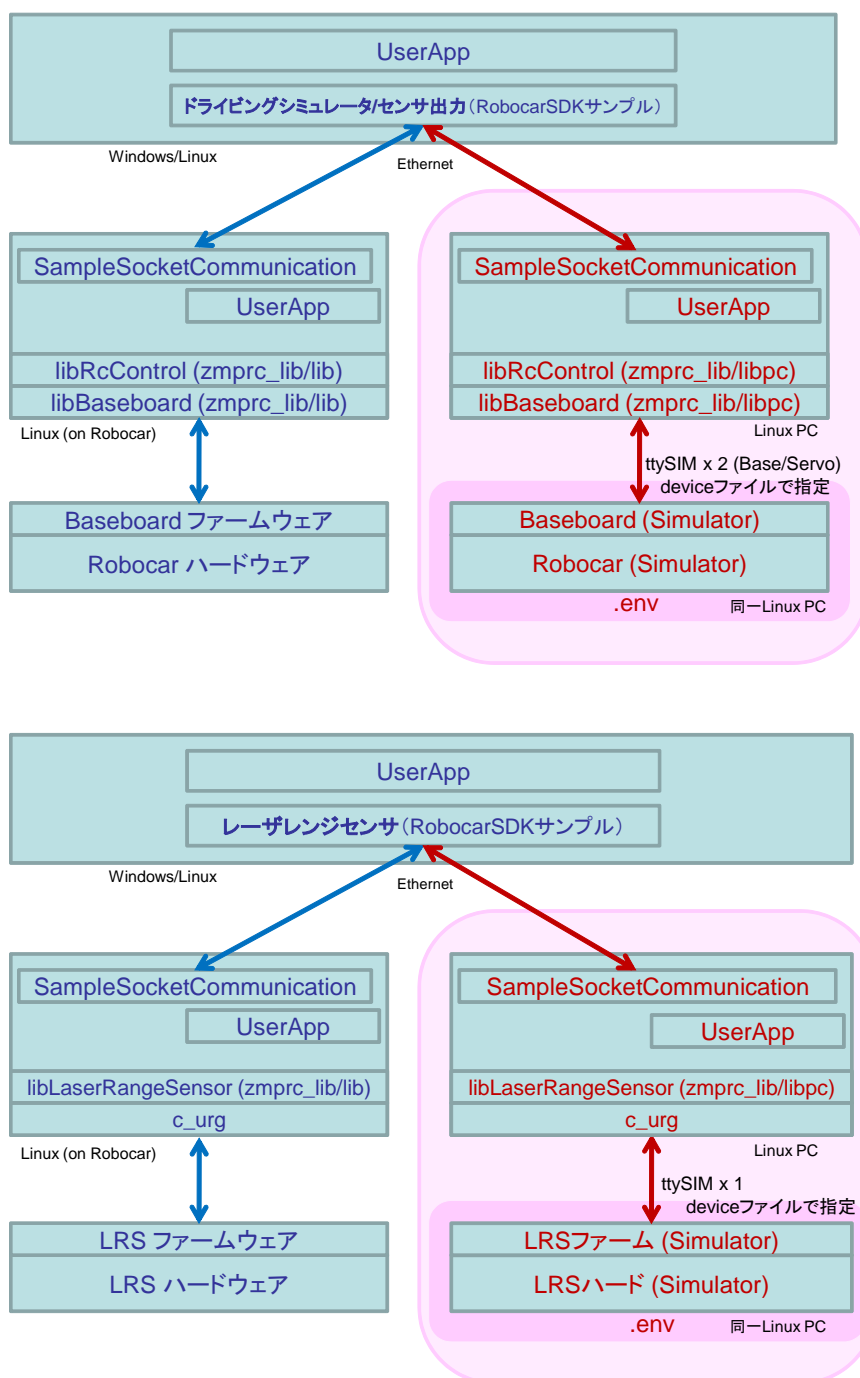
この際出力される HDOP は GPS センサ設定 HDOPlen を使用して以下のように計算されています。

$$\text{HDOP} = 1.0 + 7.0 * \text{誤差距離(m)} / \text{HDOPlen}$$

3.8. RoboCar に関して

RoboCar 搭載 CPU board の Linux 環境に相当する、LinuxPC (Ubuntu10.04 推奨) 上で動作する.env 環境シミュレータを構成します。

※RoboCar 搭載画像処理ボード及び RoboCar 搭載カメラのシミュレーションは行われません。



■ RoboCar 設定

/src/ttySIM 仮想シリアルドライバを介して、.env 環境シミュレータと RoboCar 標準ライブラリ libRcControl,libBaseboard, libLaserRengeSensor と接続します。

カーネルソースコードの入った linux にて /src/ttySIM フォルダで make を実行し、ttySIM*.ko カーネルオブジェクトを作成してください。

```
/src/robocar/Linux/zmpc_lib/include/*/
```

以下にあるファイルを以下にコピー

```
/usr/local/include
```

```
/src/robocar/Linux/zmpc_lib/lib/*/
```

以下にあるファイルを以下にコピー

```
/usr/local/lib
```

```
/src/robocar/Linux/zmpc_lib/libpc/*/
```

以下にあるファイルを以下に上書きコピー

```
/usr/local/lib
```

例として、/src/robocar/Linux/zmpc_sample/SampleDriveControl/Debug へ移動し make を実行して実行ファイルが出来たら環境の設定は完了です。

使用するデバイス名が書かれた「device」ファイルを同じフォルダに置きます。

device ファイルの内容は、

RS232 と ROBOCAR 本体

RS485 と ROBOCARSERVO

LRS と SCIP2CU が対応されます。

たとえば.env での robocar の設定が以下の場合には

```
-----  
ROBOCARSERVO    /dev/ttySIMb1  
ROBOCAR         /dev/ttySIMa1  
SCIP2CU         /dev/ttySIMc1 0 0 0.1075 0 0.13  
-----
```

device ファイルには以下のように記述します。

RS232	/dev/ttySIMa0
RS485	/dev/ttySIMb0
LRS	/dev/ttySIMc0

使用するデバイスをあらかじめインストールしておきます。

```
insmod ttySIM*.ko
```

上記の通りデバイスは1台の RoboCar につき 3 つ使用します。

lsmod コマンドを実行してインストールしたデバイスが表示されていることを確認してください。

.env を起動後に、サンプルの実行ファイルを実行することで RoboCar 標準サンプルが .env 環境シミュレータにて動作します。

/src/robocar/Windows/ 以下は RoboCar 標準の Windows サンプルソフトウェアになります。

/src/robocar/Linux/zmprc_sample/SampleSocketCommunication と Ethernet 経由で接続される Windows 版のサンプルソフトウェアになります。

RoboCar 標準のサンプルに関しては、RoboCar 標準のドキュメント

/src/robocar/Documents/ReferenceManual.zip を参照してください。

【注意】

src/robocar 以下に収録されているライブラリ・サンプルプログラム・マニュアル等

ReferenceManual

zmprc_lib

zmprc_samples

RczHostApp

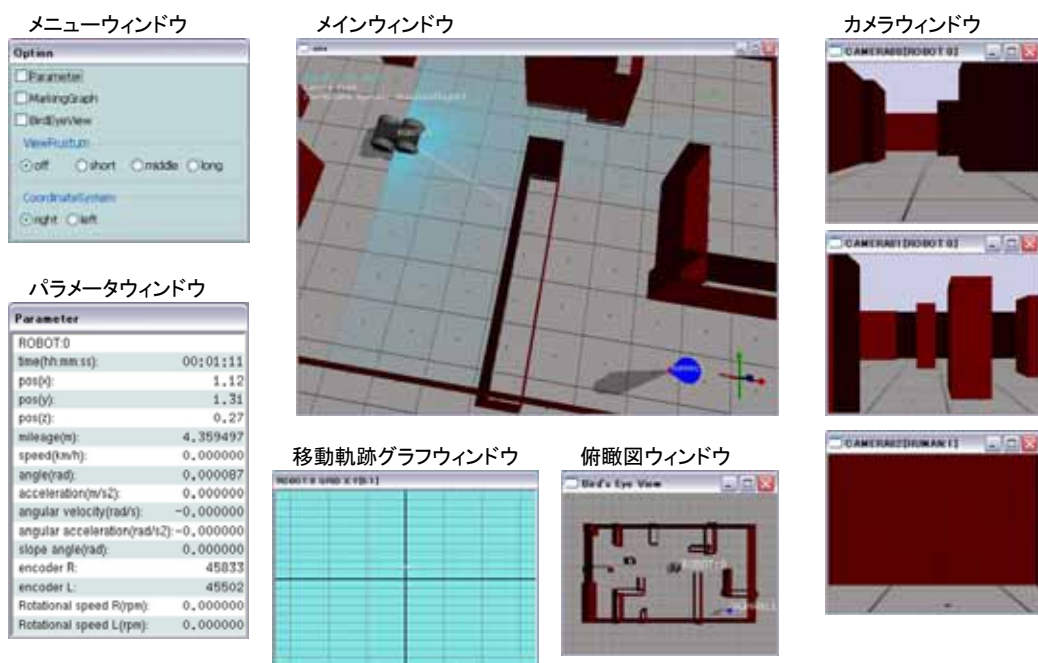
は株式会社ゼットエムピーの著作物になります。

株式会社ゼットエムピーはこれらのライブラリ・サンプルプログラムの使用時のサポートは行いません。

またこれらの再配布を行ってはいけません。

4. 画面・操作説明

.env 環境シミュレータを起動すると以下のウィンドウが表示されます。



■メインウィンドウ

キーボード・マウスでの操作が行えます。

カメラの視点を自由に操作が行えます。

SegwayPT・歩行者選択時に画面をクリックするとその地点へ移動します。

■マウス操作

マウスで操作が行える内容です。

- ・左ボタンを押したままドラッグ : 視点の向きを変えられます。
- ・右ボタンを押したままドラッグ : 視点を前後左右に移動できます。
- ・ホイールを押したままドラッグ : 視点を上下左右に移動できます。

■キーボード操作

メインウィンドウがカレント状態のときキー入力で操作ができます。

- ・Q :メニューウィンドウの表示・非表示を切り替えます。
- ・0～9 :番号に対応したオブジェクトにカメラの視点を合わせます。
- ・P+0～9 :番号に対応したオブジェクトのパッドで操作出来るようにします。
- ・R+0～9 :番号に対応した SegwayPT・歩行者をランダムに移動させます。
- ・M+0～9 :番号に対応した SegwayPT・歩行者をマニュアルで操作します。
- ・C+0～9 :番号に対応したカメラ視点をメインウィンドウに表示します。
- ・D+0～9 :番号に対応したオブジェクトのパラメータを表示します
- ・¥ :移動軌跡情報の保存(path ファイルで保存され、gnuplot でグラフを表示できます)
- ・N :メインウィンドウのカメラを切り替えます。
自由視点→注視点(オブジェクト 1)→(オブジェクト 2)→…→自由視点
- ・@ :表示座標系を切り替えます。
絶対座標系→相対座標系(オブジェクト 1)→(オブジェクト 2)→…→絶対座標系

■カメラウィンドウ

設定ファイルで設定されたカメラが表示されます。

■メニューウィンドウ

オプションで機能の設定が行えます。

- ・Parameter :パラメータウィンドウの表示・非表示が切り替えられます。
- ・MarkingGraph :移動軌跡グラフウィンドウの表示・非表示が切り替えられます。
- ・BirdEyeView :俯瞰図ウィンドウの表示・非表示が切り替えられます。
- ・ViewFrustum :視錐台の可視化を設定します。
- ・CoordinateSystem :右手・左手座標系の設定が行えます。

■パラメータウィンドウ

オブジェクトのパラメータを表示します。

•name	: 現在表示しているオブジェクト名と番号
•time	: シミュレータを起動してからの時間
•pos	: 表示座標系の設定に応じた座標
•mileage	: 初期位置からの移動距離
•speed	: 現在の速度
•angle	: 表示座標系の設定に応じたオブジェクトの左右の角度
•acceleration	: 現在の加速度
•angular velocity	: 現在の角速度
•angular acceleration	: 現在の角加速度
•slope angle	: 表示座標系の設定に応じたオブジェクトの上下の角度
•encoder	: エンコーダ値
•Rotational speed	: 車輪の回転速度

■移動軌跡グラフウィンドウ

移動した軌跡をグラフ表示します。

表示座標系によって軌跡の位置が変更します。

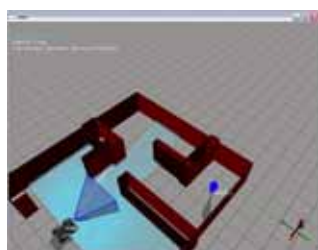
グラフを保存することが出来ます。※キーボード操作を参照ください。

■俯瞰図ウィンドウ

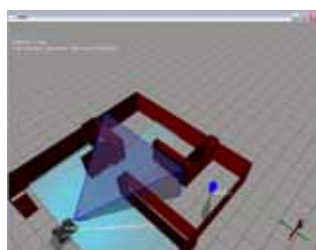
シミュレート環境の俯瞰図を表示します。

■視錐台の設定

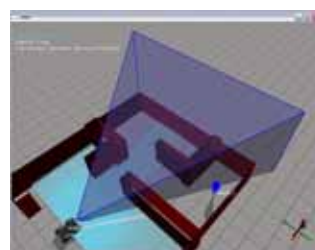
視錐台の可視化を設定します。



short



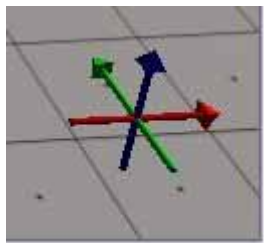
middle



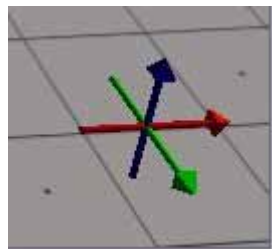
long

■ 右手・左手座標系設定

右手・左手座標系の設定が行えます。(赤:X、 緑:Y、 青:Z)



右手座標系



左手座標系

5. チュートリアル

5.1. サンプルインターフェイスの動作

「.env」に標準で付属するサンプルインターフェイスを動かしてみます。

サンプルインターフェイスを使うことで、「.env」内の仮想デバイスと、それに対応する実機とに接続することができます。ここでは例として Blackship2 輪版を使います。

(1) 「.env」の起動。

ゲームパッドが装着されていることを確認した後に、

Windows の場合には

```
bin¥sample¥startsocko.bat
```

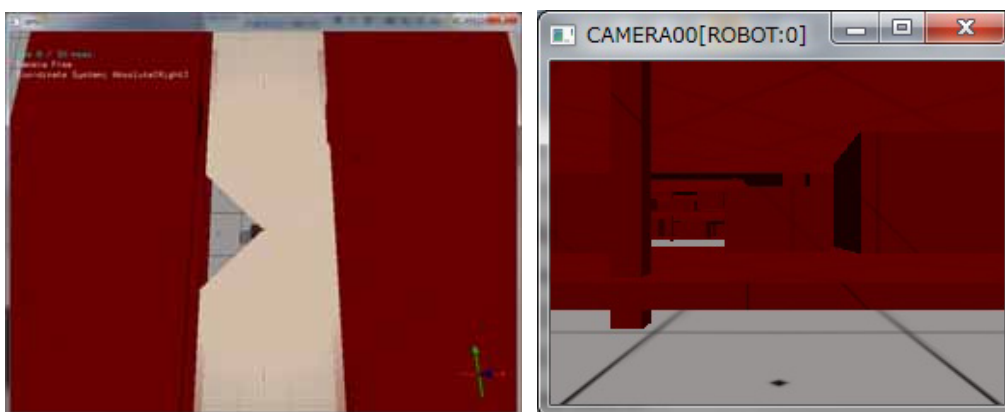
をダブルクリックして実行してください。

Linux の場合には bin/sample/ へ移動して

```
sh ./startsocko.sh
```

を実行してください。

倉庫の環境と、Blackship2 輪版(BS2W)の移動台車にレーザーレンジセンサ(SCIP2TU)と、カメラ(CAMERA)が搭載されている状態になります。



(2) Blackship2 輪版を制御するために、サンプルインターフェイスを使ったテストプログラムの実行ファイルを作成します。

```
src¥blackship¥blackshiptest
```

のフォルダへ移動し、

Windows の場合には blackshiptest.sln を開きビルドしてください。

Linux の場合にはターミナルにて make を実行して実行ファイルを作成してください。

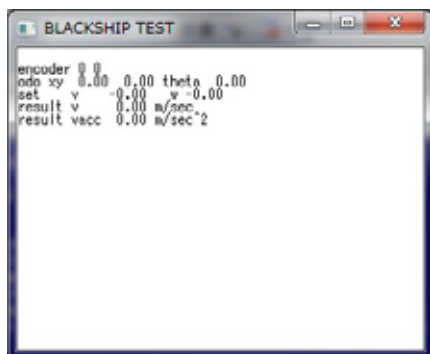
blackship と同様に src 以下に各デバイスのサンプルインターフェイスがあります。

- (3) 作成した実行ファイルを起動します。

実行する際に、パラメータに

127.0.0.1:55550

を指定して起動してください。



```
BLACKSHIP TEST
encoder 0 0
pos xy 0.00 0.00 theta 0.00
set v -0.00 w -0.00
result v 0.00 m/sec
result vacc 0.00 m/sec^2
```

テストプログラムのウィンドウが表示されます。

「.env」との接続が成功していれば、この状態でゲームパッドで「.env」内の Blackship2 輪版を操作することができます。また「.env」内の Blackship2 輪版から取得したステータスが表示されます。

デバイスに対するポートが、「.env」を起動する際に読み込まれている settingblackship2w.env に指定されています。同様に他のパラメータも 環境設定ファイルである .env ファイルで指定されます。デバイス名、ポート、位置、パラメータ等が指定されていますのでテキストエディタで開き、内容を確認しておいてください。

- (4) 実機に接続する。

サンプルの起動時に指定するパラメータを、Blackship2 輪版の実機のデバイス名に変更してください。

Windows の場合には COM*

Linux の場合には /dev/ttyS*や/dev/ttyUSB*

などになるとおもいます。

実機との接続に成功していれば、「.env」内の Blackship2 輪版と同様に、動作指示とステータスの取得が出来ることが確認できます。

(おわり)

5.2. RT-Middleware による人追従機能

次世代ロボット知能化技術開発プロジェクトで開発された、RT-Middleware([URL](#))で動作する人追従機能を「.env」で動かしてみます。

(1) intellipj¥Following¥following¥sj¥doc にあります「人追従機能モジュール群.pdf」の 10 ページ や、上記 RT-Middleware の URL を参考にして、C++版の RT-Middleware とその GUI ツール環境である RTSystemEditor をインストールしてください。

(2) インストールが完了しましたら、RT-Middleware のネームサーバーを起動します。

Windows の場合には

[スタートメニュー]→[すべてのプログラム]→[OpenRTM-aist]→[C++]→[tools] →[Start Naming Service] を実行します。

Linux の場合にはデフォルトで OS 起動時にネームサーバー(omniORB)が起動しています。

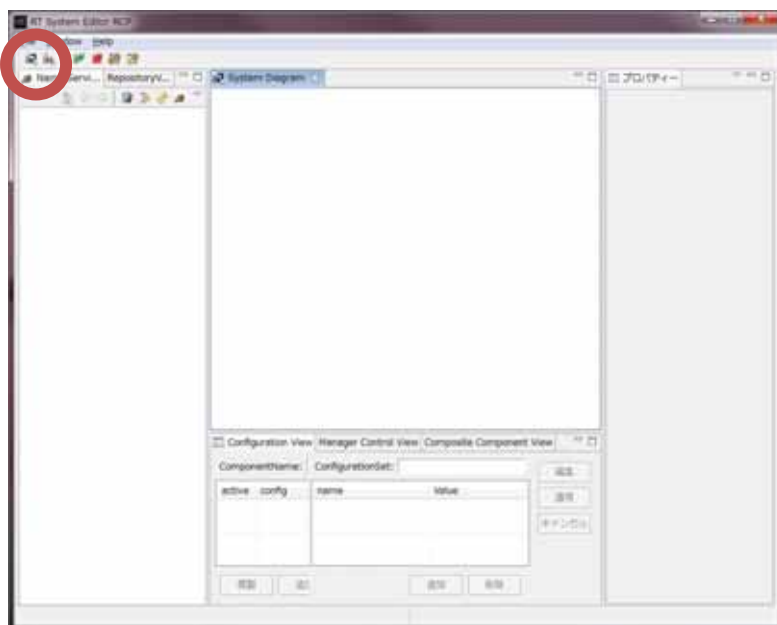
※デフォルトでのネームサーバーのホストとポートは 127.0.0.1:2809 となっています。

(3) RT-SystemEditor を起動します。

Windows の場合には

[スタートメニュー]→[すべてのプログラム]→[OpenRTM-aist]→[C++]→[tools] →[RT System Editor] を実行します。

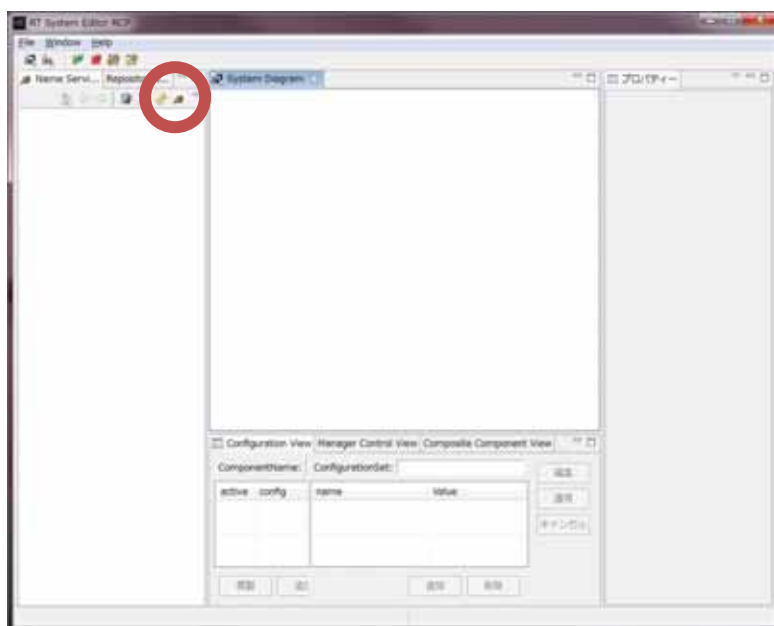
Linux の場合には(1)にて RTSystemEditor と共にインストールした eclipse を直接起動します。



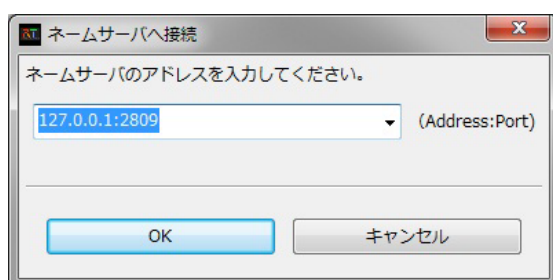
起動しましたら、左上の[on]のボタンを押して中央の SystemDiagram を表示してください。

(4) ネームサーバーを指定します。

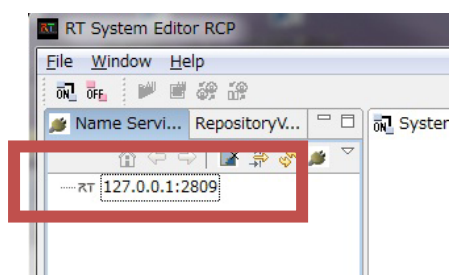
上部にある「ネームサーバを追加」ボタンを押します。



以下IPとポートを入力するウィンドウが表示されますので、127.0.0.1:2809 と入力し OK 押します。



成功すると左側ウィンドウに[RT 127.0.0.1:2809] と表示されます。



以上で RT-Middleware と RTSystemEditor の準備が完了します。

- (5) 「.env」と、RT-Middleware ソフトウェアモジュール(RTC:RT コンポーネント)を起動します
ゲームパッドが装着されていることを確認した後に、

Windows の場合には

```
bin¥sample¥startfollowing.bat
```

を実行してください。

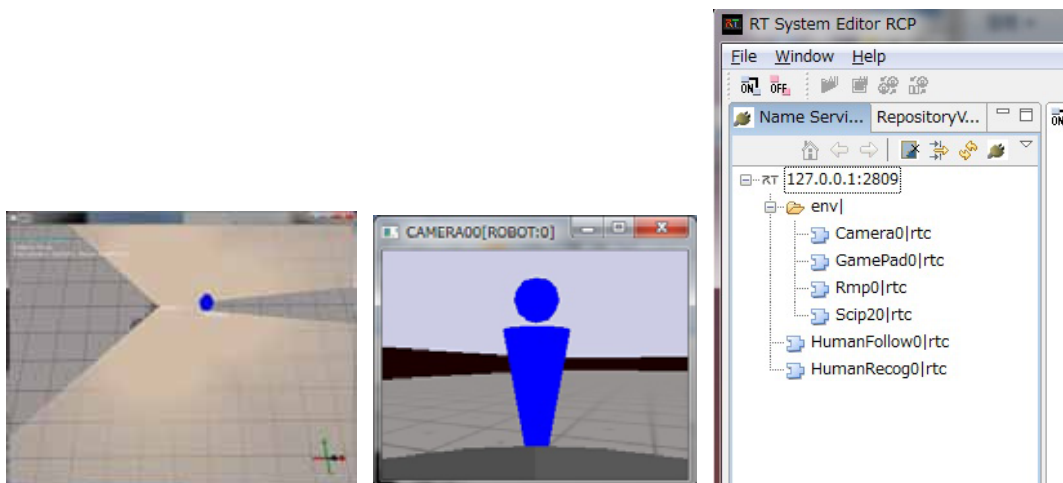
Linux の場合には bin/sample へ移動して

```
sh ./startfollowing.sh
```

を実行してください。

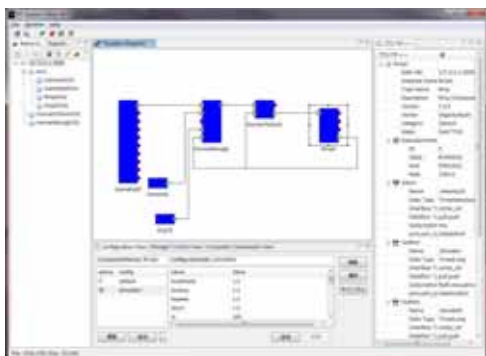
「.env」と共に「デバイス RTC」と「人追従機能 RTC」が同時に起動されます。

「.env」の画面と、RTSystemEditor で以下のように表示されれば成功です。 env 以下にリストされているものが「.env」付属の「デバイス RTC」、Human*の名前が付いているものが「人追従機能 RTC」になります。

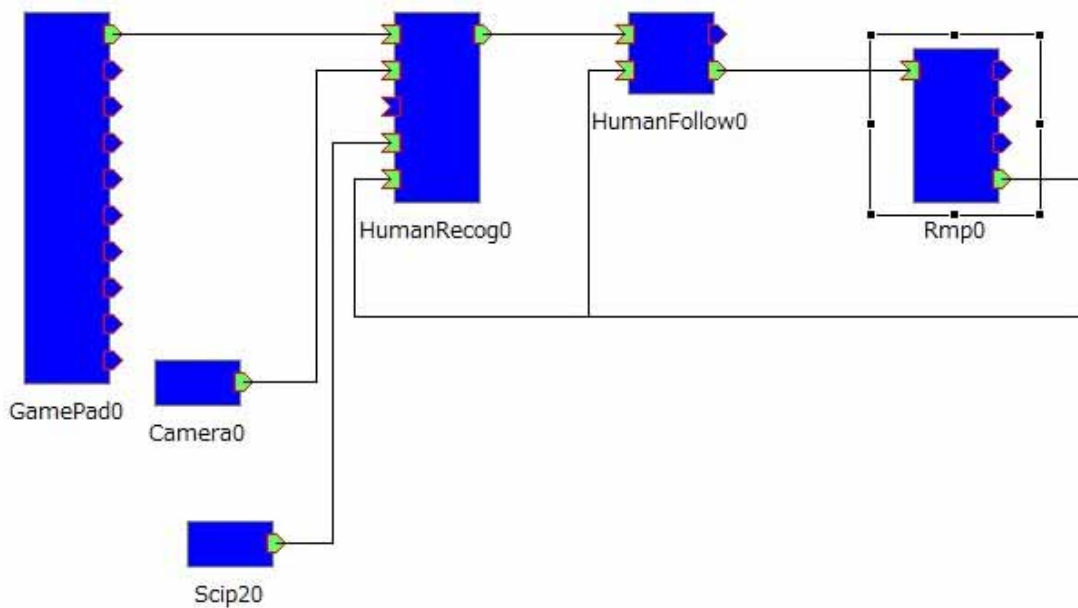


- (6) RTSystemEditor で結線します。

左のリストより各 RTC をドラッグアンドドロップして SystemDiagram 上に配置します。



配置をしたら、下図の通りに配線をしてください。

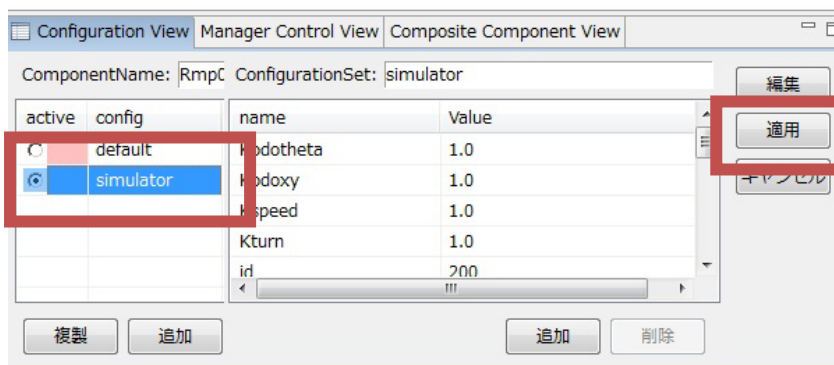
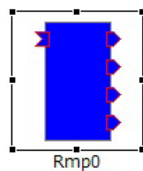


(7) 「デバイス RTC」のコンフィギュレーションにて[simulator]を選択し、適用します。

Camera, Scip2, Rmp

をそれぞれ選択し、下段の ConfigurationView にて[simulator]を選択、[適用]ボタンを押します。

3つの「デバイス RTC」に対して行います。



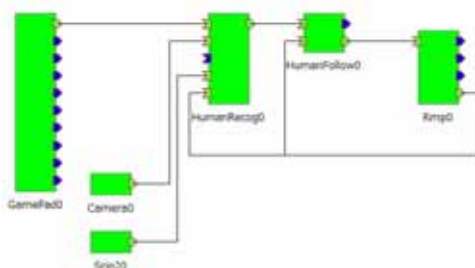
なお、「.env」付属の「デバイス RTC」は前述のチュートリアルで説明したサンプルインターフェイスを用いて作られているため、実機にもそのまま接続することができます。実機に接続する際には、Configuration の str_port に実機のデバイス名を入れて、[適用]ボタンを押してください。

(8) Activate をして動作開始します。

SystemDiagram 上で右クリックをして、[All Activate]を選択します。



SystemDiagram がすべて緑色になれば成功です。

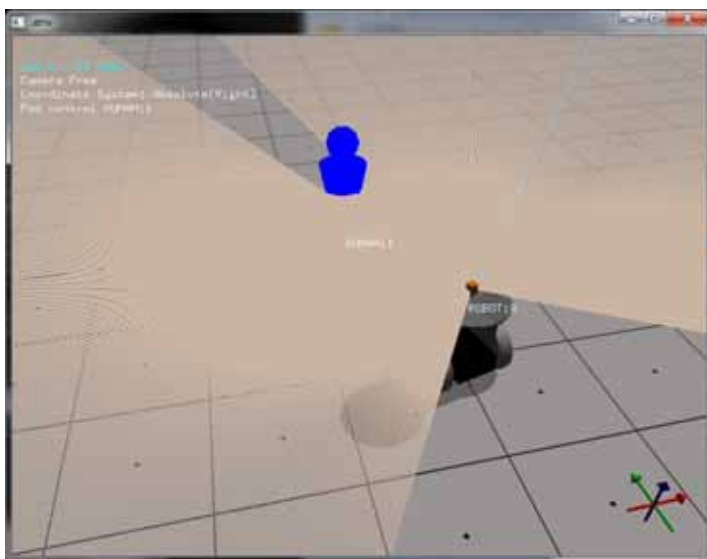


(9) ゲームパッドのボタンを押して、追従開始になります。ボタンを押した際にカメラ画像真ん中付近の色ヒストグラムが登録されます。HumanRecogComp の出力で match 1 となっていることを確認してください。



```
RPosIIS 0.348446 0.613358 1.976653  
HPosIIS -0.244157 1.992424 1.976653  
human 0.000000 1.501000 1.570796 match 1 24095.906250 / 17202.494141
```

(10) シミュレータ内の人をゲームパッドで操作してロボットが人の後を追従するのを確認してください。



(おわり)

5.3. SRI KARTO SDK の動作

SRI KARTO SDK Windows 版の地図生成機能(Mapper)を「.env」で動かしてみます。

ここではSRI KARTO SDKをSRI Internationalのサイト([URL](#))よりしかるべき形式でダウンロードしてインストールしてあることを前提とし、また「.env」の置かれている場所と並列にあるkartoというフォルダの中にインストールされているとして進めます。(src¥KARTOtest内にあるWindowsのプロジェクト構成もそのような形で収録されています。Linux環境では、SRI KARTO SDK のLinux版のインストールドキュメントを参照してください)

※ここでは Microsoft Visual C++2010 Express にて動作確認しています。

- (1) SRI KARTO SDK を動作させるのに必要な DLL やライセンスファイルを、src¥KARTOtest へコピーしてください。
- (2) 「.env」を起動します。

Windows の場合には

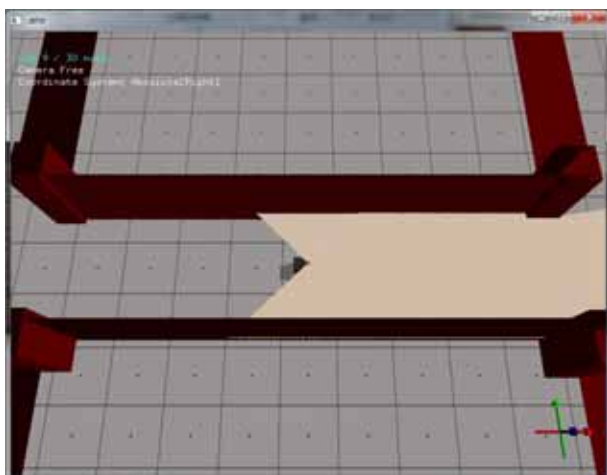
```
bin¥sample¥startkartotest.bat
```

を実行してください。

Linux の場合には bin/sample/ へ移動して

```
sh ./startkartotest.sh
```

を実行してください。



BS2W と SCIP2TU がそれぞれ

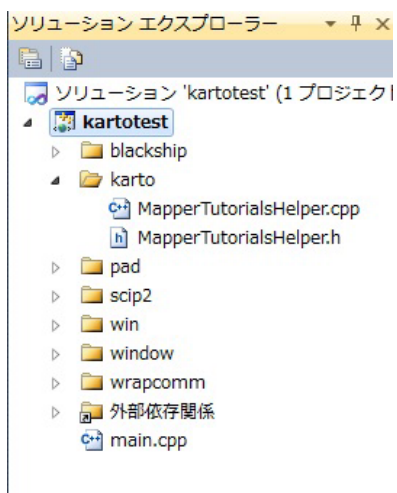
```
127.0.0.1:55550
```

```
127.0.0.1:55553
```

のポートにて起動されます。

- (3) テストプログラムをビルド、起動します。

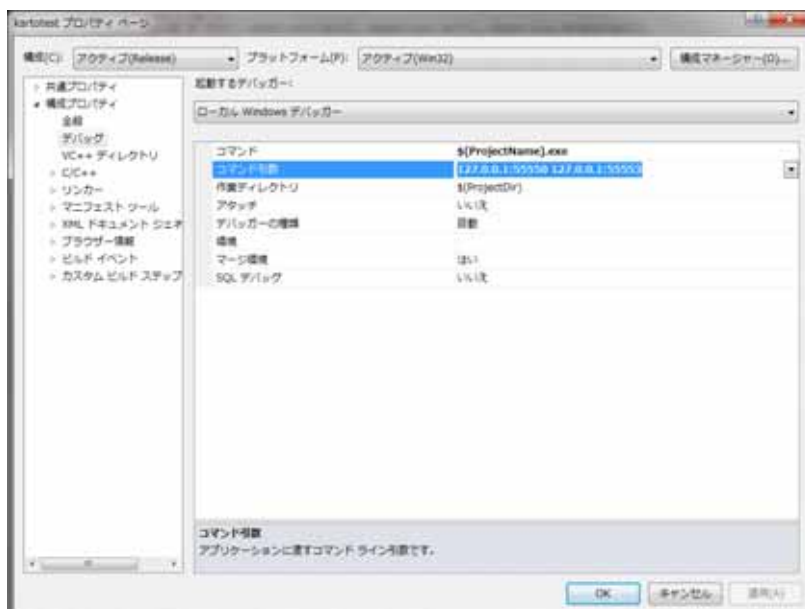
src\KARTOtest にある、kartotest.sln を開くとソリューションエクスプローラーには以下のように表示されます。そのままビルドしてください。SRI KARTO SDK を別の場所にインストールした場合にはプロジェクトの設定を適宜変更してください。



- (4) ポートを指定します。

プロジェクトのプロパティの「デバッグ」→「コマンド引数」にて、さきほどのポートを指定します。

「127.0.0.1:55550 127.0.0.1:55553」

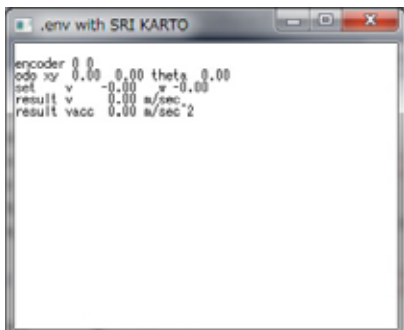


なお、コマンドプロンプトで実行ファイルを直接実行する場合には、2つのポートをパラメータに指定してください。

➤ kartotest.exe 127.0.0.1:55550 127.0.0.1:55553

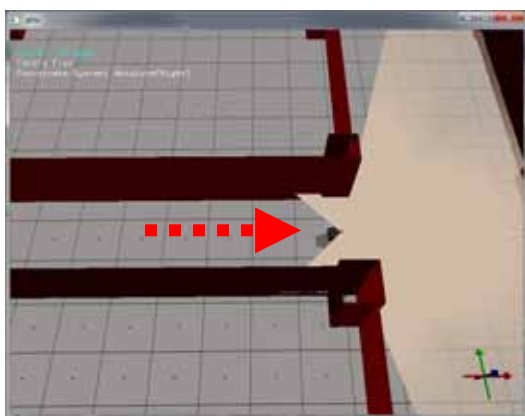
(5) [F5]を押して、実行します。

以下のようなウィンドウが表示されれば成功です。



```
.env with SRI KARTO
encoder 0 0
ode xy 0.00 0.00 theta 0.00
set v -0.00 x -0.00
result v 0.00 m/sec
result vacc 0.00 m/sec^2
```

ゲームパッドを使って Blackship2 輪版を移動させて、

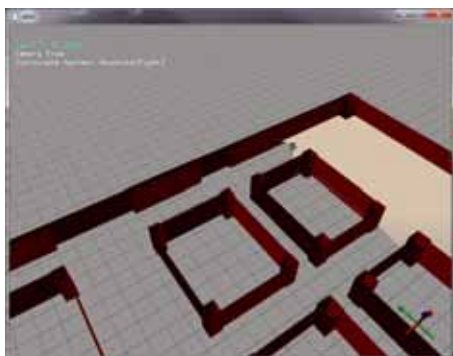


適当な場所にて、ゲームパッドの「ボタンO」を押すと、KARTO SDK がそれまでのレーザーレンジセンサと Blackship2 輪版のオドメトリの値を使って地図画像を生成します。



このような画像が Map.png というファイル名で生成されたら成功です。

そのまま広く移動しながら地図を更新していきましょう。生成される地図画像も広い範囲になっていきます。



SRI KARTO SDK は地図生成機能(Map Generation)に始まり、それを使用した位置推定機能(Localization)や未開地探索機能(Exploration), 経路計画機能(PathPlanning)など、移動に関するいろいろな機能を備えていますので、それらを「.env」と合わせて動作、検証していくことで、開発の効率やアプローチの仕方が向上することでしょう。

(おわり)

6. お問い合わせ

「.env」 お問い合わせ窓口

dotenv@segway-japan.co.jp