RTミドルウエア
コンテスト2007

プレゼンテーション資料集

http://www.is.aist.go.jp/rt/RTMcontest

SICE システムインテグレーション部門講演会（SI2007）

2007年12月22日

広 島国際大学 広島キャンパス ＆ RCC文化センター

2008.01.06版

分散制御ロボットにおける CANコンポーネント

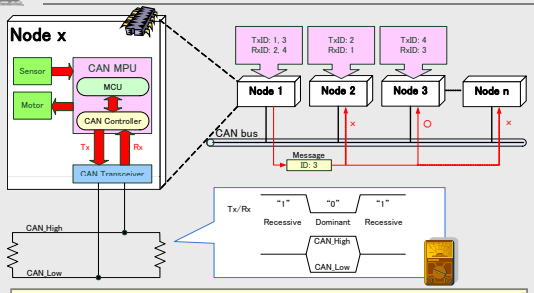
芝浦工業大学　水川研究室

○ 三浦俊宏 (Toshihiro MIURA)
水川真 (Makoto MIZUKAWA)

目次

1. CANとは

1. CANとは



通信方式採用によるメリット

- ✓ 配線本数の削減
- ✓ コントローラ(Node)の小型化
- ✓ 設計自由度の向上
- ✓ 電気的信頼性の向上
- ✓ データ通信以外への用途拡大

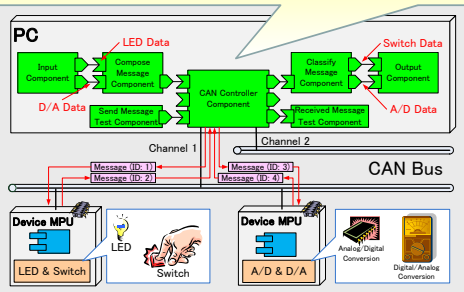
2. 目的

CAN通信技術の共有・蓄積への貢献を目的とし実装を行った.

3. CAN Componentsの概要

CAN Componentsとは？

CAN通信機能要素をRT-Component化したCAN Controller Componentとこれの動作確認及び評価を行うために作成した他のRT-Componentを含めたもの.



4.1. CAN Controller Componentの仕様

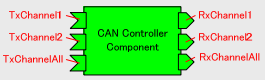
- 概要 -
2つのCANバスにCANメッセージを送受信することが可能で動的に設定できる受信メッセージのフィルタ機能を実装している.

使用機器
CAN通信機能要素をRT-Component化するにあたり，ベクター・ジャパン社製XLファミリー製品を使用しCAN Controller Componentの実装を行った.


CANcardXL

CANcardXL の特徴
> PCカード (Type II)
> 64MHzの32ビット マイクロコントローラ
> 2つの完全に独立したチャンネル
> CAN 2.0BおよびLIN
> CANcab/LINcabでのバス トランシーバ
> プラグ&プレイ

4.1. CAN Controller Componentの仕様



- Portの仕様 -
各InPort, OutPortは，CANバスに送受信するメッセージデータを入出力としており，このデータ長はCANメッセージ長を示すDLCの値によって変わる.

Port の仕様

ポート種別	ポート名称	データ型	説明
InPort	TxChannel1	TimedOctetSeq	入力されたデータをCANバスのチャンネル1に送信する.
	TxChannel2	TimedOctetSeq	入力されたデータをCANバスのチャンネル2に送信する.
	TxChannelAll	TimedOctetSeq	入力されたデータをCANバスのチャンネル1と2に送信する.
OutPort	RxChannel1	TimedOctetSeq	CANバスのチャンネル1で受信したデータを出力する.
	RxChannel2	TimedOctetSeq	CANバスのチャンネル2で受信したデータを出力する.
	RxChannelAll	TimedOctetSeq	CANバスのチャンネル1及び2で受信したデータを出力する.

InPort の仕様 (DLC=8 の場合)

データ名称	Bit数			
	8bit	8bit	8bit	8bit
ID	data[3]	data[2]	data[1]	data[0]
DLC	data[4]			
D0	data[5]			
D1	data[6]			
D2	data[7]			
D3	data[8]			
D4	data[9]			
D5	data[10]			
D6	data[11]			
D7	data[12]			

OutPort の仕様 (DLC=8 の場合)

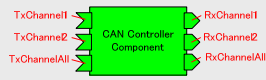
データ名称	Bit数							
	8bit	8bit	8bit	8bit	8bit	8bit	8bit	8bit
	data[16]	data[15]	data[14]					

InPort の仕様 (DLC=4 の場合)

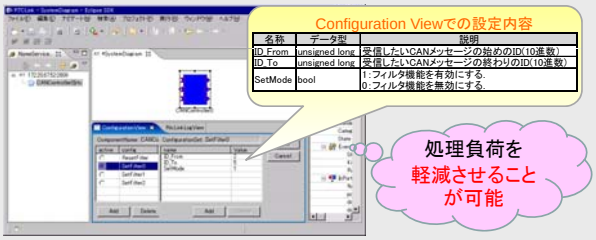
データ名称	Bit数			
	8bit	8bit	8bit	8bit
ID	data[3]	data[2]	data[1]	data[0]
DLC	data[4]			
D0	data[5]			
D1	data[6]			
D2	data[7]			
D3	data[8]			

って変化する

4.1. CAN Controller Componentの仕様

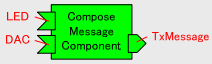


- 受信メッセージのフィルタ機能 -
RTCLink上で受信したいCANメッセージのIDを動的に設定することが可能となっている.



名称	データ型	説明
ID_From	unsigned long	受信したいCANメッセージの始めのID(10進数)
ID_To	unsigned long	受信したいCANメッセージの終わりのID(10進数)
SetMode	bool	1:フィルタ機能を有効にする. 0:フィルタ機能を無効にする.

4.2. Compose Message Componentsの仕様



- 概要 -
入力されたデータに対し，あらかじめ設定しておいたID及びDLCの設定を行いメッセージデータの生成を行う.

TxMessageの仕様 (DLC=8 の場合)

データ名称	Bit数			
	8bit	8bit	8bit	8bit
ID	data[3]	data[2]	data[1]	data[0]
DLC	data[4]			
D0	data[5]			
D1	data[6]			
D2	data[7]			
D3	data[8]			
D4	data[9]			
D5	data[10]			
D6	data[11]			
D7	data[12]			

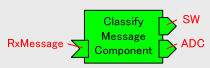
- プログラム -

```
RTC::ReturnCode_t ComposeMessage::onExecute(RTC::UniqueId ec_id)
{
                :
                :
  // ID=1, DLC=1のCANメッセージを生成してTxMessageから出力したい場合
  // (LEDのデータからCANメッセージを生成して出力する場合)
  m_LEDIn.read();
  CANTxMessageFrame CANTxMessageTmp;
  memset(&CANTxMessageTmp, 0, sizeof(CANTxMessageTmp));// 0クリア

  CANTxMessageTmp.ID = 1;          // IDの設定
  CANTxMessageTmp.DLC = 1;         // DLCの設定
  CANTxMessageTmp.Data[0].ByteAccess = m_LED.data; // Dataの設定

  m_TxMessage = ComposeTxMessage(CANTxMessageTmp); // 生成
  m_TxMessageOut.write();          // LEDのデータを出力
                :
                :
  return RTC::RTC_OK;
}
```


4.3. Classify Message Componentsの仕様



- 概要 -
CAN Controller Componentから出力されたメッセージデータを振り分けて対応するOutPortへ出力する.

RxMessageの仕様 (DLC=8 の場合)

データ名称	Bit数							
	1bit	1bit	1bit	1bit	1bit	1bit	1bit	1bit
ID	data[31]	data[30]	data[29]	data[28]				
DLC	data[4]							
D0	data[5]							
D1	data[6]							
D2	data[7]							
D3	data[8]							
D4	data[9]							
D5	data[10]							
D6	data[11]							
D7	data[12]							
MessageType	data[13]							
TimeStamp	data[20]	data[19]	data[18]	data[17]	data[16]	data[15]	data[14]	

データ名称	Bit数							
	1bit	1bit	1bit	1bit	1bit	1bit	1bit	1bit
Message Type			Tx/Rx	Channel種別				

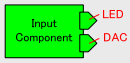
データ名称	説明
Tx/Rx	"1"の場合は送信、"0"の場合は受信したメッセージである.
Channel種別	"0"の場合は"CANcardXL Channel1"からのメッセージ. "1"の場合は"CANcardXL Channel2"からもメッセージ. "2"の場合は"Virtual Channel1"からのメッセージ. "3"の場合は"Virtual Channel2"からのメッセージ (※これはCANcardXLだけを使用した場合)

- プログラム -

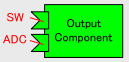
```
RTC::ReturnCode_t ClassifyMessage::onExecute(RTC::UniqueId ec_id)
{
                :
  // 受信したCANメッセージを振り分ける
  // (スイッチのデータを振り分けて出力する場合)
  CANRxMessageFrame CANRxMessageTmp;
  memset(&CANRxMessageTmp, 0, sizeof(CANRxMessageTmp));
  m_RxMessageIn.read();          // 受信したデータの読み込み
  CANRxMessageTmp = ClassifyRxMessage(m_RxMessage);  // 生成

  switch (CANRxMessageTmp.ID) // IDごとに分ける
  {
    case 2:                      // ID=2の場合
      m_SW.data = CANRxMessageTmp.Data[0].ByteAccess; // データの設定
      m_SWOut.write();           // スイッチのデータを出力
      break;
                :
  }
                :
  return RTC::RTC_OK;
}
```

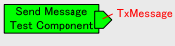

4.4. その他のRT-Compoentの仕様



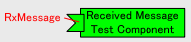
Input Component
コンソール画面で入力したLED及びD/A変換の指令値を出力する.



Output Component
入力されたデータをコンソール画面に出力する.



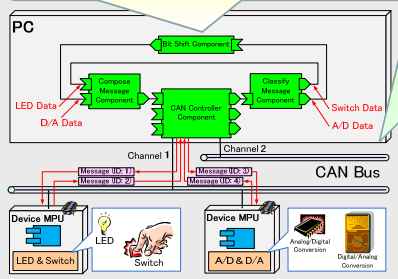
Send Message Test Component
コンソール画面で入力したID及びDLC，データからCAN Controller Componentに出力するメッセージデータの生成を行い出力する.



Received Message Test Component
CAN Controller Componentで出力されたメッセージデータをコンソール画面に出力する.

5. 動作確認

Bit Shift Component
PSoCは，A/D変換を12[Bit]でD/A変換を9[Bit]で行っている．そのため，"Bit Shift Component"では入力されたデータを3[Bit]シフトさせ（下位3[Bit]切捨て）出力している．



使用機器
◆ ノートPC (W2)
Pentium M : 900[MHz]
Memory : 512[MB]

◆ CANcardXL

◆ PSoC CAN
PSoC : CY8C29466
CAN Controller Chip : MCP2515

6. 使用例

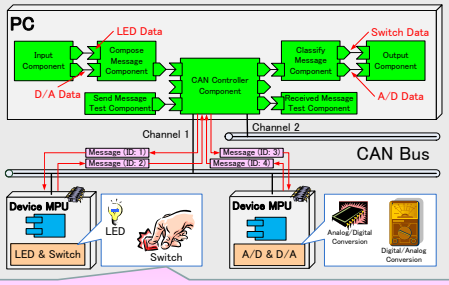

PAR-NE07

PAR-NE07 (Physical Agent Robot for Natural Environment)
屋外環境でロボットを運用するための技術獲得を目的に本研究室で開発を行い『つくばチャレンジ』に参加したロボット．
同研究室からコンテストに参加している『屋外ナビゲーションにおけるGPSコンポーネント』のRT-Componentと組み合わせることで目的地まで自律でロボットが動く．


クリスマスツリー

クリスマスツリー
クリスマスツリーをCAN通信で制御している．
LED1個ずつを周期的に制御し光の強度を変化させている．

7. 公開資料関連



公開資料
✓ CAN Componentsのソースコード ・・・ Visual Studio 2005
✓ ユーザーマニュアル ・・・ CAN Componentsの仕様・使用方法等を記載
✓ 概要資料 ・・・ CAN Componentsの概要について記載

8. ライセンス・環境について

ライセンス等ついて
CANインターフェイスを制御するAPIのソースコード等を除き，CAN Componentの著作権は，芝浦工業大学水川研究室に帰属します．
そして，APIのソースコード等の著作権は開発元の"Vector Informatik"にあります．

開発環境
- Windows XP
- Microsoft Visual Studio 2005 (VC++8.0)
- RT-Middleware (OpenRTM-aist-0.4.0)
- CANcardXL（ベクター・ジャパン社製XLファミリー製品）

動作確認環境
- Windows XP (Visual Studio 2005でコンパイル)
- RT-Middlware (OpenRTM-aist-0.4.0)
- CANcardXL（ベクター・ジャパン社製XLファミリー製品）
- CANboradXL（ベクター・ジャパン社製XLファミリー製品）

9. まとめ

◆ CAN通信機能要素をRT-Component化するにあたり，ベクター・ジャパン社製XLファミリー製品を使用しRT-Component化した．

◆ CAN Componentsの仕様・構成について述べた．

◆ CAN ComponentsのCompose Message ComponentとClassify Message Componentを作りかえることによって容易にDevice MPUの変更・機能拡張などに対応することが可能である．

◆ 以下の資料を提供することを述べた．
✓ ソースコード ・・・ Visual Studio 2005のプロジェクトで提供
✓ ユーザーマニュアル ・・・ 仕様及び使用方法等を記載
✓ 概要資料 ・・・ CAN Componentsの概要について記載

分散制御ロボットにおけるCANコンポーネント

芝浦工業大学　水川研究室

○ 三浦俊宏 (Toshihiro MIURA)
水川真 (Makoto MIZUKAWA)

PSoC CAN

PSoC CAN

PSoC CANのコンセプト

✓回路を学生が簡単に作成することが可能
✓部品は低価格であり入手性が良いものを使用
✓簡単にCAN通信をすることが可能
✓PSoCを使用しているため容易にセンサを使用したノードを作成することが可能

PSoC CANの仕様

PSoC：CY8C29466(他のPSoCにも変更可能)
CAN Controller Chip：MPC2515
CAN Transceiver Chip：PCA82C250
入力電圧：5.2～12[V]

PSoCとは

PSoCはプログラマブルアナログ・デジタル混載マイコンである.
各機能はモジュールの組み合わせにより実現可能, 動的に再構成することも可能

PSoC CAN開発について

芝浦工業大学　水川研究室
指導教員:水川　真
ソフト開発:九十九里　洋介
回路設計:高木　和貴

OpenCV を使った画像処理コンポーネントの作成例

○田窪　朋仁（大阪大学）

発表概要
- ・初心者のRTM導入例（開発環境紹介）
- ・OpenCVライブラリのコンポーネント化
- ・コンポーネント郡を組み合わる（実演）

初心者のRTM導入例

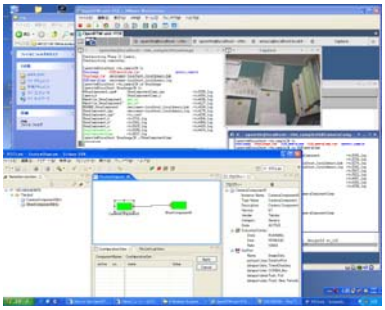
- 超初心者のRTミドルウェア導入例

手軽にRTMの魅力を体験できる例題を提供
- OpenCVライブラリを使ったUSBカメラ画像取得と映像表示
- http://www-arailab.sys.es.osaka-u.ac.jp/~takubo/howto_rtminst.html

開発環境：

・産総研オフィシャルHP提供VMWareイメージ FedraCore6
- ・OpenCVをインストール
- ・USBカメラ

初心者のRTM導入例



Windows上での実行例

コンポーネントの構成

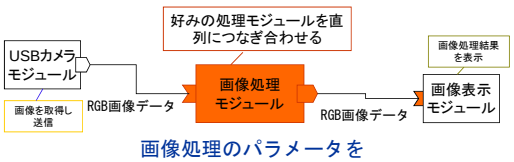
- USBカメラコンポーネント
  - 出力ポート１：TimedCharSeq でRGBカラー画像
  - 画像をキャプチャし、送信する機能のみ
- 画像表示コンポーネント
  - 入力ポート１：TimedCharSeq でRGBカラー画像
  - 画像を受け取りウィンドウに表示する機能のみ

OpenCVライブラリのコンポーネント化

１３種類のコンポーネントを作成
- **カメラキャリブレーション**：カメラの内部パラメータ計算
- **背景差分：モジュール**を起動した時を基準とした差分
- **フレーム差分**：フレーム間差分により動体を表示
- **テンプレートマッチング**：テンプレートに近い画像を探索
- **オプティカルフロー**：区間画像の動きをベクトルで示す
- **ハフ変換**：画像内の直線らしき場所を検出
- **閾値処理**：カラー画像をある輝度値で２値化する．
- **回転**：画像の回転だけ．あまり使い道はない．
- **膨張と拡大**：モフォロジー処理を行える．
- **平滑化**：雑音の除去．このあとに他の処理をするといいことがあるかもしれません．エッジは甘くなる．
- **エッジ画像**：カラー画像を入れるとモノクロのエッジ画像になります．
- **グレイ画像**：カラー画像をグレイ画像に変換します．
- **輪郭抽出**：閾値処理後の画像を入れると輪郭を抽出してくれます．

OpenCVライブラリのコンポーネント化

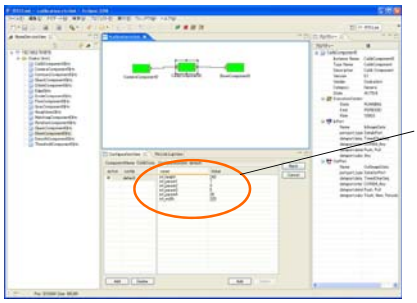
- 各モジュールのデータ送受信は画像情報のみ
  - 出力ポート１：TimedCharSeq でRGBカラー画像
  - 入力ポート１：TimedCharSeq でRGBカラー画像



画像処理のパラメータを
RTCLinkのConfiguration View
で修正可能にする

OpenCVライブラリのコンポーネント化

- RTCLinkの表示例



コンポーネント郡を組み合わる

- 実演？！

まとめ

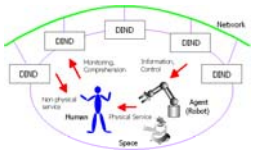
- 特徴：
  - 広く利用されている画像処理ライブラリOpenCVをRTMコンポーネントにすることで作成例を学ぶ.
  - 複数の実用性のある画像処理を簡単に連結できる.
  - 主要な処理パラメータをRTCLinkから調整できるようにすることでライブラリの試用ができる.
  - VMWareイメージにより仮想環境を配布．すぐに試せる.

分散レーザレンジファインダのキャリブレーション支援コンポーネント群

佐々木毅
東京大学生産技術研究所
橋本研究室

開発の背景

- 知能化空間
  - センサやアクチュエータを空間に分散配置し、ネットワーク化
- センサのキャリブレーションが必要
  - 各センサの座標系（ローカル座標系）において得られたデータを知能化空間の座標系（ワールド座標系）へ変換
  - 多数のセンサのキャリブレーションには多くの手間を要する



- 分散センサのキャリブレーション支援コンポーネント群を開発する

開発の背景

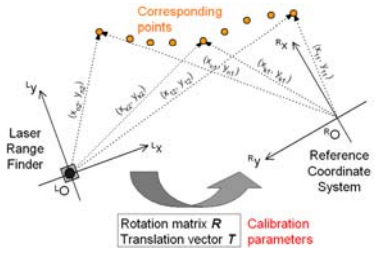
- レーザレンジファインダ(LRF)は主要なセンサの1つとして期待される
  - 設置が比較的容易
  - 知能化空間の基本機能の1つである物体トラッキングにおいても対象に特別なタグを持たせる必要がない
  - 空間の地図も獲得可能
- 本コンポーネント群では特にLRFの位置・姿勢のキャリブレーションに着目

開発環境

- OS: Ubuntu Linux 7.0.4
- RTミドルウェア: OpenRTM-aist-0.4.1-RELEASE
- コンパイラ: gcc 4.1.2
- CORBA : omniORB 4.0.7
- ACE : ACE 5.4.7-12
- Eclipse : Eclipse 3.2
- Java実行環境: Sun Java 1.5.0-11-1

LRFの位置・姿勢のキャリブレーション

- 環境中の2点以上の点について、それらのローカル座標系とワールド座標系の位置が既知ならばその対応から計算可能



開発したコンポーネント群

- LRFコンポーネント (LRFComponent)
  - 北陽電機（株）のLRF (URG-04LX)をRTコンポーネント化※
- 移動体トラッキングコンポーネント (SimpleTracker)
  - LRFのスキャンデータから移動物体の位置を出力
- キャリブレーションコンポーネント (LRFCalibration)
  - 同一物体の2つの座標系での位置を入力として受け取り、その対応点から2つの座標系の位置・姿勢の関係を出力
- 座標変換コンポーネント (CoordTrans2D)
  - キャリブレーションパラメータに基づいてセンサ座標系から基準座標系への座標変換を実行
- 入力コンポーネント (ConsoleIn2)
  - コンソールから入力した値を順に2つのOutPortlに出力

※デバイス処理には商品付属のサンプルプログラムの一部を使用

LRFCalibrationの使用例

- キャリブレーションオブジェクトを用いた絶対的な位置・姿勢の手動キャリブレーション（※）
- 移動物体を用いたLRF間の相対的な位置・姿勢の自動キャリブレーション（※）
- 移動ロボットを用いた絶対的な位置・姿勢の自動キャリブレーション
- 天井カメラの位置・姿勢のキャリブレーション

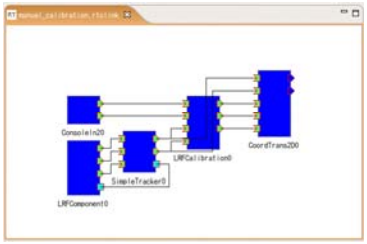
- （※）は本コンポーネント群のみで実現可能な機能

キャリブレーションオブジェクトを用いた絶対的な位置・姿勢の手動キャリブレーション

- レーザレンジファインダで検知しやすい物体を環境内の既知の位置に順番に置き、その位置情報に基づいてレーザレンジファインダの絶対的な位置・姿勢を計算
- 高精度なキャリブレーション
- 使用するコンポーネント
  - LRFComponent …… 1つ
  - SimpleTracker …… 1つ
  - LRFCalibration …… 1つ
  - ConsoleIn2 …… 1つ
  - CoordTrans2D …… 1つ

システム構築

- コンポーネントの接続



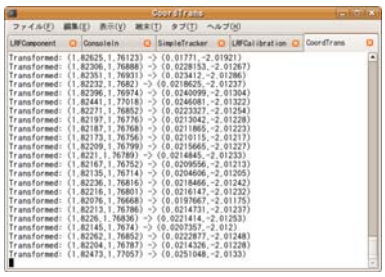
- パラメータ設定
- アクティブ化

キャリブレーション

- キャリブレーションオブジェクト（LRFで検知しやすい物体）を既知の位置に配置
- ConsoleIn2のコンソールからキャリブレーションオブジェクトの位置の座標を入力

結果の確認

- 物体を既知の位置に置き、結果を確認
  - (0, -2.0) に物体を置いた場合

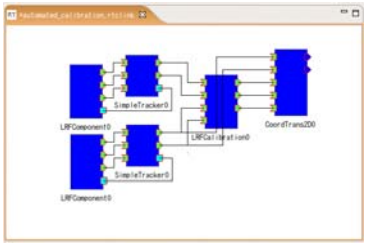


移動物体を用いたLRF間の相対的な位置・姿勢の自動キャリブレーション

- 2台のレーザレンジファインダの観測領域の重なりを利用し、一方のレーザレンジファインダを基準とした相対的な位置・姿勢を獲得
- 手動キャリブレーションと比べると精度は劣るが、簡単にキャリブレーションが可能
- 使用するコンポーネント
  - LRFComponent …… 2つ
  - SimpleTracker …… 2つ
  - LRFCalibration …… 1つ
  - CoordTrans2D …… 1つ

システム構築

- コンポーネントの接続



- パラメータ設定
- アクティブ化

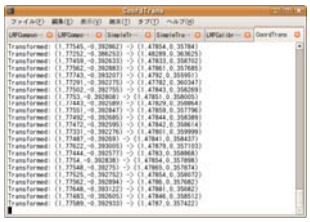
キャリブレーション

- 2台のLRFの観測領域の重なり部分で、人間やロボットなどを移動させる

結果の確認

- 物体を適当な位置に置き結果を比較

基準となるLRFでの物体の位置



キャリブレーションを行ったLRFでの物体の位置

その他の使用例

- 移動ロボットを用いた絶対的な位置・姿勢の自動キャリブレーション
  - 移動物体を用いた自動キャリブレーションでは、移動物体の絶対的な位置情報がわからないため、キャリブレーションも相対的
  - 基準となるLRFの代わりに移動ロボットの自己位置推定モジュールから得られる位置情報を利用すれば、絶対的な位置・姿勢を計算可能
- 天井カメラの位置・姿勢のキャリブレーション
  - LRFCalibrationはLRFのキャリブレーション用モジュールとして開発されたが、2次元平面上での位置を獲得するセンサであれば本コンポーネント群を同じように適用可能

まとめ

- 分散レーザレンジファインダ（LRF）のキャリブレーション支援コンポーネント群の開発
  - 各機能要素のコンポーネント化
    - LRF(北陽電機URG04-LX)
    - 移動物体トラッキング
    - キャリブレーション
- コンポーネント群の使用例の提案と動作確認
  - キャリブレーションオブジェクトを用いた絶対的な位置・姿勢の手動キャリブレーション
  - 移動物体を用いたLRF間の相対的な位置・姿勢の自動キャリブレーション

Thank you for your attention!

知的制御システム 橋本研究室
Intelligent Control System Laboratory – Hashimoto Lab.
http://dfs.iis.u-tokyo.ac.jp/

謝辞

- レーザレンジファインダコンポーネント (LRFComponent) の開発に当たりましては、東京大学生産技術研究所橋本研究室の川路浩平氏、Drazen Brscic氏、佐々木毅が作成したC++ UrgLaserクラスのコードの一部を利用しております。川路浩平氏、Drazen Brscic氏の両名に感謝申し上げます。

ライセンス（公開条件）について

- 著作権の放棄はしませんが、非商用利用であれば自由にご利用ください
- LRFコンポーネント (LRFComponent)について
  - URG-04LX付属のサンプルコードの一部を使用しているため、UrgLaserクラスはライブラリとして提供
  - 追記：成果発表会の会場にて北陽電機様よりソースコード公開の許可をいただきました

参考資料
各コンポーネントのインタフェース概説

（詳細はマニュアルを参照）

LRFComponent

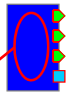
- 北陽電機（株）のLRF (URG-04LX)をRTコンポーネント化

OutPort

名称	型	説明
ScanData	TimedShortSeq	センサが獲得した距離データ [mm]
StartPoint	TimedShort	スキャン開始点（ステップ）
EndPoint	TimedShort	スキャン終了点（ステップ）

LRFComponent

- 北陽電機（株）のLRF (URG-04LX)をRTコンポーネント化

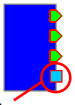


Configuration変数

名称	型	デフォルト値	説明
DeviceName	char*	“/dev/ttyACM0”	デバイスが接続されたポートの名前
ScanRate	int	19200	通信速度 [bps]
ScanStart	int	0	スキャン開始点（ステップ）
ScanEnd	int	768	スキャン終了点（ステップ）
ScanStep	int	1	ScanStartとScanEndの間のステップ数

LRFComponent

- 北陽電機（株）のLRF (URG-04LX)をRTコンポーネント化



サービスポート (provider)

関数名	引数	戻り値の型	説明
getMaxRange	なし	short	センサの最大計測可能距離[mm]を返す
getMinAngle	なし	short	最小ステップに対応する角度[deg]を返す
getMaxAngle	なし	short	最大ステップに対応する角度[deg]を返す
getMaxStep	なし	short	最大ステップを返す

SimpleTracker

- LRFのスキャンデータから移動物体の位置を出力

InPort

名称	型	説明
ScanData	TimedShortSeq	LRFComponent が出力した距離データ [mm]
StartPoint	TimedShort	スキャン開始点（ステップ）
EndPoint	TimedShort	スキャン終了点（ステップ）

SimpleTracker

- LRFのスキャンデータから移動物体の位置を出力

OutPort

名称	型	説明
X	TimedDouble	観測領域内で最大の大きさをもつ移動物体の位置のx座標 [m]
Y	TimedDouble	観測領域内で最大の大きさをもつ移動物体の位置のy座標 [m]

SimpleTracker

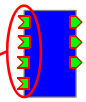
- LRFのスキャンデータから移動物体の位置を出力

サービスポート (consumer)

提供される関数についてはレーザレンジファインダコンポーネント (LRFComponent) のサービスポートを参照

LRFCalibration

- 同一物体の2つの座標系での位置を入力として受け取り、その対応点から2つの座標系の位置・姿勢の関係を出力

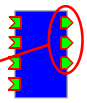


InPort

名称	型	説明
GlobalX	TimedDouble	物体の基準となる座標系での位置のx座標 [m]
GlobalY	TimedDouble	物体の基準となる座標系での位置のy座標 [m]
LocalX	TimedDouble	物体のセンサ座標系での位置のx座標 [m]
LocalY	TimedDouble	物体のセンサ座標系での位置のy座標 [m]

LRFCalibration

- 同一物体の2つの座標系での位置を入力として受け取り、その対応点から2つの座標系の位置・姿勢の関係を出力



OutPort

名称	型	説明
Tx	TimedDouble	基準座標系に対するセンサ座標系のx方向の並進量 [m]
Ty	TimedDouble	基準座標系に対するセンサ座標系のy方向の並進量 [m]
Theta	TimedDouble	基準座標系に対するセンサ座標系の回転角度 [rad]

LRFCalibration

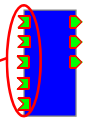
- 同一物体の2つの座標系での位置を入力として受け取り、その対応点から2つの座標系の位置・姿勢の関係を出力

Configuration変数

名称	型	デフォルト値	説明
data_num	int	0	キャリブレーションに用いる対応点の数

CoordTrans2D

- キャリブレーションパラメータに基づいてセンサ座標系から基準座標系への座標変換を実行

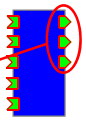


InPort

名称	型	説明
Xin	TimedDouble	センサ座標系での位置のx座標 [m]
Yin	TimedDouble	センサ座標系での位置のy座標 [m]
Tx	TimedDouble	基準座標系に対するセンサ座標系のx方向の並進量 [m]
Ty	TimedDouble	基準座標系に対するセンサ座標系のy方向の並進量 [m]
Theta	TimedDouble	基準座標系に対するセンサ座標系の回転角度 [rad]

CoordTrans2D

- キャリブレーションパラメータに基づいてセンサ座標系から基準座標系への座標変換を実行

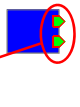


OutPort

名称	型	説明
Xout	TimedDouble	基準座標系に変換した位置のx座標 [m]
Yout	TimedDouble	基準座標系に変換した位置のy座標 [m]

ConsoleIn2

- コンソールから入力した値を順に2つのOutPortに出力



OutPort

名称	型	説明
out	TimedDouble	コンソールから最初に入力された数値
out2	TimedDouble	コンソールから2番目に入力された数値