

次世代ロボット知能化技術開発プロジェクト  
ロボット知能ソフトウェア再利用性向上技術の開発

機能仕様書  
ロボットアーム(PA10)分解運動速度制御モジュール  
(Windows)

V e r . 1 . 0

2009年11月2日

R T C 再利用技術研究センター

## 改版履歷

[illegible]

# 目次

1. はじめに .....	1
1. 1. 本書の適用範囲 .....	1
1. 2. 関連文書 .....	1
1. 3. 本書を読むにあたって .....	1
2. 機能仕様 .....	2
2. 1. 機能概要 .....	2
2. 2. モジュール構成 .....	2
2. 3. ターゲットハードウェア .....	7
3. RTC 仕様 .....	10
3. 1. frm_ctrl (軌跡制御コンポーネント) .....	10
3. 2. mixer (ヤコビシーケンス生成コンポーネント) .....	15
3. 3. jinv (逆ヤコビ行列変換コンポーネント) .....	18
3. 4. pa10fk (PA10 順運動学計算コンポーネント) .....	20
3. 5. vel_7dof (PA10 シミュレータコンポーネント) .....	24
3. 6. pa10disp (PA10 幾何モデル描画コンポーネント) .....	27
3. 7. move (操作制御コンポーネント) .....	28
3. 8. pa10vel (PA10 実機制御コンポーネント) .....	36
4. 特記事項 .....	42

# 1. はじめに

## 1. 1. 本書の適用範囲

本書はロボット向けミドルウェア OpenRTM 上で三菱重工業製汎用ロボット PA10 の分解運動速度制御を行う知能モジュールの仕様について記述した文書である。

## 1. 2. 関連文書

本書の関連文書は下表の通り。

表 1-1 関連文書

No.	文書名	備考
1	ロボットアーム(PA10)分解運動速度制御モジュール(Windows) 操作手順書	ロボットアーム(PA10)分解運動速度制御モジュールのシミュレータ及び実機環境の操作手順について記載。

## 1. 3. 本書を読むにあたって

本書は RT ミドルウェア、RT コンポーネント(以下、RTC)に関する基本知識を備えた利用者を対象としている。RT ミドルウェア、RTC については下記を参照のこと。

OpenRTM-aist Official Website:

<http://www.is.aist.go.jp/rt/OpenRTM-aist/>

## 2. 機能仕様

### 2. 1. 機能概要

本知能モジュールは PA10 の分解運動速度制御を実現する。分解運動速度制御とは、手先の並進運動および姿勢の更新を行うために、PA10 の各関節の運動速度を制御することである。

本知能モジュールでは、目標とする手先位置・姿勢の指定、動作速度の上限指定、手先に取り付けるツール長とその姿勢の設定がサービスとして提供されている。

### 2. 2. モジュール構成

#### 2. 2. 1. シミュレータ(VPython)動作環境

シミュレータ動作時の本知能モジュールは図 2-1 に表されるように、

- ① 軌跡制御コンポーネント (frm\_ctrl)
- ② ヤコビシーケンス生成コンポーネント (mixer)
- ③ 逆ヤコビ行列変換コンポーネント (jinv)
- ④ PA10 順運動学計算コンポーネント (pa10fk)
- ⑤ PA10 シミュレータコンポーネント (vel\_7dof)
- ⑥ PA10 幾何モデル描画コンポーネント (pa10disp)
- ⑦ 操作制御コンポーネント (move)

で構成される。手先の目標位置・姿勢の設定は操作制御コンポーネントで、動作速度の上限設定は軌跡制御コンポーネントで、手先ツール長の設定は PA10 順運動学計算コンポーネントでサービスとして提供されている。図中の点線で囲んだ箇所は動作環境に合わせて付け替える部分であり、ここでは VPython によるシミュレータ環境を接続したものである。図 2-2 が VPython シミュレータ表示画面になる。

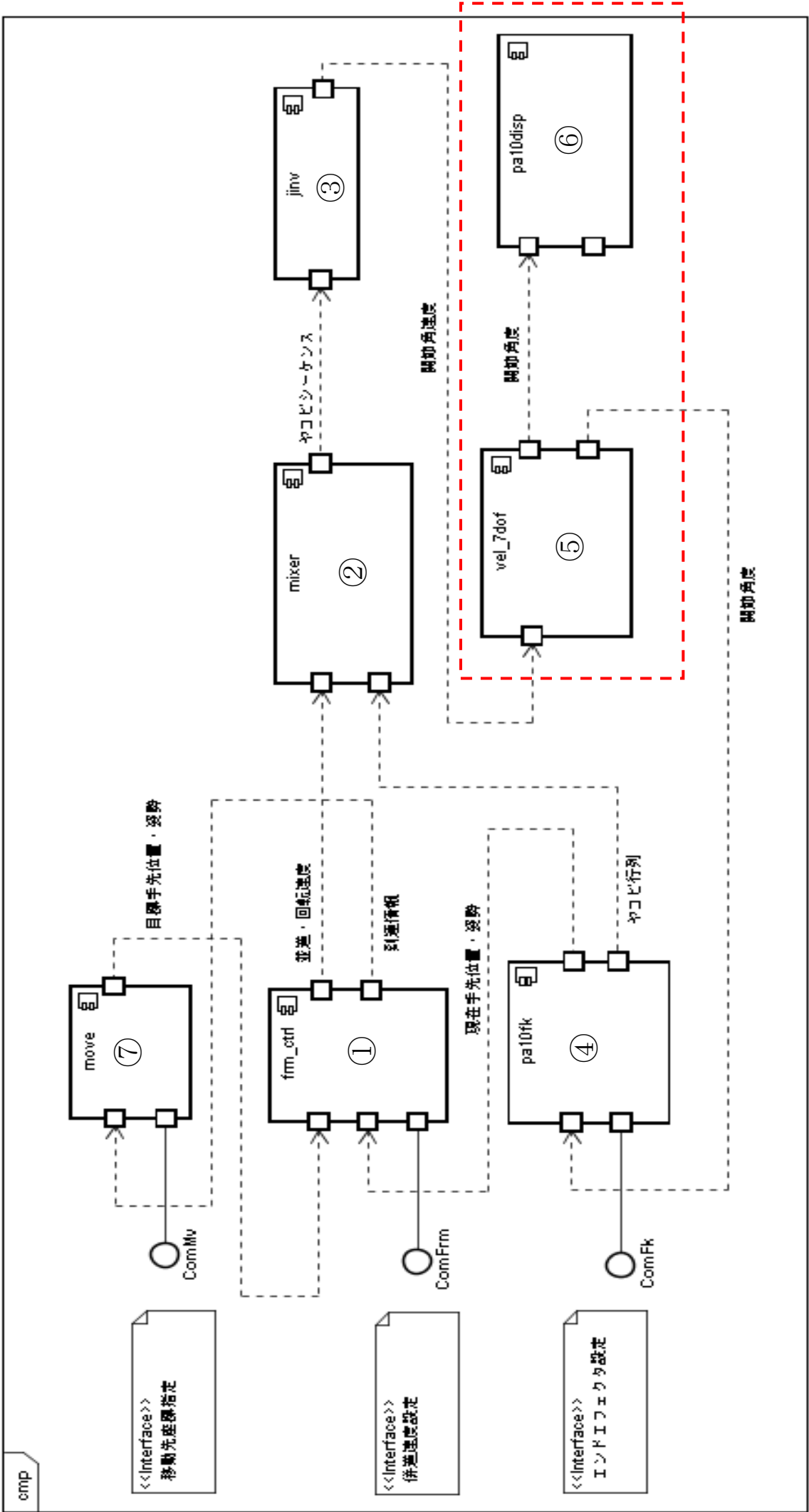


図 2-1 シミュレータ接続時モジュール構成図

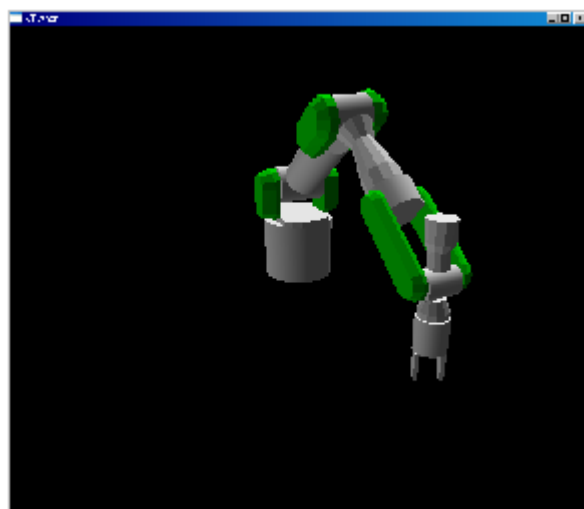


図 2-2 VPython シミュレータ表示画面

## 2. 2. 2. 実機動作環境

実機動作時のモジュール構成は、前節のシミュレータ関連コンポーネント（⑤、⑥）を実機制御コンポーネント（pa10vel、⑧）に置き換えたものになる。図 2-3 にそのモジュール構成図を示す。図 2-3 と図 2-1 の違いは図中の点線で囲んだ部分のみである。本モジュールに接続した実機の写真を図 2-4 に示す。







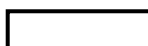
図 2-4 接続した実機

## 2. 3. ターゲットハードウェア

本知能モジュールの対象となるハードウェアは、三菱重工業製汎用ロボット PA10 である。ハードウェア構成及びその仕様は以下の通りである。

### 2. 3. 1. ハードウェア構成

表記： ——— 接続機器



ハードウェア

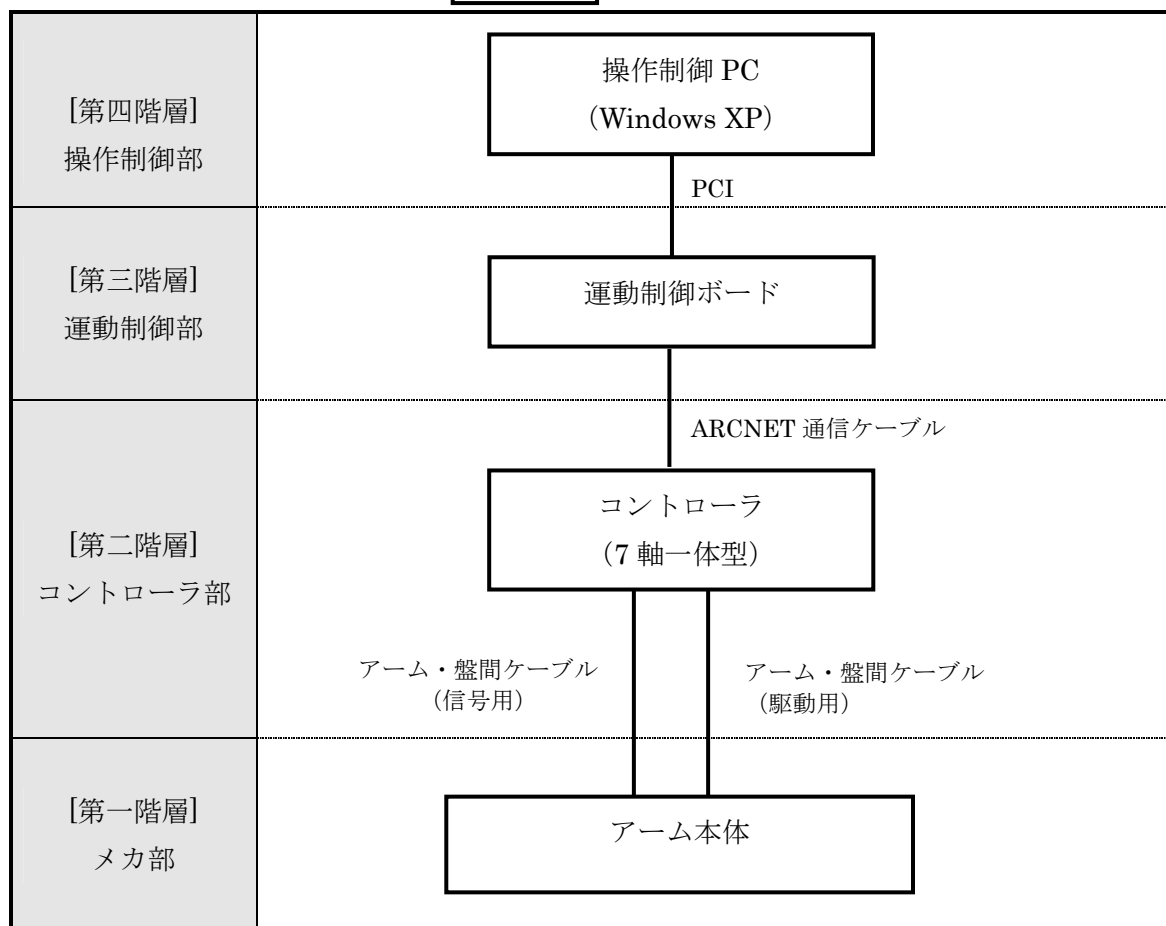


図 2-5 ハードウェア構成



図 2-6 アーム本体

## 2. 3. 2. ハードウェア仕様

### 2. 3. 2. 1. アーム本体仕様

表 2-1 アーム仕様

項目	内容				
名称	三菱重工汎用ロボット アーム				
型 名	PA10-7C-ARM				
関節数	7				
関節構成	ロボット取り付け面より R-P-R-P-R-P-R (R は回転関節, P は旋回関節を示す)				
関節名称	ロボット取り付け面より S1-S2-S3-E1-E2-W1-W2 (Sは肩関節, Eは肘関節, Wは手首関節を表す)				
アーム長	肩リーチ : 317mm (ベース面～S2 間) 上腕 : 450mm (S2～E1 軸間) 下腕 : 480mm (E1～W1 軸間) 手首リーチ : 80mm (W1～メカニカルインタフェース面間)				
関節動作範囲 及び 最高動作速度	軸名称	リミット (度)			
		メカリミット	サーボ リミット	ソフトウェア リミット	最高動作速度 (rad/sec)
	S1 (回転)	±180	±178	±177	±1
	S2 (旋回)	± 94	± 92	±91	±1
	S3 (回転)	±180	±175	±174	±1
	E1 (旋回)	±143	±138	±137	±2
	E2 (回転)	±270	±256	±255	±2π
	W1 (旋回)	±180	±166	±165	±2π
	W2 (回転)	±無限回転	±361	±360	±2π

(詳細は「三菱重工 汎用ロボット PA10 シリーズ 総合取扱説明書 P2-2」を参照)

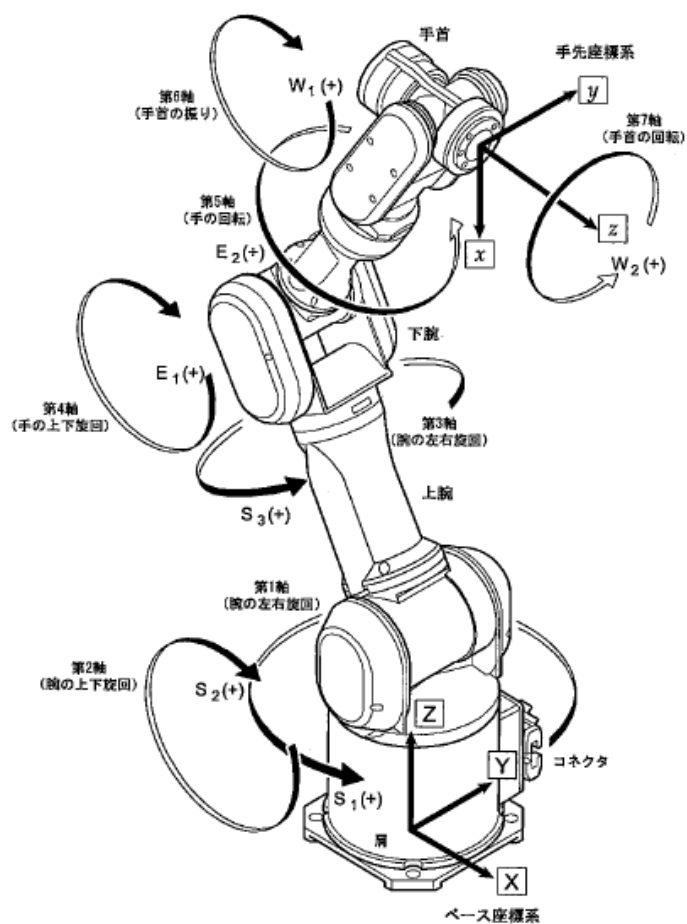


図 2-7 アームの動作軸と軸名称(総合取扱説明書より)

## 2. 3. 2. 2. コントローラ・運動制御ボード仕様

表 2-2 コントローラ・運動制御ボード仕様

項目	型番
コントローラ	PA10-7C-CNT
運動制御ボード	MHI-D7281

(詳細は「三菱重工 汎用ロボット PA10 シリーズ 総合取扱説明書 P2-13,16(コントローラ) P2-17(運動制御ボード)」を参照)

## 3. RTC 仕様

### 3. 1. frm\_ctrl (軌跡制御コンポーネント)

#### 3. 1. 1. 機能概要

本コンポーネントはロボットアームの手先位置・姿勢の軌跡制御を行う。手先の現在位置・姿勢及び手先の目標位置・姿勢から、ロボットアームの動作完了の判定、または、ロボットアームの手先並進速度、回転速度の算出を行う。

本コンポーネントでは速度計算に必要なパラメータの設定も行う。パラメータはコンフィグレーションとして用意してある。

#### 3. 1. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	C++
コンパイラ	Microsoft Visual C++ 2008 Express Edition
RT ミドルウェア／バージョン	OpenRTM-aist-0.4.2-RELEASE (C++版)
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	ACE-5.6
依存ライブラリ (その他)	行列演算ライブラリ (自作)

#### 3. 1. 3. ポート情報

##### A) データポート (InPort)

名称	型	データ長	説明
ref_frm	TimedFloatSeq	1 2	目標手先位置 (単位: mm) ・ 姿勢
cur_frm	TimedFloatSeq	1 2	現在手先位置 (単位: mm) ・ 姿勢

##### B) データポート (OutPort)

名称	型	データ長	説明
vel	TimedFloatSeq	6	並進速度 (単位: mm/sec) 回転速度 (単位: rad/sec)
state	TimedLong	1	到達情報 (1: 到着 0: 未到着)

## C) サービスポート (Provider)

サービス名	インターフェース名	説明
ComFrmCtrl	com_frm_ctrl	手先位置・姿勢の軌跡制御

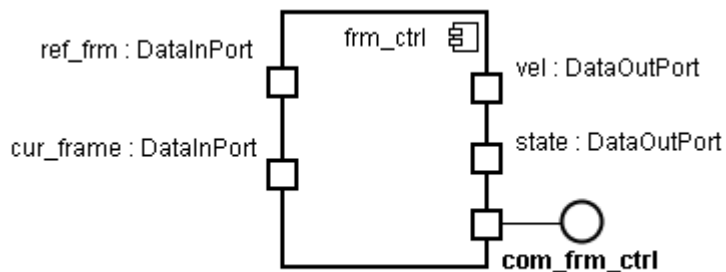


図 3-1 frm\_ctrl コンポーネント

## 3. 1. 4. コンフィグレーション情報

名称	型	デフォルト値	説明
k_gain	float	1.0	時定数
v_max	float	200.0	最大速度 (単位 : mm/sec)
k_pos_rot	float	200.0	操作オブジェクトのスケール

備考 : コンフィグレーションの更新は、一旦本コンポーネントを非活性化した状態でコンフィグレーション値を変更し、再び本コンポーネントを活性化させることで行える。

### 3. 1. 5. 入出力データフォーマット

#### 3. 1. 5. 1. 入力 : **ref\_frm**、**cur\_frm**

データ位置	格納値
0	手先の姿勢を表す回転行列の 1 行 1 列目
1	手先の姿勢を表す回転行列の 1 行 2 列目
2	手先の姿勢を表す回転行列の 1 行 3 列目
3	手先の姿勢を表す回転行列の 2 行 1 列目
4	手先の姿勢を表す回転行列の 2 行 2 列目
5	手先の姿勢を表す回転行列の 2 行 3 列目
6	手先の姿勢を表す回転行列の 3 行 1 列目
7	手先の姿勢を表す回転行列の 3 行 2 列目
8	手先の姿勢を表す回転行列の 3 行 3 列目
9	手先の位置ベクトルの X 成分 (ベース座標系)
1 0	手先の位置ベクトルの Y 成分 (ベース座標系)
1 1	手先の位置ベクトルの Z 成分 (ベース座標系)

#### 3. 1. 5. 2. 出力 : **vel** (指令値)

データ位置	格納値
0	手先の x 軸方向への並進速度
1	手先の y 軸方向への並進速度
2	手先の z 軸方向への並進速度
3	手先の x 軸回りの回転速度
4	手先の y 軸回りの回転速度
5	手先の z 軸回りの回転速度

#### 3. 1. 5. 3. 出力 : **state**

データ位置	格納値
0	到達情報

### 3. 1. 6. サービスポート I/F 仕様

#### 3. 1. 6. 1. ComFrmCtrl

##### (1) set\_param

関数名	set_param			
引数	名称	型	I/O	説明
	k_pv	float	入力	時定数 (k_gain) として設定される
	v_m	float	入力	最大速度 (v_max) として設定される (単位 : mm/sec)
	k_pr	float	入力	オブジェクトのサイズ (k_pos_rot) として設定される
戻り値	値			説明
	なし			-
説明	並進速度、回転速度計算に使用するパラメータの設定を行う。			
備考	デフォルトでは k_pv = 1.0, v_m = 200.0, k_pr = 200.0 であり、基本的に k_pv と k_pr は変更を行わない。必要に応じて最大速度の調整を行う場合には、十分注意して適切なパラメータ設定をすること。			

##### パラメータ設定実装例

```

RTC::ReturnCode_t frm_ctrl::onInitialize()
{
    bindParameter("k_gain", m_k_gain, "1.0");
    bindParameter("v_max", m_v_max, "200.0");
    bindParameter("k_pos_rot", m_k_pos_rot, "200.0");
    :
RTC::ReturnCode_t frm_ctrl::onActivated(RTC::UniqueId ec_id)
{
    // default
    // k_gain = 1.0, means time constant is 1 sec.
    // v_max = 200.0[mm/sec]
    // k_pos_rot = 200.0, means that sizes of objects are about this size.
    m_com_frm_ctrl.set_param(m_k_gain, m_v_max, m_k_pos_rot);

```



## (2) calc\_vel

関数名	calc_vel			
引数	名称	型	I/O	説明
	geo_ref	FRAME	入力	手先の目標位置と姿勢
	geo_cur	FRAME	入力	手先の現在位置と姿勢
	p_vel	VECTOR	出力	並進速度
	r_vel	VECTOR	出力	回転速度
戻り値	値			説明
	0			正常終了
	その他			異常終了
説明	手先の目標位置・姿勢、手先の現在位置・姿勢を入力として、目標到達判定及び並進速度、回転速度の計算を行う。			
備考 1	本関数は内部処理用として使用している。			
備考 2	FRAME型は $3 \times 3$ 行列と3次元ベクトルを並べた $3 \times 4$ 行列で構成されている。			

FRAME 型の定義 (myRTC/tools/geo/geo\_std.h 内)

```
#define REAL float
typedef REAL VECTOR[3];
typedef REAL MATRIX[3][3];
typedef struct {
    MATRIX mtr;
    VECTOR vtr;
} FRAME;
```

## 3. 2. mixer（ヤコビシーケンス生成コンポーネント）

### 3. 2. 1. 機能概要

本コンポーネントは指令値とヤコビ行列の成分を一行に並べたヤコビシーケンスを作成する。ビジュアルフィードバック等他の制御に対応したヤコビ行列を計算または推定することも可能である（現在は未対応）。

### 3. 2. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	C++
コンパイラ	Microsoft Visual C++ 2008 Express Edition
RT ミドルウェア／バージョン	OpenRTM-aist-0.4.2-RELEASE（C++版）
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	ACE-5.6

### 3. 2. 3. ポート情報

#### A) データポート（InPort）

名称	型	データ長	説明
vel	TimedFloatSeq	6	並進速度（単位：mm/sec） 回転速度（単位：rad/sec）
jacob	TimedFloatSeq	4 2	ヤコビ行列

#### B) データポート（OutPort）

名称	型	データ長	説明
j_list	TimedFloatSeq	4 8	ヤコビシーケンス

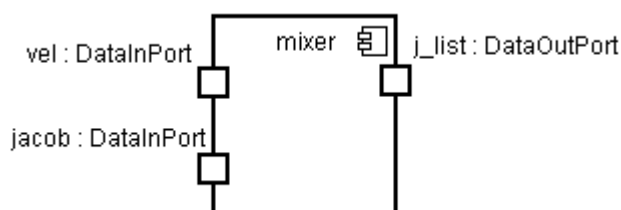


図 3-2 mixer コンポーネント

### 3. 2. 4. 入出力データフォーマット

3. 2. 4. 1. 入力：**vel**（3. 1. 5. 2と同様）

3. 2. 4. 2. 出力：**jacob**

ヤコビ行列は  $6 \times 7$  の行列である。

データ位置	格納値
0	ヤコビ行列の 1 行 1 列目
1	ヤコビ行列の 1 行 2 列目
2	ヤコビ行列の 1 行 3 列目
3	ヤコビ行列の 1 行 4 列目
4	ヤコビ行列の 1 行 5 列目
5	ヤコビ行列の 1 行 6 列目
6	ヤコビ行列の 1 行 7 列目
7	ヤコビ行列の 2 行 1 列目
	...
4 0	ヤコビ行列の 6 行 6 列目
4 1	ヤコビ行列の 6 行 7 列目

### 3. 2. 4. 3. j\_list

ヤコビシーケンスは指令値とヤコビ行列の成分で構成されたシーケンスである。

データ位置	格納値
0	手先の $x$ 軸方向への並進速度
1	ヤコビ行列の 1 行 1 列目
2	ヤコビ行列の 1 行 2 列目
3	ヤコビ行列の 1 行 3 列目
4	ヤコビ行列の 1 行 4 列目
5	ヤコビ行列の 1 行 5 列目
6	ヤコビ行列の 1 行 6 列目
7	ヤコビ行列の 1 行 7 列目
8	手先の $y$ 軸方向への並進速度
9	ヤコビ行列の 2 行 1 列目
	...
4 0	手先の $z$ 軸回りの回転速度
4 1	ヤコビ行列の 6 行 1 列目
4 2	ヤコビ行列の 6 行 2 列目
4 3	ヤコビ行列の 6 行 3 列目
4 4	ヤコビ行列の 6 行 4 列目
4 5	ヤコビ行列の 6 行 5 列目
4 6	ヤコビ行列の 6 行 6 列目
4 7	ヤコビ行列の 6 行 7 列目

### 3. 3. jinv（逆ヤコビ行列変換コンポーネント）

#### 3. 3. 1. 機能概要

本コンポーネントはロボットアームの手先の並進速度、回転速度とヤコビ行列を用いて、各軸の角速度の算出を行う。尚、7 軸のアームに対応するため、ヤコビ行列の逆行列計算には特異値分解（SVD）を用いている。

#### 3. 3. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	C++
コンパイラ	Microsoft Visual C++ 2008 Express Edition
RT ミドルウェア／バージョン	OpenRTM-aist-0.4.2-RELEASE（C++版）
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	ACE-5.6
依存ライブラリ (その他)	行列演算ライブラリ（自作）

#### 3. 3. 3. ポート情報

##### A) データポート（InPort）

名称	型	データ長	説明
y_jacob	TimedFloatSeq	4 8	ヤコビシーケンス

##### B) データポート（OutPort）

名称	型	データ長	説明
vel	TimedDoubleSeq	7	各軸（7 軸）の動作角速度 （単位：rad/sec）

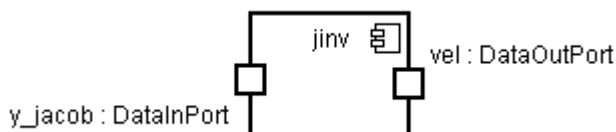


図 3-3 jinv コンポーネント

### 3. 3. 4. コンフィグレーション情報

名称	型	デフォルト値	説明
n_size	int	7	関節数 (≧1) 本モジュールではアーム自由度を7に固定するものとし、基本的に変更は行わない。
m_max	int	100	特異値分解行列の最大行数 (≧1)

### 3. 3. 5. 入出力データフォーマット

3. 3. 5. 1. 入力 : **y\_jacob** (3. 2. 4. 3 と同様)

3. 3. 5. 2. 出力 : **vel** (関節角速度)

軸名称はアーム本体使用参照。

データ位置	格納値
0	S1 軸の角速度 (図 2-7 参照)
1	S2 軸の角速度
2	S3 軸の角速度
3	E1 軸の角速度
4	E2 軸の角速度
5	W1 軸の角速度
6	W2 軸の角速度

### 3. 4. pa10fk（PA10 順運動学計算コンポーネント）

#### 3. 4. 1. 機能概要

本コンポーネントは三菱重工業製汎用ロボット PA10 の順運動学計算を行う。

またアーム先端に取り付けるツールの設定も行う。ツール設定値はコンフィグレーションに用意しておく。

#### 3. 4. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	C++
コンパイラ	Microsoft Visual C++ 2008 Express Edition
RT ミドルウェア／バージョン	OpenRTM-aist-0.4.2-RELEASE（C++版）
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	ACE-5.6
依存ライブラリ(その他)	行列演算ライブラリ（自作）

#### 3. 4. 3. ポート情報

##### A) データポート（InPort）

名称	型	データ長	説明
th	TimedDoubleSeq	7	現在の各軸角度情報（単位：rad）

##### B) データポート（OutPort）

名称	型	データ長	説明
frame	TimedFloatSeq	1 2	手先位置（単位：mm）・姿勢
jacob	TimedFloatSeq	4 2	ヤコビ行列

## C) サービスポート (Provider)

サービス名	インターフェース名	説明
ComFk	com_fk	PA10 の順運動学計算

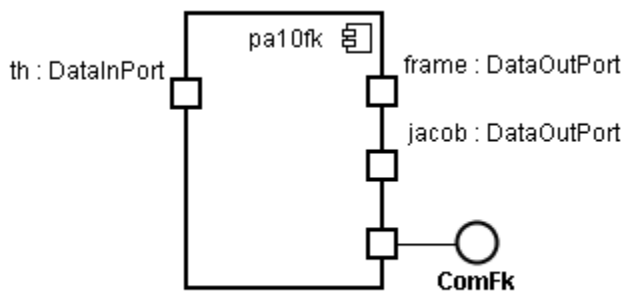


図 3-4 pa10fk コンポーネント

## 3. 4. 4. コンフィグレーション情報

名称	型	デフォルト値	説明
tool_x	float	0.0	手先座標系(図 2-7 参照)から見た手先の x 座標
tool_y	float	0.0	手先座標系から見た手先の y 座標
tool_z	float	95.0	手先座標系から見た手先の z 座標
tool_a	float	0.0	手先座標系から見たツールの x 軸回転角 (Roll)
tool_b	float	0.0	手先座標系から見たツールの y 軸回転角 (Pitch)
tool_c	float	0.0	手先座標系から見たツールの z 軸回転角 (Yaw)

備考：コンフィグレーションの更新は、一旦本コンポーネントを非活性化した状態でコンフィグレーション値を変更し、再び本コンポーネントを活性化させることで行える。



### 3. 4. 5. 入出力データフォーマット

#### 3. 4. 5. 1. 入力 : **th**

関節名称はアーム本体仕様に記載してある。

データ位置	格納値
0	S1 軸の角度
1	S2 軸の角度
2	S3 軸の角度
3	E1 軸の角度
4	E2 軸の角度
5	W1 軸の角度
6	W2 軸の角度

#### 3. 4. 5. 2. 出力 : **frame** (3. 1. 5. 1 と同様)

#### 3. 4. 5. 3. 出力 : **jacob** (3. 2. 4. 2 と同様)

### 3. 4. 6. サービスポート I/F 仕様

#### 3. 4. 6. 1. **ComFk**

##### (1) **set\_tool**

関数名	set_tool			
引数	名称	型	I/O	説明
	frm	const Frame&	入力	手先の位置とツールの姿勢
戻り値	値			説明
	なし			-
説明	ツール長と手先座標系から見たツールの姿勢を設定する。			

Frame 型の IDL 定義 (myRTC/pa10fk/com\_fk.idl 内)

```
struct Frame{
    float mat[3][3];
    float pos[3];
}
```

## 手先の位置と姿勢の設定実装例

```

RTC::ReturnCode_t pa10fk::onInitialize()
{
    bindParameter("tool_x", m_tool_x, "0.0");
    bindParameter("tool_y", m_tool_y, "0.0");
    bindParameter("tool_z", m_tool_z, "150.0");
    bindParameter("tool_a", m_tool_a, "0.0");
    bindParameter("tool_b", m_tool_b, "0.0");
    bindParameter("tool_c", m_tool_c, "0.0");
    :
    :
}

RTC::ReturnCode_t pa10fk::onActivated(RTC::UniqueId ec_id)
{
    FRAME tmp;
    MATRIX tmp_mtr;
    MATRIX tmp_amtr;
    MATRIX tmp_bmtr;
    MATRIX tmp_cmtr;

    //set position of apex of a tool
    tmp.vtr[0] = m_tool_x;
    tmp.vtr[1] = m_tool_y;
    tmp.vtr[2] = m_tool_z;

    //calculate attitude of a tool
    ma(m_tool_a,tmp_amtr);
    mb(m_tool_b,tmp_bmtr);
    mc(m_tool_c,tmp_cmtr);
    m_mul(tmp_amtr,tmp_bmtr,tmp_mtr);
    m_mul(tmp_mtr,tmp_cmtr,tmp_mtr);

    //set attitude of a tool
    for(int i=0;i<3;i++) {
        for(int j=0;j<3;j++) {
            tmp.mtr[i][j] = tmp_mtr[i][j];
        }
    }
    m_com_fk.f_kinema.set_tool(&tmp);
}

```

### 3. 5. vel\_7dof (PA10 シミュレータコンポーネント)

#### 3. 5. 1. 機能概要

本コンポーネントは順運動学計算に基づいた 7 自由度アーム (PA10) のシミュレーションを行う。尚、幾何モデル表示機能は別コンポーネント (pa10disp) で用意してある。

#### 3. 5. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	C++
コンパイラ	Microsoft Visual C++ 2008 Express Edition
RT ミドルウェア／バージョン	OpenRTM-aist-0.4.2-RELEASE (C++版)
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	ACE-5.6

#### 3. 5. 3. ポート情報

##### A) データポート (InPort)

名称	型	データ長	説明
vel	TimedDoubleSeq	7	各軸 (7 軸) の動作角速度 (単位 : rad/sec)

##### B) データポート (OutPort)

名称	型	データ長	説明
th	TimedDoubleSeq	7	pa10disp へ出力する関節角度 (単位 : rad)
th2	TimedDoubleSeq	7	pa10fk へ出力する関節角度 (単位 : rad)

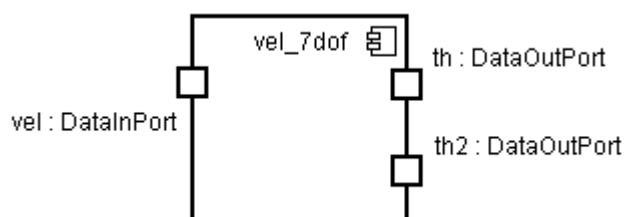


図 3-5 vel\_7dof コンポーネント

### 3. 5. 4. コンフィグレーション情報

名称	型	デフォルト値	説明
th0	float	0.0	S1 軸角度
th1	float	0.5	S2 軸角度
th2	float	0.0	S3 軸角度
th3	float	1.6	E1 軸角度
th4	float	0.0	E2 軸角度
th5	float	1.0	W1 軸角度
th6	float	0.0	W2 軸角度

### 3. 5. 5. 入出力データフォーマット

3. 5. 5. 1.      入力：**vel**（関節角速度）（3. 3. 5. 2と同様）

3. 5. 5. 2.      出力：**th**（3. 4. 5. 1と同様）

3. 5. 5. 3.      出力：**th2**（3. 4. 5. 1と同様）

### 3. 5. 6. PA10 幾何モデル待機姿勢（初期関節角度）の設定

手先の目標位置・姿勢が入力される前に、vel\_7dof コンポーネントが活性化された時点で VPython による PA10 幾何モデルは待機姿勢をとる。待機姿勢を決める初期関節角度は、コンフィグレーションとしてあらかじめ決めておく。

```
RTC::ReturnCode_t vel_7dof::onInitialize()
{
    // Bind variables and configuration variable
    bindParameter("Th0", m_Th0, "0.0");
    bindParameter("Th1", m_Th1, "0.5");
    bindParameter("Th2", m_Th2, "0.0");
    bindParameter("Th3", m_Th3, "1.6");
    bindParameter("Th4", m_Th4, "0.0");
    bindParameter("Th5", m_Th5, "1.0");
    bindParameter("Th6", m_Th6, "0.0");
    :
    :
    RTC::ReturnCode_t vel_7dof::onActivated(RTC::UniqueId ec_id)
    {
        :
        :
        th[0] = m_Th0;
        th[1] = m_Th1;
        th[2] = m_Th2;
        th[3] = m_Th3;
        th[4] = m_Th4;
        th[5] = m_Th5;
        th[6] = m_Th6;
```

### 3. 6. pa10disp (PA10 幾何モデル描画コンポーネント)

#### 3. 6. 1. 機能概要

本コンポーネントは PA10 シミュレータの一部であり、幾何モデルの表示を行う。

#### 3. 6. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	Python
インタプリタ	Python2.5.1
RT ミドルウェア／バージョン	OpenRTM-aist-Python2.5-0.4.1-RC1
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	OmniORBpy-3.1
	ACE-5.6
依存ライブラリ (その他)	VPython3.2.11

#### 3. 6. 3. ポート情報

##### A) データポート (InPort)

名称	型	データ長	説明
th	TimedDoubleSeq	7	関節角度 (単位 : rad)
handwidth	TimedFloat	1	ハンドの開閉幅 (単位 : mm)

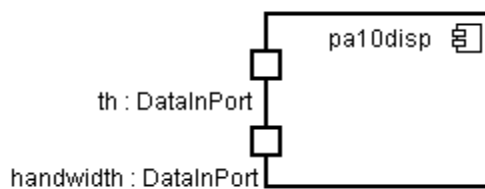


図 3-6 pa10disp コンポーネント

### 3. 6. 4. 入出力データフォーマット

3. 6. 4. 1. 入力 : **th** (3. 4. 4. 1 と同様)

3. 6. 4. 2. 入力 : **handwidth**

データ位置	格納値
0	ハンドの開閉幅

## 3. 7. move (操作制御コンポーネント)

move コンポーネントには C++ で実装された move(C++版) と、Python で実装された move(Python 版) がある。move(C++版) では目標手先位置・姿勢の入力はサービスとして実装されていないが、move(Python 版) にはそれが実装されている。

### 3. 7. 1. C++版

#### 3. 7. 1. 1. 機能概要

本コンポーネントは、ロボットアームの目標手先位置 (X, Y, Z) と目標姿勢をなすための回転角度 Roll(X 軸周り)、Pitch(Y 軸周り)、Yaw(Z 軸周り) を指定し、それらを用いて算出した目標手先位置と姿勢を軌跡制御コンポーネントへ送出する。ここで指定する X、Y、Z、Roll、Pitch、Yaw はベース座標系(図 2-7 参照) から見た手先の位置と回転角度である。

#### 3. 7. 1. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	C++
コンパイラ	Microsoft Visual C++ 2008 Express Edition
RT ミドルウェア／バージョン	OpenRTM-aist-0.4.2-RELEASE (C++版)
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	ACE-5.6
依存ライブラリ (その他)	行列演算ライブラリ

### 3. 7. 1. 3. ポート情報

#### A) データポート (InPort)

名称	型	データ長	説明
state	TimedLong	1	到達情報 (1 : 到着 0 : 未到着)

#### B) データポート (OutPort)

名称	型	データ長	説明
frame	TimedFloatSeq	1 2	目標手先位置 (単位 : mm) ・姿勢



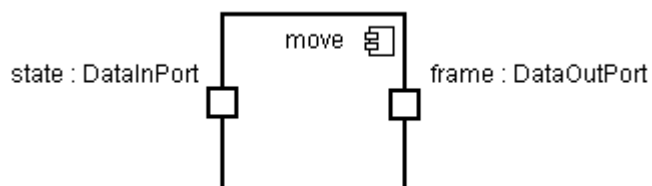


図 3-7 move コンポーネント

### 3. 7. 1. 4.      コンフィグレーション情報

名称	型	デフォルト値	説明
pos_x	float	600.0	手先座標系から見た手先の x 座標
pos_y	float	0.0	手先座標系から見た手先の y 座標
pos_z	float	300.0	手先座標系から見た手先の z 座標
rot_a	float	0.0	手先座標系から見たツールの x 軸 回転角 (Roll)
rot_b	float	180.0	手先座標系から見たツールの y 軸 回転角 (Pitch)
rot_c	float	0.0	手先座標系から見たツールの z 軸 回転角 (Yaw)

### 3. 7. 1. 5.      入出力データフォーマット

3. 7. 1.5.1      入力：**state** (3. 1. 4. 3 と同様)

3. 7. 1.5.2      出力：**frame** (3. 1. 4. 1 と同様)

### 3. 7. 1. 6. ユーザインターフェース

本コンポーネントでは、(X, Y, Z, Roll, Pitch, Yaw) は標準入力で指定される。以下に入力コンソール画面と、入力例を示す。

```
Please input x,y,z [mm] and a,b,c [deg]:
```

図 3-8 入力コンソール画面

コンポーネントを起動し活性化すると、図 3-8 に示したコンソール画面が立ち上がり、先頭に “Please input x,y,z [mm] and a,b,c [deg]: ” と表示される。座標 x,y,z は「ミリメートル[mm]」単位、回転角度 a,b,c の単位は「度」で指定する。

```
Please input x,y,z [mm] and a,b,c [deg]: 600.0 0.0 500.0 0.0 180.0 0.0
vtr 600 0 500
abc 0 3.14159 0
write
end
state = 1
Please input x,y,z [mm] and a,b,c [deg]:
```

図 3-9 入力例（入力数値例：X=600.0, Y=0.0, Z=500.0, Roll=0.0, Pitch=180.0, Yaw=0.0）

コンソールへは入力値を X,Y,Z,Roll,Pitch,Yaw の順にスペース区切りで打ち込み、最後に Enter キーを押す。正しく入力が行われた場合は、入力した座標（vtr）と回転角度（abc）がコンソールに表示される。このとき、回転角度は「度」から「ラジアン」で表現したものへ変換される。“write”、“end” は送出データのポート（Outport）登録開始と終了を示す。そして、手先の位置と姿勢が目標値に達すると、再度入力が可能になる。

```
Please input x,y,z [mm] and a,b,c [deg]: default
vtr 600 0 300
abc 0 3.14159 0
write
end
state = 1
Please input x,y,z [mm] and a,b,c [deg]:
```

図 3-10 入力デフォルト値の使用例

また、本コンポーネントではコンフィグレーションにデフォルトの目標位置と姿勢が用意されている。これを使用するときは入力値として **default** と打ち込む。

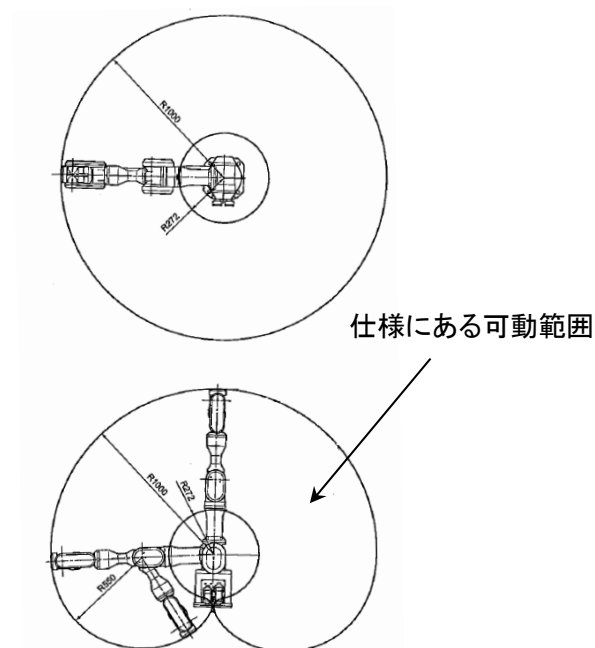
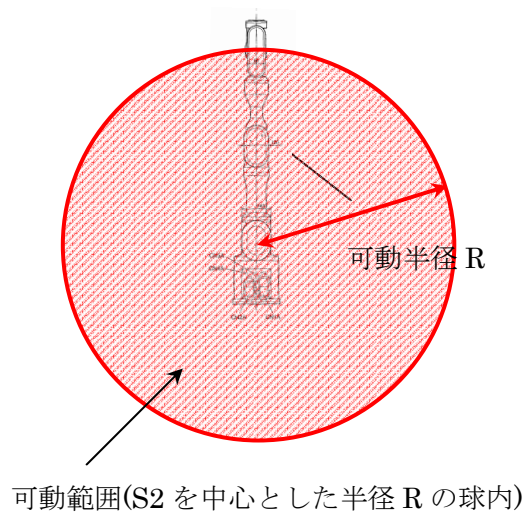


図 3-11 本コンポーネント許容可動範囲 図 3-12 PA10 動作範囲(総合取扱説明書より)

```
Please input x,y,z [mm] and a,b,c [deg]: 10000.0 0.0 500.0 0.0 180.0
0.0
Out of movable area!
Please input x,y,z [mm] and a,b,c [deg]:
```

図 3-13 可動範囲外を指定した例

目標座標が可動範囲外である場合はエラーとし、再入力が要求される。ここで決める可動範囲(図 3-11)は、PA10 の仕様に沿った可動範囲 (図 3-12)ではなく、単にアームが一直線に伸びきってしまうことを回避するために定めた領域である。ここでは関節角度が限界値を超えてしまう場合やアームの内部干渉は考えられていない。以下に PA10 用に設定した例を示す。

```
#ifndef MOVE_H
#define MOVE_H
:
#define MOVABLE_RAD 860
:
```

←可動範囲を決める可動半径 [mm] の設定

図 3-14 可動範囲設定例

### 3. 7. 2. Python 版

#### 3. 7. 2. 1. 機能概要

本コンポーネントは目標手先位置・姿勢の入力をサービスとして提供している。

本知能モジュール制御用 Python スクリプトには、move(C++版)ではなく、本コンポーネントが使用されている。

#### 3. 7. 2. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	Python
インタプリタ	Python2.5.1
RT ミドルウェア／バージョン	OpenRTM-aist-Python2.5-0.4.1-RC1
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	OmniORBpy-3.1
	ACE-5.6
依存ライブラリ(その他)	行列演算ライブラリ (自作)

### 3. 7. 2. 3. ポート情報 (Python 版)

#### A) データポート (InPort)

名称	型	データ長	説明
state	TimedLong	1	到達情報 (1 : 到着 0 : 未到着)

#### B) データポート (OutPort)

名称	型	データ長	説明
frame	TimedFloatSeq	1 2	目標手先位置 (単位 : mm) ・姿勢

#### C) サービスポート (Provider)

サービス名	インターフェース名	説明
ComMv	com_move	算出した目標手先位置・姿勢をデータポート (OutPort)へ書き込む

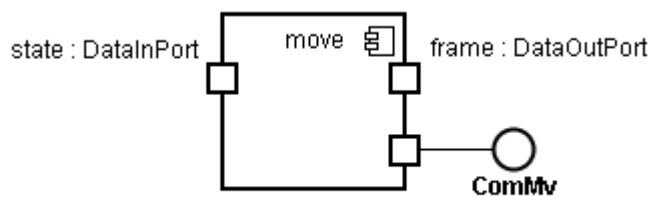


図 3-8 move コンポーネント

### 3. 7. 2. 4. 入出力データフォーマット

3. 7. 2.4.1 入力 : **state** (3. 1. 4. 3 と同様)

3. 7. 2.4.2 出力 : **frame** (3. 1. 4. 1 と同様)

### 3. 7. 2. 5. サービスポート I/F 仕様

#### 3. 7. 2.5.1 ComMv

##### (1) com\_move

関数名	com_move			
引数	名称	型	I/O	説明
	self	float	入力	自身クラスのインスタンス
	frm	FRAME	入力	目標手先位置・姿勢
戻り値	値			説明
	なし			-
説明	引数として渡された frm をデータポート(OutPort)に書き込む。			

## 3. 8. pa10vel (PA10 実機制御コンポーネント)

### 3. 8. 1. 機能概要

本コンポーネントは三菱重工業製汎用ロボット PA10 の動作制御を行う。  
主な機能は以下の通りである。

- ・ PA10 の各軸角度制御動作（動作待機姿勢への移行時のみ）
- ・ PA10 の各軸角速度制御動作
- ・ PA10 の現在の各軸角度情報取得

### 3. 8. 2. 動作環境

コンポーネント動作環境を以下に示す。

動作 OS	Windows XP
開発言語	C++
コンパイラ	Microsoft Visual C++ 2008 Express Edition
RT ミドルウェア／バージョン	OpenRTM-aist-0.4.2-RELEASE (C++版)
依存ライブラリ (OpenRTM)	OmniORB-4.1.2
	ACE-5.6
依存ライブラリ (その他)	行列演算ライブラリ
	三菱重工業製 PA ライブラリ (商用)

### 3. 8. 3. ポート情報

#### A) データポート (InPort)

名称	型	値の範囲	説明
vel	TimedDoubleSeq	[※1]	各軸（7 軸）の動作角速度 (単位 : rad/sec)

[※1] PA10 の制御パラメータ情報より速度リミットを取得し、判断を行っている。リミット値のデフォルトの設定はアーム本体仕様を参照のこと。

#### B) データポート (OutPort)

名称	型	値の範囲	説明
th	TimedDoubleSeq	[※2]	現在の各軸（7 軸）角度 (単位 : rad)

[※2] PA10 の制御パラメータ情報より各軸の角度情報を取得し、出力パラメータとしている。リミット値のデフォルトの設定はアーム本体仕様を参照のこと。

#### C) サービスポート (Provider)

サービス名	インターフェース名	説明
ComPA10	com_pa10	PA10 の各種動作制御

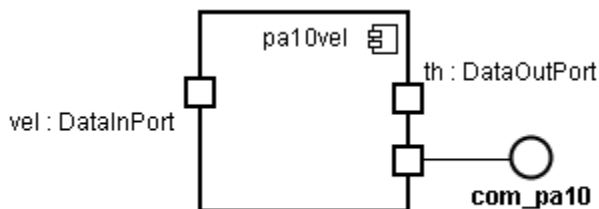


図 3-9 pa10vel コンポーネント



### 3. 8. 4. 入出力データフォーマット

3. 8. 4. 1. 入力：**vel**（関節角速度）（3. 3. 4. 2と同様）  
関節角速度のデータ構造を下表に示す。

データ位置	格納値
0	S1 軸の角速度
1	S2 軸の角速度
2	S3 軸の角速度
3	E1 軸の角速度
4	E2 軸の角速度
5	W1 軸の角速度
6	W2 軸の角速度

3. 8. 4. 1. 出力：**th**（3. 4. 4. 1と同様）

### 3. 8. 5. サービスポート I/F 仕様

#### 3. 8. 5. 1. ComPA10

##### (1) move\_here\_pa10

関数名	move_here_pa10			
引数	名称	型	I/O	説明
	なし	-	-	-
戻り値	値			説明
	0			正常終了
	その他			異常終了
説明	PA10 実機の現在の各軸角度を取得後、その角度へ各軸角度制御を行う。 これにより PA10 実機の各軸角度とコンポーネントが保持する現在角度との同期をとることが可能である。			
備考	本 API は reset_pa10 で呼び出される			

## (2) reset\_pa10

関数名	reset_pa10			
引数	名称	型	I/O	説明
	なし	-	-	-
戻り値	値			説明
	0			正常終了
	その他			異常終了
説明	PA ライブラリの初期化及びアームのリセット処理を行う。			
備考	本 API は init_pa10 で呼び出される。			

## (3) init\_pa10

関数名	init_pa10			
引数	名称	型	I/O	説明
	なし	-	-	-
戻り値	値			説明
	0			正常終了
	その他			異常終了
説明	待機姿勢として指定された角度へ各軸制御を行う。			

## (4) on\_pa10

関数名	on_pa10			
引数	名称	型	I/O	説明
	なし	-	-	-
戻り値	値			説明
	0			正常終了
	その他			異常終了
説明	PA10 の各軸速度制御の開始処理を行う。			

## (5) off\_pa10

関数名	off_pa10			
引数	名称	型	I/O	説明
	なし	-	-	-
戻り値	値			説明
	0			正常終了
	その他			異常終了
説明	PA10 の各軸速度制御の一時停止処理を行う。			

## (6) end\_pa10

関数名	end_pa10			
引数	名称	型	I/O	説明
	なし	-	-	-
戻り値	値			説明
	0			正常終了
	その他			異常終了
説明	PA ライブラリの終了及びアームのクローズ処理を行う。			

## (7) set\_flg

関数名	set_flg			
引数	名称	型	I/O	説明
	flg	long	入力	割り込みフラグ 1 : ON 各軸角速度制御を中断する 2 : OFF 各軸角速度制御を再開する
戻り値	値			説明
	なし			-
説明	各軸角速度制御への割り込みフラグを設定する。			
備考	本コーポメントがアクティブ状態であるときは、必ず本フラグの設定をおこなうこと。実装例は下記の通りである。			

### set\_flg 呼び出し例

```
m_com_pa10->set_flg(1); // 割り込みフラグ設定 (フラグ ON)
usleep(1000);

m_com_pa10->off_pa10(); // PA10 の停止処理
m_com_pa10->init_pa10(); // PA10 の初期処理
m_com_pa10->on_pa10(); // PA10 の開始処理
usleep(1000);

m_com_pa10->set_flg(0); // 割り込みフラグ設定 (フラグ OFF)
```

## 3. 8. 6. PA10 待機姿勢（初期関節角度）の設定

手先の目標位置・姿勢が入力される前に、pa10vel コンポーネント活性化後、PA10 はまず待機姿勢へ移行する。待機姿勢は初期関節角度の値で決まり、PA ライブラリ pa\_exe\_axs が呼び出されることでそれへ移行する。待機姿勢設定例を以下に示す。

### 待機姿勢設定実装例

```
ANGLE an;

// init position setting
an.s1 = 0.0;
an.s2 = M_PI/4;
an.s3 = 0.0;
an.e1 = M_PI/4;
an.e2 = 0.0;
an.w1 = M_PI/2;
an.w2 = M_PI;
err = pa_exe_axs(ARM0, AXISALL, &an, WM_WAIT);
```

マクロ定義に関しては、「三菱重工 汎用ロボット PA10 シリーズ プログラミングマニュアル」を参照のこと。

## 4. 特記事項

本モジュールをご利用される場合には、以下の記載事項・条件にご同意いただいたものとします。

- 本モジュールは独立行政法人 新エネルギー・産業技術総合開発機構の「次世代ロボット知能化技術開発プロジェクト」内実施者向けに評価を目的として提供するものであり、商用利用など他の目的で使用することを禁じます。
- ドキュメントに情報を掲載する際には万全を期していますが、それらの情報の正確性またはお客様にとっての有用性等については一切保証いたしません。
- 利用者が本モジュールを利用することにより生じたいかなる損害についても一切責任を負いません。
- 本モジュールの変更、削除等は、原則として利用者への予告なしに行います。また、止むを得ない事由により公開を中断あるいは中止させていただくことがあります。
- 本モジュールの情報の変更、削除、公開の中断、中止により、利用者に生じたいかなる損害についても一切責任を負いません。
- PA ライブラリは、三菱重工業株式会社の製品であり、権利は三菱重工業株式会社に帰属します。

### 【連絡先】

RTC 再利用技術研究センター

〒101-0021 東京都千代田区外神田 1-18-13 秋葉原ダイビル 1303 号室

Tel/Fax : 03-3256-6353 E-Mail : [contact@rtc-center.jp](mailto:contact@rtc-center.jp)