平成24年度修士論文

# 身体パーツの組み換えを考慮した低コストな家庭用ヒューマノイドロボットの開発


学　籍　番　号　1132075

氏　　名　丸山　恭平

知能機械工学専攻　先端ロボティクスコース

主　指　導　教　員　長井隆行　准教授

副　指　導　教　員　金子正秀　教授



提　　出　　日　平成25年2月28日

平成24年度修士論文

# 身体パーツの組み換えを考慮した低コストな家庭用ヒューマノイドロボットの開発


学　籍　番　号　1132075

氏　　名　丸山　恭平

知能機械工学専攻　先端ロボティクスコース

主　指　導　教　員　長井隆行　准教授

副　指　導　教　員　金子正秀　教授



提　出　日　平成25年2月28日

# 概要

本稿では，身体パーツの組み換えを考慮した低コストな家庭用ヒューマノイドロボットのプラットホームを開発することをを目的とする．低コストなロボットを実現するために，市販されている部品や規格品の組み合わせによるロボット開発が有効ではあるが，ロボットの性能低下は避ける事ができない．しかし，ロボットを構成するパーツを用途に応じて組み換えや追加を行うことができればロボットの性能低下をカバーできると考えられる．本論文ではロボットを構成するパーツを身体パーツとして幾つかに分けて設計・製作を行いそれらを組み合わせることで1台のロボットを構築する．また，製作したロボットにおいてパーツの交換や用途によって身体構造の組み換えといったことを可能にするためにRTミドルウェアによるロボットシステムを実装を行う．これにより，身体パーツの組み換えによりロボットの身体構造が変化した場合でもシステムの変更が最小限で済むようにする．システムの評価のためにロボカップ@ホームリーグにおける基本タスクであるモバイルマニピュレーションタスクを実装し，実験を通してロボットの性能評価とシステムの有効性を示す．

# 目次

**1 序論** **1**

1.1 はじめに . . . . . 1

1.2 関連研究 . . . . . 3

1.3 本論文の構成 . . . . . 4

**2 理論** **5**

2.1 全方位移動機構 . . . . . 5

2.2 台車の運動学 . . . . . 6

2.3 地図生成と自己位置推定 . . . . . 7

2.4 ロボットアームの構造 . . . . . 10

2.5 順運動学 . . . . . 10

2.6 アームの逆運動学 . . . . . 14

2.6.1 オイラー角 . . . . . 14

2.6.2 解析的導出 . . . . . 16

2.6.3 ヤコビ行列を用いた導出 . . . . . 20

2.7 平面の検出 . . . . . 23

2.7.1 ハフ変換 . . . . . 23

2.7.2 3 次元ハフ変換による平面の検出 . . . . . 24

2.8 RT ミドルウェア . . . . . 25

2.8.1 組み込み用 RTM . . . . . 27

**3 提案するロボットハードウェア** **28**

3.1 ロボットの基本構成 . . . . . 28

3.1.1 全方位移動台車の設計 . . . . . 30
3.1.2 7 自由度アームの設計 . . . . . 31
3.1.3 伸縮機構の設計 . . . . . 33
3.1.4 ロボットハンドの設計 . . . . . 34
3.1.5 ロボット頭部の設計 . . . . . 35
3.1.6 製作したロボットの全体構成 . . . . . 36

**4 ソフトウェアシステム** **38**
4.1 パーツ組み換えを実現するシステム . . . . . 38
4.1.1 システムの概要 . . . . . 38
4.1.2 ロボットパーツ記述言語 (RPDL) . . . . . 39
4.1.3 動作記述言語 (ADL) . . . . . 41
4.1.4 アクションプログラム . . . . . 41
4.1.5 タスクプログラム . . . . . 41
4.2 RT コンポーネントによるシステムの実装 . . . . . 43
4.2.1 アクション RTC . . . . . 43

**5 実験** **47**
5.1 実験内容 . . . . . 47
5.1.1 マニピュレーションタスクにおける物体の把持範囲 . . . . . 47
5.1.2 モバイルマニピュレーションタスク . . . . . 51
5.1.3 身体パーツを組み換えた場合の動作 . . . . . 52
5.2 結果と考察 . . . . . 54

**6 結論** **55**

**謝辞** **57**

**参考文献** **58**

# 図一覧


1.1 DiGORO . . . . . 2

2.1 全方位移動台車 . . . . . 5
2.2 全方位台車モデル . . . . . 6
2.3 ICP アルゴリズム . . . . . 8
2.4 使用する対応点群 . . . . . 9
2.5 点群 $M$ のほうが多い場合 . . . . . 9
2.6 点群 $S$ のほうが多い場合 . . . . . 9
2.7 アームの構造 . . . . . 10
2.8 アームのリンクベクトル . . . . . 10
2.9 アーム初期姿勢 . . . . . 13
2.10 アーム回転後 . . . . . 13
2.11 4 通りの解 . . . . . 16
2.12 逆運動学の解析的解法 . . . . . 16
2.13 逆運動学による $\theta_1$〜$\theta_3$ の導出 . . . . . 17
2.14 x-y 空間から a-b 空間への変換 . . . . . 23
2.15 平面の 3 次元極座標表示 . . . . . 24
2.16 RTC ミドルウェアおよび RTC コンポーネント . . . . . 26
2.17 RTC ミドルウェアによる分散システム . . . . . 26

3.1 ロボット（最大身長時） . . . . . 29
3.2 ロボット（最小身長時） . . . . . 29
3.3 全方位移動台車 . . . . . 30

3.4 7 自由度アーム . . . . . 31
3.5 アームのための減速機 . . . . . 32
3.6 上半身の構成 . . . . . 33
3.7 伸縮機構（最小時） . . . . . 34
3.8 伸縮機構（最大時） . . . . . 34
3.9 ロボットハンド . . . . . 35
3.10 ロボットの頭部 . . . . . 35
3.11 製作したロボット . . . . . 36
3.12 ハードウェア構成 . . . . . 36

4.1 アームに対する RPDL の例 (紙面の都合で一部のみ表示) . . . . . 40
4.2 頷き動作生成のための ADL の例 (紙面の都合で一部のみ表示) . . . 42
4.3 RTC によるロボットシステム . . . . . 43
4.4 RTno による台車制御用 RTC の実装 . . . . . 44

5.1 マニピュレーションタスクに使用する机 . . . . . 48
5.2 実験環境 . . . . . 48
5.3 物体検出結果のズレ . . . . . 49
5.4 モバイルマニピュレーションタスク環境 . . . . . 51
5.5 モバイルマニピュレーションタスク . . . . . 52
5.6 パーツ組み換え . . . . . 53
5.7 パーツ組み換え時のマニピュレーションタスク . . . . . 53

# 表一覧

3.1 ロボットの仕様 . . . . . 29
3.2 PC スペック . . . . . 37

4.1 主要 RTC 一覧 . . . . . 44

5.1 マニピュレーションタスク実験結果（身長の可変なし） . . . . . 49
5.2 マニピュレーションタスク実験結果（身長の可変あり） . . . . . 50
5.3 モバイルマニピュレーションタスクの結果 . . . . . 52

6.1 ロボット製作費 . . . . . 56

# 第 1 章

# 序論

## 1.1 はじめに

近年，エレクトロニクスの進歩と共に多種多様なロボットの研究・開発がされてきた．中でも，人間と同じ環境で作業を行うことが可能なロボットへの期待が膨らみつつあり，家庭や病院，オフィスなどで人間をサポートすることができる家庭用ロボットやサービスロボットの研究開発が様々な研究機関で行われている [1]-[3]. こうした研究で使用されているヒューマノイドタイプのロボットに共通して言えることの一つは，ロボットプラットフォームが非常に高価だということである．筆者らのグループがこれまで開発してきた家庭内サービスロボット “DiGORO” (図 1.1) も例外ではなく，1 台製作するためのハードウェアコストは約 1 千万円である．

DiGORO の家庭用ロボットとしての性能は、ロボカップ@ホーム [18] への参加を通して評価を行ってきた．ロボカップ@ホームは，家庭用サービスロボットの性能を競う世界的な大会であり，モバイルマニピュレーションを基本として，人の探索・認識・追跡，物体の探索・認識・把持，自然言語理解などを組み合わせたタスクが複数設定されている．実際 DiGORO はこうした世界的な大会で高い成績を修めており，家庭用ロボットとして高いポテンシャルを有している．しかし，決められているタスクのすべてが実行できるわけではなく，性能に対する価格はかなり高いと言わざるを得ない.

ロボットの低コスト化を考えた場合，用途に応じたハードウェア性能を持ったロボットの構築が重要となる．例えば，サービスロボットとして物体操作が必要なロ

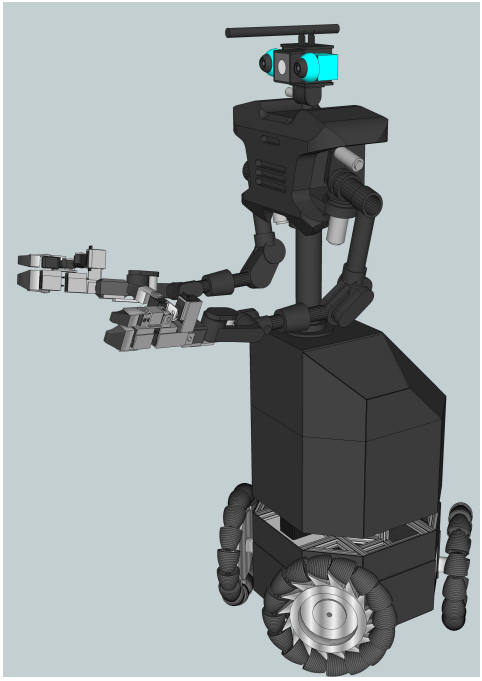

図 1.1 DiGORO

ボットにはアームが必要であるが，インタラクション中心のロボットにおいてそういったものは不要である．ロボットを用途に応じて機能を絞って設計することは低コスト化に繋がると考えられる．しかし，時として現在のロボットに新しく機能を追加したい場合がある．例えば机の上の物体操作を行うロボットにおいて，新たに床の上や棚の上のものを操作したといった場合，予め機能を絞って設計されたロボットにおいてこれを実現するには大変な労力をもって改造するか，新たに設計し直すかのどちらかとなる．いずれの方法でもコストが掛かってっしまう．

こうした問題を解決する方法として主要なロボットパーツの分割設計を行い，用途に応じてそれらを組み換えが行えるようなロボットを本研究では提案する．ロボットを構成するパーツを身体パーツとして設計し，ロボットの目的によりそれらを組み合わせたり組み換えたりすることでロボットを構築する．本研究では家庭内サービスロボットとしてロボカップ@ホームリーグで行われるタスクが行えるようなロボットをベースとして考え，必要となる身体パーツを設計し，実際に製作を行う．DiGORO の性能は維持しつつ，ロボットは市販品や規格品の組み合わせにより低コストに実現することを目的とする．

## 1.2 関連研究

サーボモーターなど市販部品を用いた低コストなヒューマノイドロボットの実現は，Stuckler らによって試みられている [4]．また，文献 [5] では低コストでありながら精度を向上させた 7 自由度のロボットアームを実現している．本論文ではさらに，パーツレベルで柔軟に身体構造の変更を可能にするために，組み込み型の RT ミドルウェアである RTno [12] を利用することを検討する．各パーツにはマイコンが搭載されており，そのマイコン上で動作するモジュールが各パーツを制御することでパーツの自由な脱着が可能となる．こうしたパーツの脱着を自由かつ簡単に行うためには，ロボット全体を制御するためのプログラムについて考える必要がある．これは，多くの行動がロボットの身体的なコンフィギュレーションに依存しているためであり，一般にパーツの脱着や取り付け位置の変更によって，ソフトウェアの大きな変更を余儀なくされる．この問題に対して本論文では，各パーツに関する記述や行動制御プログラムの記述を工夫することで解決することを提案する．

近年こうした問題は徐々に注目を集めつつある．例えば，文献 [6] では，ロボットのハードウェア向け記述言語 Semantic Robot Description Language (SRDL) の提案を行っている．これは，様々なロボットのハードウェアに対する記述の標準化を目指したものであり，ソフトウェア側はこの記述を参照することで様々なロボットに対して共通の行動プログラムを生成することが可能となると考えられる．本研究においても，同様の考え方でハードウェアを記述することを提案するが，SRDL はパーツ毎の記述やその組み換えなどは考慮されていない．

一方，ロボットのタスクを実行するためのソフトウェアについては，クラウドロボティクスの考え方が広がりを見せる中，例えば Web に記述されているタスクに関する知識をいかにロボットの行動プログラムに変換するかといった問題が議論されている．また，自然言語の命令からいかにロボットの行動を生成するかといった問題も様々検討されている．本研究の問題は，ハードウェアの構成が変化した際にも問題なく（物理的に可能な範囲）で，現在動作している行動プログラムを実行できるかということであり，これは前述のハードの記述とソフトウェアの記述をうまく融合させることで可能となると考えられる．

## 1.3 本論文の構成

本論文は以下，次のように構成されている．**2.** 章で基本的な原理を述べ，**3.** 章で提案するロボットのハードウェアについて述べる．**4.** 章では実装するロボットシステムの詳細について述べ，**5.** 章では提案手法を用いた実験の結果と考察を示し **6.** 章で結論と今後の課題について述べる．

# 第 2 章

# 理論

本章ではロボット開発における基本事項やシステムの構成手法を示す.

## 2.1　全方位移動機構

車輪を用いたロボットの移動方法に全方位移動というものがある. これは台車の移動方向が拘束されず, 任意の方向へ向きを変えることなく移動できるものである. 全方位移動を実現するための手法はいくつか挙げられるが, ここではオムニホールを用いた全方位移動機構について説明する.

オムニホイールとは車輪の円周上に横方向に回転可能な駒が配置されている車輪である. これを正方形の各頂点に配置することで全方移動が可能となる. 図 2.4 にモデル化したものを示す. 車輪の回転力が赤色の矢印で表されている. これを x,y 成分ごとに分けると青い矢印の成分がオムニホールの駒によって打ち消される. 結果として x 成分のみが残るため台車は緑の矢印に進むことになる. 各車輪の速度を調整することで任意の方向への移動が可能となる.

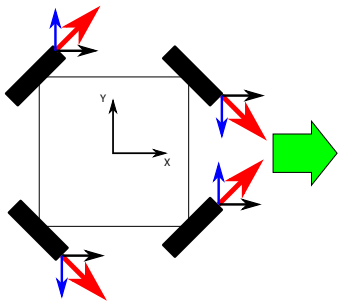


図 2.1 全方位移動台車

## 2.2 台車の運動学

台車の制御には速度制御が必要である. 最低限ロボットに必要な制御としては指定方向に指定速度で移動させることである. これを実現するには, ロボット座標を基準として方向と速度が与えられたときに各車輪対する速度を計算する必要がある. 以下に台車のモデルと台車の移動速度の関係を示す.

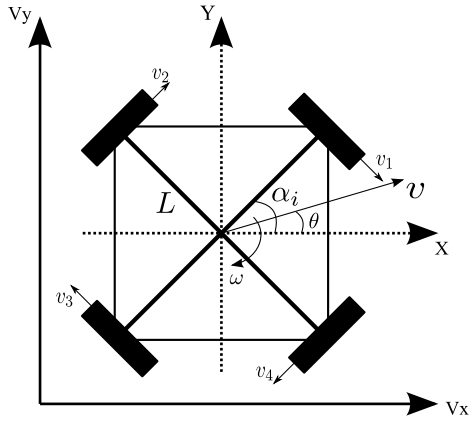


図 2.2 全方位台車モデル

- $v$:台車の移動速度
- $\theta$:ロボット座標を基準とした移動方向
- $v_i$:車輪の回転速度 ($i$=1,2,3,4)
- $\omega$:台車の回転速度
- $\alpha_i$:車輪の取り付け角 ($i$=1,2,3,4)
- $L$:車輪から台車中心までの距離

図より, 各車輪の回転速度 $v_i$ は次式によって表すことができる.

$$\begin{aligned} v_i &= v_x \cos(\alpha_i - \frac{\pi}{2}) + v_y \sin(\alpha_i - \frac{\pi}{2}) + \omega L \\ &= v_x \sin\alpha_i - v_y \cos\alpha_i + \omega L \end{aligned} \tag{2.1}$$

式 2.1 に $\alpha_i$ をそれぞれ代入することで台車の運動学を求めることができ, 以下のように表すことができる.

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} \sin\alpha_1 & -\cos\alpha_1 & L \\ \sin\alpha_2 & -\cos\alpha_2 & L \\ \sin\alpha_3 & -\cos\alpha_3 & L \\ \sin\alpha_4 & -\cos\alpha_4 & L \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{2.2}$$

ここで,$\alpha_i$ を台車のモデルより以下のように仮定する.

$$\begin{cases} \alpha_1 = \frac{1}{4}\pi & \alpha_2 = \frac{3}{4}\pi \\ \alpha_3 = -\frac{3}{4}\pi & \alpha_4 = -\frac{1}{4}\pi \end{cases} \tag{2.3}$$

式 2.3 を式 2.2 に代入することで, 各車輪の速度は次式で表される.

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} \sin\frac{1}{4}\pi & -\cos\frac{1}{4}\pi & L \\ \sin\frac{3}{4}\pi & -\cos\frac{3}{4}\pi & L \\ \sin-\frac{3}{4}\pi & -\cos-\frac{3}{4}\pi & L \\ \sin-\frac{1}{4}\pi & -\cos-\frac{1}{4}\pi & L \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{2.4}$$

$$= \frac{\sqrt{2}}{2} \begin{bmatrix} 1 & 1 & \sqrt{2}L \\ 1 & -1 & \sqrt{2}L \\ -1 & -1 & \sqrt{2}L \\ -1 & 1 & \sqrt{2}L \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{2.5}$$

ここで $v_x$,$v_y$ は $X$,$Y$ 方向の並進速度であり $v$ と $\theta$ を用いて表すならば次式となる.

$$v_x = v\cos(\theta) \tag{2.6}$$

$$v_y = v\sin(\theta) \tag{2.7}$$

以上の式を用いることで, ロボット座標を基準とした移動方向、移動速度および回転速度指定による台車の制御が可能となる.

## 2.3 地図生成と自己位置推定

地図の生成は LRF により得られたデータを用いて行う. LRF には計測できる範囲に限りがあり, 一度の計測だけで周囲環境すべての地図情報を生成することがで

きない. そこで移動ロボットにより周囲の情報を複数の複数の異なった位置から計測し, 各計測データを一つに統合することで地図生成を行う. LRF で計測したデータを一つに統合するためには, 各計測データの位置関係を推定する必要がある. 車輪移動型ロボットの自己位置推定の方法には, 内界センサを用いた手法としてデッドレコニングがある. しかし, デットレコニングは単純に車輪の回転数を計測しているため, 車輪のスリップなどにより位置誤差が生じてしまう, さらに, 生じた位置誤差を補正できないため, 走行距離が増えるほど誤差が累積されていく. よってデットレコニングのみで地図生成を行うと, 各計測データの位置関係に誤差が生じ, 歪んだ地図が生成されてしまう. そこでデッドレコニングによる自己位置推定で生じた位置誤差を補正するため, 外界センサとして LRF を用いる.LRF により計測されたデータの重なった部分を用いて, 各計測データ間の位置合わせを行うことで, 自己位置の補正を行う.

位置合わせの手法として ICP(Iterative Closest Point) アルゴリズムを用いる. ICP アルゴリズムは, 複数の距離画像間で重複して計測された部分を利用して, 繰り返し計算により誤差関数を最小化する関数を求める手法である.

具体的には, まず 2 つの点群 $M$,$S$ があるとき, 点群 $S$ 中の各点 $s_i(1 \leq i \leq N)$ について, 点群 $M$ 中で最も近い点 $m_j(1 \leq j \leq N)$ を対応点とする.(図 2.3) このとき式 (2.8) で表される各点間の距離の 2 乗和 $E_1$ が最小となる移動パラメータ $(R, t)$ を求める. ここで $R$ は回転行列で,$t$ は並進移動ベクトルである.

$$E_1(R,t) = \sum \|m_i - (Rs_i + t)\|^2 \tag{2.8}$$

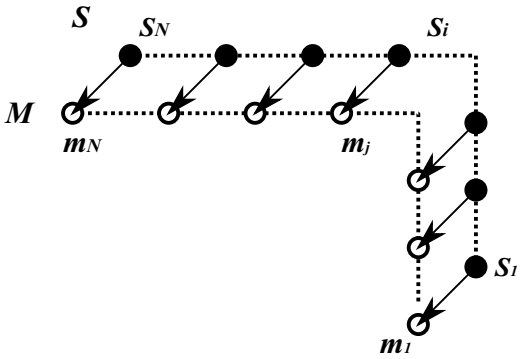


図 2.3 ICP アルゴリズム

実際にマッチングを行うときに使用する対応点群は対応点間が一定の距離以下のもののを使用する.(図 2.4) そして, その対応点群と式 (2.8) によって求めた移動パラメータ $(R, t)$ を用いて, すべての点群を修正する.

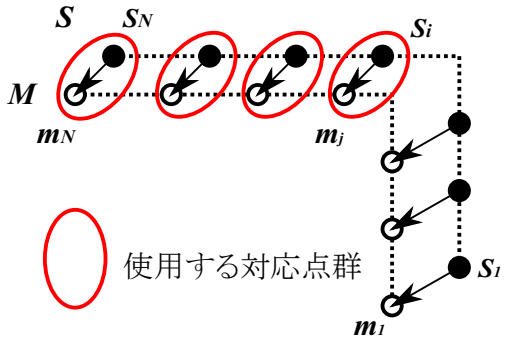


図 2.4 使用する対応点群

また, 点群 $S$ と点群 $M$ の数が異なっている場合もこの方法で修正を行う. その場合, 例えば点群 $M$ のほうが多い時には, 点群 $M$ 中で点群 $S$ の対応点に選ばれない点が存在した状態で修正を行う.(図 2.5)

逆に, 点群 $S$ のほうが多いときは, 点群 $M$ において点群 $S$ 中で最も近い点を選び, その対応点間の距離が最小となるように点群 $S$ を移動する.(図 2.6)

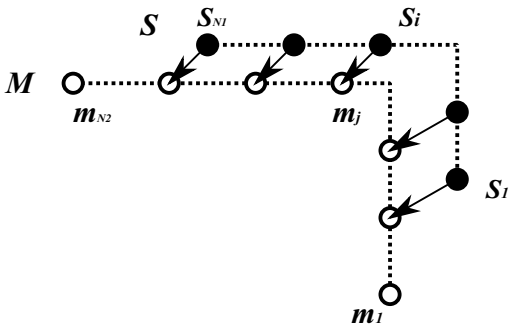


図 2.5 点群 $M$ のほうが多い場合

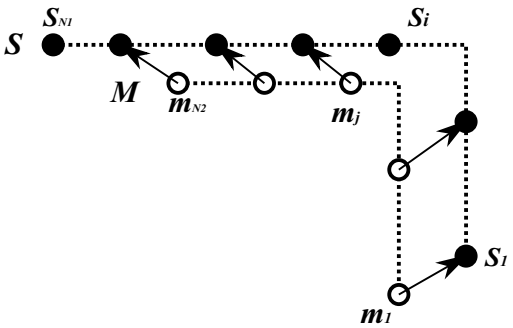


図 2.6 点群 $S$ のほうが多い場合

## 2.4 ロボットアームの構造

一般的に, ロボットのアームは図 2.7 のように，複数のリンクがジョイントによって結合されて構成されている.

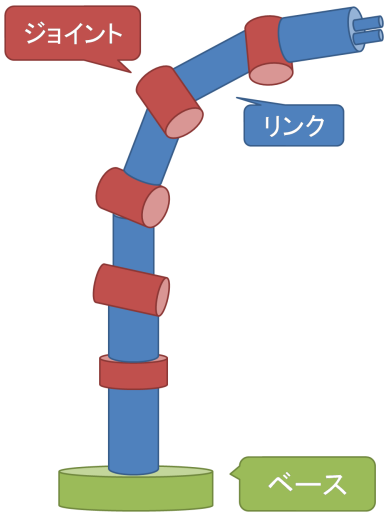


図 2.7 アームの構造

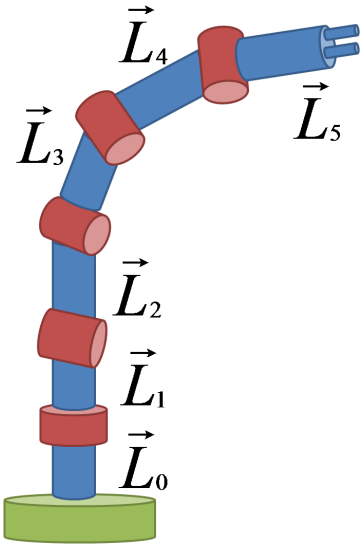


図 2.8 アームのリンクベクトル

各ジョイントが回転することにより姿勢が変化し，様々な位置へ動かすことができる．一般に，3 次元空間の任意の位置へ動かすためにはジョイントの数が 6 つ (6 自由度) 必要ということが知られている.

## 2.5 順運動学

アームの各リンクが回転した時の，手先の位置は一意に求めることができる.

図 2.8 のように，アームの根元から順番に，各ジョイントから次のジョイントへのリンクを $\mathbf{L_0}, \mathbf{L_1}, \cdots, \mathbf{L_n}$ とする．各リンクが回転したときの，そのリンクを基準とした相対的な回転行列を $R_n$ とすると手先の位置 $\mathbf{P}$ は

$$\mathbf{P} = \mathbf{L_0} + \mathbf{L_1} + \mathbf{L_2} + \mathbf{L_3} + \cdots + \mathbf{L_{n-1}} + \mathbf{L_n} \tag{2.9}$$

$$= \mathbf{L_0} + {}^0R_1(\hat{\mathbf{L}}_\mathbf{1}) + {}^0R_2(\hat{\mathbf{L}}_\mathbf{2}) + {}^0R_3(\hat{\mathbf{L}}_\mathbf{3}) + {}^0R_{n-1}(\hat{\mathbf{L}}_\mathbf{n-1}) + {}^0R_n(\hat{\mathbf{L}}_\mathbf{n}) \tag{2.10}$$

$$= \mathbf{L_0} + R_1(\hat{\mathbf{L}}_\mathbf{1}) + R_1R_2(\hat{\mathbf{L}}_\mathbf{2}) + R_1R_2R_3(\hat{\mathbf{L}}_\mathbf{3}) + R_1R_2R_3R_{n-1}R_n(\hat{\mathbf{L}}_\mathbf{n}) \tag{2.11}$$

$$= \hat{\mathbf{L}}_\mathbf{0} + R_1(\hat{\mathbf{L}}_\mathbf{1} + R_2(\hat{\mathbf{L}}_\mathbf{2} + R_2(\hat{\mathbf{L}}_\mathbf{3} + \cdots + R_{n-1}(\hat{\mathbf{L}}_\mathbf{n-1} + R_n(\hat{\mathbf{L}}_\mathbf{n}))))) \tag{2.12}$$

となる. ただし, 初期姿勢とそのリンクベクトルを $\hat{\mathbf{L}}_\mathbf{n}$ とし, ${}^0R_i = R_1R_2\cdots R_{i-1}R_i$ であるとする.

任意の座標 $(x, y, z)$ を x 軸を基準に $\alpha$ 回転させて $(x', y', z')$ に移動したとき, 回転行列を $\mathbf{R}_x$ とすると, 次式で表される.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \mathbf{R}_x \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.13}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.14}$$

同様に y 軸を基準に $\beta$ 回転させたときの回転行列を $\mathbf{R}_y$ とすると,

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \mathbf{R}_y \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.15}$$

$$= \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.16}$$

となり，z 軸を基準に $\gamma$ 回転させたときの回転行列を $\mathbf{R}_z$ とすると，

$$\begin{bmatrix} x^{'} \\ y^{'} \\ z^{'} \\ 1 \end{bmatrix} = \mathbf{R}_z \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.17}$$

$$= \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 & 0 \\ \sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.18}$$

となる．さらに，任意の単位ベクトル $(n_x, n_y, n_z)$ まわりに $\theta$ 回転した時の回転行列 $\mathbf{R_a}$ を用いて，

$$\begin{bmatrix} x^{'} \\ y^{'} \\ z^{'} \\ 1 \end{bmatrix} = \mathbf{R}_a \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.19}$$

$$= \begin{bmatrix} R_{a11} & R_{a12} & R_{a13} & 0 \\ R_{a21} & R_{a22} & R_{a23} & 0 \\ R_{a31} & R_{a32} & R_{a33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.20}$$

と一般化することができる．ただし，

$$
\begin{aligned}
R_{a11} &= n_x^2(1-\cos\theta)+\cos\theta \\
R_{a12} &= n_x n_y(1-\cos\theta)-n_z\sin\theta \\
R_{a13} &= n_z n_x(1-\cos\theta)+n_y\sin\theta \\
R_{a21} &= n_x n_y(1-\cos\theta)+n_z\sin\theta \\
R_{a22} &= n_y^2(1-\cos\theta)+\cos\theta \\
R_{a23} &= n_y n_z(1-\cos\theta)-n_x\sin\theta \\
R_{a31} &= n_z n_x(1-\cos\theta)-n_y\sin\theta \\
R_{a23} &= n_y n_z(1-\cos\theta)+n_x\sin\theta \\
R_{a33} &= n_z^2(1-\cos\theta)+\cos\theta
\end{aligned}
$$

とする.

図 2.9,2.10 の場合，第 0 リンク，第 1 リンク共に z 軸を回転軸にもつので,

$$
\begin{aligned}
&\mathbf{P} = \mathbf{L_0} + \mathbf{L_1} \\
&\mathbf{L_0} = R_0\hat{\mathbf{L}}_\mathbf{0} = \mathrm{R_z}(\theta_0) \\
&\mathbf{L_1} = R_0R_1\hat{\mathbf{L}}_\mathbf{1} = \mathrm{R_z}(\theta_0)\mathrm{R_z}(\theta_1)\hat{\mathbf{L}}_\mathbf{1}
\end{aligned}
$$

と求めることができる.

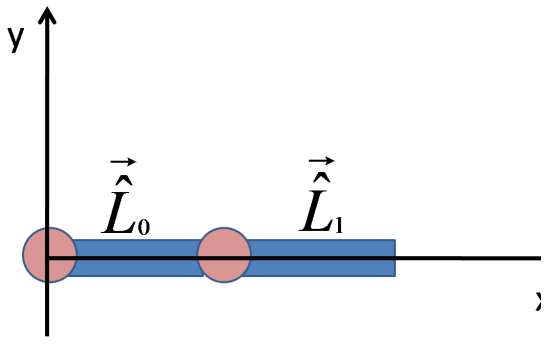


図 2.9 アーム初期姿勢

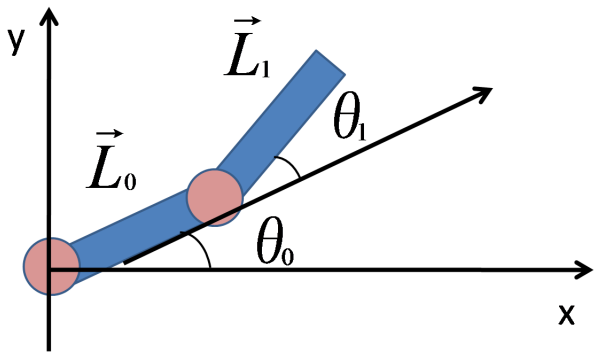


図 2.10 アーム回転後

## 2.6 アームの逆運動学

手先の位置から各関節を求める場合を逆運動学という．順運動学と違い，解が一つに決まらない場合がある．

### 2.6.1 オイラー角

まず，姿勢を表すためのオイラー角を示す．基準座標系を z 軸回りに $\phi$ 回転させ，次に回転後の座標系を回転後の y' 軸回りに $\theta$ 回転させ，最後に z" 軸回りに $\psi$ 回転させると，任意の姿勢にある座行系と必ず重ねあわせることができる．このときに利用した 3 つの回転，すなわち z,y',z" 軸回りに行った $\phi, \theta, \psi$ の 3 つの回転角の組をオイラー角という．この 3 つの回転を行った後の座標系は，

$$\begin{aligned}
&\mathbf{R_z}(\phi)\mathbf{R_y}(\theta)\mathbf{R_z}(\psi) \\
&= \begin{bmatrix} \mathrm{C}_\phi & -\mathrm{S}_\phi & 0 \\ \mathrm{S}_\phi & \mathrm{C}_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{C}_\theta & 0 & \mathrm{S}_\theta \\ 0 & 1 & 0 \\ -\mathrm{S}_\theta & 0 & \mathrm{C}_\theta \end{bmatrix} \begin{bmatrix} \mathrm{C}_\psi & -\mathrm{S}_\psi & 0 \\ \mathrm{S}_\psi & \mathrm{C}_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.21) \\
&= \begin{bmatrix} \mathrm{C}_\phi\mathrm{C}_\theta\mathrm{C}_\psi - \mathrm{S}_\phi\mathrm{S}_\psi & -\mathrm{C}_\phi\mathrm{C}_\theta\mathrm{S}_\psi - \mathrm{S}_\phi\mathrm{C}_\psi & \mathrm{C}_\phi\mathrm{S}_\theta \\ \mathrm{S}_\phi\mathrm{C}_\theta\mathrm{C}_\psi + \mathrm{C}_\phi\mathrm{S}_\psi & -\mathrm{S}_\phi\mathrm{C}_\theta\mathrm{S}_\psi + \mathrm{C}_\phi\mathrm{C}_\psi & \mathrm{S}_\phi\mathrm{S}_\theta \\ -\mathrm{S}_\theta\mathrm{C}_\psi & \mathrm{S}_\theta\mathrm{S}_\psi & \mathrm{C}_\theta \end{bmatrix} \quad (2.22)
\end{aligned}$$

となる．ただし，

$$\begin{aligned}
\mathrm{S}_\phi &= \sin\phi \\
\mathrm{C}_\phi &= \cos\phi \\
\mathrm{S}_\theta &= \sin\theta \\
\mathrm{C}_\theta &= \cos\theta \\
\mathrm{S}_\psi &= \sin\psi \\
\mathrm{C}_\psi &= \cos\psi
\end{aligned}$$

とする．つまり任意の回転行列 $\mathbf{E}$ に対して

$$\mathbf{E} = \begin{bmatrix} e_{xx} & e_{yx} & e_{zx} \\ e_{xy} & e_{yy} & e_{zy} \\ e_{xz} & e_{yz} & e_{zz} \end{bmatrix} \tag{2.23}$$

$$= \begin{bmatrix} \mathrm{C}_\phi\mathrm{C}_\theta\mathrm{C}_\psi - \mathrm{S}_\phi\mathrm{S}_\psi & -\mathrm{C}_\phi\mathrm{C}_\theta\mathrm{S}_\psi - \mathrm{S}_\phi\mathrm{C}_\psi & \mathrm{C}_\phi\mathrm{S}_\theta \\ \mathrm{S}_\phi\mathrm{C}_\theta\mathrm{C}_\psi + \mathrm{C}_\phi\mathrm{S}_\psi & -\mathrm{S}_\phi\mathrm{C}_\theta\mathrm{S}_\psi + \mathrm{C}_\phi\mathrm{C}_\psi & \mathrm{S}_\phi\mathrm{S}_\theta \\ -\mathrm{S}_\theta\mathrm{C}_\psi & \mathrm{S}_\theta\mathrm{S}_\psi & \mathrm{C}_\theta \end{bmatrix} \tag{2.24}$$

となるように $\phi, \theta, \psi$ を決定すれば，任意の姿勢がオイラー角で表わせたことになる．

$$\frac{e_{zx}^2 + e_{zy}^2}{e_{zz}^2} = \tan^2\theta \tag{2.25}$$

より

$$\theta = \mathrm{atan2}(\pm\sqrt{e_{zx}^2 + e_{zy}^2}, e_{zz}) \tag{2.26}$$

とかける．

$$\psi = \mathrm{atan2}(\pm e_{yz}, \mp e_{xz}) \tag{2.27}$$

と求まり (複合同順)，同様に $\phi$ も

$$\phi = \mathrm{atan2}(\pm e_{zy}, \pm e_{zx}) \tag{2.28}$$

として求まる (複合同順)．ここで用いた $\mathrm{atan2}(y, x)$ は C 言語で用いられる関数を利用した表記法であり，括弧内でカンマに区切られたはじめの値を $y$，後ろの値を $x$ としたとき，ベクトル $[\ x \quad y\ ]^T$ が x 軸となす角度を表す．これをもちいると第 1 象限から第 4 象限までの解を $-\pi$〜$\pi$ の範囲ですべて表すことができるので大変便利である．以下，本論文でも atan2 を利用する．

### 2.6.2　解析的導出

実際にアームの姿勢から関節角を解析的に導出する手段を示す．例えば図 2.11 のように，同じ手先の位置でも様々な姿勢をとる可能性がある．

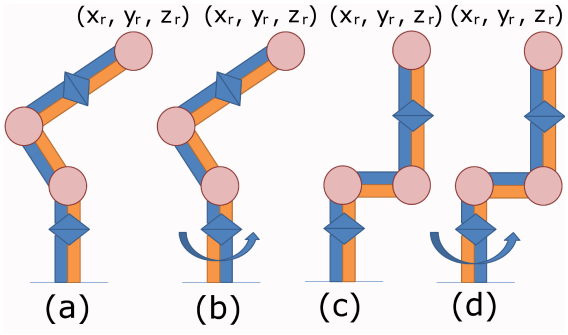


図 2.11 4 通りの解

空間の好きな位置に動かすために必要な 6 自由度マニピュレータの全てのパターンを解析的に逆運動学を解くのは難しい．そこで，マニピュレータを設計するときに，あらかじめ逆運動学が解けるように関節のタイプや軸の向き，リンクパラメータを決めておく必要がある．図 2.12 に代表的な 6 自由度のアームのモデルを示す．

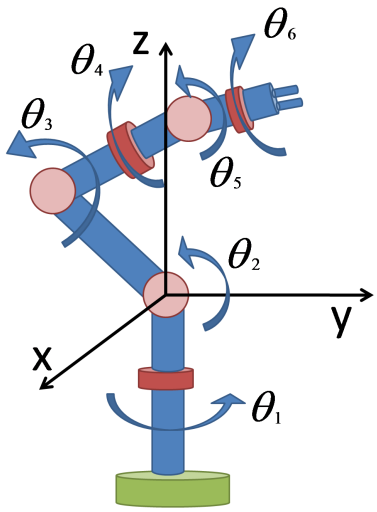


図 2.12 逆運動学の解析的解法

この構造のロボットでは，根元から数えて 4 番目以降の手先側 3 つの関節の回転軸が第 5 関節で交わっており，しかもこの交点は手先に目標の位置・姿勢が与えら

れると，手先側 3 つの関節の関節角の値によらず，一意に定まってしまう．この 5 番目の関節の位置を手首と呼ぶとすると，この手首の位置は，根元から 3 つめまでの関節の値だけで決まる．この関係を使うと逆運動学計算が簡単になる．手首位置が根元側 3 関節で決定される特徴をもつマニピュレータについて，逆運動学の計算方法を示す．まず，達成した位置・姿勢に仮想的に手先を置き，そのときに手首の位置 $(x_r, y_r, z_r)$ がどこになるのかを求める．そして，根元からの 3 関節の値 $\theta_1$〜$\theta_3$ を決定する．そしてこれらの値を使い，まずは $\theta_4 = 0$ として第 4 リンクの座標系を求める．この座標系は第 5 関節の位置を原点にもつ．$\theta_4$〜$\theta_6$ の値は，$\theta_4 = 0$ として暫定的に求まった座標系から見た手先目標座標系を求め，$\theta_4$〜$\theta_6$ を決定すればよい．

実際に図 2.12 にあてはめてみる．図 2.12 は第 4 リンク座標系から第 6 リンク座標系までの座標系はすべて原点が等しく，第 5 関節上にある．さらに，第 4 関節の位置は第 1〜3 関節の関節角だけで決まってしまう．ただし図 2.11 のように解は 4 通り存在する．そこでまずは図 2.11 の (a) を考える．

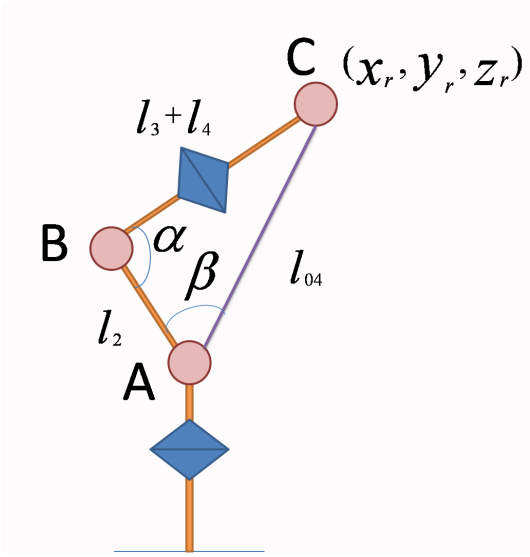


図 2.13 逆運動学による $\theta_1$〜$\theta_3$ の導出

図 2.13 に示す A 点 (基準座標系原点)，B 点 (第 3 関節)，C 点 (第 4 関節) を結んだ三角形を考える．この三角形は 3 辺の長さが全て求まるので，3 つの角も簡単に求めることができる．図 2.13 のマニピュレータを真上から見下ろすと，第 2，第 3，

第 4 リンクは，xy 平面の原点を通り，$(x_r, y_r)$ に向かう線分として見える．そして，この線分が x 軸となす角が $\theta_1$ となる．ただし逆運動学には図 2.11 の (a) と (b) の場合があるので，次式となる．

$$\theta_1 = \mathrm{atan2}(-x_r, y_r) \pm \frac{\pi}{2} \tag{2.29}$$

AC の長さを $l_{04}$ とすると

$$l_{04} = \sqrt{x_r^2 + y_r^2 + z_r^2}$$

となるので，余弦定理より，

$$2l_2 l_{34} \cos\alpha = l_2^2 + l_{34}^2 - l_{04}^2$$

が得られる．ただし $s_{34} = l_3 l_3 + s_4$ であるさらに $1 - \cos^2\alpha = \sin^2\alpha$ より

$$2l_2 l_{34} \sin\alpha = \sqrt{(l_2^2 + l_{34}^2 + l_{04}^2)^2 - 2(l_2^4 + l_{34}^4 + l_{04}^4)}$$

となる．よって

$$\alpha = \mathrm{atan2}(\sin\alpha, \cos\alpha) = atan2(\kappa, l_2^2 + l_{34}^2 - l_{04}^2)$$

として $\alpha$ が求まる．ただし

$$\kappa = \sqrt{(l_2^2 + l_{34}^2 + l_{04}^2)^2 - 2(l_2^4 + l_{34}^4 + l_{04}^4)}$$

である．これより，

$$\theta_3 = \pm\alpha - \frac{\pi}{2} \tag{2.30}$$

となる．+，-の符号はそれぞれ，図 2.11(a),(d) と (b),(c) の場合に対応している．$\beta$ に関しても余弦定理より

$$2l_2l_{34}\cos\beta = l_2^2 + l_{34}^2 - l_{04}^2$$

が成り立ち

$$2l_2l_{04}\sin\beta = \kappa$$

となるので

$$\beta = \mathrm{atan2}(\sin\beta, \cos\beta) = \mathrm{atan2}(\kappa, l_2^2 + l_{34}^2 - l_{04}^2)$$

となる．よって $\theta_2$ は次のように求まる．

$$\text{図 2.11}(a) \quad \theta_2 = \mathrm{atan2}(z_r, \sqrt{(x_r)_2 + (y_r)_2}) + \beta \tag{2.31}$$

$$\text{図 2.11}(b) \quad \theta_2 = \pi - \{atan2(z_r, \sqrt{x_r^2 + y_r^2}) + \beta\} \tag{2.32}$$

$$\text{図 2.11}(c) \quad \theta_2 = \mathrm{atan2}(z_r, \sqrt{(x_r)_2 + (y_r)_2}) - \beta \tag{2.33}$$

$$\text{図 2.11}(d) \quad \theta_2 = \pi - \{atan2(z_r, \sqrt{x_r^2 + y_r^2}) - \beta\} \tag{2.34}$$

残りの 3 つ $\theta_4$〜$\theta_6$ を求める手順は以下の通りである．運動学の計算により $\theta_4 = 0$ とした場合の第 4 リンク座標系 $\Sigma_{4'}$ の姿勢が求まる．このときの姿勢を表す回転行列を ${}^w\mathbf{R}_{\mathbf{4'}}$ とする．また，第 6 リンク座標系 $\Sigma_6$ が達成すべき基準座標系対する回転行列を ${}^w\mathbf{R}_{\mathbf{6'}}$ とする．$\Sigma_6$ の $\Sigma_{4'}$ に対する回転行列 ${}^{4'}\mathbf{R}_{\mathbf{6'}}$ は

$${}^{4'}\mathbf{R}_{\mathbf{6'}} = {}^w\mathbf{R}_{\mathbf{6'}-1}{}^w\mathbf{R}_{\mathbf{6'}} \tag{2.35}$$

で求めることができる．そして $\theta_4$〜$\theta_6$ はそれぞれ z,y,z 軸を基準に回転しているので，

$$\begin{aligned}\theta_4 &= \phi \\ \theta_5 &= \theta \\ \theta_6 &= \psi\end{aligned}$$

としてオイラー角の式2.28〜2.27 として求めることができる.

### 2.6.3 ヤコビ行列を用いた導出

解析的に求めることができない場合，ヤコビ行列を利用した計算方法が必要となる．ヤコビ行列は，手先の微小変位と関節の微小変位の関係を表す行列である.

#### 2.6.3.1 ヤコビ行列

マニピュレータとして 6 自由度のものを考え，関節の値を $q_1$〜$q_6$ ，手先の位置・姿勢を表す変数を $x, y, z, \phi, \theta, \psi$ とする．運動学計算結果から，

$$x = \mathrm{f_x}(q_1, q_2, q_3, q_4, q_5, q_6) = \mathrm{f_x}(\mathbf{q}) \tag{2.36}$$

ただし

$$\mathbf{q} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix}^T \tag{2.37}$$

とかける．よって，$q_1$〜$q_6$ の各関節がそれぞれ微小点に $\Delta q_1$〜$\Delta q_6$ を発生したとすると，

$$\Delta x = \frac{\partial f_x}{\partial q_1}\Delta q_1 + \frac{\partial f_x}{\partial q_2}\Delta q_2 + \frac{\partial f_x}{\partial q_3}\Delta q_3 + \frac{\partial f_x}{\partial q_4}\Delta q_4 + \frac{\partial f_x}{\partial q_5}\Delta q_5 + \frac{\partial f_x}{\partial q_6}\Delta q_6 \tag{2.38}$$

で x 方向の移動量を求めることができる．同様に $y, z, \phi, \theta, \psi$ についての運動学計算結果を

$$y = f_y(\mathbf{q}), z = f_z(\mathbf{q}), \phi = f_\phi(\mathbf{q}), \theta = f_\theta(\mathbf{q}), \psi = f_\psi(\mathbf{q})$$

とし，これらが $q_1$〜$q_6$ によって，どれだけずつ変化するかをすべて並べてかくと，行列を用いて

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \phi \\ \Delta \theta \\ \Delta \psi \end{bmatrix} = \begin{bmatrix} \frac{\partial f_x}{\partial q_1} & \frac{\partial f_x}{\partial q_2} & \frac{\partial f_x}{\partial q_3} & \frac{\partial f_x}{\partial q_4} & \frac{\partial f_x}{\partial q_5} & \frac{\partial f_x}{\partial q_6} \\ \frac{\partial f_y}{\partial q_1} & \frac{\partial f_y}{\partial q_2} & \frac{\partial f_y}{\partial q_3} & \frac{\partial f_y}{\partial q_4} & \frac{\partial f_y}{\partial q_5} & \frac{\partial f_y}{\partial q_6} \\ \frac{\partial f_z}{\partial q_1} & \frac{\partial f_z}{\partial q_2} & \frac{\partial f_z}{\partial q_3} & \frac{\partial f_z}{\partial q_4} & \frac{\partial f_z}{\partial q_5} & \frac{\partial f_z}{\partial q_6} \\ \frac{\partial f_\phi}{\partial q_1} & \frac{\partial f_\phi}{\partial q_2} & \frac{\partial f_\phi}{\partial q_3} & \frac{\partial f_\phi}{\partial q_4} & \frac{\partial f_\phi}{\partial q_5} & \frac{\partial f_\phi}{\partial q_6} \\ \frac{\partial f_\theta}{\partial q_1} & \frac{\partial f_\theta}{\partial q_2} & \frac{\partial f_\theta}{\partial q_3} & \frac{\partial f_\theta}{\partial q_4} & \frac{\partial f_\theta}{\partial q_5} & \frac{\partial f_\theta}{\partial q_6} \\ \frac{\partial f_\psi}{\partial q_1} & \frac{\partial f_\psi}{\partial q_2} & \frac{\partial f_\psi}{\partial q_3} & \frac{\partial f_\psi}{\partial q_4} & \frac{\partial f_\psi}{\partial q_5} & \frac{\partial f_\psi}{\partial q_6} \end{bmatrix} \begin{bmatrix} \Delta q_1 \\ \Delta q_2 \\ \Delta q_3 \\ \Delta q_4 \\ \Delta q_5 \\ \Delta q_6 \end{bmatrix} \tag{2.39}$$

と表すことができる．さらに，

$$\Delta \mathbf{x} = \begin{bmatrix} \Delta x & \Delta y & \Delta z & \Delta \phi & \Delta \theta & \Delta \psi \end{bmatrix}^T$$

$$\Delta \mathbf{q} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix}^T$$

とし，$\Delta \mathbf{q}$ の係数となっている行列を $\mathbf{J}$ とおくと

$$\Delta \mathbf{x} = \mathbf{J} \Delta \mathbf{q} \tag{2.40}$$

となる．この行列 $\mathbf{J}$ をこの 6 自由度マニピュレータで手先の位置・姿勢 6 成分を制御する場合のヤコビ行列と呼ぶ．もしこのマニピュレータの制御対象が (x,y,z) 座標だけで姿勢が問われなければ，

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} \frac{\partial f_x}{\partial q_1} & \frac{\partial f_x}{\partial q_2} & \frac{\partial f_x}{\partial q_3} & \frac{\partial f_x}{\partial q_4} & \frac{\partial f_x}{\partial q_5} & \frac{\partial f_x}{\partial q_6} \\ \frac{\partial f_y}{\partial q_1} & \frac{\partial f_y}{\partial q_2} & \frac{\partial f_y}{\partial q_3} & \frac{\partial f_y}{\partial q_4} & \frac{\partial f_y}{\partial q_5} & \frac{\partial f_y}{\partial q_6} \\ \frac{\partial f_z}{\partial q_1} & \frac{\partial f_z}{\partial q_2} & \frac{\partial f_z}{\partial q_3} & \frac{\partial f_z}{\partial q_4} & \frac{\partial f_z}{\partial q_5} & \frac{\partial f_z}{\partial q_6} \end{bmatrix} \begin{bmatrix} \Delta q_1 \\ \Delta q_2 \\ \Delta q_3 \\ \Delta q_4 \\ \Delta q_5 \\ \Delta q_6 \end{bmatrix}$$

より，この場合のヤコビ行列 $\mathbf{J}$ は

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_x}{\partial q_1} & \frac{\partial f_x}{\partial q_2} & \frac{\partial f_x}{\partial q_3} & \frac{\partial f_x}{\partial q_4} & \frac{\partial f_x}{\partial q_5} & \frac{\partial f_x}{\partial q_6} \\ \frac{\partial f_y}{\partial q_1} & \frac{\partial f_y}{\partial q_2} & \frac{\partial f_y}{\partial q_3} & \frac{\partial f_y}{\partial q_4} & \frac{\partial f_y}{\partial q_5} & \frac{\partial f_y}{\partial q_6} \\ \frac{\partial f_z}{\partial q_1} & \frac{\partial f_z}{\partial q_2} & \frac{\partial f_z}{\partial q_3} & \frac{\partial f_z}{\partial q_4} & \frac{\partial f_z}{\partial q_5} & \frac{\partial f_z}{\partial q_6} \end{bmatrix} \tag{2.41}$$

となる．また，関節が 6 より多くても少なくても，ヤコビ行列はつくることができる．つまり，ヤコビ行列は，行わせたい作業に必要となる変数の数 (作業空間の次元) とマニピュレータの関節の数 (マニピュレータの自由度) に応じて決定される．

#### 2.6.3.2 導出

どんなに複雑な構造のマニピュレータも，運動学の計算は機械的に行うことができる．そこで，正確ではないけれども，手先の目標の位置・姿勢の近くまでもっていくことのできる関節角を適当に決める．そして，この関節角で達成される手先位置・姿勢と目標の位置・姿勢の誤差の分だけ手先を動かすことのできる関節角の微小変位を求め，いまの変位を修正する．この補正量は次式で表される．

$$\Delta\boldsymbol{\theta} = \mathbf{J}(\boldsymbol{\theta})^{-1}(\mathbf{x}_d - \mathbf{x}(\boldsymbol{\theta})) \tag{2.42}$$

ただし，$\mathbf{J}$ はヤコビ行列，$\mathbf{x}_d$ はマニピュレータの手先の目標位置・姿勢である．そして現在の関節変位ベクトル $\boldsymbol{\theta}$ を

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$$

として修正する．ヤコビ行列が線形近似を前提としているため，一度の修正では手先を正確に目標にもっていくことはできない．目標の位置・姿勢との誤差が十分小さくなるまで繰り返し計算し，逆運動学の解とする．

## 2.7 平面の検出

取得した 3 次元情報の中から平面を検出するのに，3 次元ハフ変換が利用可能である．まず最初に基本となる 2 次元でのハフ変換の説明を行い，次に 3 次元ハフ変換を用いた平面検出の手法について説明する．

### 2.7.1 ハフ変換

ハフ変換はディジタル画像処理の分野において，入力画像中の直線を検出するために頻繁に用いられている．x-y 空間のある点を通る直線は，傾き $a$，切片 $b$ をパラメータとして表すことができ，a-b パラメータ空間では，x-y 空間の 1 点が直線として表される．ここで，図 2.14 に示すように，x-y 空間の点 $(x_i, y_i), i = 1, 2, , , N$ が 1 本直線上に乗っている場合，$a$-$b$ パラメータ空間ではそれらの点は N 本の直線で表わされ，$(a_0, b_0)$ の 1 点で交差する．

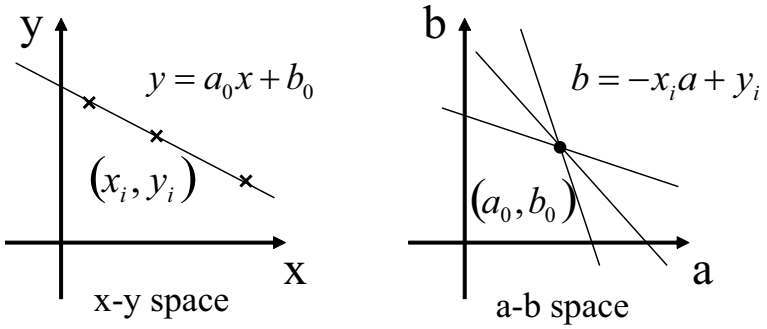


図 2.14 x-y 空間から a-b 空間への変換

つまり，x-y 空間上で最も多くの点を通過する直線を求めるためには，$a$-$b$ パラメータ空間上で最も多く直線が交差している点を見つければ良いということになる．これがハフ変換の基本原理である．ただし，一般的には $\theta$, $\rho$ の極座標空間を用いることが多く，x-y 空間上の点はハフ変換によって $\theta$-$\rho$ パラメータ空間上では曲

線として表されることになる．実際に，x-y 空間上の 2 点からパラメータ $\theta$, $\rho$ を計算し，そのパラメータが対応するビンに投票を行うという処理を繰り返し行うことで，パラメータのピークを求めることができる．

### 2.7.2 3 次元ハフ変換による平面の検出

ハフ変換を 3 次元に拡張したものが，3 次元ハフ変換である．3 次元ハフ変換を用いることで 3 次元情報中の平面を求めることができる．以下に 3 次元ハフ変換による平面の検出を説明する．

X-Y-Z 空間上の平面は, x 軸との角度 $\theta$, y 軸との角度 $\phi$, 原点からの距離 $\rho$ の 3 つのパラメータで表わされ, 次の式で与えられる．

$$\rho = x\cos\theta\cos\phi + y\sin\theta\cos\phi + z\sin\phi \tag{2.43}$$

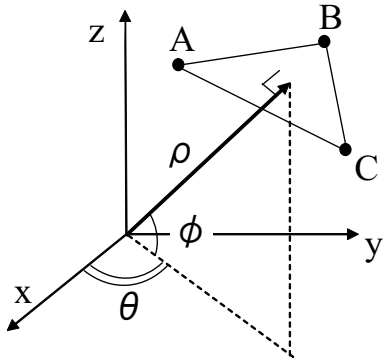


図 2.15 平面の 3 次元極座標表示

$\theta$, $\phi$, $\rho$ の 3 つのパラメータで表わされる 3 次元空間を考えるとき，X-Y-Z 空間上の平面は全て $\theta$-$\phi$-$\rho$ 空間上の点として表わされる．つまり，平面を 3 次元ハフ変換したものは $\theta$-$\phi$-$\rho$ 空間上の点となる．

ここで，X-Y-Z 空間上の点 $P(x_p, y_p, z_p)$ を考える．この点を通る平面は無数に存在し，次の平面の式で表わすことができる．

$$\rho = x_p\cos\theta\cos\phi + y_p\sin\theta\cos\phi + z_p\sin\phi \tag{2.44}$$

また，点 $P(x_p, y_p, z_p)$ を通る全ての平面は式 2.44 によって $\theta$-$\phi$-$\rho$ 空間上で曲平面として描くことができる．

パラメータ $\theta_0$, $\phi_0$, $\rho_0$ をもつ平面上にのる 3 次元点 $P(x_i, y_i, z_i)$ がいくつか存在するとき, それぞれの点が自身を通る全ての平面を $\theta$-$\phi$-$\rho$ 空間上の 1 つの曲面で表わすことができ, それらの曲面は, $(\theta_0, \phi_0, \rho_0)$ の 1 点で交わることになる.

そこで，入力された 3 次元情報から平面を求めるために，$\theta$-$\phi$-$\rho$ 空間を任意の数のビンに量子化する．入力された 3 次元情報からランダムで 3 点を選び，パラメータ $\theta$, $\phi$, $\rho$ を計算し，求めたパラメータに対応するビンへの投票を行う．この処理を何度も繰り返し行い，最大の投票数を求めることで，最も頻度の高いパラメータを求めることができる．このパラメータで表わされる平面が，入力データ中の「もっとも平面らしい面」となる.

## 2.8 RT ミドルウェア

ロボットのソフトウェアは，モジュール群として実装される．ロボットシステムのような大規模システムにおいては分散制御システムを採用することが多い．分散制御システムにおいて異なるシステム間でデータをやり取りする際に用いられるのがミドルウェアである．ロボット用のミドルウェアとして産総研が開発している OpenRTM(RTM) がある．RTM は国際標準化団体 OMG (Object Management Group) において標準化されたロボット向けのソフトウェアである [11]．ロボット機能のソフトウェア要素をモジュール化された部品 (RT コンポーネント:RTC) とし (図 2.16)，これらの部品を組み合わせることで様々なロボットシステムを OS や言語の壁を超えて構築することが可能となる．(図 2.17)

RTC には次のような機能が実装されている.

- アクティビティ・実行コンテキスト : コンポーネントのライフサイクルや，コアロジックの実行を行う
- データポート : 連続的なデータの送受信を行うデータ指向ポートで，同規格のポート同士は動的に接続・切断が可能
- サービスポート : ユーザー定義が可能なインターフェースでコンポーネントの持つ詳細な機能にアクセスが可能

- コンフィグレーション：パラメータを保持する仕組みで，RTC 実行時に動的に変更が可能

RTC を用いることでシステムの並行開発，再利用，交換や更新，分離などが可能となり，開発効率の向上やシステムの柔軟性，拡張性，安定性の向上が期待できる．RTC 同士の通信は RTM により隠蔽されるため，ハードウェアを制御する部分を RTC 化することで，ハードウェアの知識が無くてもハードウェアの制御が容易に行える．

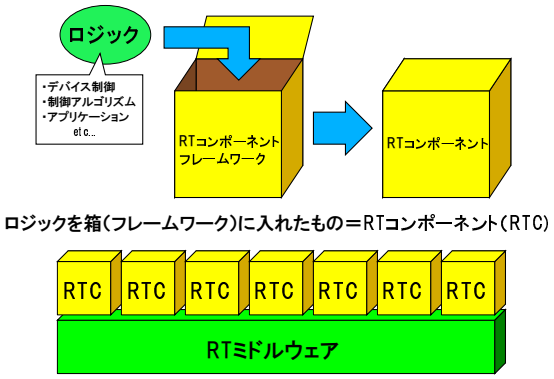


図 2.16 RTC ミドルウェアおよび RTC コンポーネント

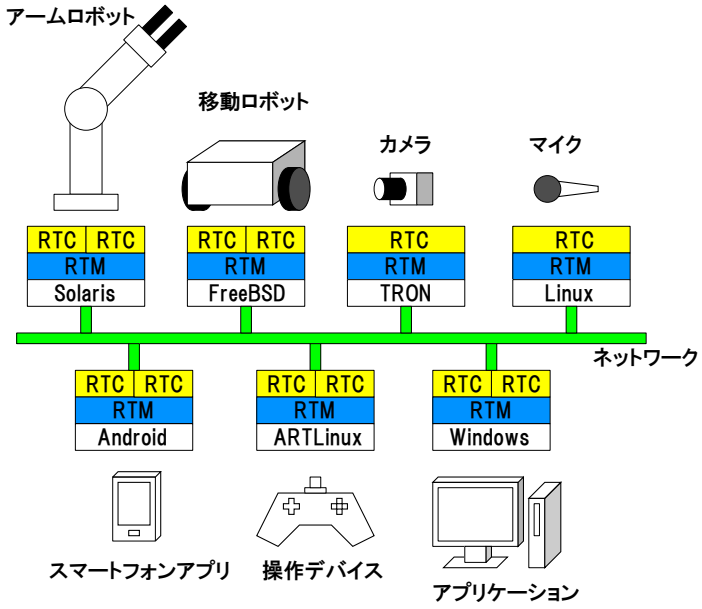


図 2.17 RTC ミドルウェアによる分散システム

### 2.8.1 組み込み用 RTM

マイコンに RTM を搭載することでマイコンが直接制御するセンサやデバイスを簡単にロボットシステムに取り込むことができる．マイコン用の RTM として RTno がある [12]．RTno はイーサネットを介した通信にも対応しているため，マイコンをネットワークに接続するだけで RTC として利用可能になる．これを利用してロボットを構成するパーツ（アーム，カメラ，移動台車等）ごとに RTM を搭載したマイコンを割り当て，それぞれのパーツをマイコンが直接制御すればネットワーク上のどこからでもパーツの制御や情報の取得が可能になる．ロボットを構成するパーツはネットワークに接続するだけで簡単に情報のやり取りができるようになる．

# 第 3 章

# 提案するロボットハードウェア

本章ではロボットを構成するハードウェアについて述べる．ロボットシステムについては 4 章でさらに詳しく述べる．

## 3.1　ロボットの基本構成

家庭用ヒューマノイドロボットのプラットフォームには以下のような条件が望まれる [13].

- できるだけ人間に近いサイズ・形態・自由度であること: 人間のために整備されているインフラストラクチャーをそのまま利用でき，また機能や動作が直感的に分かりやすい
- 軽量であること: 人間一人で取り扱うのに困難を伴うロボットでは，人間と共存するうえで危険が大きい
- 安全であること: 予期せぬ事故が起きた場合を考え，アクチュエータはあまり高出力でないほうがよく，機械的にコンプライアンスであることが望ましい
- 安価であること: ヒューマノイドロボットが普及するには，その価格が最低でも 100 万円以下であることが必要ではないかと考える

以上の条件を満たすように仕様を決定し，ロボット設計を行った．特に使用する部品に関しては可能な限り市販されているものや規格品を使いコストを抑え，保守性

に優れたシンプルな構造を目指した．表 3.1 に本研究において開発したロボットの仕様を示す．

表 3.1 ロボットの仕様

| | |
|---|---|
| 身長 | 170[cm] |
| 重量 | 35[kg] |
| アーム自由度 | 7 |
| アームペイロード | 2[kg] |
| 総関節数 | 17 |
| 搭載 PC 数 | 1 台 |

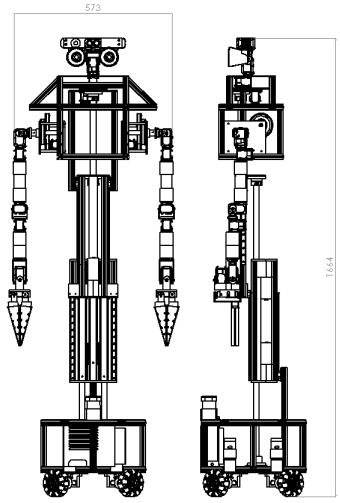


図 3.1 ロボット（最大身長時）

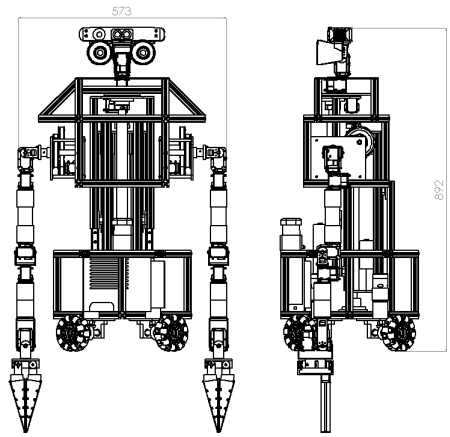


図 3.2 ロボット（最小身長時）

図 3.1，3.2 に，仕様に基づき設計を行ったロボットのモデルを示す．ロボットの身体パーツをそれぞれ，台車，伸縮機構，上半身（アーム）ごとにユニットとして設計し，それらを組み合わせることでロボットの身体を構築した．ロボットのフレームには溝付のアルミフレームを使用する．これによりロボットの身体の一部を変更したり，用途に応じて各部位の配置を変更したりすることが可能となる．基本となるロボットは，車輪による移動台車と双腕を備えた構成となっている．台車と上半身の接合部に伸縮機構のユニットを設けることで，ロボットは自由に身長を

変えることができる．これにより物体探索やマニピュレーション範囲の制約が改善され，様々な環境において，より柔軟に動作することが可能となる．また，このロボットではロボットに必要な演算処理は外部で行うクラウドロボティクスを想定し，内蔵するコンピュータの数を最小限とした．コンピュータ数を低減することで必要なバッテリーの容量を抑えることができ，軽量化・低コスト化につながる．

### 3.1.1　全方位移動台車の設計

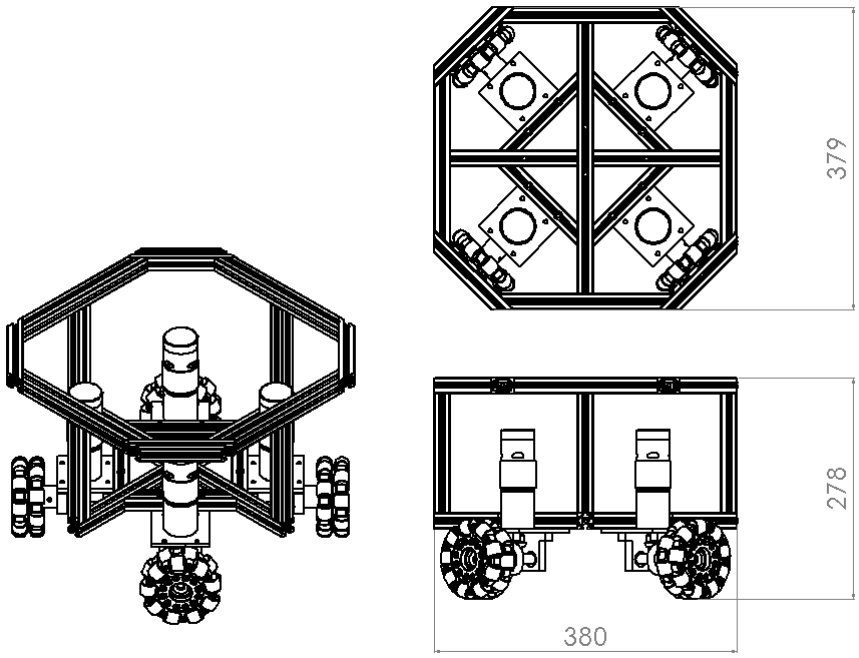


図 3.3 全方位移動台車

狭い空間で効率良く移動するために，オムニホイール [22] を使用した全方位移動台車を設計した．台車を構成する要素は大きく分けて 3 つあり，それぞれフレーム，モータ及び車輪である．

モータはモータ [23] とコントローラ [24] がセットになった市販のものを使用する．車輪はその直径と耐荷重を考えて選定した．車輪の直径が大きくなるほど段差などを乗り越えやすくなるが，車輪を回転させるのに必要なトルクが大きくなる．

本研究では家庭内で使用することを想定し，1[cm] 程度の段差を乗り越えられれば良いと考えている．フレームに関しては強度と重量を考えてアルミフレームで構成する．

図 3.3 に，設計した台車の外観を示す．台車のサイズは 380×380[mm] で，高さは約 280[mm] である．また，車輪の直径は 100[mm]，可搬重量は約 40[kg]，最高移動速度は 0.3[m/s] である．ギヤードモータからの出力は傘歯車を用いて減速させると共に，回転軸を直交させた．これにより，車輪をフレーム内に収めることができ台車全体を小型化することができる．内部のスペースには制御用の PC やバッテリー，イーサネットコンバータなどを配置する．

### 3.1.2　7 自由度アームの設計

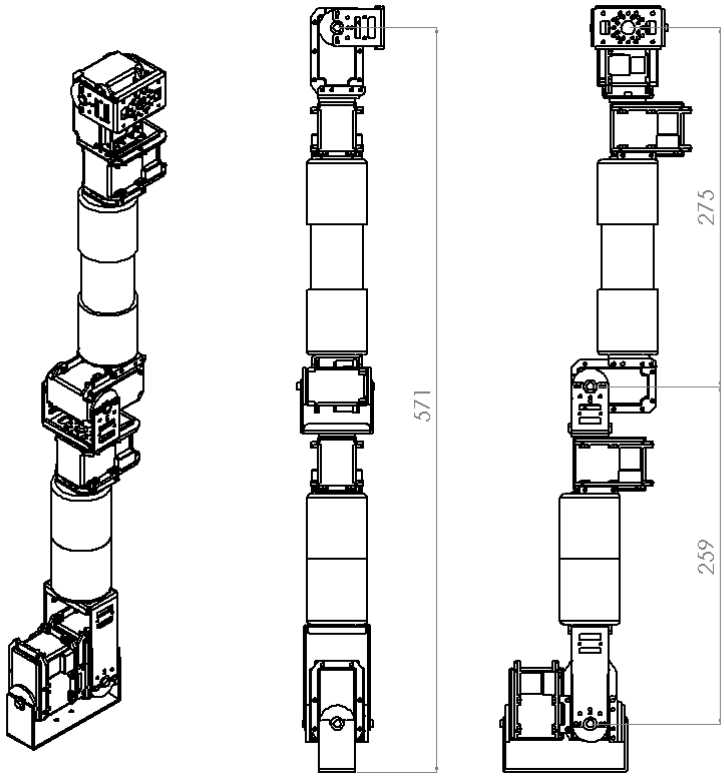


図 3.4 7 自由度アーム

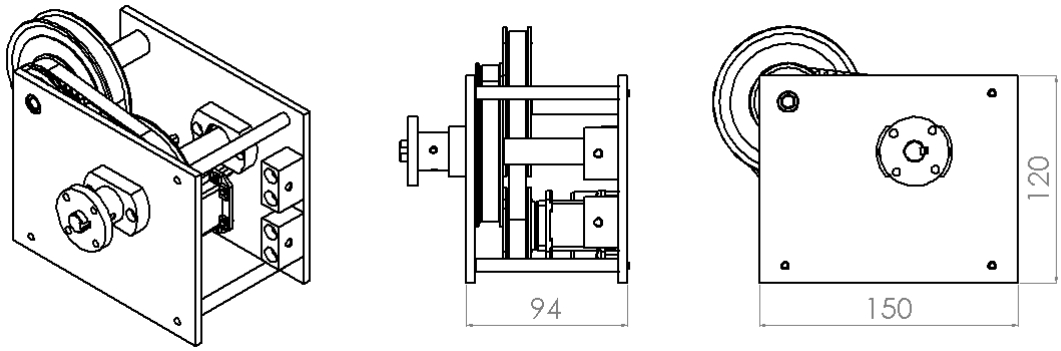


図 3.5 アームのための減速機

アームの自由度は，より人間の関節構造に近づけるため 7 自由度構成とした．アクチュエータの選定は，アームの制御精度に直接影響する．一般的なアームのアクチュエータは，バックラッシュの少ない減速機構にモータを組み合わせたものや，ステッピングモータを使用したものが多い [5]．しかし，これらを個別に選定して設計するのはコストがかかる上，メンテナンス性も悪くなる．よって，アームに使用するアクチュエータはすべて市販のサーボモータとした [19]．

図 3.4 に製作したアームを示す．アームの全長は約 570[mm]，ペイロードは約 2[kg] を想定している．アームに使用するアクチュエータは，コンプライアンス制御が可能な Dynamixel 社の EX-106 と RX-64 である．アームの長さと予想される重さ，必要となるペイロードをもとに各関節で必要となるトルクを算出し，それに適した出力を持つサーボを使用する．最も大きなトルクを必要とするのは肩の付け根の部分である．サーボの出力以上のトルクが必要となるため，図 3.5 のような減速機を用いることでより大きなトルクが出せるようにした．減速機には通常の歯車ではなくタイミングプーリーとベルトを用いることでバックラッシュを抑え，制御精度の低下を防いでいる．

#### 3.1.2.1　上半身の構成

本研究では双腕を有するロボットとするため，前述したロボットアームを 2 つ搭載する．図 3.6 に示すようなアルミフレームを用いた上半身フレームを構成し，そこに 2 本のアームを取り付ける．アームの取り付けはネジ 2 本で可能であるため容

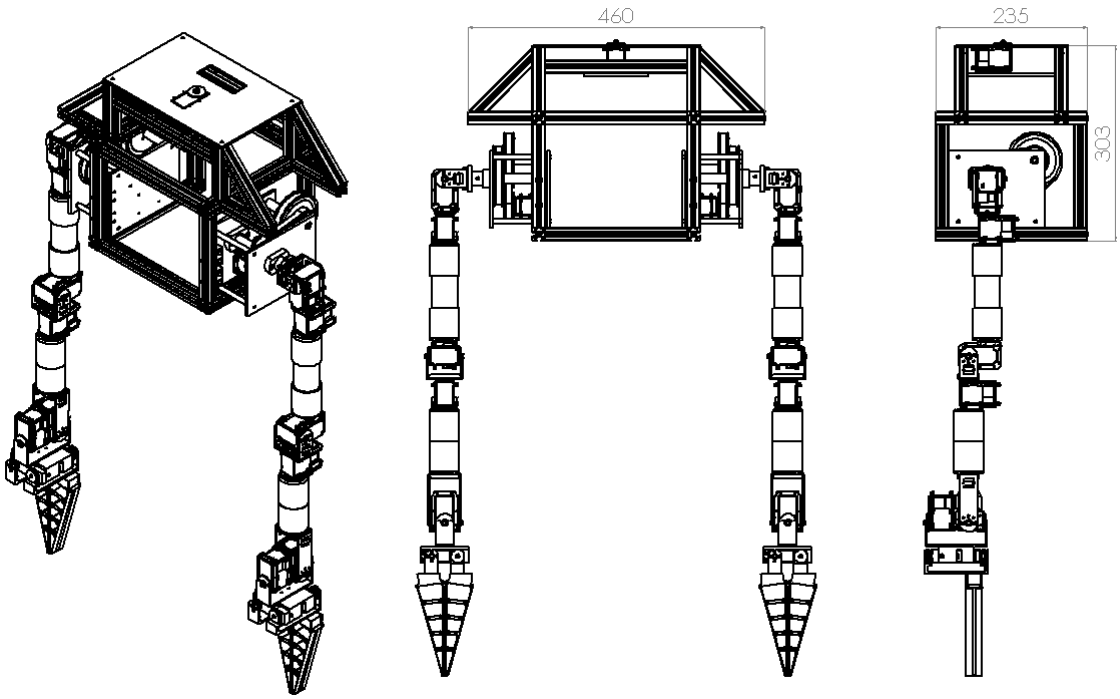


図 3.6 上半身の構成

易に取り外しが可能である．また上半身を構成するフレームを組み替えれば様々なサイズのロボットを構成することが可能である．

### 3.1.3 伸縮機構の設計

上半身を直動移動させることを考えた場合，いくつかの方法がある．本研究では，ボールネジ駆動式のリニアアクチュエータ [20] を複数用いることによりこれを実現する．リニアアクチュエータを用いることのメリットとしては，容易に高出力が得られることと，バックドライバビリティがないことである．バックドライバビリティがないということはモータに電力が供給されていなくてもその位置を保持しておくことが可能であり，重量のあるものを長時間保持することに適している．また市販品は容易に入手でき保守性にも優れている．リニアアクチュエータ単体では可動領域が狭いため，より広い可動領域を得るためには複数のリニアアクチュエータを使用する必要がある．

図 3.7 に可動領域を広くするための機構を示す．300[mm] のストロークと，58[kg]

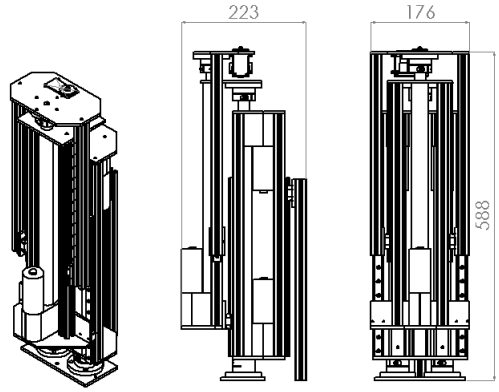


図 3.7 伸縮機構（最小時）

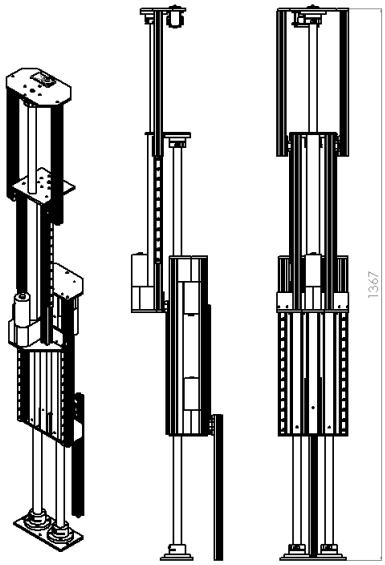


図 3.8 伸縮機構（最大時）

の可搬重量を有する図 3.8 に示すように 4 つのリニアアクチュエータを互い違いに配置することで，可動領域を広くとれるようにした．アルミフレームで構成され，要所にはリニアガイドを設け伸縮時の安定性を高めている．各リニアアクチュエータは任意の長さに制御可能であり，機構の全長は 590[mm]〜1300[mm] まで自由に変更することが可能である．

また，機構先端にはアームに用いられているサーボと同様のものが取り付けられており．上半身を載せることで腰軸として機能する．

### 3.1.4　ロボットハンドの設計

ハンドの形状には用途によって様々な形をしたものが考えられる．例えば，複雑な物体操作を必要とする場合はより人間の手に近い自由度を持ったハンドが必要である．本研究においてはロボットがハンドを使って行う物体操作は簡単なものを想定しており，そこまで多くの自由度を必要としていない．よって本研究では図 3.9 に示すような 2 自由度のハンドを設計した．市販のサーボ [19] を用いて構成されており，2 本のグリッパによって主に物体の把持操作を行う．ハンドのアームへの取り付けは容易に行えるようになっており，用途に応じて別のハンドに交換するといったことも可能である．

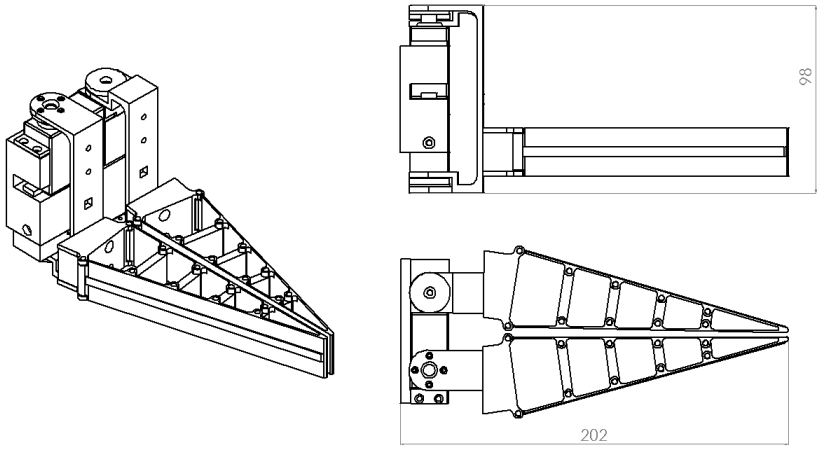


図 3.9 ロボットハンド

### 3.1.5　ロボット頭部の設計

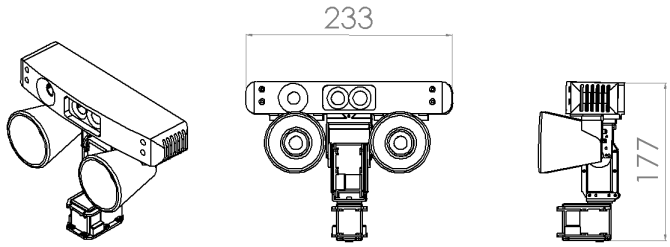


図 3.10 ロボットの頭部

ロボットの頭部には視覚情報を取得するためのセンサーを取り付ける．本研究において視覚センサには Kinect を使用した．Kinect センサを搭載することで安価に，3 次元情報，2 次元画像情報，音声情報を取得することができるだけでなく，豊富に用意された開発用の SDK やライブラリを用いることで簡単に物体の認識や人物の認識といったことが可能となる．

図 3.10 に示すように，Kinect を内蔵するカバーには別途 CCD カメラなど，別の視覚センサを取り付けられるように改良を施している．また，センサの台座にはサーボモータによる 2 自由度のパン・チルト機能を搭載し，センサの画角の狭さを補っている．

### 3.1.6　製作したロボットの全体構成

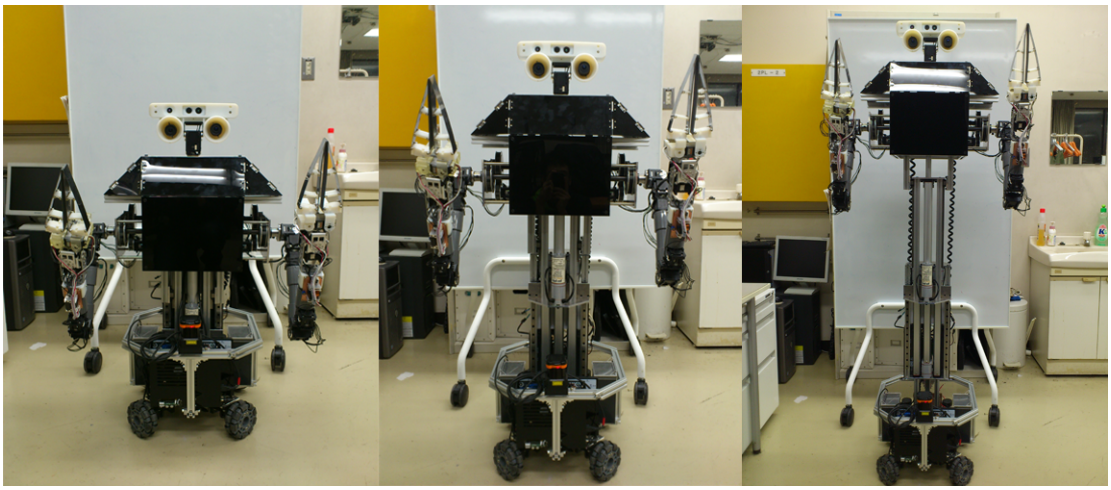

図 3.11 製作したロボット

設計に基づき製作を行ったロボットの外観を図 3.11 に示す．伸縮機構を取り入れたことで身長を自由に変えることができている．ロボットを制御するための PC やバッテリーなどは台車内部に搭載されており，自律での行動が可能となっている．

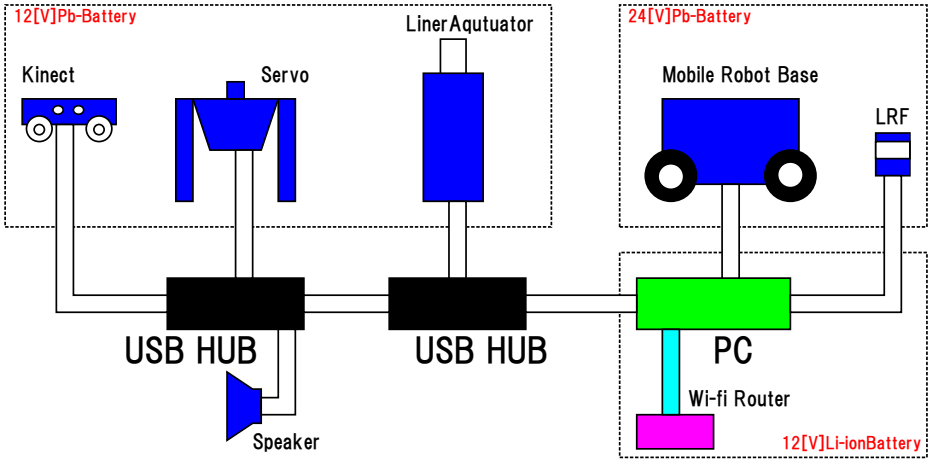


図 3.12 ハードウェア構成

電源を含めたロボットのハードウェア構成を図 3.12 に示す．搭載された一台の PC から USB ハブを経由して各種ハードウェアと PC は接続されている．また電源に関してはハードごとに供給する電圧が異なるため 2 系統に分けている．Kinect

やアーム用のサーボ，伸縮機構のリニアアクチュエータには 12[V] の 2 個の小型鉛バッテリーをを並列接続して使用している．台車の電源には 24[V] 必要となるため 12[V] の鉛バッテリー 2 個を直列接続して使用している．PC の電源には小型の Li-ion バッテリーを使用している．PC の操作はネットワークを利用したリモートアクセスで行うため，PC には無線のイーサネットコンバータ（ルーター）が接続されている．製作したロボットに搭載されている PC のスペックを表に示す．

表 3.2 PC スペック

| | |
|---|---|
| M/B | Intel DH77DF(mini-ITX) |
| CPU | Corei5 2500T 2.3Ghz |
| Mem | DDR3-1333 4GB |
| HDD | mSATA SSD128GB |
| OS | Windows7 Pro 64bit |

# 第 4 章

# ソフトウェアシステム

本章では，提案手法にて製作したロボットに実装するシステムの詳細について記す．

## 4.1　パーツ組み換えを実現するシステム

本論文の目的は，低コストなロボットを開発することだけなく，ロボットの各パーツが容易に組み換え可能となるようにシステムを設計することにもある．これには，ロボットが働くべき多様な環境に合わせてパーツを選択し組み合わせることで，考え得る多くの環境を網羅するような複雑なロボットのハードウェア構成を避け，ハードウェアのコストを抑える狙いがある．また，ユーザはロボットに要求する性能に合わせて，パーツを注文し取り付けることができ，コストと性能のトレードオフをより細かく捉えることができる．本節では，前章で設計した各ハードウェアパーツに搭載するソフトウェアシステムを検討することで，このようにフレキシブルなロボットの構成を可能とすることを考える．

### 4.1.1　システムの概要

提案するシステムは，ハードウェアの記述と動作の記述および動作アルゴリズムなどの実装を含むソフトウェアシステムから構成される．それぞれのロボットパーツにはマイコンをベースとした RT コンポーネントが搭載されており，各コンポーネントはロボットパーツ記述言語で記述されたハードウェア情報をファイルから読み込み，ポートを通じてロボット全体を管理するコンポーネントに通信することが

できる仕組みを実現する．また，各センサにも同様の仕組みが搭載されており，センサの組み換えにも対応する．さらに各パーツ RTC には、実際に効果器を動作させるための基本動作プログラムが実装されており，パーツ単位の動作が関数を呼び出すことによって実行される．こうした関数に関する情報も，ロボットパーツ記述言語によってハードウェア情報として記述されている．

動作記述言語は，ロボットの基本的な動作を汎用的に記述するためのフォーマットである．アクションプログラムは，パーツ・センサ単位の動作関数を組み合わせたもので動作記述言語によって生成される．タスクプログラムは，アクションを組み合わせたロボットのタスク全体の制御をつかさどるプログラムである．ロボットパーツ記述言語や動作記述言語によって記述される内容はメタな記述であり，アクションやタスクプログラムは，実際にロボットを動作させるためのハードウェアに依存したプログラムとなっている．

### 4.1.2 ロボットパーツ記述言語 (RPDL)

ロボットパーツ記述言語 (Robot Parts Description Language:RPDL) としては，ROS でサポートされている Unified Robot Description Format (URDF) や，文献 [6] で提案されている SRDL と同様のアイディアを用いる．つまり構成要素や関節，座標系，重さ，形状，衝突判定のためのモデルなどハードウェアとしての情報を XML などのファイル形式で保持するものである．ただし本論文で提案するものは，パーツ毎にこれを記述し，パーツ内のマイコンに内蔵させ，ミドルウェアを介して通信できるようになっている点がポイントである．また，各パーツで独立した動作や情報の取得関数もパーツ自体に内蔵されているため，この関数に関する情報もハードウェア記述言語によって記述されている．この記述には，関数名と関数に渡すべき引数，及びその動作が含まれており，自然言語ベースで記述されている．このパーツを利用する側では，オントロジー（ここではワードネットを用いる）を用いて記述を標準形式に変換し，変換された機能記述と引数のマッチングにより利用可能な関数を探索する．実際に関数を利用する場合は，サービスポートを通してアクセスる．アームの RPDL に関する XML ファイルの例を，図 4.1 に示す．

センサについても同様に記述を行い，各センサに内蔵させることとする．

```
<parts>
   <name> arm </name>
   <config> arm.conf </config>
   <function>
        <name> GetConfig </name>
        <description> obtain configuration of this part
        </description>
        <args> none </args>
        <return> struct armconf </return>
   </function>
   <function>
        <name> MovePointTo </name>
        <description> move the endpoint to a specified position
        </description>
        <args> struct 3Dpoint, specified 3D position </args>
        <return> bool, success or false </return>
   </function>
   <function>
        <name> CheckReachability </name>
        <description> check reachability of the endpoint
         to a specified position</description>
        <args> struct 3Dpoint, specified 3D position </args>
        <return> bool, reachable or not </return>
   </function>
</parts>
```

図 4.1 アームに対する RPDL の例 (紙面の都合で一部のみ表示)

### 4.1.3 動作記述言語 (ADL)

動作記述言語 (Action Description Language:ADL) は，ロボットがタスクを実行する際に基本となる動作単位のプログラムをハードウェアに合わせて生成するための汎用的な記述を行うためのフォーマットであり，前述の RPDL と同様の構造をもつ．つまり，ADL はアクションプログラムに関するメタな記述であり，接続されている RPDL を参照することで，実際のアルゴリズムの実装を組み合わせてアクションプログラムを生成するために利用される．ADL においても自然言語を用いた動作時系列の記述がポイントとなっている．動作時系列の記述は，形態素解析され，ワードネットを利用した標準形式への変換を介して，必要となるパーツ・センサとそのパーツ・センサに搭載されている関数の検索を行い，実際にそれらの関数を適切なタイミングで呼び出すアクションプログラムを生成する．図 4.2 に頷き動作生成用の ADL の例を示す.

### 4.1.4 アクションプログラム

アクションプログラムは，上述のように ADL でメタ記述されたものを基に自動生成される実行ファイル群である．パーツの変更や身体構造が変化した場合においても RPDL を参照することで，実際のアルゴリズムと組み合わせて適切なアクションが生成されるため，身体構造の変化がロボットシステムに与える影響を軽減することが可能となる.

### 4.1.5 タスクプログラム

タスクはアクションプログラムを並べることによって実現されており，これは事前に設計者によって定義されているとする．ただし，設計の際にはアクションプログラムの ADL による記述が参照されることになる．これは，実際に使われるハードウェアは設計の際には特定されていないためである．現状は，設計者が手作業でファイルを編集する必要があるが，将来的には直観的にわかり易い GUI ベースの設計ツールなどを整備する必要がある．また，シミュレーション環境を利用して，タスクのシミュレーションを行い，どのようなロボットの身体構造が有用であるかを自動的に判定することも可能であると考えられる．さらには，ロボットが利用さ

```
<action>
   <name> Nod Action</name>
   <config> action.conf </config>
   <actionsequence>
     func1,func2,func3,func2
   </actionsequence>
   <description>nod</description>
   <args> none </args> <return> bool, success or false </return>
   <func1>
     <name> Pantilt  down</name>
     <description> move the pantilt  to down</description>
     <args> none</args><return> bool, success or false </return>
   </func1>
   <func2>
     <name> Wait</name>
     <description> wait until pantilt is moving</description>
     <args> none</args><return> bool, reachable or not </return>
   </func2>
   <func3>
     <name>Pantilt  up</name>
     <description> move the pantilt  to up</description>
     <args> none</args><return> bool, reachable or not </return>
   </func3>
</action>
```

図 4.2 頷き動作生成のための ADL の例 (紙面の都合で一部のみ表示)

れる環境の記述に関する標準化を行うことで，例えば一般的なリビングであるロボットが設計したタスクを実行できる確率などを算出可能であると考えている．

## 4.2 RT コンポーネントによるシステムの実装

ロボットのソフトウェアは RT ミドルウェアによる RT コンポーネントとして実装する (RTC)．図 4.3 に RTC によるロボットシステムを示す．RTC により，ソフトウェアの機能をモジュール単位に分割することでソフトウェアの使い回しや組み換えが容易に行える．表 4.1 に本研究において作成した，ロボットを動作させるために最低限必要となる RTC と主な機能の一覧をまとめて示す．

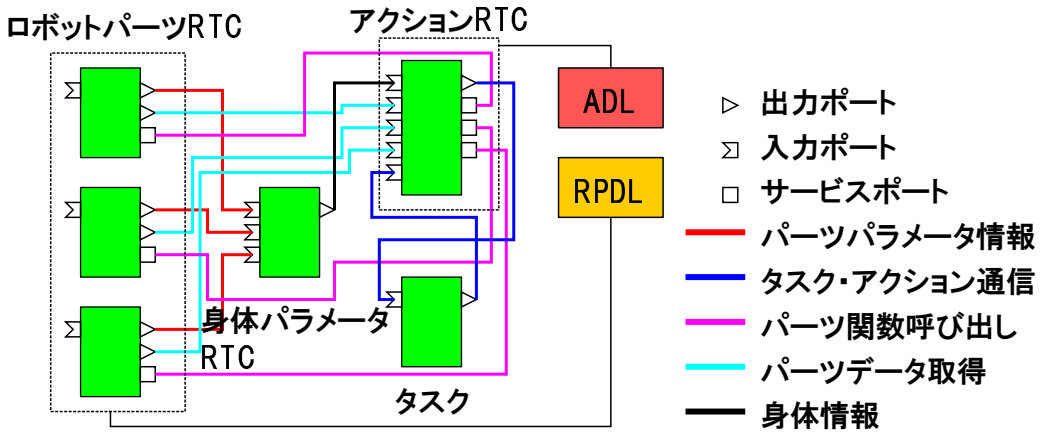


図 4.3 RTC によるロボットシステム

また組み込み用の RTM として RTno[12] を使用することで，パーツやセンサをマイコンで制御し，RTC としてロボットシステムに組み込む．図 4.4 に台車に搭載したマイコンを示す．マイコンには mbed を使用しており，この中に台車制御用のプログラム及び RPDL が記述されている．Eathernet ポートが付属しておりネットワークに接続することで RTC として利用することが可能となる．

### 4.2.1 アクション RTC

ロボットでタスクを行うには単純な動作や行動をひとまとめにアクションとし，それらを順番に実行させることで複雑なタスクを実現する．

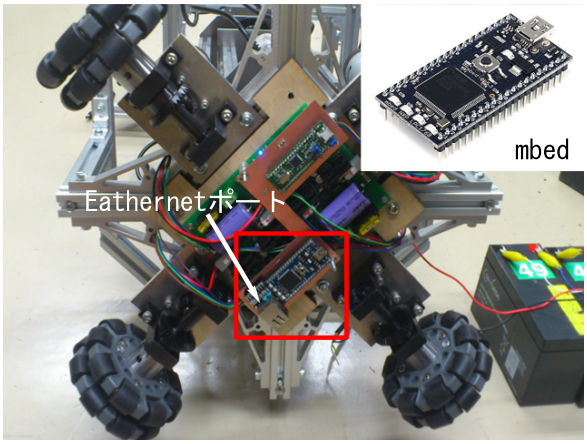


図 4.4 RTno による台車制御用 RTC の実装

表 4.1 主要 RTC 一覧

| RTC 名 | 機能 |
|---|---|
| RobotRCT | 台車の移動制御を行う RTC 内部では更に細分化された複数の RTC で構成される |
| ServoRTC | アームなどに使用されているサーボをコントロールする RTC |
| BodyCtrlRTC | ロボットの上半パーツを制御する RTC，ServoRTC とペアで使用する |
| LinerCtrlRTC | 伸縮機構を制御する RTC |
| KinectCtrlRTC | Kinect を制御する RTC，画像データと距離画像データの出力を行う |
| RobotParamRTC | ロボットのパラメータやハードウェアの状態を管理する RTC |
| InverseKinematicsRTC | アームの逆運動学を解くための RTC |
| CoordinateTransRTC | 各種座標変換を行うための RTC，RobotParam のパラメータを参照して変換を行う |
| ObjectRecogRTC | KinectRTC の情報から，平面検出を行いテクスチャ情報を用いて物体認識を行う |

本研究ではモバイルマニピュレーションの実現を目標として以下のアクションを実装する．

- GrabActionRTC
- SearchActionRTC
- LookActionActionRTC

#### 4.2.1.1 GrabActionRTC

GrabActionRTC は指定座標の把持動作を行うためのアクションである．入力として把持位置の 3 次元座標（ロボット座標系）を必要とする．把持動作のための 3 次元座標がタスク与えられると，RobotParamRTC よりアームのコンフィグレーション情報を取得する．これらの情報と把持位置座標を統合し InverseKinematicsRTC に情報を送り IK を求める．IK の解ある場合，InverseKinematicsRTC からはアームの関節角情報が出力が行われる．これを BodyCtrlRTC へ送ることでアームが動作することになる．その後は同じく BodyCtrlRTC へハンドの開閉命令を行い．アームを元の位置へ戻し動作は終了となる．動作結果を出力ポートから出力することで，アクションを制御する上位のタスクではアクションの成功，失敗を判断することができる．

#### 4.2.1.2 SearchActionRTC

SerchActionRTC は指定した物体の探索を行うためのアクションである．入力としては探索物体の ID，探索時のロボットの姿勢（チルトと腰の角度，身長）とアクションを終了するまでのタイムアウトなどの情報が必要となる．これらの情報が与えられると　，まずは BodyCtrlRTC へ姿勢情報が送られ，指定された探索姿勢をとる．次に物体の認識を行うため ObjectRecogRTC へ物体認識開始の命令が送られる．ObjectRecogRTC では検出された物体の ID と座標（カメラ座標系）が順次出力されるので，ActionRTC では指定された物体がタスクから指示されたものかどうかを判断する．もし指定された物体の検出が成功した場合，物体の座標変換を行う．座標変換のために CoordinateTransRTC にカメラ座標を入力すると

CoordinateTransRTC は RobotParamRTC より現在のコンフィグレーションを参照して与えられた座標をロボット座標系へと変換し，結果をポートから出力する．SearchActionRTC はこの変換された座標を受け取り，結果をタスク側に送信する．

#### 4.2.1.3 LookActionRTC

LookActionRTC は GrabActionRTC や SearchActionRTC において使われる単純な動作のためのアクションである．動作としてはパン・チルトを使い指定座標を見るというものである．入力として 3 次元座標（ロボット座標系）が与えられると，RobotParamRTC から現在のロボットのコンフィグレーションを参照し，指定座標を向くためのパン・チルト角度を計算する．計算された角度情報を BodyCtrlRTC へ送信することでパン・チルトを指定角度に動かす．こういった動作を SearchActionRTC や GrabActionRTC で使用することでロボットの動作をより自然にみせることができる．

# 第 5 章

# 実験

## 5.1 実験内容

製作したロボットで以下のような評価実験を行う．

- マニピュレーションタスクにおける物体の把持範囲
- モバイルマニピュレーションタスク
- 身体パーツを組み換えた場合の動作

### 5.1.1 マニピュレーションタスクにおける物体の把持範囲

物体探索アクション，物体把持アクションを用いて特定物体の把持を行うマニピュレーションタスクを作成する．作成したタスクの把持アクションにロボットの身長の可変がある場合とない場合とで把持領域の違いを調べる．

実験方法としては，図 5.1 に示すように 10[cm] 間隔で 15 個のマーカーが記された机をロボットから一定の距離に設置し，マーカー上に置かれる物体の把持を行う．

実験環境を図 5.2 に示す．アームの原点と机上の Z 軸方向の距離 D を-100[mm]〜500[mm] までロボットの伸縮機構により可変させ，それぞれの距離において 15 箇所に置かれた物体の把持を行う．把持アクションは身長の可変を用いる場合と用いない場合の 2 パターンをロボットのコンフィグレーションにより切り替える．探索アクションにおける物体探索時のパン・チルト角度はパン角度 0 度，チルト角度を

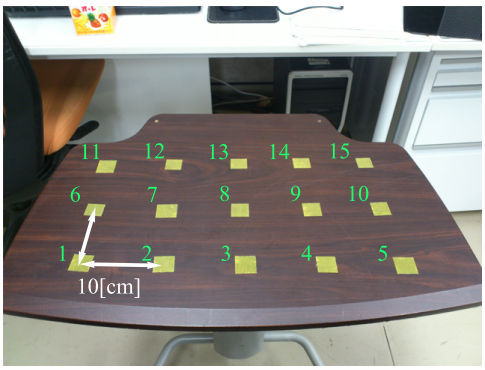


図 5.1 マニピュレーションタスクに使用する机

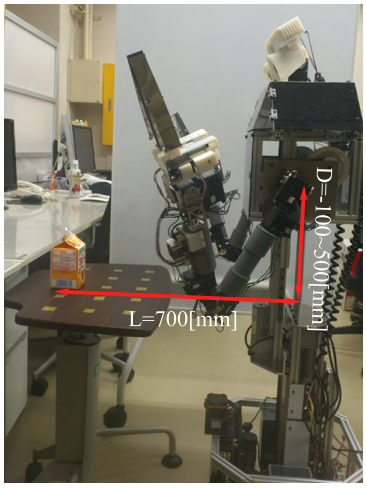


図 5.2 実験環境

30 度，腰の角度は 0 度一定，また，ロボットと机との X 方向の距離 L はロボットの中心から図 5.1 の 13 の点までを 700[mm] 一定とする．

実験結果を表 5.1，5.2 にまとめて示す．ただし,「把持に成功した場合」を”〇”,「IK が解けず失敗した場合」を”×”,「物体を途中で倒す，または腕が机と接触して失敗した場合」を”△”,「物体の検出ができなかった場合」を”-”としてある．また，D=-100[mm] および 500[mm] においては物体の検出ができなかったため結果からは省略してある．

実験結果より身長の可変がある把持アクションでは，物体の検出ができる範囲においては，ほぼ確実に物体を把持できていることが確認できる．一部で物体の把持に失敗しているが，これはハンドと物体の接触による失敗で，原因としては物体検出時の物体座標の誤差によるものが挙げられる．物体を正面以外から捉えた場合，図 5.3 に示すように物体の重心座標が左右にズレれることがあり，これにより把持位置がズレ，ハンドが物体と接触してしまったと考えられる．

身長の可変がない把持アクションにおいては物体検出が成功しても身体構造的に IK が解けずに失敗することがあった．また IK が解けた場合でも遠くに置かれ物体把持の場合は，机の縁にアームが接触してしまい把持に失敗することがあった．

把持領域の広さを考える場合，重要となるのが Z 方向の範囲の広さと，平面上の範囲の広さである．Z 方向の広さで重要なのはアーム原点と机との距離 D である．この範囲が広いほど，様々な高さに置かれた物体を把持可能であるといえる．

表 5.1 マニピュレーションタスク実験結果（身長の可変なし）

| 把持位置 | D=0[mm] | D=100[mm] | D=200[mm] | D=300[mm] | D=400[mm] |
|---|---|---|---|---|---|
| 1 | - | ○ | ○ | - | - |
| 2 | - | ○ | ○ | - | - |
| 3 | - | ○ | ○ | - | - |
| 4 | - | ○ | ○ | - | - |
| 5 | × | ○ | ○ | - | - |
| 6 | × | ○ | ○ | ○ | ○ |
| 7 | × | ○ | ○ | ○ | ○ |
| 8 | × | ○ | ○ | ○ | ○ |
| 9 | × | ○ | ○ | ○ | △ |
| 10 | × | ○ | ○ | ○ | ○ |
| 11 | × | △ | ○ | ○ | ○ |
| 12 | × | △ | ○ | ○ | ○ |
| 13 | △ | △ | ○ | ○ | ○ |
| 14 | × | △ | ○ | ○ | ○ |
| 15 | × | △ | △ | ○ | △ |

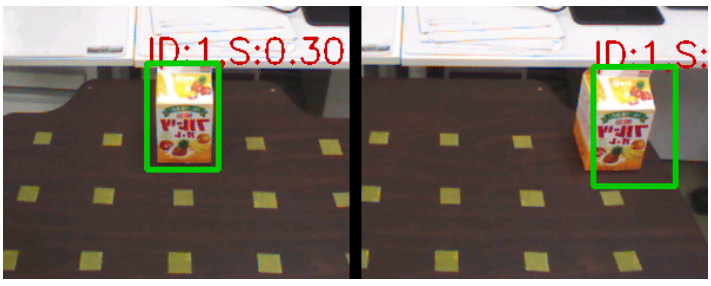


図 5.3 物体検出結果のズレ

表 5.2 マニピュレーションタスク実験結果（身長の可変あり）

| 把持位置 | D=0[mm] | D=100[mm] | D=200[mm] | D=300[mm] | D=400[mm] |
|---|---|---|---|---|---|
| 1 | - | ○ | ○ | - | - |
| 2 | - | ○ | ○ | - | - |
| 3 | - | ○ | ○ | - | - |
| 4 | - | ○ | ○ | - | - |
| 5 | ○ | ○ | ○ | - | - |
| 6 | ○ | ○ | ○ | ○ | ○ |
| 7 | ○ | ○ | ○ | ○ | ○ |
| 8 | ○ | ○ | ○ | ○ | ○ |
| 9 | ○ | ○ | ○ | ○ | ○ |
| 10 | ○ | ○ | ○ | ○ | ○ |
| 11 | ○ | ○ | ○ | ○ | ○ |
| 12 | ○ | ○ | ○ | ○ | ○ |
| 13 | ○ | ○ | ○ | ○ | ○ |
| 14 | ○ | ○ | ○ | ○ | ○ |
| 15 | ○ | △ | ○ | ○ | △ |

また，平面上の広さはロボットの移動によっても決まるが，移動を考えない場合は同一平面でより多くの場所を把持できることが重要である．

以上のことを考慮して両結果を比較する．把持位置 6〜15 のすべてが把持可能な時の D に注目すると，身長可変がない場合の D の範囲は 200[mm]（200〜400[mm]）である．一方，身長可変を用いたアクションではこの範囲は 400[mm]（0〜400[mm]）であり，およそ倍の範囲をカバーできていることがわかる．

### 5.1.2 モバイルマニピュレーションタスク

室内の環境地図を作成し，移動を含めたマニピュレーションタスクを作成する．作成したタスクでの物体の把持を行い成功率について考察する．

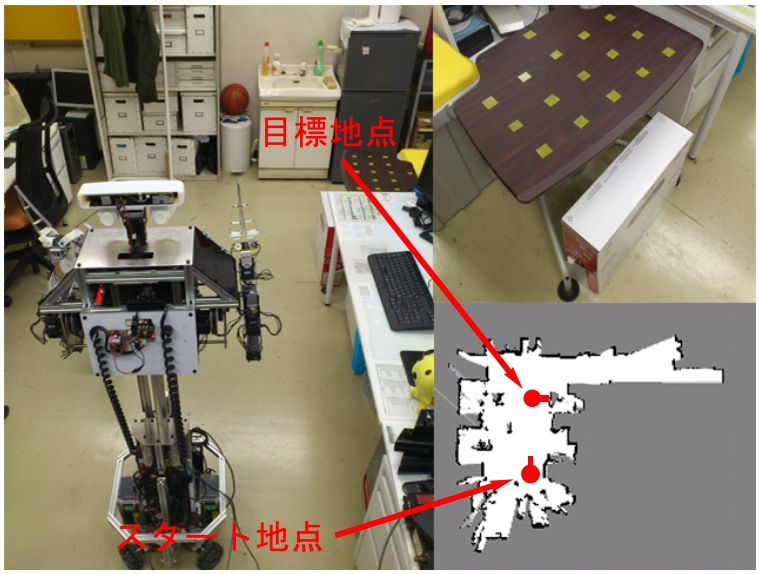


図 5.4 モバイルマニピュレーションタスク環境

実験 5.1.1 にて作成したタスクにロボットの移動を追加する．把持アクションには身長の可変を有効にしたものを使用する．物体を置く場所は実験 5.1.1 において把持が成功している範囲内でランダムに高さ，場所を変えて置くものとする．図 5.4 に実験環境と作成した環境マップを示す．タスクを 30 回実行し，その結果を表 5.3 にまとめて示す．また実験の様子を図 5.5 に示す．

実験結果より約 9 割近くが把持成功であり，移動を含めたマニピュレーションタスクにおいて高い把持率が実現できているといえる．把持失敗の主な原因としては

図 5.5 モバイルマニピュレーションタスク

表 5.3 モバイルマニピュレーションタスクの結果

| 成功回数 | ハンドの接触による失敗 [回] | 物体検出による失敗 [回] | IK による失敗 [回] |
|---|---|---|---|
| 25 | 2 | 1 | 2 |

移動誤差によって物体検出が不可能な位置に，または IK が解けない範囲にロボットが移動してしまったことが挙げられる．

### 5.1.3 身体パーツを組み換えた場合の動作

実験 5.1.1 と同様のタスクを用いて身体パーツを組み替えた場合の動作実験を行う．実験 5.1.1 においてロボットはアームの原点から机上面の Z 軸方向の距離が 0〜400[mm] 以内で，ロボットの中心から 700[mm] 以内に物体の検出がであれば把持が可能であることがわかった．ロボットは最小時でもアームの原点と床との距離は 580[mm] であるため，実験 5.1.1 のアクションやタスクでそのまま床の上に置かれた物体を把持することはできない．そこで床の上の置かれた物体を把持するため，ロボットの身体パーツの組み換えを行う．

図 5.6 に示すように上半身のアルミフレームを一部変更し，左アームを下方へ設置する．さらに，このパーツ変更を適応するためロボットのパラメータ管理でアーム取り付け位置の値を変更する．パラメータの設定後，実験 5.1.1 のタスクを実行し動作の確認を行う．

実験の様子を図 5.7 に示す．

図 5.7 より，パーツ組み換え前は把持不可能であった床の上の物体の把持が可能

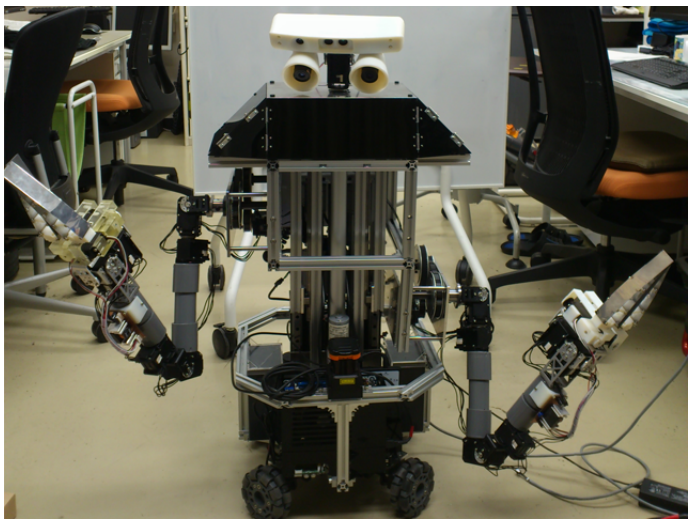

図 5.6 パーツ組み換え

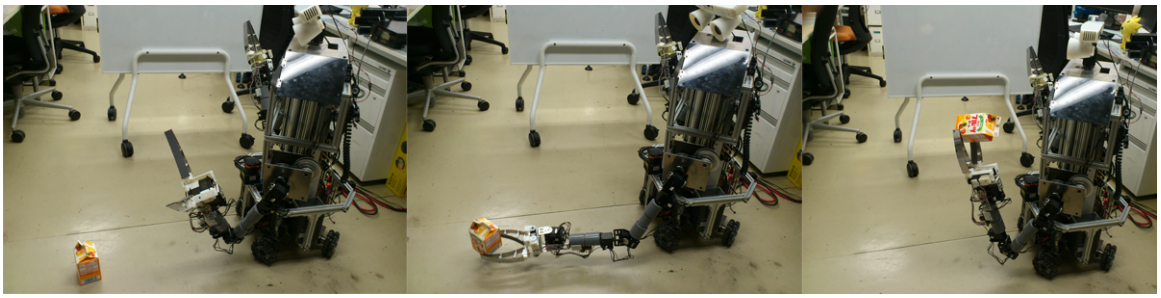

図 5.7 パーツ組み換え時のマニピュレーションタスク

になっていることが確認できる．本実験においてロボットのシステムはすべて立ち上がった状態で行い，パーツの組み換えとパラメータ変更にかかった時間は 5 分程度である．プログラムの書き換えや調整を行うことなく床上の物体把持が可能となった．また，この状態で実験 5.1.1 と同様に机の上の物体把持を行わせたが正常に動作することが確認できた．

## 5.2 結果と考察

実験 5.1.1，5.1.2 より，製作したロボットにおける物体把持は，物体の 3 次元座標がきちんと取得でき，かつ IK の解ける範囲内に位置すればほぼ確実に物体の把持を行うことができるといえる．これらの実験においては物体の探索動作は単純な動作しか行わなかったが，より，確実に物体の把持を行うには複雑な処理を行う必要があると考える．例えば物体が検出できなかった場合には，台車を移動したり身長を可変させ違った場所から探索を行うといった処理が挙げられる．また物体を検出してもそれが IK の範囲外であるならば，IK が解ける位置まで台車を移動させるといった処理も必要である．

実験 5.6 においては実際にロボットのパーツを組み換えることで，組換前の状態では実現不可能だった動作が可能となったことを示した．システムが動いている状態でもパラメータを動的に変更可能であり，タスクの調整や書き換えを行うことなく動作させることができた．また，ロボットは身体パーツの組み合わせにより構築されるため，簡単に各パーツを付け換えることが確認できた．

ロボットの性能評価として本実験ではロボカップ@ホームリーグで基本となるモバイルマニピュレーションタスクを選択したが，結果として製作したロボットの基本性能は十分実用できる性能であると考える．より詳細な性能評価を行うためにはシステム面での機能の充実やロボカップ@ホームリーグの各種タスクの実装が必要となる．

# 第 6 章

# 結論

本研究では，身体パーツの組み換えを考慮したヒューマノイドロボットを低コストで実現することを提案した．市販されている部品や規格品を組み合わせることで安価にロボットを製作することができる．ロボットを製作する上でかかった費用をTable.6.1 にまとめる．ロボットの製作費だけで 100 万円近くかかっているが，冒頭に述べたように DiGORO の開発コスト（一千万円）と比較するとおよそ 10 分の 1 である．ロボットの性能面での評価としては概ね設計通りのスペックであり，家庭内タスクにおいて基本となるモバイルマニピュレーションを実現するには十分な性能を有していることが実験により示された．また，本研究ではロボットの身体パーツを分割して設計し，それらを組み合わせることでロボットを構築した．これによりロボットの身体構造に幅をもたせることができ，パーツの組み換えにより様々なロボットが実現可能となる．実験によりロボットの身体構造を変えることでタスクの内容を変更することなくロボットのマニピュレーション範囲を拡大できることを示した．今後の課題としてはロボットをより実用的に利用するための各種アクションの実装や機能の充実が望まれる．また現在のロボットシステムではパラメータの変更はユーザーがしなければならないが，こういった部分もロボットが自動で行えるようなシステムを構築していく必要がある．

表 6.1 ロボット製作費

| 分類 | 費用 [円] |
|---|---|
| 上半身 | 570,000 |
| 移動台車 | 160,000 |
| 伸縮機構 | 130,000 |
| バッテリー | 65,000 |
| PC | 40,000 |
| センサ類 | 350,000 |
| 電装部品 | 45,000 |
| 合計 | 1,360,000 |

# 謝辞

本研究において，的確な助言や多大なるご指導を頂ました長井隆行准教授に最大限の感謝を心から述べたいと思います．研究の方向性などで悩んでいた際には，お忙しい中一緒になって考えて頂きありがとうございました．また，プログラム開発におきまして多大なるご支援を頂いた中村友昭先輩，Muhammad Attamimi 先輩には心よりお礼申し上げます．

# 参考文献


[1] 菅野ほか, “人間共存ロボットTWENDY-ONEのデザイン”, ロボティクス・メカトロニクス講演会講演概要集 2008, pp.2A1-D23(1)-2A1-D23(2), 2008

[2] U.Reiser *et al.*,“Care-O-bot®3 - Creating a product vision for service robot applications by integrating design and technology”, IROS2009, pp.1992-1998, 2009

[3] B.Pitzer *et al.*,“Making robots cheaper, more capable, and safer”, IROS2011:The PR2 Workshop

[4] J.Stuckler *et al.*,“Dynamaid, an Anthropomorphic Robot for Research on Domestic Service Application”, ECMR2009, pp.87-92, 2009

[5] M.Quigley *et al.*,“A Low-cost Compliant 7-DOF Robotic Manipulator”, ICRA2011, pp.6051-6058, 2011

[6] L.Kunze, T.Roehm, M.Beetz, “Towards Semantic Robot Description Languages”, ICRA2011, pp.5589-5595, 2011

[7] S.AMAGAI *et al.*, “Control of Omni-Directional Mobile Platform with Four Driving Wheels Using Torque Redundancy”, International Conference on Intelligent Robots and Systems, Sep.2008


—

[8] 中本ほか, “レーザーレンジファインダ搭載移動ロボットによる動的環境の 3 次元地図生成”, 電子情報通信学会技術研究報告. WIT, 福祉情報工学 106(144), pp25-30, 2006

[9] 米田 完, 大隅 久, 坪内 孝司 “ ここが知りたいロボット創造設計 ” 講談社サイエンティフィク,pp.68-79, 2005

[10] S.Kagami *et al.*,“Plane segment finder: algorithm, implementation and applications”, ICRA2001, pp.2120-2125, 2001

[11] 村永ほか, “ロボカップ@ホームのオープンプラットフォーム化”, ロボティクス・メカトロニクス講演会講演概要集 2011, pp.2P1-K16(1)-2P1-K16(3), 2011

[12] 菅佑樹, “Arduino を使って OpenRTM-aist 対応組込みシステムを簡単に作るためのライブラリ「RTno」の開発”, ロボティクス・メカトロニクス講演会 2011,2P1-K12, 2011

[13] 白田ほか, “ヒューマノイドロボット才華 4 の設計”, 計測自動制御学会東北支部第 206 回研究集会資料, pp.206-5, 2002

[14] 古田ほか, “ヒューマノイドロボット才華 3 のアーム開発”, 計測自動制御学会東北支部第 192 回研究集会資料, pp.192-9, 2000

[15] K.A.Wyrobek *et al.*, “Towards a personal robotics development platform: Rationale and design of an intrinsically safe personal robot”,ICRA2008, pp.2165-2170, 2008

[16] 金子ほか, “ヒューマノイドロボット HRP-3 の開発”, 日本ロボット学会誌,Vol.26(No.6) pp.658-666, 2008

[17] 大西ほか, “人体を外側から扱うロボットの設計”, 日本ロボット学会誌,Vol.26(No.3) pp.1-4, 2008

[18] RoboCup@HomeRuleBook2012 “`http://purl.org/holz/2012\_rulebook.pdf`”, 2012

[19] DynamixelServo “http://www.hizook.com/files/users/3/EX-106_Robotis_Dynamixel_Servo_UserGuide.pdf”, 2012

[20] 12 Stroke 150 lbs Force Linear Actuator w / Potentiometer Feedback “http://www.robotshop.com/content/PDF/manual-fa-po-150-12-12.pdf”, 2012

[21] Firgelli Technologies Linear Actuator Control Board “http://www.firgelli.com/Uploads/LAC_Datasheet.pdf”, 2012

[22] 100mm Omnidirectional Wheel “http://www.robotshop.com/content/PDF/layout-14049.pdf”, 2012

[23] Devantech24V 49:1GearMotor/Encoder “http://www.robot-electronics.co.uk/htm/emg49.htm”,2012

[24] MD49-Dual24Volt5AmpH-BridgeMotorDrive “http://www.robotshop.com/content/PDF/md49-documentation.pdf”,2012

## 発表実績・予定

(1) 丸山恭平, 中村友昭, 長井隆行, “低コストな家庭用ヒューマノイドロボットの開発”, SI2012, 3H2-2, 2012.12

## 本人発表以外の関連発表状況

(1) 日南雄貴, 中村友昭, 荒木孝弥, 丸山恭平, 長井隆行, 大木宗一, “マルチモーダル情報の取得と語意獲得をオンラインで人と協調しながら長期的に行うロボットプラットフォーム”, SI2012, 3H2-4, 2012.12

(2) 藤岡直幹, 阿部香澄, 中村友昭, 荒木孝弥, 丸山恭平, 長井隆行, “遊び相手ロボット実現のためのヒューマノイドロボット遠隔操作インターフェース”, 電気学会計測研究会, IM-12-073, 2012

(3) 日南雄貴, 中村友昭, 荒木孝弥, 丸山恭平, 長井隆行, 大木宗一, “マルチモーダル情報の取得と概念・語意獲得を長期的に人と協調しながら行うロボットプラットフォーム”, 電気学会計測研究会, IM-12-070, 2012

(4) Muhammad Attamimi, 丸山恭平, 前田泰斗, 中村友昭, 長井隆行 “物体認識と材質認識を用いた掃除タスクの実現”, 日本ロボット学会学術講演会, 2J1-2, 2011.09

## 受賞歴

(1) ロボカップジャパンオープン 2010 大阪@ホームリーグ優勝, 2010.05

(2) ロボカップロボット学会賞, 2010.05

(3) ロボカップ世界大会 2010 シンガポール@ホームリーグ優勝, 2010.06

(4) ロボカップジャパンオープン 2011 大阪@ホームリーグ優勝, 2011.05

(5) ロボカップ人工知能学会賞, 2011.05

(6) ロボカップジャパンオープン 2012 大阪@ホームリーグ優勝, 2012.05

(7) ロボカップ世界大会 2012 メキシコ@ホームリーグ準優勝, 2012.06