博士学位論文

多関節ロボットのモーション実行用
ソフトウェアアーキテクチャの提案

２０１８年

中京大学大学院 情報科学研究科 情報認知科学専攻

加 藤 央 昌

目 次

第 1 章　序論 **1**

1.1　背景 1

1.2　多関節ロボットの多機種展開に向けての問題点 3

1.3　研究目的 4

1.4　本論文の構成 5

第 2 章　多関節ロボットのモーションと提案するアーキテクチャの概要と利点 **9**

2.1　ロボットのプランニング 9

2.2　多関節ロボットのモーション生成から実行までの流れ 10

2.3　多関節ロボットのモーション実行における問題点 11

2.4　本論文で提案するモーション実行用ソフトウェアアーキテクチャ 13

2.4.1　提案するモーション実行用ソフトウェアアーキテクチャの目標 . 13

2.4.2　モーションの実行に必要な要素の抽出 15

2.4.3　メモリ・ベースト制御 15

2.4.4　提案するモーション実行用ソフトウェアアーキテクチャの概要 . 16

2.4.5　提案するアーキテクチャによるモーション実行用ソフトウェア開発における利点 18

2.5　ロボット制御システムのシステム化技術 20

2.6　第 2 章で使用する図 21

第 3 章　多関節ロボットのモーション実行用ソフトウェアアーキテクチャの提案 **29**

3.1　多関節ロボットのモーション実行用ソフトウェアアーキテクチャ 29

3.1.1　モーションデータ 30

3.1.2　モーション実行機能 30

3.1.3　モーションデータの選択機能 31

3.1.4 モーションデータの読み込み機能 31
3.2 センサフィードバック 32
3.3 多関節ロボットのモーション実行システムの試作 33
3.3.1 モーションデータの構造 34
3.3.2 モーション実行機能においてのモーション実行方法 34
3.3.3 モーションデータの読み込み例としてのモーション合成 35
3.4 動作実験 1 35
3.4.1 2 脚ロボットでのモーション実行 36
3.4.2 アームロボットでのモーション実行 36
3.5 動作実験 2 37
3.6 動作実験 3 39
3.6.1 センサフィードバック処理の説明 39
3.6.2 モーションの切り替え 40
3.6.3 屈伸運動中に押された場合の押し返し 40
3.7 まとめ 41
3.8 第 3 章で使用する図 42

第 4 章 多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性 55
4.1 多関節ロボットのモーション実行用ソフトウェアアーキテクチャの機能拡張 55
4.1.1 モーションの N 倍速実行機能を追加する背景 56
4.1.2 モーションの N 倍速実行機能 56
4.1.3 モーション間の同期機能を追加する背景 57
4.1.4 モーション間の同期機能 58
4.2 動作実験 1 60
4.2.1 N 倍速実行機能の実行結果 60
4.2.2 N 倍速実行機能の効果の考察 61
4.3 動作実験 2 62
4.3.1 モーション間の同期機能の実行結果 62
4.3.2 モーション間の同期機能の効果の考察 63
4.4 まとめ 65

4.5 第 4 章で使用する図 66

第 5 章 結論 **75**

5.1 まとめ 75

5.2 今後の展望 77

謝辞 **79**

参 考 文 献 **81**

研究業績 **89**

図 目 次

1.1 ロボットを構成する技術要素 6
1.2 ロボット制御の流れ 6
1.3 1つのモーションの生成から実行までの流れ 7
1.4 多関節ロボットのモーション実行用ソフトウェアアーキテクチャのアイディア 7
1.5 本論文の構成 8

2.1 移動ロボットの経路 22
2.2 マニピュレータの経路生成例 23
2.3 逆運動学による関節角度の不良設定 23
2.4 ロボット制御システムの考え方 24
2.5 多関節ロボットのモーション実行用ソフトウェアアーキテクチャのアイディア 25
2.6 メモリ・ベースト制御 25
2.7 多関節ロボットのモーション実行用ソフトウェアアーキテクチャ 26
2.8 モーションの実行部分とセンサフィードバック 26
2.9 多関節ロボットのモーション実行用ソフトウェアアーキテクチャの機能拡張 27
2.10 モーション実行用ソフトウェアの概要図 27

3.1 多関節ロボットのモーション実行用ソフトウェアアーキテクチャ 43
3.2 モーションの実行部分とセンサフィードバック 43
3.3 メモリ・ベースト制御 44
3.4 モーション実行用ソフトウェアの概要図 44
3.5 モーションデータの構造 45

3.6 モーションの実行方法 45
3.7 モーションの合成 (motion1 + motion2) 46
3.8 実験に使用したロボットの構成 46
3.9 2 脚ロボットの右脚屈伸モーションの実行結果 47
3.10 アームロボットの動作結果 48
3.11 腿関節を外に広げる動作 49
3.12 右脚屈伸 (図 3.9) と腿関節を外に広げる動作 (図 3.11) の合成結果 . . . 50
3.13 実行できるモーション数の比較 51
3.14 2 脚ロボットの足裏に搭載したセンサについて 51
3.15 センサ入力によるモーションの切り替え結果 52
3.16 屈伸運動中に押された場合の様子（センサフィードバックあり） 53
3.17 屈伸運動中に押された場合の様子（センサフィードバックなし） 54

4.1 多関節ロボットのモーション実行用ソフトウェアアーキテクチャ 67
4.2 多関節ロボットのモーション実行用ソフトウェアアーキテクチャの機能拡張 67
4.3 2 脚ロボットのモーションデータと実行タイミングの概念図 68
4.4 右脚屈伸の様子 69
4.5 2 倍速での右脚屈伸の実行結果 70
4.6 右脚屈伸と左脚屈伸の合成による両足屈伸の様子 71
4.7 モーションの同期機能による足踏み動作の実行結果 72
4.8 4 脚ロボットのモーションデータと実行タイミングの概念図 73
4.9 モーションの同期機能による効果 74

表 目 次

4.1　モーションデータ数の比較 62

第1章

序論

1.1 背景

ロボットに関する研究・開発は古くから行われており，様々なロボットが開発されてきた．2005年に開催された愛・地球博では，ヒューマノイド型ロボット，4脚ロボットをはじめ，人間との共生を期待させるロボットが登場し，世界中の人々を驚かせた．ヒューマノイド型ロボットでは，本田技研工業株式会社のASIMOやトヨタ自動車株式会社のパートナーロボットが挙げられる．また，エンターテインメントロボットとして，ソニー株式会社のQRIOやAIBOが挙げられる．このようなロボットの技術は近年において進化が著しく，実際に人間との共生を始めているロボットもある．災害現場などの人間が立ち入れない場所においては，レスキューロボットが活躍しており，2011年3月11日に発生した東日本大震災では原子力発電所内の情報収集のために実際に導入されたことは記憶に新しい．NASAのヒューマノイドロボット開発計画によって開発されたValkyrie（正式名称：R5）は，災害対応ロボットとして設計されたが，現在では人類の火星移住計画などでの活躍が期待されている．Aldebaran Robotics社の小型ヒューマノイドロボットであるNAOは，接客運用の試験が開始されている．NAOは，ロボカップサッカーの標準プラットフォームリーグにおいての共通プラットフォームとしても使用され,「2050年，人型ロボットでワールドカップ・チャンピオンに勝つ」という目標のもと，プログラミングの優劣が競われている．また，ソフトバンクグループ株式会社のPepperは，家庭向けやビジネス向けに販売されており，日常生活での会話やビジネス上での接客などで活躍している．

このように，ロボットは日常生活上での作業や，危険環境下での作業，さらに産業用ロボットなどによる生産現場での作業において人間の代わりを担う役割を持つ．ヒュー

マノイド型ロボットや 4 脚ロボットであれば，歩行に用いる脚が複数の関節によって構成される．また，ヒューマノイド型ロボットに限らず，ものを持つ，ものを移動させる等の作業を行うために，複数の関節によって構成されたアームを搭載するロボットも多い．以下，本論文では，回転軸による関節構造を複数個持つロボットのことを多関節ロボットと呼ぶ．多関節ロボットでは，各関節に搭載されたアクチュエータを制御することで関節角度を変化させ，その動きによって作業目的を完遂する．たとえば，アームロボットでは，各関節のアクチュエータを制御することで各関節角度を変化させ，アームの先端を把持対象まで移動させる動きを行う．また，作業目的によって多関節ロボットの関節数は異なる．そのため，作業目的を完遂するための各関節の角度変化パターンも多岐にわたり多関節ロボットの制御は複雑になる．多関節ロボットの制御の複雑さについては，第 2 章で述べる．以下，本論文では，多関節ロボットの各関節の角度変化による動きの制御および動きそのものをモーションと呼ぶ．

さて，ロボットは多くの技術の集まりであることは周知の事実である．ロボット白書 2014[1] では，ロボットの技術要素をシステム化技術，環境知能化技術，認識処理技術，センシング，コンピュータ，プランニング，制御，アクチュエータ，機構の 9 つに分類している．本論文では，ロボット白書 2014[1] での分類を参考に，ハードウェア，ソフトウェアの観点で図 1.1 のように分類した．図 1.1 中のシステム化技術とは，多数の技術要素を統合し，迅速にロボットシステムを構築する技術である．また，環境知能化技術は，ロボットが活動しやすい場を提供する技術である．図 1.1 に示すように，ロボットは作業目的に合わせたハードウェアとソフトウェアによって構成される．そのため，システム化技術と環境知能化技術はハードウェア，ソフトウェアの範囲外とした．ロボットのソフトウェアは，ロボットの行動を管理し制御する．ロボットの行動は，図 1.2 に示すロボット制御の流れのように，多様なセンサの情報をもとにした音声認識，画像認識，自己位置推定などの外界情報の認識，作業目的と認識結果から行動を選択するプランニング，選択した行動を実行するアクチュエータ制御によって行われる．行動を実行するアクチュエータ制御は，ロボット制御の中でも下位レベルの制御である [2]．近年では，外界情報の認識処理技術が向上したため，プランニングが詳細に行われるようになった．これにより，多関節ロボットのようにアクチュエータ数が多い場合においてはモーション実行用ソフトウェアが複雑になり，モーションの評価を行うことが困難になるという問題を招く．ロボットの活躍の場が日常生活上まで広がりつつある昨今では，ロボットが多機種に展開され量産されることを考慮す

ると安全性の確保が必要とされる．特に，多関節ロボットにおいては，モーションは安全性の高いものでなければならない．

1.2 多関節ロボットの多機種展開に向けての問題点

近年では，ロボットが人間の生活に関わる場面が増えてきた．しかし，自動車のように量産され，人間の生活に密着したものではない．現状では，開発するロボットの台数が少ないため，研究機関などの少数気鋭でロボット開発が行われている．ロボットの台数が少ない現状では，ロボット専用のソフトウェアを機種ごとに作成すれば対応可能である．これは，多関節ロボットも同様であり，モーションの評価もロボットの台数が少ないため対応可能である．本論文におけるモーションの評価とは，多関節ロボットにおいて所望のモーションが実行可能であるかを確認することである．今後，ロボットが日常生活上で活躍することを考えると，多関節ロボットが多機種に展開され量産されることを考慮する必要がある．多関節ロボットは，アームを有するロボット，2脚ロボット，4脚ロボット，6脚ロボットなど，その形態によって必要となるモーションが異なる．また，多関節ロボットは，アームを有するロボット，2脚ロボット，4脚ロボット，6脚ロボットなどの種類のみでなく，今後どのような形態の多関節ロボットが登場するか不明である．それらのロボットに対して，モーションを実行するための専用ソフトウェアを形態や機種ごとに作成することは困難である．さらに，多機種展開によって量産される場合においては，ロボットの行動に対して安全性が問われる．特に多関節ロボットでは，実行されるモーションを保証することで安全を確保する必要がある．たとえば，消費者にとって多関節ロボットのモーションがどのような動きをするのか分からないことは危険である．そのため，開発者側は多関節ロボットのモーションが，どのような動きであるか示すことで安全を確保する必要がある．本論文におけるモーションの保証とは，実行されるモーションの軌道の保証であり，同時にモーションの生成によって作成されたモーションを実行するために必要な情報の保証である．1つのモーションの軌道は図1.3に示すように，運動解析やモーションエディタ等によって生成される．生成されたモーションは，モーションの実行に必要な情報である各関節の角度情報とその角度に到達するまでの移動時間に変換され，その情報に基づいて関節角度制御を行うことで実行される．この際，関節角度制御の出力結果である角度情報は，アクチュエータに出力することでモーションとして具現化される．つ

まり，上述の本論文におけるモーションの評価は，モーションの実行に必要な情報を用いて，生成されたモーションが実行可能であるか確認することである．また，本論文におけるモーションの保証とは，生成されたモーションが実行できることを保証することである．なお，モーションの生成結果は，多関節ロボットの形態，アクチュエータの性能，軌道の生成手法に依存する．多関節ロボットの形態が異なれば，関節数やリンクの長さも異なる．そのため，図1.3に示すモーションの実行に必要な情報である各関節の角度情報とその角度に到達するまでの移動時間は，多関節ロボットの形態に依存する．さらに，多関節ロボットに使用するアクチュエータは様々あり，アクチュエータへの出力方法も異なる．つまり，モーションはハードウェアとソフトウェアの関連が深く，多関節ロボットの機種や形態ごとに専用のソフトウェアを作成しなければならない．

生成された多関節ロボットのモーションを保証するためには，試作された多関節ロボットを用いて，実行されるモーションを保証するためのモーションの評価が必要である．そして，評価を繰り返し行うことでモーションを保証することが可能となり量産される．これは，量産が進み人間の生活に密着している自動車と同様である．そのため，特定の多関節ロボット専用のソフトウェアを，多関節ロボットの機種や形態ごとに作成していては，今後の多関節ロボットの多機種展開において自動車のような普及は難しいだけではなく，ソフトウェアごとに構造が異なる場合においては実行されるモーションの保証に対して，信頼性が下がることが懸念される．

1.3　研究目的

本論文の目的は，多関節ロボットの多機種展開に向けての問題点の対応策として，メモリ・ベースト制御による多関節ロボットのモーション実行用ソフトウェアアーキテクチャの提案を行うことである．メモリ・ベースト制御については2.4.3節で述べるが，予め作成されたモーションを選択・実行する方法である．本提案の基本アイディアを図1.4に示す．本提案は，メモリ・ベースト制御を用いることで，モーションの生成によって作成されたモーションデータを予め用意および保持し，モーションデータを選択することでモーションを実行するソフトウェアアーキテクチャの提案である．モーションデータとは，図1.3中の1つのモーションの実行に必要な情報である各関節の角度情報とその角度に到達するまでの移動時間のことであり，ロボットの形態，アク

チュエータの性能，軌道の生成手法などに基づいて作成される．予めモーションデータを作成することで形態の違いを吸収し，保持しているモーションデータを選択・実行する．また，多関節ロボットに使用するアクチュエータの種類に応じて角度情報の出力方法を変更する．このように，ハードウェアとソフトウェアの関連を整理し，図 1.4 中の関節角度情報による関節角度制御を行うモーション実行用ソフトウェアの構造のルール化を行う．そのため，提案するアーキテクチャを多関節ロボットのモーション実行用ソフトウェアの記述ルールとして使用することで，どのような形態の多関節ロボットにおいても共通のソフトウェア構造を用いることができることから，試作ロボットでのモーションの評価が繰り返し行いやすくなり，量産される多関節ロボットにおいてモーションの保証に対しての信頼性が確保できると考える．なお，本提案は，多関節ロボットのアクチュエータ制御ソフトウェア基本構造の概念およびルールの提案であるため，図 1.1 に示す分類においてのシステム化技術とアクチュエータ制御に関連する提案である．

1.4 本論文の構成

本論文の構成を図 1.5 に示す．まず，第 2 章では，本論文での提案内容を理解する上で必要となる多関節ロボットのモーションについて，モーションの生成・実行および多機種展開に向けての問題点の具体例を述べ，多関節ロボットのモーション実行用ソフトウェアアーキテクチャの目標と概要，および，多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いたモーション実行用ソフトウェア開発における利点について述べる．続く，第 3 章では，提案する多関節ロボットのモーション実行ソフトウェアアーキテクチャについて述べ，モーション実行システムの試作とモーションの実行結果について述べる．第 4 章では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性について述べる．第 5 章では，本論文の結論を述べる．なお，各章で用いる図は，各章の末尾にまとめて記載する．

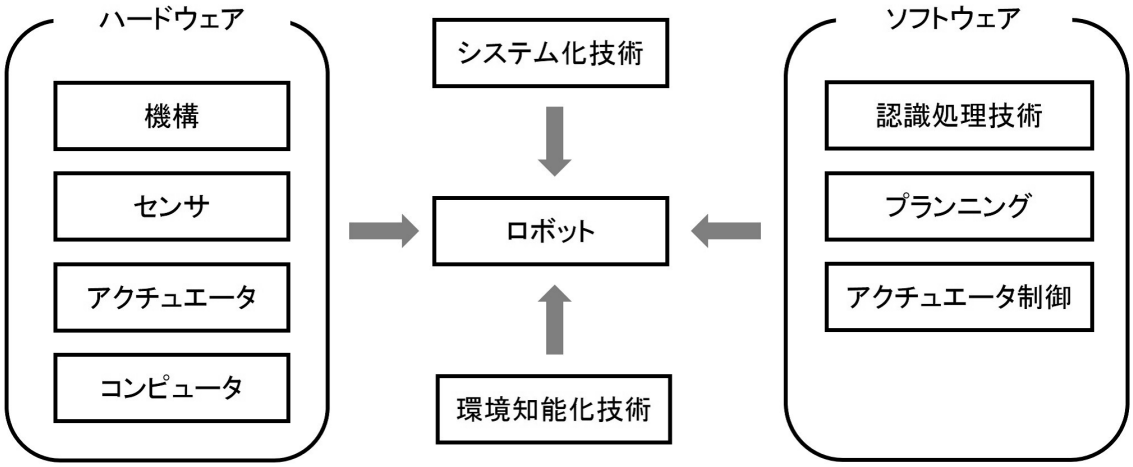


図 1.1: ロボットを構成する技術要素

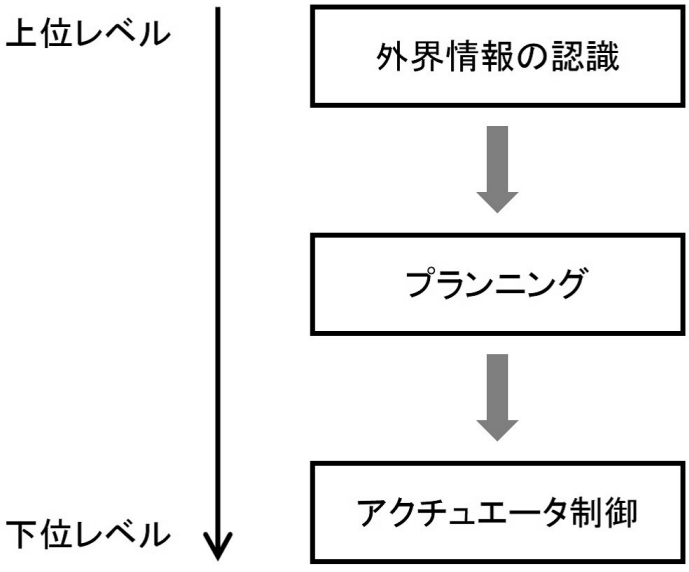


図 1.2: ロボット制御の流れ

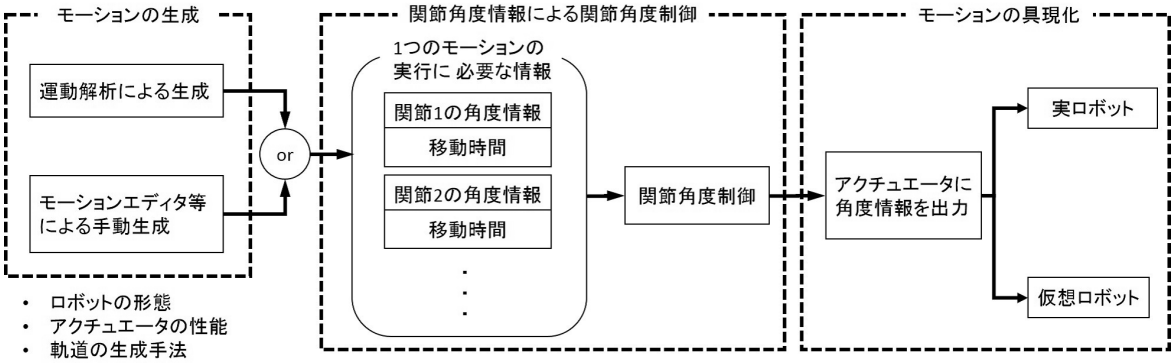


図 1.3: 1 つのモーションの生成から実行までの流れ

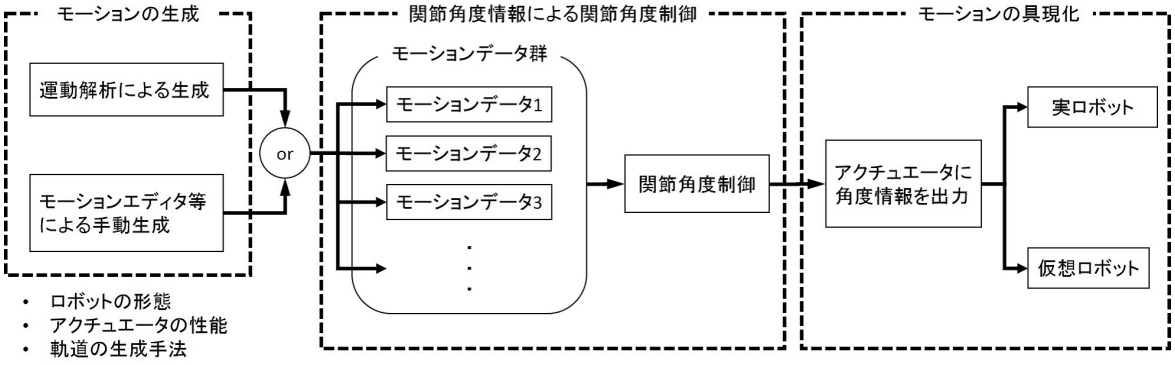


図 1.4: 多関節ロボットのモーション実行用ソフトウェアアーキテクチャのアイディア

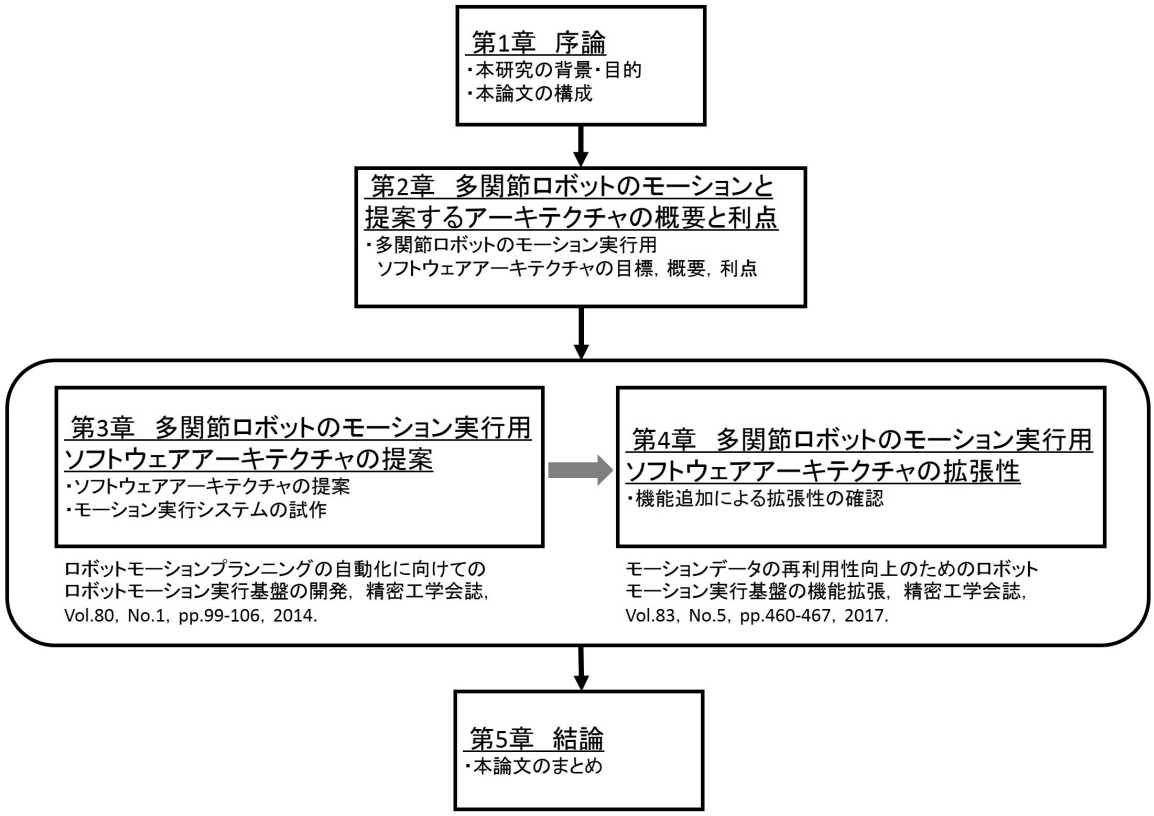

第1章　序論
・本研究の背景・目的
・本論文の構成
第2章　多関節ロボットのモーションと提案するアーキテクチャの概要と利点
・多関節ロボットのモーション実行用ソフトウェアアーキテクチャの目標，概要，利点
第3章　多関節ロボットのモーション実行用ソフトウェアアーキテクチャの提案
・ソフトウェアアーキテクチャの提案
・モーション実行システムの試作
第4章　多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性
・機能追加による拡張性の確認
ロボットモーションプランニングの自動化に向けてのロボットモーション実行基盤の開発，精密工学会誌，Vol.80，No.1，pp.99-106，2014.
モーションデータの再利用性向上のためのロボットモーション実行基盤の機能拡張，精密工学会誌，Vol.83，No.5，pp.460-467，2017.
第5章　結論
・本論文のまとめ

図 1.5: 本論文の構成

第2章

多関節ロボットのモーションと提案するアーキテクチャの概要と利点

本章では，本論文での提案内容を理解する上で必要となる多関節ロボットのモーションについて，モーションの生成からモーション実行までの流れについて述べる．また，多関節ロボットの多機種展開に向けてのモーション実行の問題点を述べ，その解決手段として多関節ロボットのモーション実行用ソフトウェアアーキテクチャを提案する．なお，本章で使用する図は，章末 p.22 よりまとめて記載する．

2.1　ロボットのプランニング

本節では，ロボットの行動を決定するプランニングについて述べる．

ロボットの行動は，センサ情報を用いた認識結果に基づいてプランニングにより決定される [1]．プランニングは，作業目的を達成するための手段を与える．そのため，車輪型の移動ロボット，脚機構の移動ロボット，マニピュレータなどのロボットの行動において，プランニングは必要不可欠な工程である．プランニングの研究は古くから行われており，移動ロボットやマニピュレータなどのロボットに対して研究が行われている．移動ロボットでは，障害物回避 [3]〜[5] や障害物を回避するための障害物領域の予測 [6]，移動ロボットの軌道を計画する手法 [7] や複数移動ロボットの行動を計画する手法 [8]〜[10] などがある．プランニングは，ロボットの行動を決定するものであるため，経路計画や軌道計画という言葉がよく使われる．

ここで，経路と軌道の違いを述べる．「製造業及び非製造業の両環境において運転するロボット及びロボティックデバイスに関する用語について規定」している日本工業規格（JIS B 0134）では，経路は「ポーズの順序集合（path)」，軌道は「経路に時間

をパラメータとして付加したもの（trajectory）」と定義されている [11]．図 2.1 に示す区間 1 から区間 5 で構成された移動ロボットの経路を使って，移動ロボットの経路と軌道について説明する．図 2.1(a) は，ロボットのスタート位置（区間 1 のスタート位置）である．区間 1 は図 2.1(b) に示す区間 2 のスタート位置まで，区間 2 は図 2.1(c) に示す区間 3 のスタート位置まで，区間 3 は図 2.1(d) に示す区間 4 のスタート位置まで，区間 4 は図 2.1(e) に示す区間 5 のスタート位置まで，区間 5 は図 2.1(f) に示す終了位置までである．このとき，移動ロボットは各区間を移動するが，そこに時間の概念はなく，終了位置に到達するまでにどれだけの時間を要してもよいという考え方が経路であり，経路を計画することを経路計画という．一方で，軌道は各区間の移動時間が考慮される．そのため，各区間の移動時間を考慮した上での区間 1 から区間 5 までの経路が軌道であり，軌道を計画することを軌道計画という．車輪型ロボットでは，経路計画や軌道計画によって生成された道筋を，アクチュエータ制御を行うことで走行する．移動ロボットを例に説明したが，マニピュレータのような多関節ロボットであっても同様である．次節では，多関節ロボットの例としてマニピュレータのモーションの生成・実行までの流れについて述べる．

2.2　多関節ロボットのモーション生成から実行までの流れ

本節では，多関節ロボットの例としてマニピュレータのモーションを生成し，実行するまでの流れについて述べる．

前節でも述べたように，プランニングの研究は古くから行われている．マニピュレータのプランニングには，作業時の障害物回避 [12]〜[17] が挙げられる．マニピュレータの作業目的を達成させるためには，エンドエフェクタを目標位置に移動させる必要がある．エンドエフェクタを目標位置に移動させるためには，マニピュレータを構成する複数の関節の角度を逆運動学によって算出する必要がある．図 2.2 にマニピュレータの経路生成の例を示す．マニピュレータのエンドエフェクタを目標位置に移動させることを考えた場合，θ_1 と θ_2 を目標位置での角度まで変化させる必要がある．このとき，障害物を考慮せずに経路を生成し，θ_1 と θ_2 を逆運動学によって算出された目標位置での角度に変化させてしまうと，図 2.2(a) のように障害物に衝突してしまう．これでは，マニピュレータとしての作業目的が達成できない．そこで，図 2.2(b) のように，中間点を設け目標位置にエンドエフェクタを移動させることにより，障害物との衝突

を避けることができる．しかし，θ_1 と θ_2 の角度変化を同時に行わなければ，図 2.2 に示すような経路をたどることはできない．たとえば，θ_1 の角度を変化させる時間が早く，θ_2 の角度を変化させる時間が遅ければ，図 2.2(a) と同様に障害物に衝突してしまう．そのため，マニピュレータのような多関節ロボットは，各関節ごとに経路に対して時間要素を付加した軌道が重要である．マニピュレータの軌道に着目した研究として，軌道の生成手法 [18]〜[22] や，より効率的な軌道の生成手法 [23]〜[27] がある．また，マニピュレータは人間の手の代わりを行うものであるため，物体操作についてのプランニング [28]〜[32] においても軌道は重要な要素である．さらに，複数台のマニピュレータによる協調動作 [33], [34] においては，各マニピュレータの障害物回避や，マニピュレータ同士の衝突回避などを考慮しなければならないため，軌道の重要度は増す．

以上のことから，マニピュレータのような多関節ロボットにおいては，生成された経路に時間要素を付加した軌道が重要であり，軌道に基づいて関節角度をアクチュエータ制御によって変化させることで，作業目的に応じたモーションを実行する．ロボットモーションは，実時間処理によるプランニングを行い，作業目的に応じたモーションを実行できることが理想であると考えられる．

2.3 多関節ロボットのモーション実行における問題点

本節では，多関節ロボットのモーション実行における問題点について述べる．

前節では，多関節ロボットのモーションを生成することにおいて軌道が重要であり，軌道を生成するために逆運動学を用いることを述べた．逆運動学を用いることで，マニピュレータのような多関節ロボットにおいて先端軌道を生成することが可能であるが，逆運動学によって算出された関節角度は一意ではない．これは逆運動学問題と呼ばれており，平面 2 リンクのアームロボットでは，図 2.3 に示すようにエンドエフェクタが目標位置に到達したときの関節角度が 2 パターンある．リンク数が増えることによって，エンドエフェクタが目標位置に到達したときの関節角度のパターンは増加する．逆運動学問題を解決するための手法 [35]〜[38] があるが，いずれの場合においても関節角度の生成に時間がかかるだけでなく，必ず解けるという保証はない [39], [40]．加えて，特異姿勢についても考慮しなければならない [41]．このことは，多関節ロボットのモーションを実時間処理で生成し実行することについて，安全なモーションが保証されないことを意味している．また，多関節ロボットの開発者自らが，開発した多

関節ロボット専用のモーションを実時間処理で生成し実行するソフトウェアを作成する場合，所望の動きを実行するためには逆運動学問題や特異姿勢に対して例外処理を行う必要がある．逆運動学問題や特異姿勢の全てに対して対応することは困難である．これは，第1章で述べたようにモーション実行用ソフトウェアが複雑になり，モーションの評価が困難になるという問題を招く．これも，安全なモーションが保証されないことを意味する．さらに，多関節ロボットはマニピュレータだけではない．文献 [42] には，脚移動を行うロボットについて述べられており，脚移動を行う多関節ロボットの脚の数には2脚，4脚，6脚などの種類がある．このように，制御対象が増えた場合においては各脚の協調動作が求められるため，モーション実行用ソフトウェアはより複雑になる．その中でも，2脚のロボットは，協調動作のみではなく，片脚支持時における転倒を防ぐために安定化を考慮したモーションが必要になり [43]〜[46]，より複雑さを増す．このことは，脚機構を有する多関節ロボットのモーションを実時間処理で生成し実行することについても，安全なモーションが保証されないことを意味している．加えて，逆運動学問題が解決され実時間処理によってモーションの生成と実行が可能な場合を仮定した場合においても，安全なモーションが保証されるとは限らない．ロボットの行動はセンサ情報による外界認識から始まる．センサ情報によって，多関節ロボットのモーションが生成されることを考えると，この場合においては，センサ情報が常に正しいとは限らないということが懸念事項として挙げられる．これは，どれだけ高精度な認識処理技術が存在しても，センサ情報が正しくない場合は正しいモーションが生成されないということであり，安全なモーションが保証されないことを意味している．

ロボットの活躍の場が日常生活上まで広がりつつある昨今においては，ロボットが多機種に展開され量産されることを考慮しなければならない．多関節ロボットのモーションが保証されないことは，作業目的を達成できないことと同じである．また，多関節ロボットは行動そのものがモーションを伴うため，安全なモーションを保証しなければならない．そのため，実時間処理によるモーションの生成・実行は量産多関節ロボットには向かないといえる．モーションが保証されない場合は，実運用の場である日常生活上がロボットモーションの試験場になることを意味しており，非常に危険である．昨今ではCPUの計算能力が向上したことや，センサ技術の向上，認識処理技術の向上により実時間処理でのモーションの生成や実行に対して，前向きに検討すべき項目である．しかし，多機種に展開され量産される場合においては安全性が最も

重要視されるべきであると考える．そのため，向上した計算能力やセンサ技術は，ロボットの安全性向上のために認識処理技術に利用することがよいと考える．

2.4 本論文で提案するモーション実行用ソフトウェアアーキテクチャ

本節では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの目標について述べる．その後，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの概要を説明するために，モーションの実行に必要な要素，メモリ・ベースト制御について述べる．また，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いたモーション実行用ソフトウェア開発における利点について述べる．

2.4.1 提案するモーション実行用ソフトウェアアーキテクチャの目標

ロボットが多機種に展開され量産される場合においては，ロボットの動作が安全であることが最も重要である．多関節ロボットにおいては，モーションの保証が安全性を左右する．多関節ロボットは，各関節に搭載されているアクチュエータを制御することによってモーションを実行する．モーションを保証するためには，モーションを十分に評価する必要があるが，形態や機種の異なる多関節ロボットの全てに対してモーションを実行し，モーションの評価を行うことは多大なコストを要するであろうことは容易に想像がつく．自動車業界においては，多岐にわたる車種においてそれぞれ数台の試作車を製作し，十分な評価を行った後に量産車が製作され販売される．今後，活躍が期待される多関節ロボットにおいても同様の措置が必要であると考えられる．また，2.3 節で述べたように，多関節ロボットはロボットごとにアクチュエータ数が異なり，モーションも異なるため，モーション実行用ソフトウェアも複雑になる．そのため，モーションの評価を行うことは容易ではない．これは，図 2.4(a) に示すように，ロボットをひとつのシステムとして考えた場合において，ロボット制御のうち下位制御に当たるアクチュエータ制御は，上位制御の認識処理，プランニングと相互に影響しあうからであると考えられる．近年では，ロボットの評価に物理演算エンジンを搭

載したシミュレータを用いて，仮想空間での評価を行うことも容易になってきたため，実際のロボットを用意することなく評価を行うことが可能となった [1]．そのため，評価にかかるコストは低く見積もることができるが，モーションを実行するソフトウェア部分の記述方法は定まっていないため，上述と同様の複雑さを持つ．また，近年では，ロボットを一つのシステムとして捉えるのではなく，図 2.4(b) に示すように個々のシステムの集合体として考え，ロボットを構成する技術要素を組み合わせることでロボットのシステムを構築するシステム化技術に注目が集まっている．システム化技術については 2.5 節で述べるが，多関節ロボットのモーションを実行するソフトウェア部分の記述方法は定まっていないため，上述と同様の複雑さを持つことについては変わりない．

そこで，本論文では，システム化技術と同様に，ロボットのシステムを図 2.4(b) に示すような個々のシステムの集合体として考え，モーションを実行する部分であるアクチュエータ制御システムのソフトウェアアーキテクチャを提案する．具体的には，多関節ロボットのモーション実行に必要な要素，機能を選別し，それらを組み合わせることにより，多関節ロボットのモーション実行用ソフトウェアアーキテクチャの提案を行う．これは，1.3 節で述べたように，図 2.5（図 1.4 と同じ）中の関節角度情報による関節角度制御を行うモーション実行用ソフトウェアの構造のルール化である．また，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，どのような形態の多関節ロボットにおいても共通のソフトウェア構造を用いることができ，試作ロボットでの評価の繰り返しや量産される多関節ロボットにおいてモーションの保証に対しての信頼性を確保するために，作成されたモーションを評価しやすく，実際に運用される量産多関節ロボットにおいても同じアーキテクチャを利用できる必要がある．そのため，予め作成されたモーションデータを選択・実行するメモリ・ベースト制御を利用する．これにより，多関節ロボットが多機種に展開され量産される場合において，試作多関節ロボットで評価されたモーションもしくはシミュレータ内で評価されたモーションを，量産された多関節ロボットで利用することが可能となるため，保証されたモーションを量産ロボットで実行可能となる．しかし，単純なメモリ・ベースト制御では，すべての状況下において適切なモーションを用意しなければならず，モーションデータとして保持しなければならない．これは，実運用において現実的ではないため，モーションの組み合わせによるモーションの合成やモーション間の同期，モーションの速さなどを考慮した場合の拡張性についても考慮する．本提案は，

多関節ロボットにおいて関節角度情報による関節角度制御を行うモーション実行用ソフトウェアの設計指針となるアクチュエータ制御システムの基本構造としての提案である．

2.4.2 モーションの実行に必要な要素の抽出

文献 [47] では，アームロボットの制御について紹介すると同時に，ロボットは基本的に何らかの駆動部分を有してることを明言している．多関節ロボットの駆動部分は，各関節のアクチュエータであり，アクチュエータを制御することにより関節の角度を変化させ，モーションを実行している．2.2 節で述べたが，必要となるのは所望のモーションを実行するための各関節の角度情報と，各関節の角度変化に要する時間である．本論文では，所望のモーションを実行するための各関節の角度情報と，各関節の角度変化に要する時間を合わせてモーションデータと呼ぶ．また，保証されたモーションを実行するために，メモリ・ベースト制御を利用する．そのため，複数用意されたモーションデータを選択できる機能が必要である．さらに，モーションデータに記載されている関節角度情報を，角度変化に要する時間要素に基づいて実行する機能が必要である．

以上のことから，保証されたモーションの実行に必要な要素は以下の通りである．

(1) モーションデータ
(2) モーションデータの選択機能
(3) モーションの実行機能

2.4.3 メモリ・ベースト制御

メモリ・ベースト制御 [48] とは，図 2.6 のように予め用意したモーション（軌道の関数）をロボット自身が持ち，状況の変化に応じたモーションを選択・実行する方法である．事例として，ジャグリングロボット [49] やエアホッケーロボット [50] などがある．また，2 足歩行ロボットの運動制御に利用した事例 [51], [52] もある．この事例では，一般的に困難であるといわれている 2 足歩行ロボットの運動制御において，状況の変化に応じた 2 足歩行のモーションを多数用意および保持し選択・実行することで 2 足歩行を実現している．

メモリ・ベースト制御を利用することの利点として，状況の変化に応じたモーショ

ンを保持し選択・実行することでモーションを実現可能であるため，保証されたモーションを確実に選択･実行することが可能であるということである．一方で，すべての状況変化に応じたモーションを予めデータベースとして保持し，状況の変化に応じて最良のモーションを選択しなければならないため，モーション数が膨大になる．しかし，本論文で提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいて多関節ロボットのアクチュエータ制御システムを構築した場合では，評価によって保証されたモーションの使用方法を考慮することにより，用意したモーションの数よりも実行できるモーション数を多くすることが可能である．たとえば，モーションの組み合わせによる合成やモーション間の同期が挙げられる．そのため，保証されたモーションを最小単位で保持することが可能であり，モーションの組み合わせによるモーションの合成やモーション間の同期によって得られるモーションは保持しない．ただし，保証されているモーションの使用方法を考慮し合成したモーションやモーション間の同期によるモーションは，別途モーション評価を行いモーションを保証する必要がある．

2.4.4　提案するモーション実行用ソフトウェアアーキテクチャの概要

本論文で提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャを図 2.7 に示す．多関節ロボットのモーション実行用ソフトウェアアーキテクチャの構成は，2.4.2 項で述べたモーションの実行に必要な要素，機能として抽出したモーションデータ，モーションデータの選択機能，モーションの実行機能に加え，モーションデータの使用方法を考慮できる要素としてのモーションデータの読み込み機能を追加したものである．図 2.7 中において，モーションデータ群 (MotionData Group) がモーションデータ，main 処理 (Main Processing) がモーションデータの選択機能，モーション実行部 (Motion Execution Part) がモーションの実行機能，モーションデータ読み込み部 (Reading MotionData Part) がモーションデータの読み込み機能に相当する．提案するモーション実行用ソフトウェアアーキテクチャでは，事前にモーション生成によるモーションデータの作成が行われていることを前提としている．そのため，モーションデータを選択・実行するという点では，メモリ・ベースト制御である．また，図 2.7 中のモーションの実行部内には，図 2.8 に示すようにセンサフィードバック処理が行え

る仕組みを搭載している．つまり，提案するモーション実行用ソフトウェアアーキテクチャでは，モーションの実行機能はモーションの実行に必要な関節角度制御と，センサフィードバック処理が行える仕組みを持つ．モーションの実行は，多関節ロボットの制御において下位制御に当たる．そのため，モーション実行に関する下位レベルでのセンサフィードバックによって，モーションの切り替えやモーションの修正などが可能である．モーションの実行部内に搭載したセンサフィードバック処理は，メモリ・ベースト制御から独立したものである．第 3 章では，多関節ロボットのモーション実行用ソフトウェアアーキテクチャの提案と題し，モーション実行システムの試作と動作実験について述べる．また，モーションデータの使用方法を考慮できる要素として追加したモーションデータ読み込み部について，モーションデータの使用方法の例としてモーションの合成を取り上げ，その効果を述べる．

図 2.9 は，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャにおいて機能拡張したものである．モーションデータの使用方法を考慮できる要素であるモーション読み込み部に与える情報の付加や，モーションデータの時間要素の変更を行うなどの機能を追加している．提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャでは，図 2.7 の基本構造を変えることなく機能追加が可能という柔軟な拡張性を持つ．第 4 章では，提案する関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性について，モーション読み込み部に与える情報の付加やモーションデータの時間要素の変更を行うなどの機能を例として，その効果を述べる．

なお，提案する多関節ロボットのモーション実行ソフトウェアアーキテクチャは，モーション生成と実行を含むモーション実行用ソフトウェアの複雑さは回避されない．しかし，図 2.5 のアイディアに基づいた図 2.7 の多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いることで，モーションデータの作成後においてのアクチュエータ制御システムのソフトウェアの記述方法が統一される．そのため，試作多関節ロボットにおけるモーションの評価が行いやすいだけではなく，モーションの変更による評価項目の選定や問題発生時の原因の追究も行いやすい．この点については，次項で述べる．また，試作多関節ロボットでのモーションの評価を行ったソフトウェア構造を量産多関節ロボットに利用できるためモーションの保証に対しての信頼性が高い．

2.4.5　提案するアーキテクチャによるモーション実行用ソフトウェア開発における利点

本項では，多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いたモーション実行用ソフトウェアを作成した場合について，想定できる状況においての評価内容や利点について述べる．

図 2.10 は，図 2.7 に示す多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いたモーション実行用ソフトウェアの概要図である．なお，モーション実行用ソフトウェアへの入力となるプランニングの出力結果は，選択すべきモーションデータへの情報である．多関節ロボットのモーション実行用ソフトウェアの作成において想定できる場合は以下の 3 点である．

(1) 新機種の多関節ロボットでモーション実行用ソフトウェアを作成する場合
(2) （1）のソフトウェアを用いた同機種の多関節ロボットでモーションに変更が生じた場合
(3) 他の機種の多関節ロボットで（1）のソフトウェアを用いる場合

（1）の場合について述べる．新機種の多関節ロボットでモーション実行用ソフトウェアを作成しモーションを評価する場合，モーションデータの構造や形式に基づいてモーションデータの選択機能，モーションデータの読み込み機能，モーション実行部内の関節角度制御やセンサフィードバック処理をソフトウェアとして記述しなければならない．そのため，各機能の処理の評価も行わなければならない．しかし，各機能の評価が完了しモーションの評価のみになった場合は，モーションデータの追加や修正等のみの変更でモーションの評価を行えるため，モーションを保証するための評価が繰り返し行いやすい．また，モーション評価に用いたモーション実行用ソフトウェアは，そのまま量産多関節ロボットに利用できる．そのため，モーションの保証に対しての信頼性が確保できる．

（2）の場合について述べる．これは，(1）のソフトウェアを用いた同機種の多関節ロボットにおいて，使用するモーションのみが変更される場合である．この場合においては，モーションデータの選択機能，モーションデータの読み込み機能，モーション実行部内の関節角度制御やセンサフィードバック処理等は確定しており評価は完了しているため，モーションデータの追加や修正等のみの変更でモーションの評価を行

える．したがって,(2）の場合においても，モーションを保証するための評価の繰り返しが行いやすく，モーションの評価に用いたモーション実行用ソフトウェアは，そのまま量産多関節ロボットに利用できるため，モーションの保証に対しての信頼性が確保できる．

（3）の場合について述べる．これは,(1）のソフトウェアを用いた多関節ロボットとモーションデータの選択機能，モーションデータの読み込み機能，モーション実行部内の関節角度制御の仕様は同じだが，他の機種の多関節ロボットを量産したい場合である．たとえば，自動車であれば横展開車両である．この場合は,(1）のソフトウェアを用いた多関節ロボットと同じ形態のときと別の形態のときがある．たとえば，同じ形態であれば4脚ロボット同士，別の形態であれば4脚ロボットと6脚ロボットなどの場合である．同じ形態である場合，別の形態である場合のどちらにおいても，モーションデータの選択機能，モーションデータの読み込み機能，モーション実行部内の関節角度制御の仕様は同じあるため，これらの機能の評価は行う必要がない．したがって，多関節ロボットとして必要となるモーションのモーションデータの追加および修正等のみ変更によってモーションの評価を行える．そのため，モーションを保証するための評価の繰り返しが行いやすく，モーションの評価に用いたモーション実行用ソフトウェアは，そのまま量産多関節ロボットに利用できるため，モーションの保証に対しての信頼性が確保できる．ただし，別の形態の多関節ロボットである場合，必要となるセンサフィードバック処理が異なることもあるため，センサフィードバック処理については評価する必要がある．

以上のことから,(1）において多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いたモーション実行用ソフトウェアを作成することで，モーションの変更点に着目した評価のみによってモーションを保証することができる．モーションの組み合わせによるモーションの合成やモーション間の同期，モーションの速さ変更によって具現化されるモーションは，モーションの変更点として着目し，モーションの評価を行う必要がある．なお，モーションの変更に伴う処理の変化点や，多関節ロボットを構成するアクチュエータやセンサ等のハードウェアの仕様変更に伴う処理の変化点については，変化点に着目した評価が必要になる．たとえば，モーションデータの構造や形式に変更が生じた場合は，関節角度制御にも変更が生じる場合がある．このような変更に伴う変更点のことを変化点と呼ぶが，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いたモーション実行用ソフトウェアで

は，変化点を特定しやすい．また，多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いたモーション実行用ソフトウェアでは，モーション実行における問題が発生した場合において問題の原因追究が行いやすい．たとえば，図 2.5 に示すモーションの生成段階で発生している問題なのか，モーションデータやその作成方法に問題があるのか，関節角度制御方法に問題があるのかを特定しやすいといえる．

2.5　ロボット制御システムのシステム化技術

本節では，ロボット制御システムに関連する技術として，システム化技術について述べる．システム化技術とは，多数の技術要素を迅速に統合し，目的に沿ったシステムを構築する手法のことである [1]．以下，システム化技術として RT ミドルウェアと ROS について述べる．また，システム化技術に対しての本論文で提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの位置づけについて述べる．

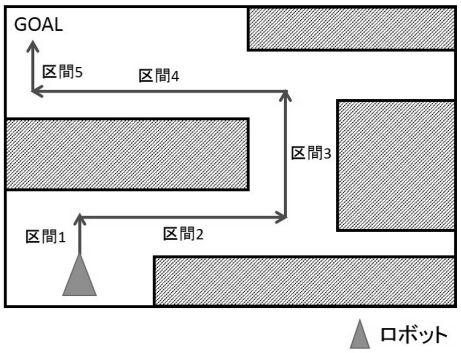
まず，RT ミドルウェアについて述べる．RT ミドルウェアは，技術の共有と再利用を目的としたロボットシステム開発用のシステム化技術のことである．RT ミドルウェアは，ロボットの機能部品（センサ，サーボ，モータ等）をモジュール化し，モジュール化された機能部品を組み合わせることにより，ユーザの幅広いニーズに合わせた新しいロボットシステムを，設計者が容易に効率よく構築することを可能にするものである [53]〜[56]．この技術を用いて実現されたシステムの事例として，移動ロボットシステム [57]，コミュニケーションシステム [58]，マニピュレーションシステム [59], [60]，シミュレータ [61], [62] などがある．また，RT ミドルウェアによる技術の共有と再利用を目的としたプロジェクトは，RT ミドルウェアプロジェクト [63] と呼ばれており，RT ミドルウェアの公開 [64] を初め，知能モジュール構築プロジェクト [65]，ロボットサービスの標準化 [66]，ロボット産業の展開 [67], [68]，ソフトウェア技術の標準化 [69], [70] などの活動が行われている．その他にも，ロボットが活動する環境に RT ミドルウェアを利用する事例として，空間知能化 [71]，ユビキタス [72]，ユニバーサルデザイン [73] などがある．続いて，ROS について述べる．RT ミドルウェアと同様に，ROS も技術の共有と再利用を目的としたロボットシステム開発用のシステム化技術である [74]〜[77]．そのため，ロボットをセンサ，サーボ，モータ等に関する技術に細分化することで機能部品を作成し，それらを組み合わせることによってロボットシステムを構築することが可能である．ROS を活用した事例は，自律移動ロボット [78]，災害対応ロボッ

ト [79]，石油プラント点検ロボット [80]，生活支援 [81], [82]，産業用ロボット [83]，ロボットが活動する環境の知能化 [84]，工学教育への応用 [85] などがある．また，ROS の機能拡張 [86], [87] も積極的に行われており，ロボット技術の発展に向けてより利用しやすい環境も整えられてきている．

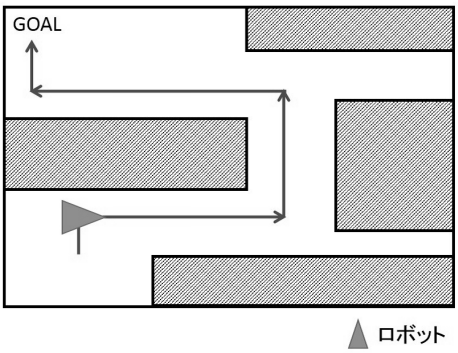
RT ミドルウェアや ROS を用いて多関節ロボットの制御システムを構築したい場合，その利点は，機能部品を組み合わせるだけで多関節ロボットの制御システムを構築することが可能なことである．しかし，多関節ロボットのモーション実行用の機能部品を作成することは容易ではない．なぜならば，2.3 節で述べたように多関節ロボットのモーションを実行するためには複雑な関節角度のパターンの組み合わせを網羅する必要があるため，汎用性のあるモーション実行用の機能部品を作成することはきわめて困難である．本論文で提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは RT ミドルウェアや ROS と同様にシステム化技術であるといえる．しかし，RT ミドルウェアや ROS のようにロボットシステム全体を網羅するものではない．多関節ロボットのモーション実行用ソフトウェアアーキテクチャの対象は，関節角度制御（アクチュエータ制御）によるモーションの実行である．たとえば，RT ミドルウェアや ROS においてのモーション実行の役割を持つ機能部品作成においての概念やルールに相当する．

2.6 第 2 章で使用する図

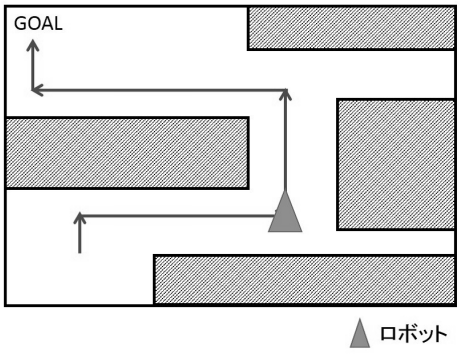
本章で使用する図を次頁より示す．



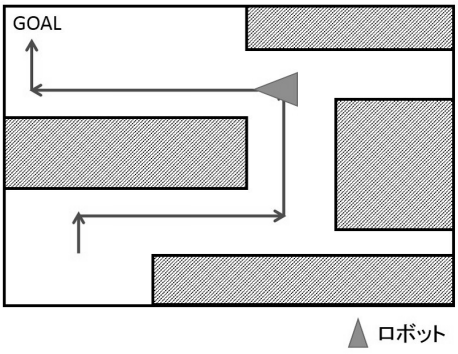
(a) 移動経路とスタート位置
（区間 1 のスタート位置）



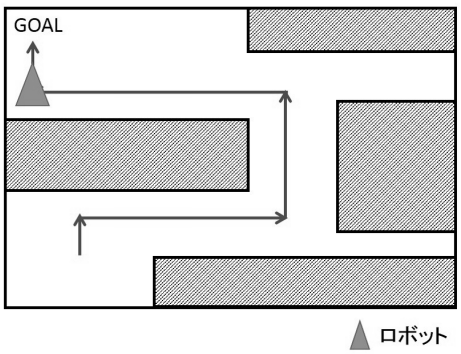
(b) 区間 2 のスタート位置



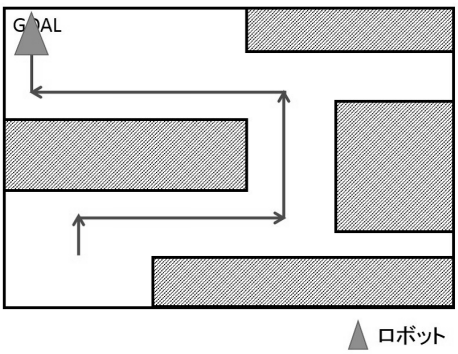
(c) 区間 3 のスタート位置



(d) 区間 4 のスタート位置

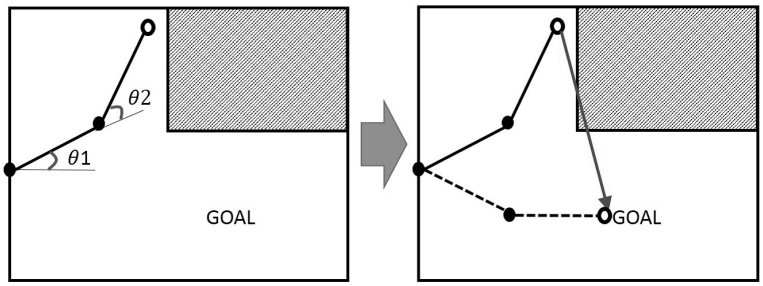


(e) 区間 5 のスタート位置

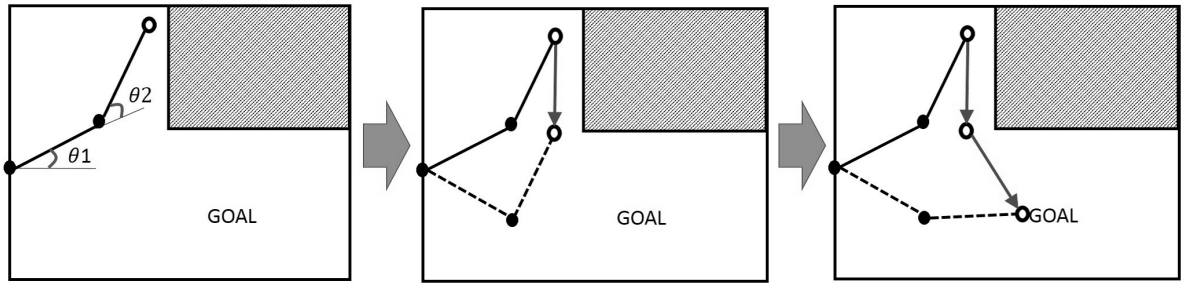


(f) 終了位置

図 2.1: 移動ロボットの経路



(a) 障害物を考慮しない場合



(b) 障害物を考慮した場合

図 2.2: マニピュレータの経路生成例

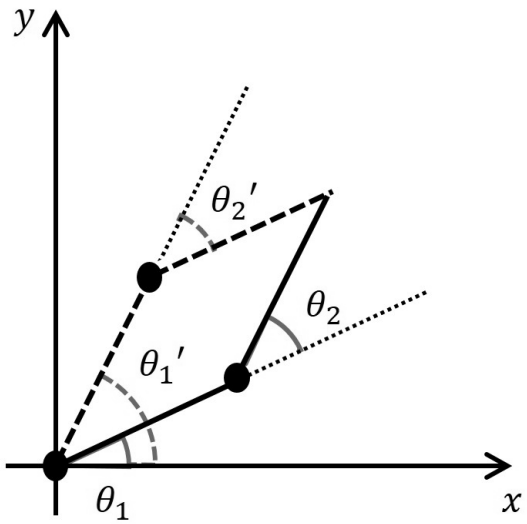
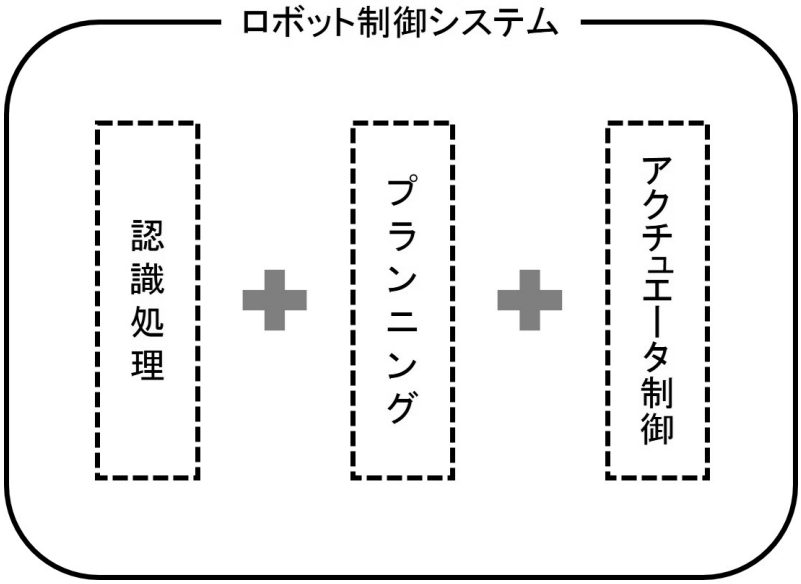
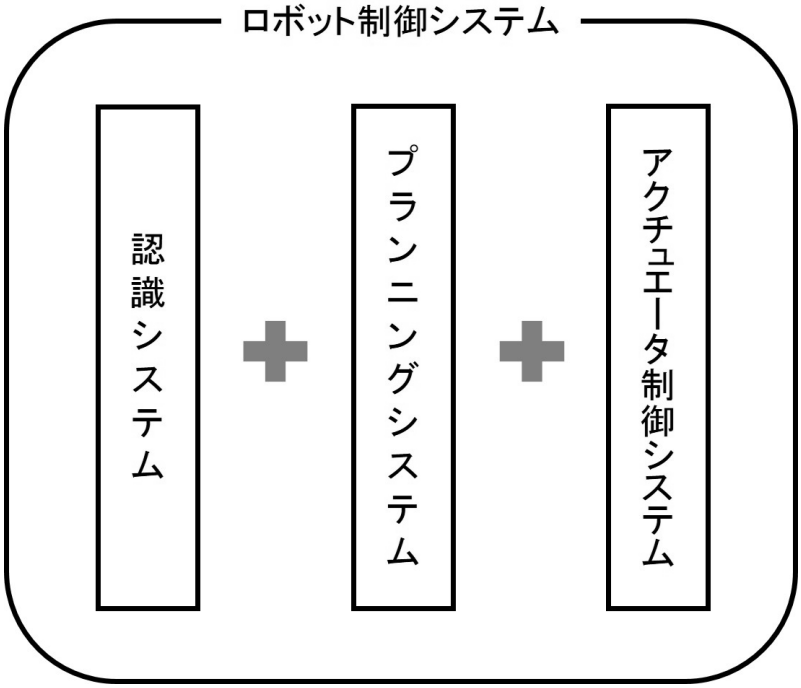


図 2.3: 逆運動学による関節角度の不良設定



(a) ロボットを 1 つのシステムとして考える場合



(b) ロボットを個々のシステムの集合体として考える場合

図 2.4: ロボット制御システムの考え方

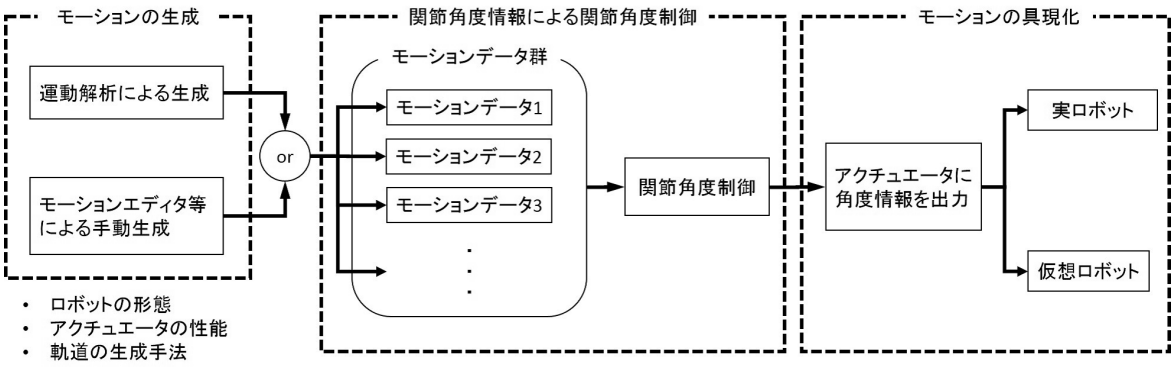


図 2.5: 多関節ロボットのモーション実行用ソフトウェアアーキテクチャのアイディア

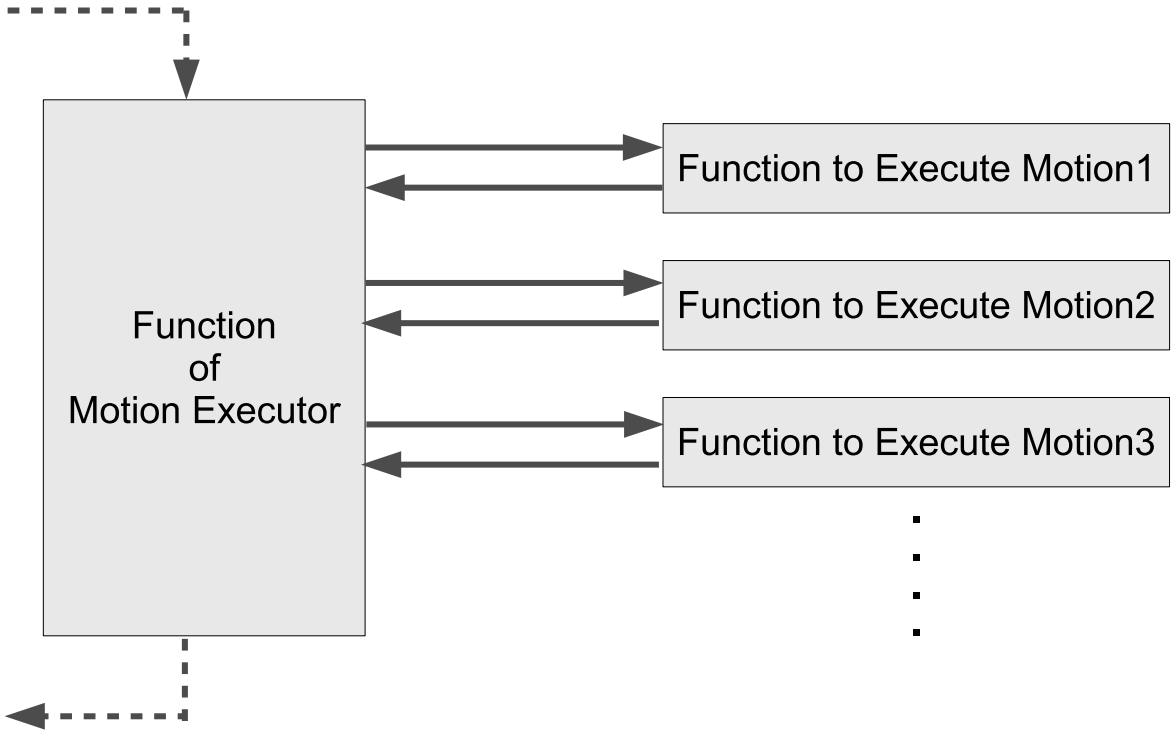


図 2.6: メモリ・ベースト制御

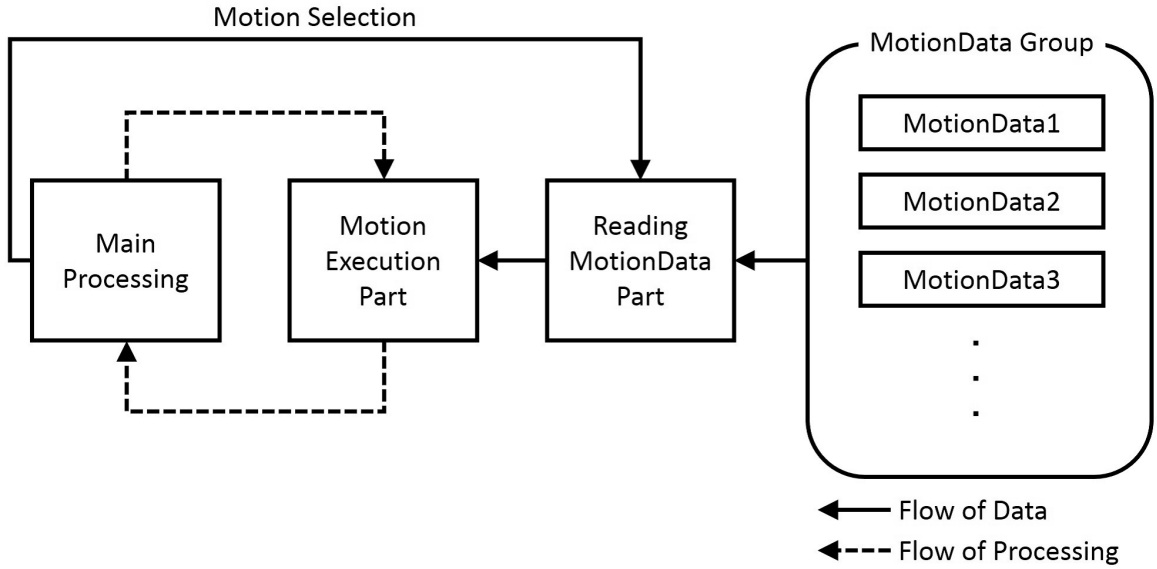


図 2.7: 多関節ロボットのモーション実行用ソフトウェアアーキテクチャ

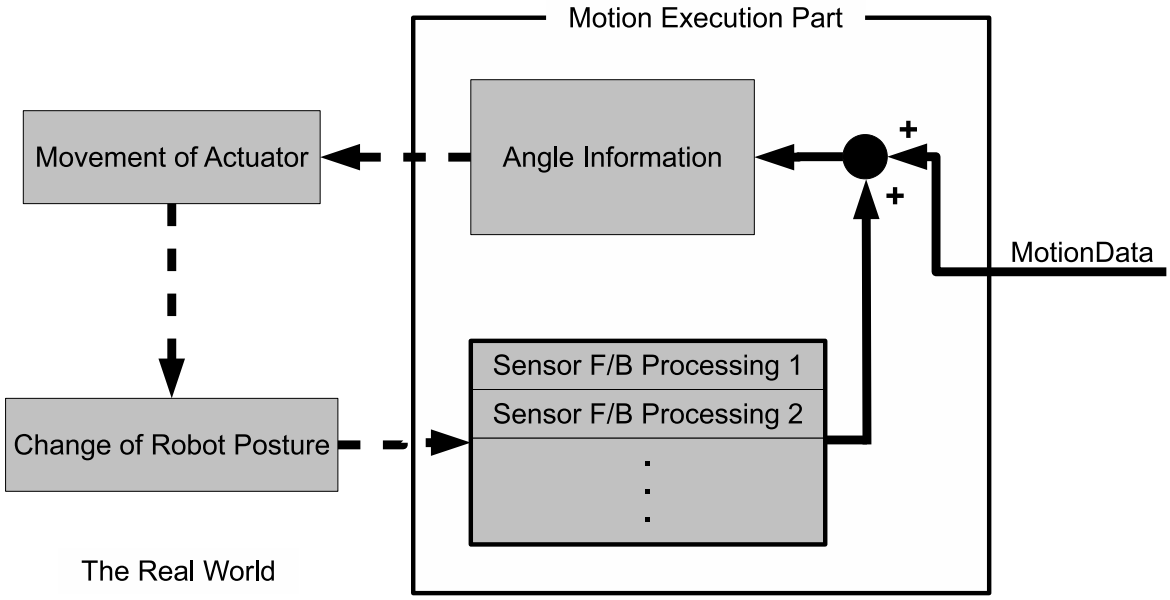


図 2.8: モーションの実行部分とセンサフィードバック

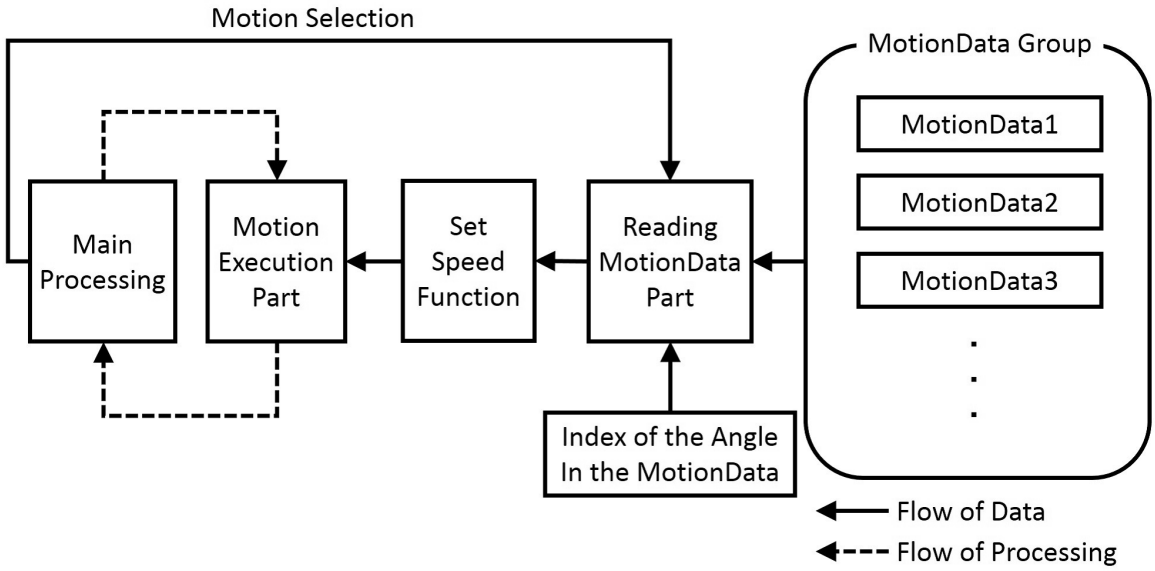


図 2.9: 多関節ロボットのモーション実行用ソフトウェアアーキテクチャの機能拡張

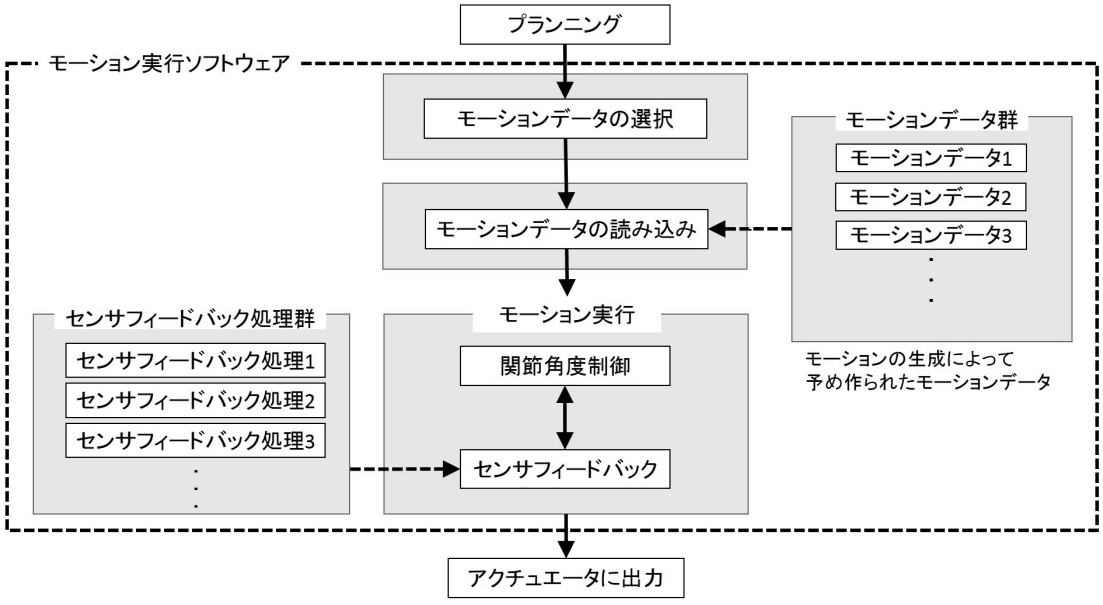


図 2.10: モーション実行用ソフトウェアの概要図

第3章

多関節ロボットのモーション実行用ソフトウェアアーキテクチャの提案

本章では，2.4.4 節で述べた提案するモーション実行用ソフトウェアアーキテクチャの概要をより詳細に述べることで，改めて多関節ロボットのモーション実行用ソフトウェアアーキテクチャを提案する．また，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいてモーション実行システムを構築可能であることを示し，2.4.5 節で述べた提案するアーキテクチャによるモーション実行用ソフトウェア開発における利点について見解を述べる．なお，本章で使用する図は，章末 p.43 よりまとめて記載する．

3.1 多関節ロボットのモーション実行用ソフトウェアアーキテクチャ

本節では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャについて述べる．

提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，多関節ロボットのモーション実行用ソフトウェアの設計指針となるアクチュエータ制御システムの基本構造である．図 3.1 に提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャを示す．提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，2.4.2 節で述べた保証されたモーションを実行するために必要な要素であるモーションデータ，モーションデータの選択機能，モーションの実行機能に加えて，モーションデータの使用方法を考慮できる要素としてのモーションデータの読み込み機能を追加したものである．また，モーションの実行機能には，モーショ

ン実行に関する下位レベルでのセンサフィードバックが可能な仕組みを搭載している．以下，3.1.1 項ではモーションデータについて，3.1.2 項ではモーション実行機能，3.1.3 項ではモーションデータの選択機能，3.1.4 項ではモーションデータの読み込み機能について説明する．なお，モーション実行に関する下位レベルでのセンサフィードバックが可能な仕組みについては次節で述べる．

3.1.1　モーションデータ

提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャには，量産された多関節ロボットにおいて保証されたモーションを実行できる仕組みが必要である．そのため，モーションデータは事前にモーション生成によってモーションデータを作成し，それらを選択・実行するメモリ・ベースト制御を採用した．事前に作成されたモーションデータは，図 3.1 のモーションデータ群 (MotionData Part) として保持する．

モーションの実行には軌道が重要であることを 2.2 節で述べた．そのため，モーションデータの形式はモーションの生成手法に依存するが，モーションデータは関節角度と移動時間の組み合わせであればよい．たとえば，マニピュレータのエンドエフェクタを目標位置に到達させたときの各関節の角度と移動時間の組み合わせや，初期位置での各関節の角度と移動時間の組み合わせである．

3.1.2　モーション実行機能

モーションの実行機能は，図 3.1 中のモーション実行部 (Motion Execution Part) が役割を担う．具体的な役割は，関節角度制御とモーション実行に関するセンサフィードバックである．図 3.1 のモーションデータ群 (MotionData Part) に保持されているモーションデータに対して，どのモーションデータであっても一つのモーション実行部 (Motion Execution Part) でモーションを実行する．そのため，ソフトウェア作成時はモーションデータの構造や形式に基づいて，モーション実行部 (Motion Execution Part) を記述する必要がある．また，モーション実行部 (Motion Execution Part) においてモーションを実行するということは，保持しているモーションデータをモーションとして具現化することである．つまり，アクチュエータ制御においての出力部分にあたる．そのため，ソフトウェア作成時には使用するアクチュエータの情報に基づい

て，アクチュエータに信号を入力するための記述が必要である．モーション実行に関するセンサフィードバックを可能にする仕組みについては 3.2 節で述べる．

3.1.3 モーションデータの選択機能

多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，多関節ロボットのアクチュエータ制御システムのアーキテクチャである．アクチュエータ制御はロボット制御において下位レベルの制御であり，アクチュエータの制御指令はプランニングからの出力結果である．多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いたアクチュエータ制御システムでは，プランニングからの出力結果は使用するモーションデータの指定であり，アクチュエータ制御システムへ入力される．図 3.1 中では，main 処理 (Main Processing) がシステムの入力部分にあたり，モーションデータ群 (MotionData Part) よりモーションデータが選択され，モーション実行部 (Motion Execution Part) においてモーションが実行される．

3.1.4 モーションデータの読み込み機能

モーションデータの読み込み機能は，図 3.1 中のモーションデータ読み込み部 (Reading MotionData Part) が役割を担う．メモリ・ベースト制御を利用してモーションを選択・実行するのみであれば，図 3.1 中のモーションデータ読み込み部 (Reading MotionData Part) は必要ない．しかし，メモリ・ベースト制御を利用して選択・実行できるモーションは，保持しているモーションの数のみである．つまり，メモリ・ベースト制御を利用している以上，選択・実行に必要なモーションは全て保持しなければならない．そのため，多くのモーションを選択・実行するためには，必然的にモーションデータの数が膨大になる．そこで，保持しているモーションデータの使用方法を考慮できる機能として，保証されたモーションを実行するために必要な要素に加えて，モーションデータの読み込み機能を多関節ロボットのモーション実行用ソフトウェアアーキテクチャに追加した．たとえば，保持している複数のモーションデータを用いてモーションを合成することで，保持しているモーション数が少なくても多くのモーションを実行することが可能になる．ただし，保持している保証されたモーションのモーションデータの合成によって得られるモーションは，別途モーションの評価が必要である．モーションの合成によって実行可能となるモーション数については 3.5 節で述べる．

3.2　センサフィードバック

本節では，モーション実行に関するセンサフィードバックの必要性と，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャにおけるセンサフィードバックについて述べる．

モーションの実行は，アクチュエータを制御することによって行われる．アクチュエータ制御はロボット制御においては下位レベルの制御であり，アクチュエータへの制御指令はプランニングによって与えられる．ロボット制御全体としての流れは，第 1 章の図 1.2 の通りであり，モーションの実行中はセンサ情報を用いた認識結果に基づいたプランニングの結果が更新されない限り，モーションの切り替えや修正を行うことができない．つまり，モーション実行中の接触による緊急停止や，別のモーションへの切り替え，モーションの修正による姿勢制御等が実時間処理で行えないということである．しかし，多関節ロボットのモーションの安全性を考慮するならば，モーション実行に関するセンサフィードバックは下位レベルで行う必要がある．また，モーションの保証という観点では，アクチュエータ制御システムとしての自己診断機能によってモーション実行の可否を行う場合が考えられる．自己診断機能は，センサ情報に基づいてシステムの異常を検知するものである．アクチュエータ制御システムにおいては，アクチュエータの異常の検知やその他の要因によるシステムの異常を検知することでモーション実行の可否を行い，モーションを保証する必要がある．アクチュエータの異常やアクチュエータ制御システムの異常が発生した場合はモーションが保証されないため，モーション実行に関するセンサフィードバックは下位レベルで行う必要があるといえる．

そこで，モーション実行に関するセンサフィードバックの仕組みとして，図 3.1 中のモーション実行部 (Motion Execution Part) 内に，図 3.2 のように搭載した．モーションに加えるセンサフィードバック処理は，複数のセンサフィードバック処理に対応できるようにするために，モーション実行部 (Motion Execution Part) の内部にセンサフィードバック処理群として保持する構造とした．そのため，どのモーションに対しても統一的にセンサフィードバックを加えることが可能である．この構造は，センサフィードバック処理がメモリ・ベースト制御から独立しているため，モーションデータはセンサフィードバックに関する情報を持たない．つまり，メモリ・ベースト制御による保証されたモーションの実行が可能でありながら，センサフィードバック処理

はモーションに対して一つあるいは複数のセンサフィードバック処理を追加していくことが可能となる．たとえば，センサフィードバックによるモーションの切り替えや，微小角度変化によるモーションの修正等のユーザが任意に加えた処理を，実行したいモーションのモーションデータと関連付けすることで実行できる．モーションに対して加えるセンサフィードバック処理は，ユーザが任意に加えた処理であるため，ロボットごとの作業目的に応じたセンサフィードバック処理を加えることにより，環境に対応したモーションを実行することが可能である．また，このセンサフィードバック処理を行う仕組みは，メモリ・ベースト制御から独立したものであるため，プログラムが煩雑になることの軽減や省メモリ化に貢献できる．たとえば，図 3.3 のように，メモリ・ベースト制御において軌道の関数としてモーションを保持しているときに，実行したい全てのモーションに対して同じセンサフィードバック処理を加えることを考える．この場合，軌道の関数として保持している各モーションの各関節の角度情報に対して詳細なセンサフィードバック処理を記述する必要がある．そのため，モーションの数だけセンサフィードバック処理の記述が必要になり，ソフトウェアが煩雑になることが考えられる．さらに，メモリ効率においても良いとはいえない．提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャでは，これらのことを軽減することが可能である．

3.3 多関節ロボットのモーション実行システムの試作

本節では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチに基づいた多関節ロボットのモーション実行システムの試作について述べる．多関節ロボットのモーション実行システムの試作は，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャを用いることで，多関節ロボットのモーション実行システムの構築が可能であることを示すために行った．試作として構築した多関節ロボットのモーション実行システムは，モーション実行用ソフトウェアとしての多関節ロボットのサーボモータ制御プラットフォームであり，その概略図として図 2.10 を図 3.4 として改めて掲載する．以下，多関節ロボットのサーボモータ制御プラットフォームにおけるモーションデータの構造について 3.3.1 項で述べ，モーション実行機能においてのモーション実行方法（関節角度制御）について 3.3.2 項で述べる．モーションの読み込み機能においては，3.1.4 項でモーションデータの使用方法の例として挙げたモー

ションの合成方法について 3.3.3 項で述べる．なお，モーションデータの選択機能については，実行したいモーションデータを選択するのみであるため割愛する．

3.3.1　モーションデータの構造

モーションデータの構造を図 3.5 に示す．多関節ロボットのサーボモータ制御プラットフォームのモーションデータは，各関節 (Motor 1, Motor 2, ⋯ , Motor m) においてのモーションのキーとなる点の角度情報と移動時間のサイクル数を保持する．モーションのキーとなる点の角度情報について述べる．モーションのキーとなる角度情報とは，アニメーションのキーフレーム法のように動きのキーフレームにおいての関節角度である．例えば，必要となる角度情報は，始点角度 (Angle 1)，中間点角度 (Angle 2, Angle 3, ⋯ , Angle n-1)，終点角度 (Angle n) である．そして，それぞれの角度情報には移動時間のサイクル数（Movement Cycle）を付加しており，一つのモーションデータは図 3.5 のように管理される．

3.3.2　モーション実行機能においてのモーション実行方法

モーションの実行機能においてのモーションの実行方法（関節角度制御）について述べる．モーションの実行方法は，モーションデータに記述されているモーションのキーとなる点と点の角度の差分を移動時間のサイクル数によって細分化し，細分化された角度を順次実行することでモーションを実行する．例として，キーとなる点が始点角度と終点角度の 2 点の場合において，モーションの実行方法について述べる．この場合，モーションデータは始点角度として図 3.6(a)，終点角度として図 3.6(b) の角度を移動時間のサイクル数とともに保持している．モーションを実行する際には，始点角度 (図 3.6(a)) と終点角度 (図 3.6(b)) の差分を図 3.6(c) のように移動時間のサイクル数によって細分化し，細分化された角度を順次サーボモータに信号を入力することでモーションを実行する．サーボーモータに信号を入力する手段は，使用するサーボモータの情報に基づいて記述を行った．

3.3.3 モーションデータの読み込み例としてのモーション合成

モーションデータの使用方法を考慮できる機能として搭載したモーションデータの読み込み機能において，モーションデータの読み込み例としてのモーション合成の方法について述べる．モーションの合成は，合成したいモーションのモーションデータを複数個選択し，モーションデータの角度情報および移動時間のサイクル数を図 3.7 のように足し合わせることで行う．具体的には，それぞれのモーションが保持する各関節の中から同じ関節の角度情報を足し合わせ，その角度情報を用いてモーションを実行する．この際，各関節で足し合わせたモーション数で，足し合わせた角度を割った値を角度情報として使用する．合成によって生成されたモーションは，一つのモーションとして管理することが可能となるようにした．

3.4 動作実験 1

本節では，多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームにおいて，形態の異なる多関節ロボットでのモーション実行が可能であることを示す．また，モーションを保証するためのモーションの評価について見解を述べる．

実験に用いたロボットは，2 脚ロボットとアームロボットである．2 脚ロボットとアームロボットの構成を図 3.8 に示す．2 脚ロボットは，PWM 方式のサーボモータ 12 個によって構成されている．アームロボットは，アーム部分とハンド部分で通信方式の異なるサーボモータを使用してる．アーム部分は PWM 方式のサーボモータ 4 個で構成されており，ハンド部分はシリアルサーボモータ 2 個で構成されている．PWM 方式のサーボモータとシリアルサーボモータへの角度情報を送出する信号の変換は，3.1.2 項および 3.3.2 項で述べたように，多関節ロボットのモーション実行用ソフトウェアアーキテクチャのモーション実行機能内で行っている．このため，扱う信号の異なるサーボモータが混在している場合においてもモーションの実行が可能である．制御 CPU はいずれも株式会社ルネサステクノロジ製の SH7145F である．なお，3.5 節および 3.6 節の動作実験においても，本節の動作実験で用いた 2 脚ロボットを用いる．3.6 節の動作実験では，2 脚ロボットの足裏に搭載した感圧ゴムを用いた圧力センサからの値によっ

てセンサフィードバック処理を行う．

3.4.1　2脚ロボットでのモーション実行

本実験では，2脚ロボットでのモーション実行が可能であることを示す．本実験において使用するモーションデータは，始点角度と終点角度のみで構成される．その内容は，始点角度として足首以外を伸ばした状態の各サーボモータの角度情報，終点角度として膝関節と腿関節を曲げた状態の各サーボモータの角度情報である．

図 3.9 は，2脚ロボットを用いて片脚屈伸モーションを実行した結果である．(a) のスタート位置から (b) (c) (d) (e) へと脚を曲げていき，(e) から (f) (g) (h) (i) へと脚を伸ばしていく様子が分かる．このことから，多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームにおいて，2脚ロボットでのモーション実行が可能であることを確認できた．

3.4.2　アームロボットでのモーション実行

本実験では，アームロボットでのモーション実行が可能であることを示す．また，前項の実験結果と本実験の結果から，モーションを保証するためのモーションの評価について見解を述べる．本実験において使用するモーションデータの内容は次の通りである．始点角度は，ハンドを広げアームを下ろした状態の各サーボモータの角度情報である．中間点角度は，中間点角度 1，中間点角度 2，中間点角度 3 を用意した．中間点角度 1 は，ハンドを閉じアームを上げた状態の各サーボモータの角度情報である．中間点角度 2 は，アームを下ろしハンドを閉じた状態の各サーボモータの角度情報である．中間点角度 3 は，ハンドを広げアームを上げた状態の各サーボモータの角度情報である．終点角度は，ハンドを広げアームを下ろした状態の各サーボモータの角度情報である．

図 3.10 は，アームロボットを用いてモーションを実行した結果である．(a) のスタート位置から (b) (c) へとアームを持ち上げながらハンドを閉じている様子が分かる．続く (d) (e) では，アームを下ろしながらハンドを閉じている様子が分かる．さらに，続く (f) (g) では，アームを持ち上げながらハンドを開いていき，(h) (i) ではアームを下ろしながらハンドを開いている様子が分かる．このことから，多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいた多関節ロボットのサーボ

モータ制御プラットフォームにおいて，アームロボットでのモーション実行が可能であることを確認できた．

また，前項の 2 脚ロボットでのモーション実行と本項のアームロボットでのモーションの実行においては，モーションデータの選択機能，モーションデータの読み込み機能，モーション実行における関節角度制御に変更はなく，モーションデータの変更のみでモーションを実行している．また，実験で使用した 2 脚ロボットやアームロボットのように，形式の異なるサーボモータを使用している場合においては，サーボモータに信号を入力する手段を，使用するサーボモータの情報に基づいて記述するのみでモーションを実行可能としている．以上のことは，モーションデータの変更のみで，機種や形態の異なる多関節ロボットにおいて同一のモーション実行用ソフトウェアによるモーションの実行が可能であることを示している．よって，モーションを保証するための評価はモーションの変更点に着目した評価となり，モーションデータの追加や修正等の変更のみでモーションの評価を行えるため，評価が繰り返し行いやすいといえる．また，共通のソフトウェア構造を用いてモーションの実行を可能にしているため，モーションの保証に対しての信頼性が確保されるといえる．

3.5 動作実験 2

本実験では，多関節ロボットのモーション実行用ソフトウェアアーキテクチャにモーションデータの使用方法を考慮できる機能として搭載したモーションデータの読み込み機能において，モーションデータの読み込み例としてのモーション合成が正しく機能していることを確認する．また，モーションを保証するためのモーションの評価について見解を述べる．実験に使用するロボットは，3.4 節の 2 脚ロボットと同じである．

モーションの合成に使用するモーションは，図 3.9 に示したモーションと図 3.11 に示すモーションである．図 3.11 に示すモーションのモーションデータの内容は，始点角度として足首以外を伸ばした状態の各サーボモータの角度情報，終点情報として腿関節を外に広げた状態の各サーボモータの角度情報である．そのため，実行することで脚全体を開閉するモーションが行われる．

図 3.12 は，図 3.9 の右脚屈伸モーションと，図 3.11 の右脚の開閉モーションを合成した結果である．(a) のスタート位置から (b) (c) (d) (e) へと右脚を外側へ広げながら右脚を曲げていき，(e) から (f) (g) (h) (i) へと右脚を内側に戻しながら右

脚を伸ばしていることから，二つのモーションが合成され実行されていることがわかる．このことから，モーションの合成が正しく機能していることを確認できた．また，モーションの評価については，モーションの評価によって保証されたモーションを用いた合成であることを前提とすれば，合成によって作られるモーションが保証されたモーションからの変更点である．そのため，モーションの合成によって所望のモーションが実現可能であることに着目したモーションの評価のみでよい．

モーションの合成が可能であることから，合成によって実行可能となるモーション数について次のことが言える．モーションデータとして保持しているモーションの数を n，n 個のモーションの中からモーションの合成に使用するモーションの数を m とすると，生成できるモーションは

$$ {}_nC_m \qquad (n \geqq 2) \tag{3.1} $$

である．そのため，生成できるモーションの総数は，

$$ \sum_{m=2}^{n} ({}_nC_m) \qquad (n \geqq 2) \tag{3.2} $$

であり，実行できるモーションの総数は，

$$ \sum_{m=2}^{n} ({}_nC_m) + n \quad (n \geqq 2) \tag{3.3} $$

である．図 3.3 のようにメモリ・ベースト制御において軌道の関数としてモーションを保持しているとき，実行できるモーションは保持しているモーションの数に依存する．一方で，式 (3.2) と式 (3.3) より，モーションデータの使用方法を考慮しモーションの合成を行うことで，多くのモーションを実行することが可能であることが分かる．図 3.13 は，純粋に実行できるモーションの数を比較したものである．保持しているモーションが簡単なものであるか複雑なものであるかは関係ない．この結果は保持しているモーションの数が 0〜5 個の場合において，モーションの合成が可能な場合に実行できるモーションの数と，図 3.3 のようにメモリ・ベースト制御において軌道の関数としてモーションを保持しているときに実行できるモーションの数を比較したことによる．この結果から，保持している n 個のモーションのみではなく，$\sum_{m=2}^{n}({}_n\mathrm{C}_m) + n$ 個 ($n \geqq 2$) のモーションが実行可能なことが分かる．このことから，モーションデータの使用方法を充実させることにより，モーションデータ数が少なくても多くのモーショ

ンを実行できることが分かる．また，モーションデータの読み込み機能によってモーションデータを組み合わせることで，実行できるモーション数を増やすことが可能であることを示唆している．

3.6 動作実験 3

本実験では，メモリ・ベースト制御から独立したセンサフィードバック処理が正しく機能することを確認する．また，モーションの保証について見解を述べる．センサフィードバック処理の例として以下のセンサフィードバック処理をセンサフィードバック処理群として保持している．なお，モーションデータはセンサフィードバックに関する情報を持たない．

(1) モーションの切り替え
(2) 屈伸運動中に押された場合の押し返し

実験に使用するロボットは，3.4 節の 2 脚ロボットと同じである．

3.6.1 センサフィードバック処理の説明

2 脚ロボットの足裏には図 3.14(a) のように圧力センサを設置している．3.6.2 項のモーションの切り替えの実験では，片脚の屈伸運動のモーションと脚を上げるのみのモーションを保持しており，片脚の屈伸運動中にセンサ f0 の値が変化した場合に脚を上げるのみのモーションに切り替えを行う．片脚の屈伸運動のモーションと脚を上げるのみのモーションでは移動時間のサイクル数が異なる．脚を上げるのみのモーションの移動時間のサイクル数は，片脚の屈伸運動のモーションよりも短く設定している．そのため，片脚の屈伸運動のモーションよりも脚を上げるのみのモーションの方が動きが速い．

また，両足接地時においては，すべてのセンサを使用するのではなく，図 3.14(b) に示すようにセンサ f0，f2，f5，f7 のみを使用している．そこで，3.6.3 項の実験では，以下の式 (3.4)〜式 (3.7) によって各方向のセンサの合計値を計算し，条件 (3.8)〜条件 (3.13) によって重心移動を判断している．これによって，前方向に重心移動した場合は後ろへ，後ろ方向へ重心移動した場合は前へ，左へ重心移動した場合は右へ，右へ重心移動した場合は左へとアクチュエータの微小角度調整によって重心を移動する．アクチュ

エータの微小角度調整を行うことによって前後方向および左右方向に同時に重心移動が行われた場合においても対応することが可能である．

$$front = f0 + f5 \tag{3.4}$$

$$back = f2 + f7 \tag{3.5}$$

$$right = f0 + f2 \tag{3.6}$$

$$left = f5 + f7 \tag{3.7}$$

$$right > left \tag{3.8}$$

$$right < left \tag{3.9}$$

$$right = left \tag{3.10}$$

$$front > back \tag{3.11}$$

$$front < back \tag{3.12}$$

$$front = back \tag{3.13}$$

3.6.2　モーションの切り替え

センサフィードバック処理の例として，モーションの切り替えが可能なことを確認する．図 3.15 は，センサフィードバックによるモーションの切り替えを実行した結果である．各フレームは約 0.3 秒おきにキャプチャしたものである．(a) ～ (e) にかけて脚を上げ，(f) (g) と脚を下ろして (h) でセンサに触れた．これによりセンサの値が変わりモーションの切り替えが行われた結果，(a) ～ (h) までとは異なる速さで脚を上げた．以上より，センサフィードバック処理群として保持しているセンサフィードバック処理のうち，モーションの切り替えが選択・実行可能であることを確認できた．

3.6.3　屈伸運動中に押された場合の押し返し

センサフィードバック処理の例として，屈伸運動中に押された場合の押し返しが実行可能なことを確認する．図 3.16 は，センサフィードバックによる屈伸運動中に押された場合の押し返しを実行した結果である．(a) ～ (c) にかけて 2 脚ロボットを押すことによりセンサの値が変化する．その結果，(d) ～ (f) にかけて 2 脚ロボットが押し

ている様子がわかる．また,(g) ～ (i) にかけて押す力を緩めると，センサの値が変化する．その結果，2脚ロボットがもとの姿勢に戻っていく様子がわかる．センサフィードバック処理を行っていない状態との比較のために，図 3.17 を用意した．図 3.17 では,(a) ～ (c) にかけて2脚ロボットを押しているが,(d) ～ (f) にかけて2脚ロボットが押し返すことなく，押された方向へ姿勢を崩していく様子がわかる．また，2脚ロボットを押し続けることによる転倒を防ぐため (g) ～ (i) にかけては，指で支えることによりもとの姿勢に戻している．以上のことから，センサフィードバック処理群として保持しているセンサフィードバック処理のうち，屈伸運動中に押された場合の押し返しが選択・実行可能なことを確認できた．

また，モーションデータは，前項のモーションの切り替えや本項の屈伸運動中に押された場合の押し返し等のセンサフィードバックに関する情報を持たない．つまり，前項のモーションの切り替えや本項の屈伸運動中に押された場合の押し返しの実験において，モーションデータ，および，モーションデータのみによって実行されるモーションの保証は保たれたまま，センサフィードバック処理を加えることが可能であることを示した．そのため，メモリ・ベースト制御による保証されたモーションの実行が可能でありながら，センサフィードバック処理はモーションに対して追加していくことが可能である．このことから，保証されたモーションからの変更点はセンサフィードバック処理によるものであるといえる．よって，センサフィードバック処理の評価は，保証されたモーションからの変更点であるセンサフィードバック処理の処理内容に着目した評価が可能であるといえる．

3.7 まとめ

本章では，多関節ロボットのモーション実行用ソフトウェアアーキテクチャを提案した．提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，量産された多関節ロボットにおいて保証されたモーションを実行可能とするため，メモリ・ベースト制御を用いている．メモリ・ベースト制御を用いることで，モーションを保証するための評価はモーションの変更点に着目した評価となり，モーションデータの追加や修正等の変更のみでモーションの評価を行えるため，評価が繰り返し行いやすい．また，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，共通のソフトウェア構造を用いて機種や形態の異なる多関節ロボットにおい

て，モーションの実行を可能にしている．そのため，評価が繰り返し行いやすいのみではなく，モーションの保証に対しての信頼性が確保される．メモリ・ベースト制御においてモーションデータ数が膨大になるという問題があるが，モーションデータの読み込み機能によってモーションデータの使用方法を考慮することで新たなモーションを作ることができる．これによって，純粋なメモリ・ベースト制御よりも多くのモーションを実行可能である．また，モーションデータの使用方法を考慮することで作られるモーションは，モーションの評価によって保証されたモーションを用いることを前提とすれば，作られるモーションが所望のモーションであることに着目したモーションの評価のみでよい．さらに，ロボット制御全体のうち下位レベル制御に当たるモーションの実行に関して，メモリ・ベースト制御による保証されたモーションの実行が可能でありながら，センサフィードバック処理を処理群として保持することでセンサフィードバック処理をモーションに対して追加していくことを可能とする仕組みを有している．これによって，センサフィードバック処理の評価は，保証されたモーションからの変更点であるセンサフィードバック処理の処理内容に着目した評価でよい．以上のことを示すために，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいた多関節ロボットのモーション実行システムを試作し動作実験を行った．

提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，多関節ロボットのモーション実行用ソフトウェアの設計指針となるアクチュエータ制御システムの基本構造である．そのため，基本構造を変更することなく機能拡張が行える拡張性を供えていなければならない．次章では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性について述べる．

3.8　第3章で使用する図

本章で使用する図を次頁より示す．

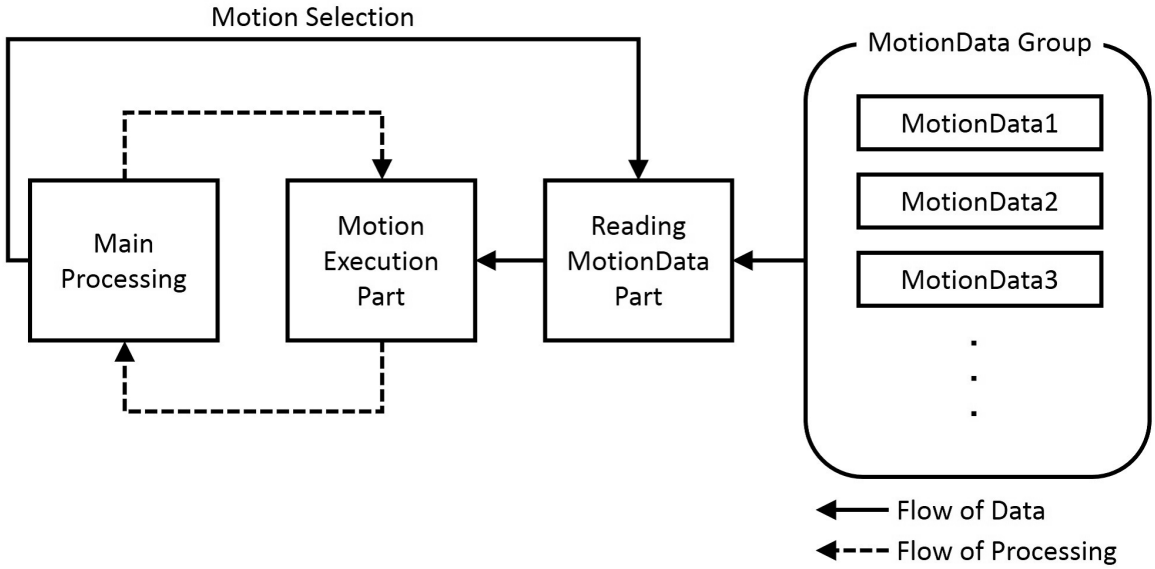


図 3.1: 多関節ロボットのモーション実行用ソフトウェアアーキテクチャ

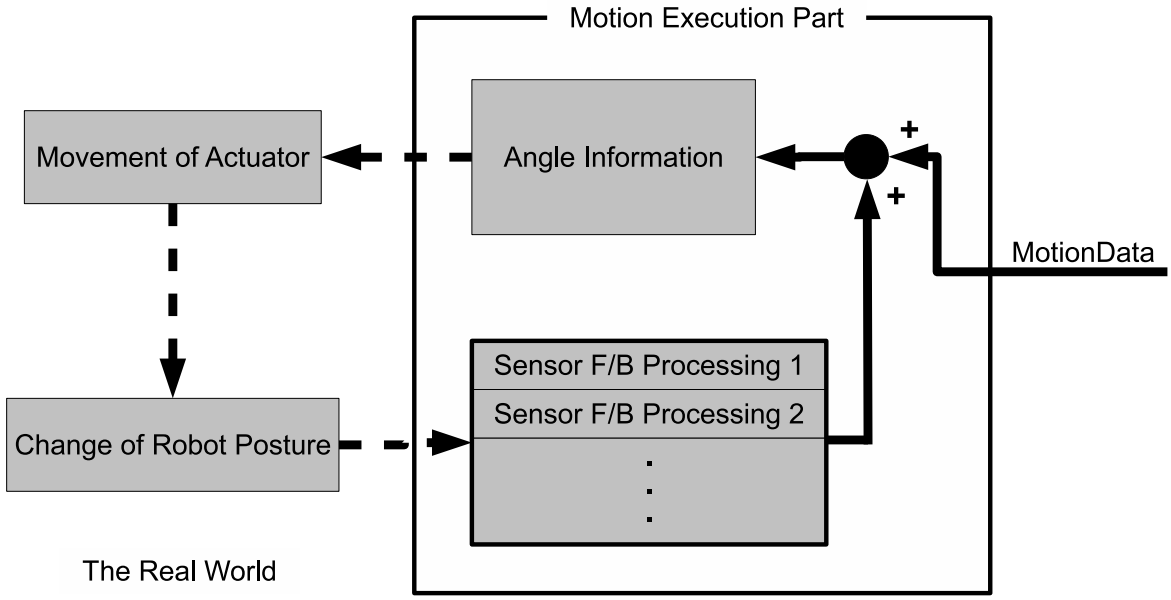


図 3.2: モーションの実行部分とセンサフィードバック

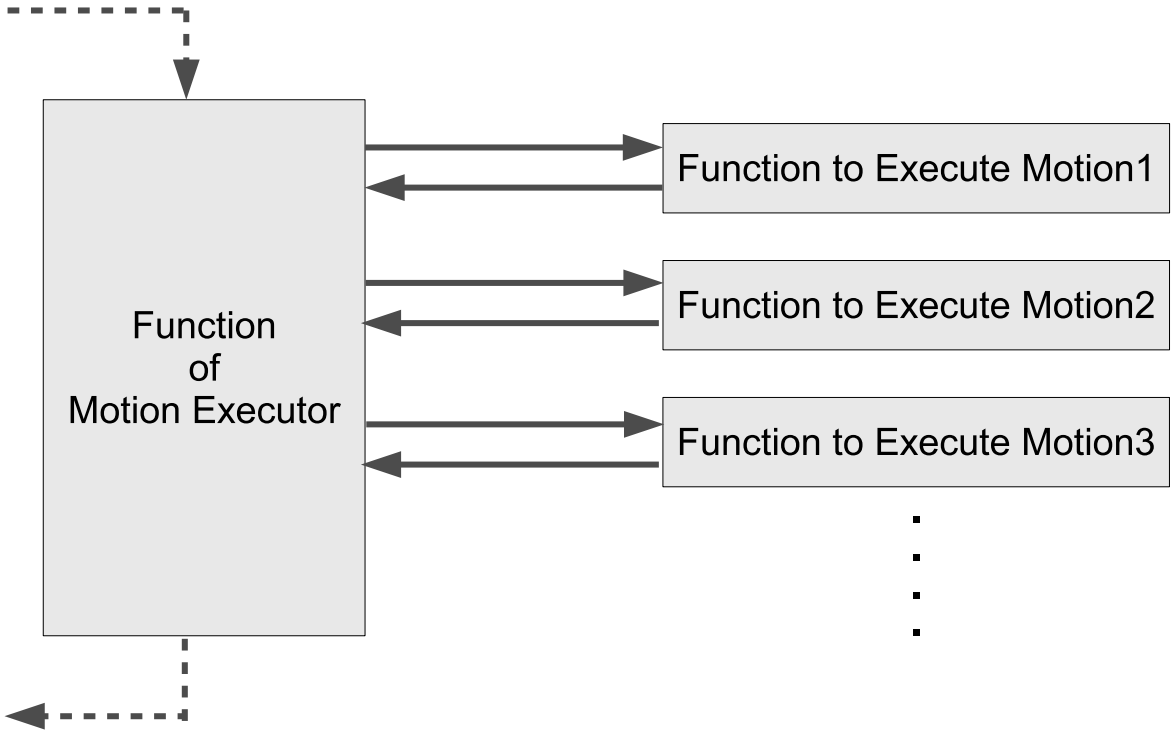


図 3.3: メモリ・ベースト制御

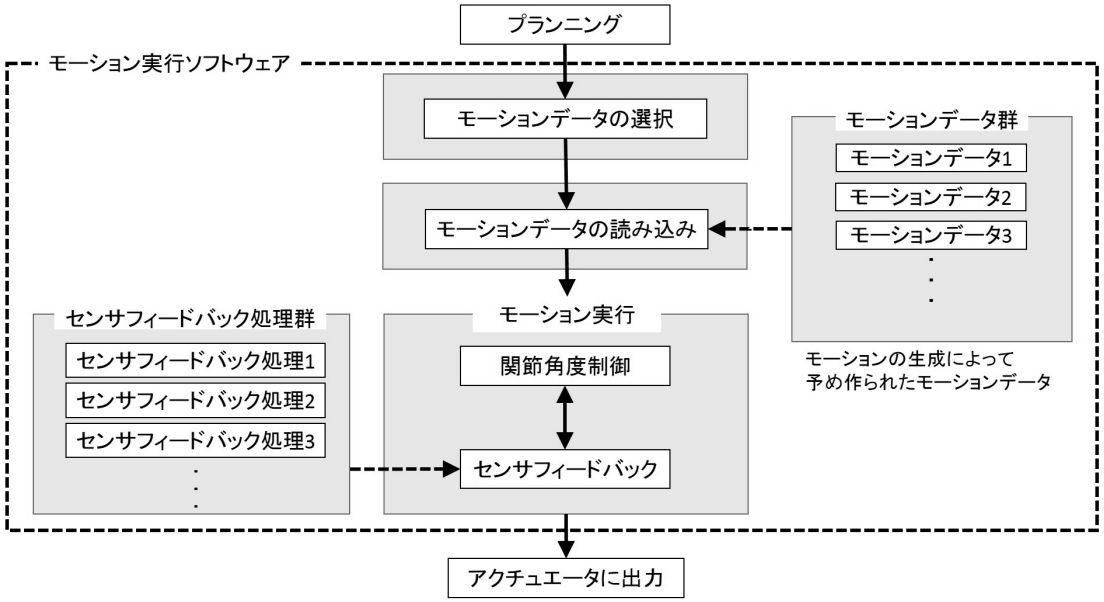


図 3.4: モーション実行用ソフトウェアの概要図

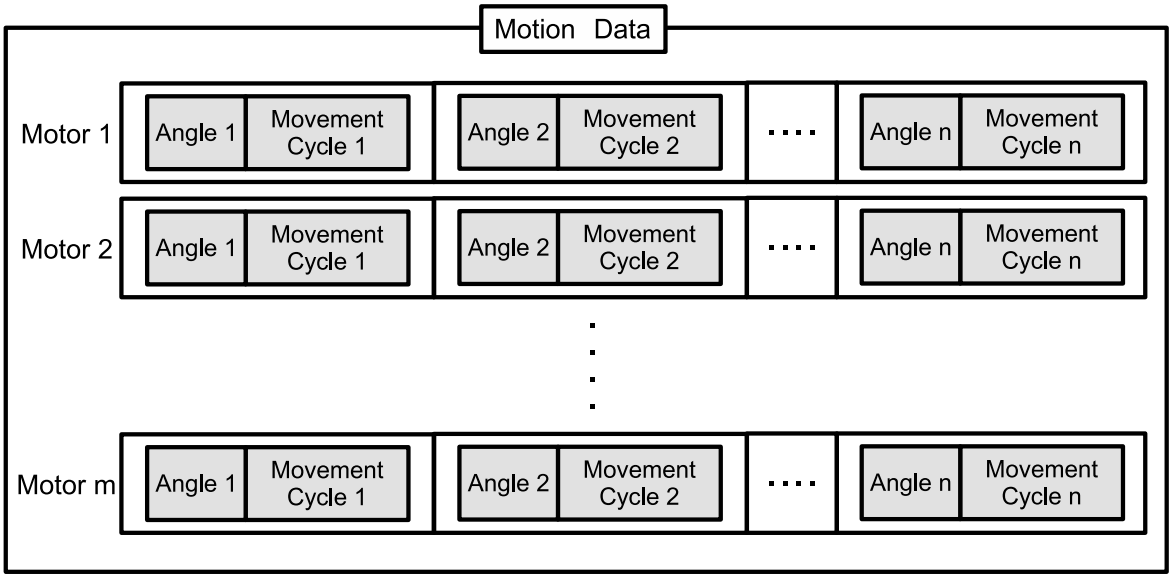
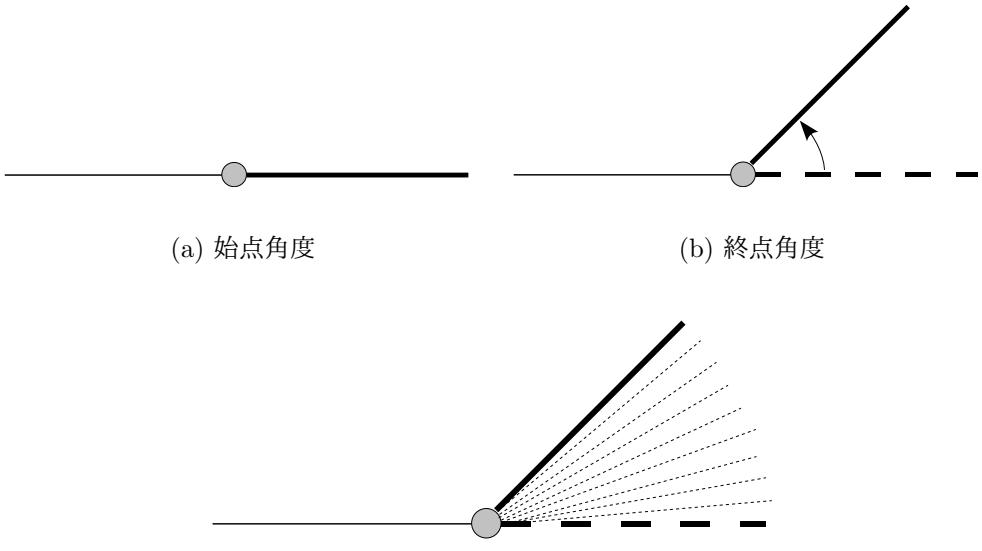


図 3.5: モーションデータの構造



(a) 始点角度

(b) 終点角度

(c) 角度の細分化

図 3.6: モーションの実行方法

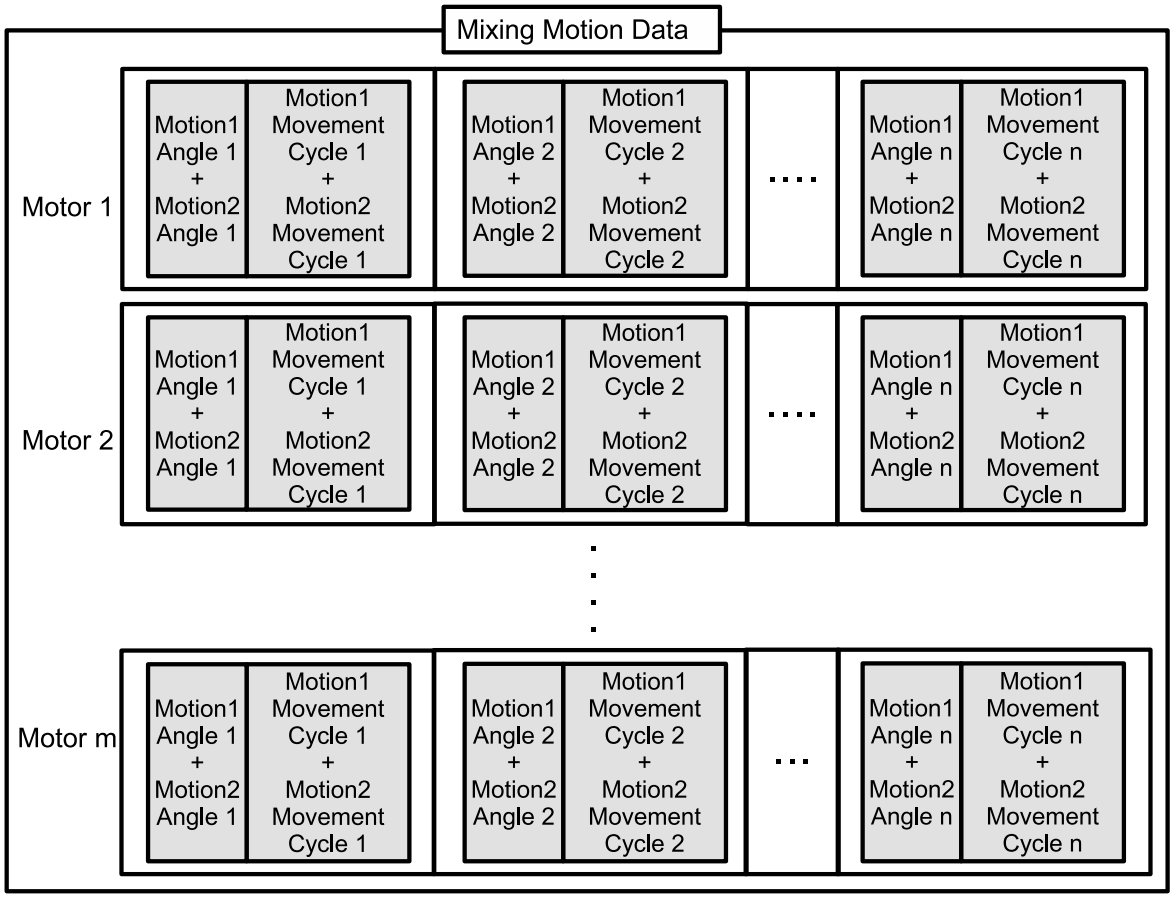
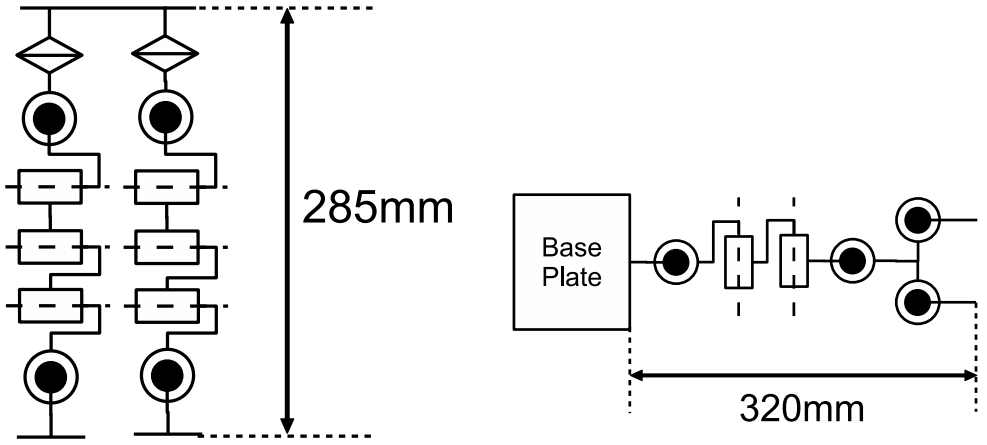
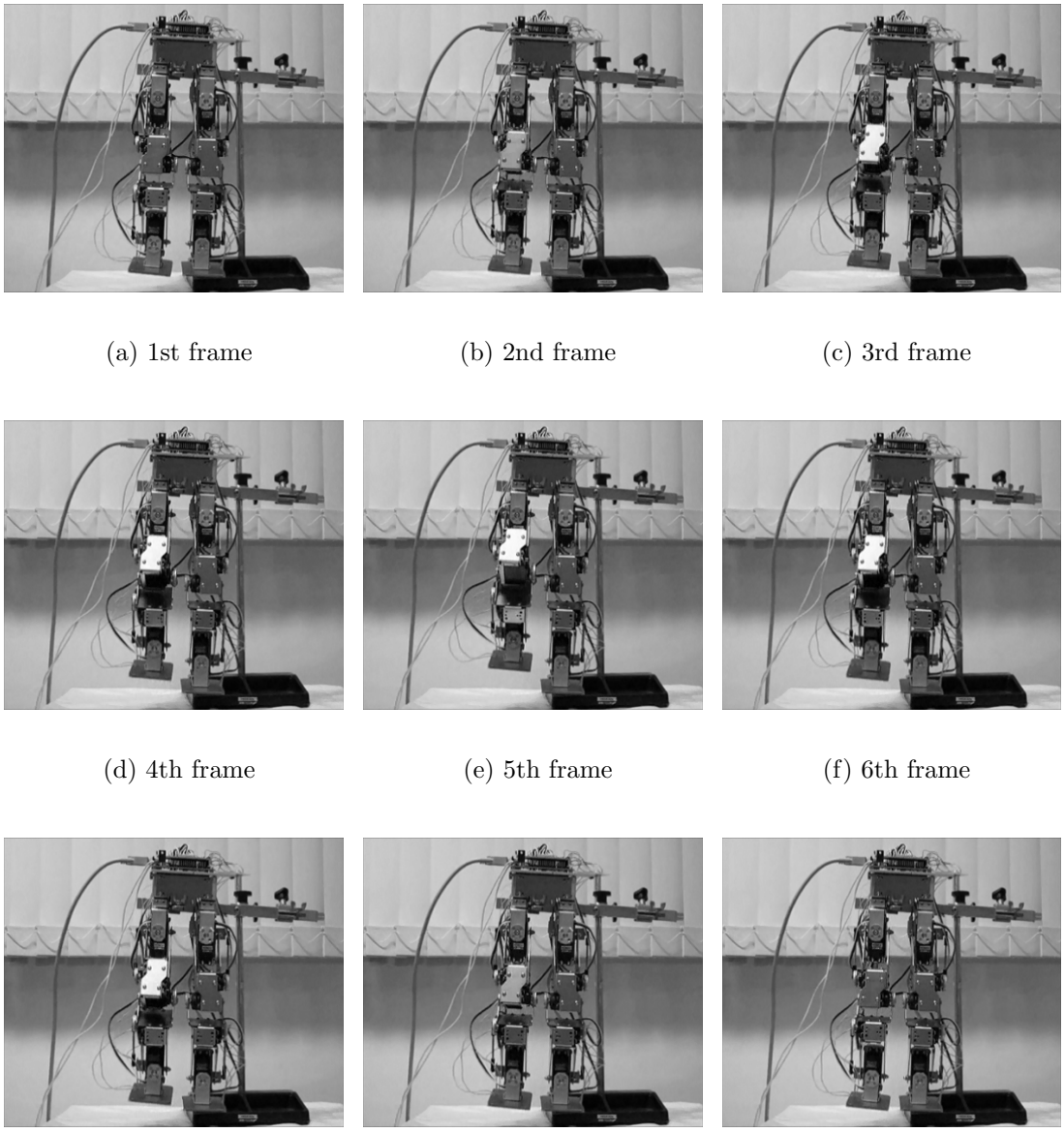


図 3.7: モーションの合成 (motion1 + motion2)



(a) 2 脚ロボット　　(b) アームロボット

図 3.8: 実験に使用したロボットの構成

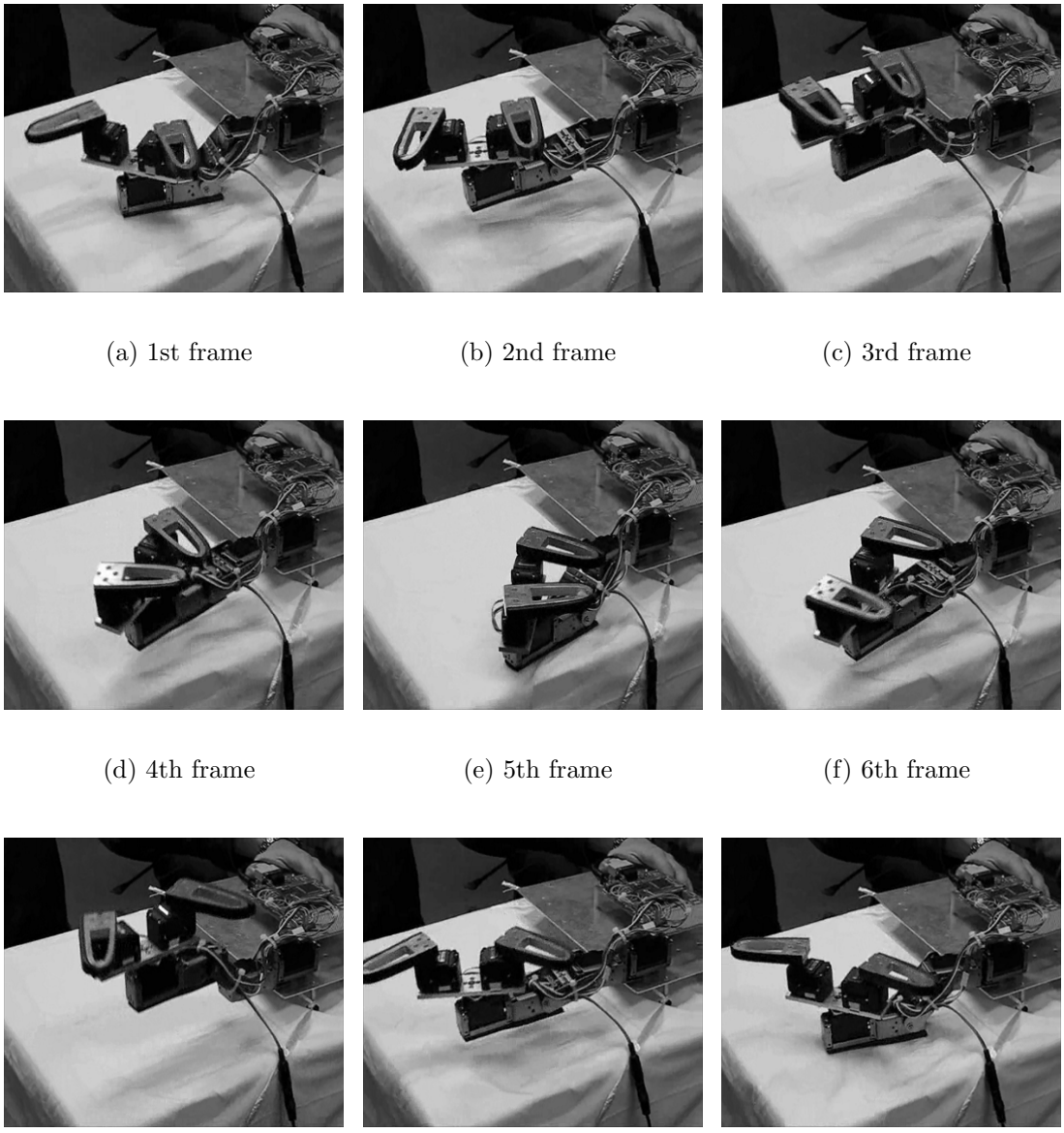


(a) 1st frame (b) 2nd frame (c) 3rd frame

(d) 4th frame (e) 5th frame (f) 6th frame

(g) 7th frame (h) 8th frame (i) 9th frame

図 3.9: 2 脚ロボットの右脚屈伸モーションの実行結果

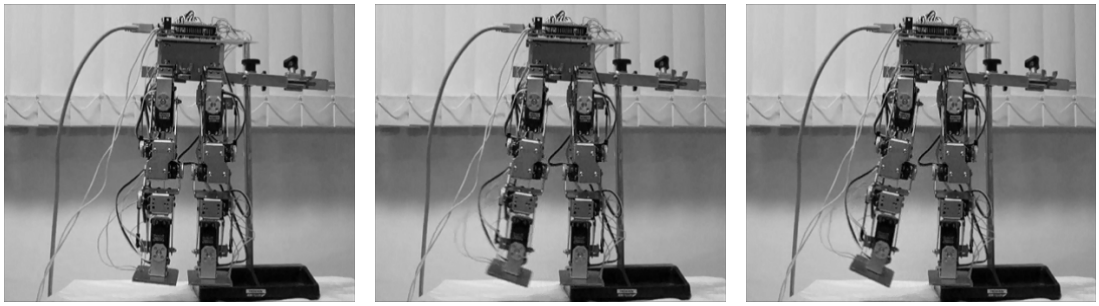


(a) 1st frame　(b) 2nd frame　(c) 3rd frame

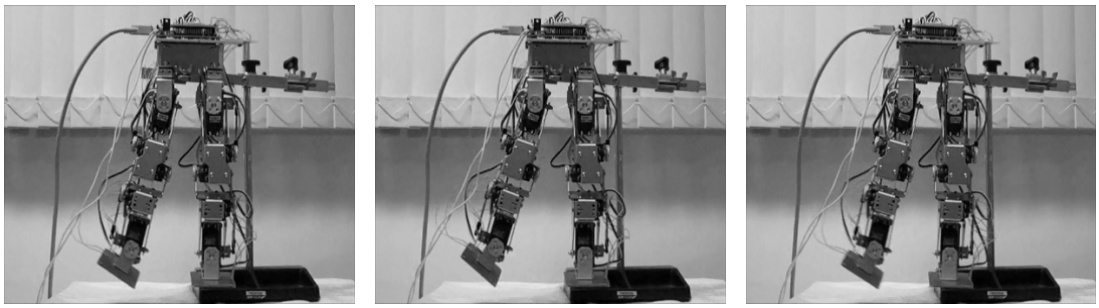
(d) 4th frame　(e) 5th frame　(f) 6th frame

(g) 7th frame　(h) 8th frame　(i) 9th frame

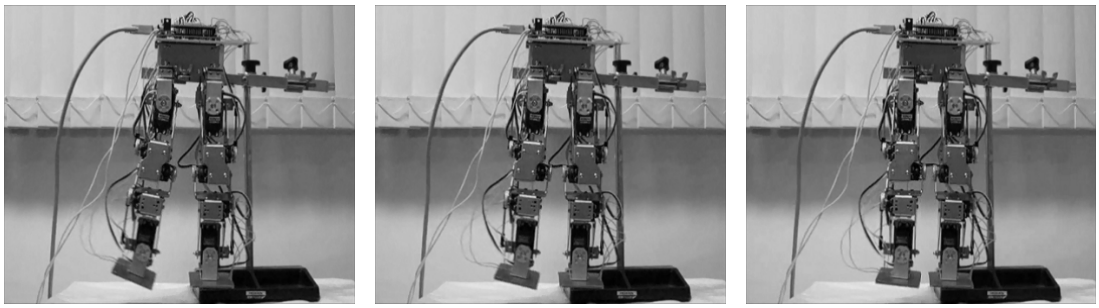
図 3.10: アームロボットの動作結果



(a) 1st frame (b) 2nd frame (c) 3rd frame

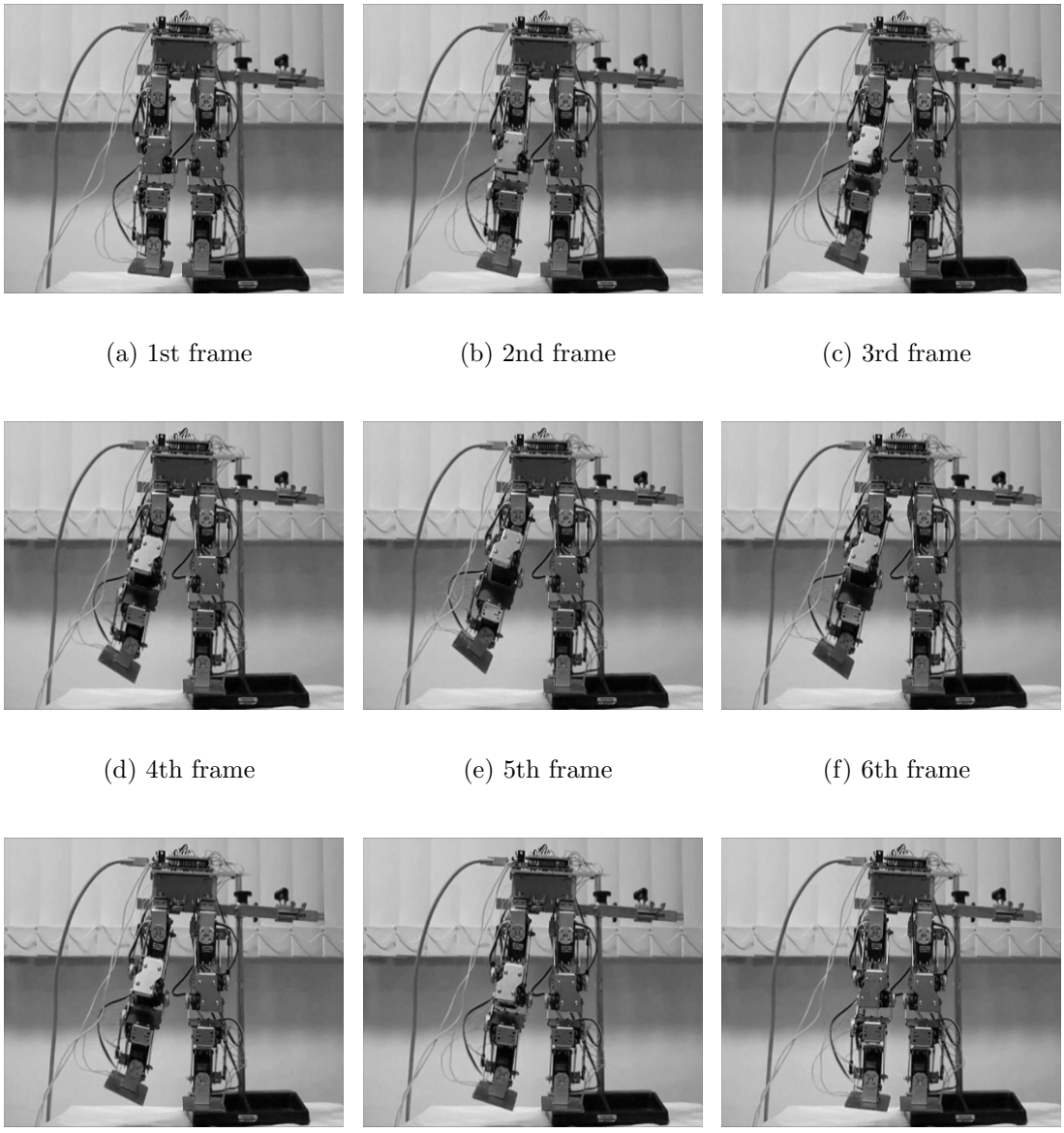


(d) 4th frame (e) 5th frame (f) 6th frame



(g) 7th frame (h) 8th frame (i) 9th frame

図 3.11: 腿関節を外に広げる動作



(a) 1st frame　(b) 2nd frame　(c) 3rd frame

(d) 4th frame　(e) 5th frame　(f) 6th frame

(g) 7th frame　(h) 8th frame　(i) 9th frame

図 3.12: 右脚屈伸 (図 3.9) と腿関節を外に広げる動作 (図 3.11) の合成結果

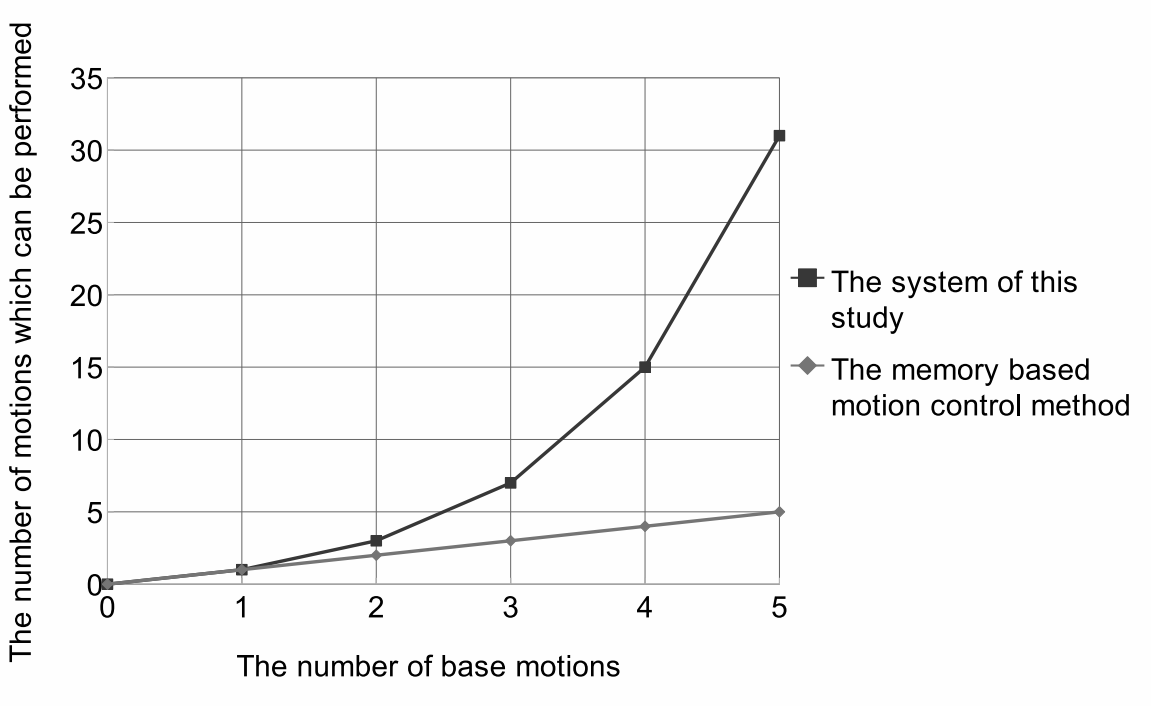
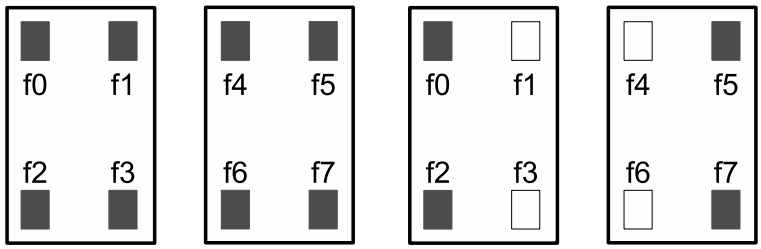


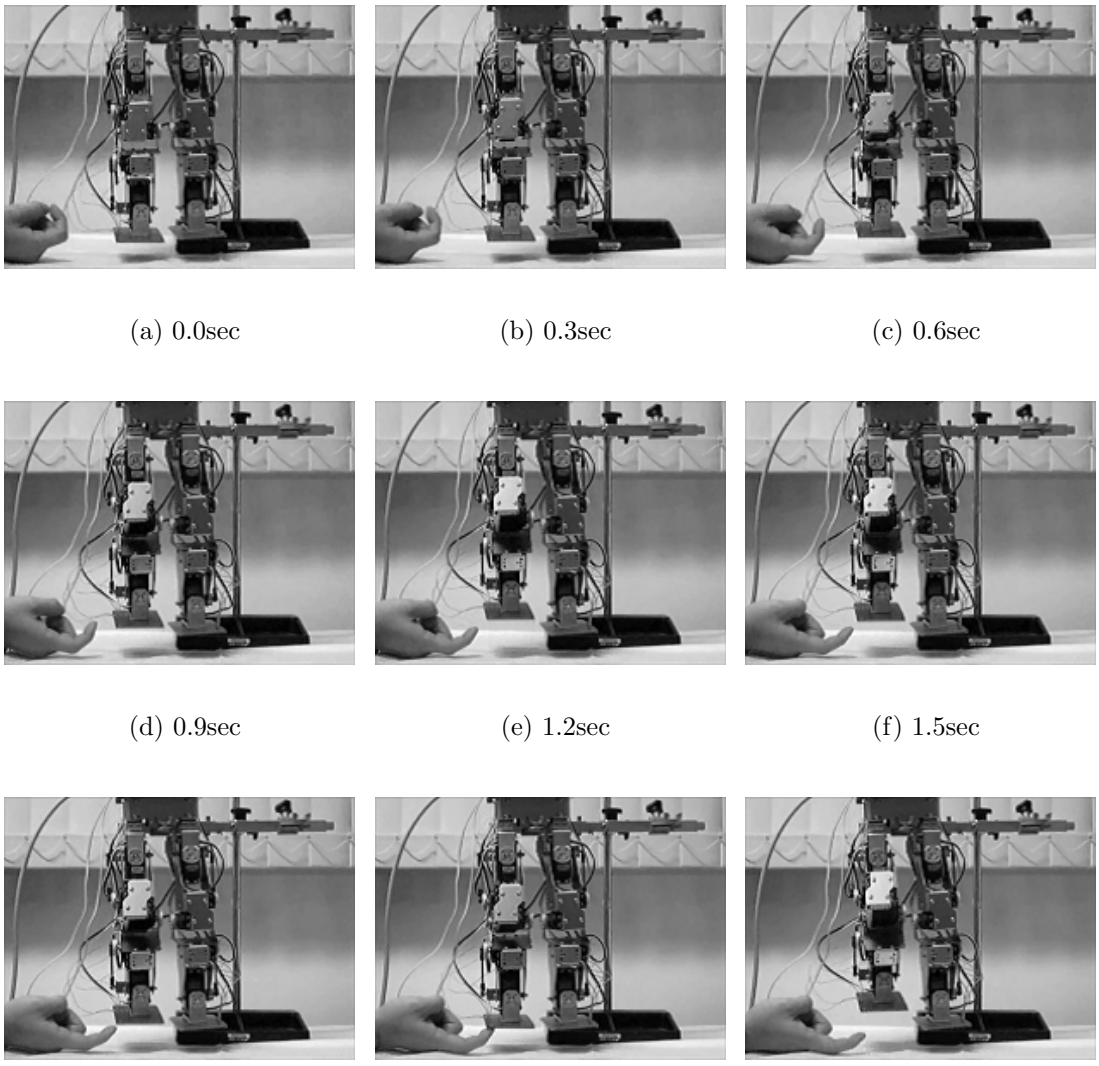
図 3.13: 実行できるモーション数の比較



(a) センサの配置

(b) 両足接地時に利用するセンサ

図 3.14: 2 脚ロボットの足裏に搭載したセンサについて

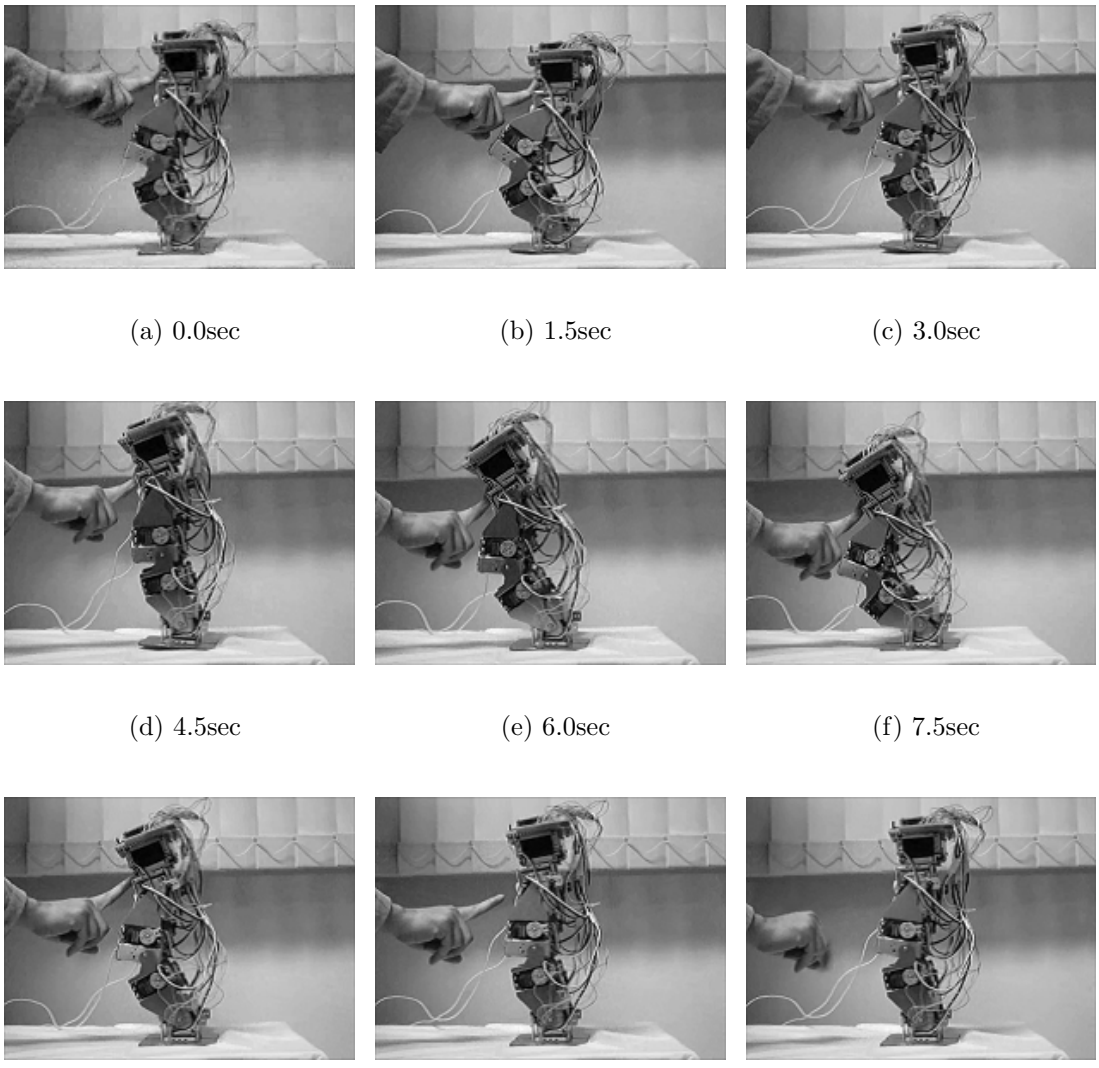


(a) 0.0sec　(b) 0.3sec　(c) 0.6sec

(d) 0.9sec　(e) 1.2sec　(f) 1.5sec

(g) 1.8sec　(h) 2.1sec (touching sensor)　(i) 2.4sec

図 3.15: センサ入力によるモーションの切り替え結果

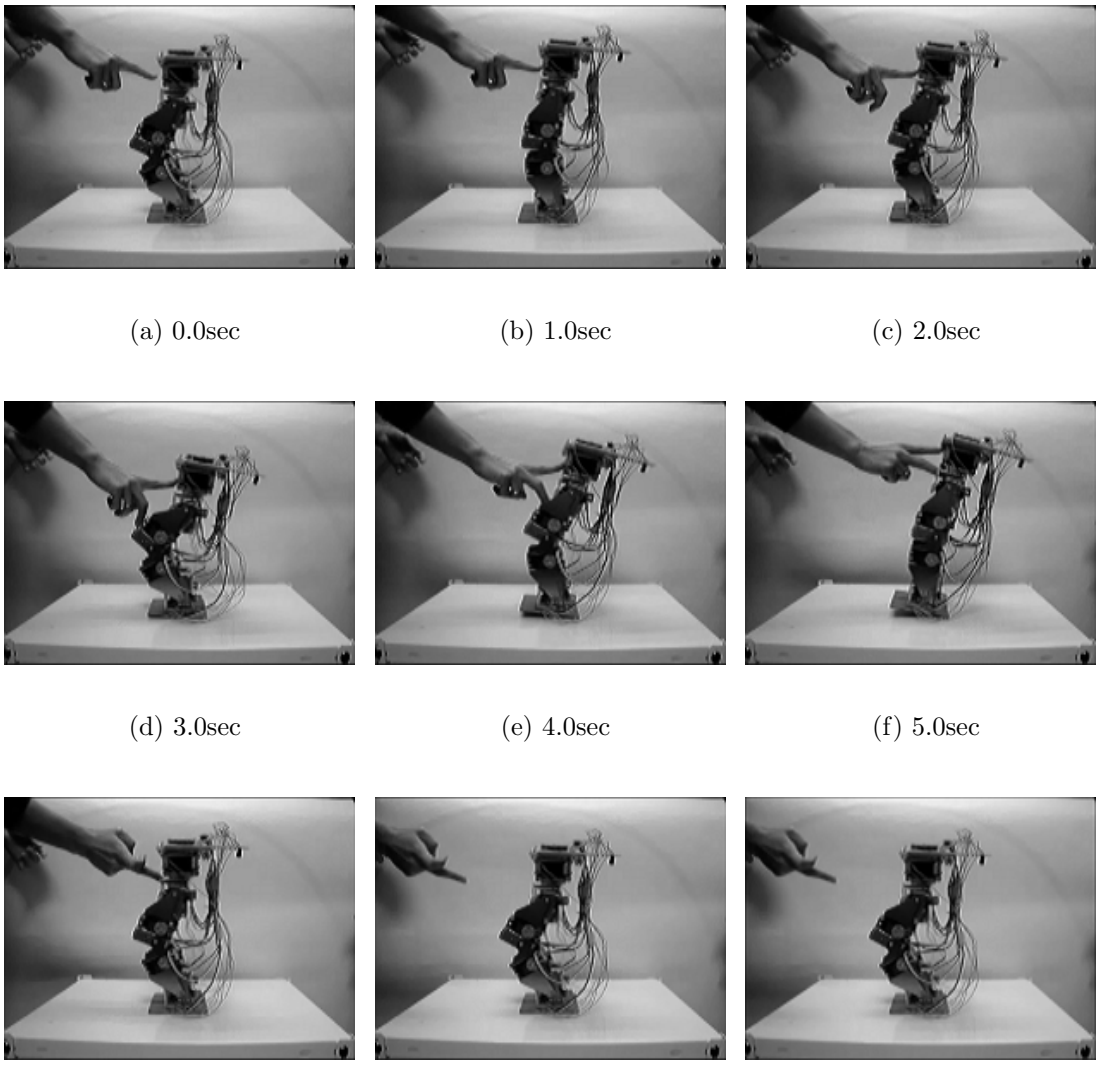


(a) 0.0sec (b) 1.5sec (c) 3.0sec

(d) 4.5sec (e) 6.0sec (f) 7.5sec

(g) 9.0sec (h) 10.5sec (i) 12.0sec

図 3.16: 屈伸運動中に押された場合の様子（センサフィードバックあり）



(a) 0.0sec　(b) 1.0sec　(c) 2.0sec

(d) 3.0sec　(e) 4.0sec　(f) 5.0sec

(g) 6.0sec　(h) 7.0sec　(i) 8.0sec

図 3.17: 屈伸運動中に押された場合の様子（センサフィードバックなし）

第4章

多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性

本章では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性について述べる．拡張性について述べるために，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに対して機能拡張した構造に基づいて，前章で試作した多関節ロボットのサーボモータ制御プラットフォームに機能追加を行い，その効果を確認した．なお，本章で使用する図は，章末 p.67 よりまとめて記載する．

4.1 多関節ロボットのモーション実行用ソフトウェアアーキテクチャの機能拡張

ソフトウェアにおける拡張性とは，ソフトウェアの基本構造に大きな変更や影響を与えることなく，機能追加や性能向上を行えることである．提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，量産された多関節ロボットにおいて保証されたモーションを実行するためのアーキテクチャである．量産されるためには，十分な評価を試作ロボットやシミュレータで行う必要があるが，その際において機能追加や性能向上を行うことは開発において避けられないことである．しかし，機能追加や性能向上を行うことでソフトウェアの基本構造に大きな変更や影響を与えてしまうことは，ソフトウェアの品質に影響を及ぼす．図 4.1 に示す提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，多関節ロボットのモーション実行用ソフトウェアの設計指針となるアクチュエータ制御システムの基本構造としての提案である．そこで，提案する多関節ロボットのモーション実行用ソフトウェアアー

キテクチャが，基本構造を変更することなく機能拡張が行えることを確認する．追加する機能は，モーションの N倍速実行機能とモーションの同期機能である．第3章で図4.1に示す多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいて試作した多関節ロボットのサーボモータ制御プラットフォームを基本システムとして機能を追加した．以下，4.1.1項ではモーションの N倍速実行機能を追加する背景を述べ，4.1.2項では N倍速実行機能について述べる．続く，4.1.3項ではモーション間の同期機能を追加する背景を述べ，4.1.4項ではモーションの同期機能について述べる．

4.1.1　モーションの N倍速実行機能を追加する背景

提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいて試作した多関節ロボットのサーボモータ制御プラットフォームでは，モーションの実行方法は3.3.2節で述べたように，各関節におけるモーションのキーとなる点と点の角度の差分を移動時間のサイクル数によって細分化し，細分化された角度を順次実行することによってモーションを実現している．そのため，モーションの速さは，モーションデータに関節の角度情報とともに保持している移動時間のサイクル数に依存する．このことから，モーションの速さのみを変更したい場合，モーションのキーとなる各関節の角度情報は変わらないにもかかわらず，新たにモーションデータを記述しなければならない．つまり，同じ動きであるにもかかわらず速さごとにモーションデータが必要となり，際限なくモーションデータが増加してしまう．これは，同じ動きであるにもかかわらず速さが異なるだけでモーションデータの信頼性が落ち，モーションが保証されないことを意味する．そのため，保証されたモーションを実行するために，動きと速さの両方に着目した評価を行わなければならない．また，システムとしてメモリ効率も良いとはいえない．そこで，既存モーションデータを再利用することに注目し，モーションの速さだけを変更する機能を追加する．

4.1.2　モーションの N倍速実行機能

モーションの N倍速実行機能は，図4.1に示す提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに対して，図4.2に示すSet Speed Functionを追加することにより実現した．Set Speed Functionは，モーションデータの読み込み後，モーション実行部(Motion Execution Part)にデータを受け渡す前に読み込む関数

であり，モーションデータの移動時間のサイクル数にのみ作用する．また，引数は倍速要素となる数値 N（$N \geqq 1$）であり，モーションデータとして保持している移動時間のサイクル数を C_{base} とすると，

$$C_{new} = C_{base}/N \quad (1 \leqq N \leqq C_{base}) \tag{4.1}$$

によって移動時間のサイクル数を C_{new} に更新する．その後，移動時間のサイクル数 C_{new} を持ったモーションデータをモーション実行部 (Motion Execution Part) に受け渡すことで N 倍速のモーションを実行する．数値 N は $N \geqq 1$ の整数のみではなく浮動小数点でも問題はない*． 移動時間のサイクル数を 450 としてモーションデータを作成した場合を例として，数値 N=2 および数値 N=3 としたときの Set Speed Function の機能を説明する．

- 数値 N=2 のとき，(4.1) 式によって移動時間のサイクル数は 225 に更新される
- 数値 N=3 のとき，(4.1) 式によって移動時間のサイクル数は 150 に更新される

モーションの速度を変更したくなければ，数値 N=1 とすればよい．このように，モーションは Set Speed Function に与えた数値 N 倍の速さになる．なお，式 (4.1) に示すように，N の上限値はモーションデータとして保持している移動時間のサイクル数である．また，モーションの N 倍速実行機能は，モーションデータ読み込み部 (Reading MotionData Part) でのモーションデータの読み込み後においてモーション実行部 (Motion Execution Part) にデータを受け渡す前に読み込む機能であるため，3.3.3 項のモーションデータの読み込み例としてのモーションの合成や後述（4.1.4 項）のモーション間の同期のように，モーションデータの読み込み機能において加工したモーションに対しても速さ変更が可能である．N 倍速実行機能における静的，動的な問題に関して述べると，サーボモータに加わるモーメントが静的，動的ともに許容範囲内であれば，N 倍速実行機能は有効であるといえる．ただし，モーションを N 倍速にする場合は，別途モーションの速さのみに着目したモーションを保証するための評価行う必要がある．

4.1.3 モーション間の同期機能を追加する背景

提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいて試作した多関節ロボットのサーボモータ制御プラットフォームでは，モーションデータ

*浮動小数点を使用する場合は，浮動小数点演算処理装置が搭載されているマイコンに限る

の読み込み機能においてモーションデータの使用方法の例としてモーションの合成を行うことが出来る．モーションの合成は，3.5節で示したように，純粋なメモリ・ベースト制御でのモーション実行よりも多くのモーションを実行することが可能である．3.5節で示した合成の例では，1つの制御対象に対して新たなモーションを生成することに有効である．一方で，異なる制御対象同士のモーションの合成も可能である．例えば，2脚ロボットにおいて異なる制御対象である左脚の片脚屈伸と右脚の片脚屈伸を合成することによって両脚屈伸を実現できる．これは，左右の片脚屈伸のモーションデータを保持するのみで，左右の片脚屈伸のみではなく両脚屈伸を実行可能であることを示している．しかし，左右の片脚屈伸の組み合わせである足踏み動作を実行したい場合は，単純にモーションを合成するだけではできない．そのため，左右の片脚屈伸と同じ動きであるにもかかわらず，新たにモーションデータを用意しなければならない．また，左脚，右脚といった複数の制御対象による足踏み動作のような動きは，動きの同期をとる必要がある．足踏み動作のような複数の制御対象の動きにおいて同期をとる場合，同期をとるタイミングによって動きの見た目が変わる．そのため，モーションデータの作成が難しいだけではなく，足踏み動作を1つのモーションとして扱わなければならないだけではなく，同期をとるタイミングの数だけモーションデータが必要になり，際限なくモーションデータが増加してしまう．これは，同じ動きの組み合わせであるにもかかわらずモーションデータの信頼性が落ち，モーションが保証されないことを意味する．そのため，保証されたモーションを実行するために，動きと同期の両方に着目した評価を行わなければならない．さらに，システムとしてメモリ効率も良いとはいえない．そこで，既存モーションデータを再利用することに注目し，モーションデータの読み込み機能においてモーションデータの使用方法の例としてモーション間の同期をとる機能を追加する．

4.1.4　モーション間の同期機能

モーション間の同期機能は，モーションデータの使用方法を考慮するためのモーションデータの読み込み部 (Reading MotionData Part) 内での機能追加である．これは，モーションの合成において合成に用いるモーションのうち1つを基準モーションとして，その他のモーションを基準モーションのどのタイミングで開始するのかを設定する機能である．具体的には，基準モーションの始点角度，中間点角度，終点角度のう

ち，どの角度に達したときにその他のモーションを開始するのか設定する機能である．この機能は，図 4.1 に示す提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに対して，図 4.2 に示すように，モーションデータの読み込み機能に動きの同期をとるための情報を与えることにより実現した．与える情報は，合成に用いるモーションのうち「基準となるモーション」とその他のモーションのそれぞれの開始点となる「基準モーションデータ中の位置」のペアである．この情報は，合成に用いるモーションの全てに対して与える．与える情報のペアとは，{ 基準モーション，基準モーションデータ中の位置 } である．また，モーション間の同期機能は図 4.2 に示すモーションデータ読み込み部 (Reading MotionData Part) で処理が行われるため，4.1.2 節の N 倍速実行機能はモーション間の同期機能によって作られたモーションに対しても使用可能である．なお，モーション間の同期機能によって作られたモーションは，別途モーション間の同期のみに着目したモーションを保証するための評価を行う必要がある．

2 脚ロボットでのモーション間の同期を例に述べる．図 4.3(a) は，2 脚ロボットの左脚屈伸のモーションデータ（モーション名：Left leg) と右脚屈伸のモーションデータ（モーション名：Right leg）を抽象的に示した例である．左脚屈伸，右脚屈伸ともに始点角度（Start Angle）は脚を伸ばした状態の関節角度，中間点角度（Middle Angle）は屈伸において脚を最大に曲げた状態の関節角度，終点角度（End Angle）は始点角度と同じである．なお，図 4.3(a) 中の角度情報は左脚屈伸，右脚屈伸ともに同一の表現となっているが，制御対象が右脚，左脚と異なることによって上述の状態の角度は異なる．この 2 つのモーションを図 4.2 のモーションデータ読み込み部 (Reading MotionData Part) において合成することにより，図 4.1 に示すアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームでは両脚屈伸を実現していた．図 4.2 のようにモーション間の同期機能を追加することにより，この 2 つのモーションの合成によって足踏み動作を実現可能にした．基準モーションとして左脚屈伸（Left leg）を設定し，図 4.3(b) に示すように左脚屈伸（Left leg）は始点角度である Start Angle から開始させ，右脚屈伸（Right leg）は基準モーションである左脚屈伸（Left leg）が Middle Angle 到達時から開始させることによって足踏み動作が実現可能である．このとき，モーションデータ読み込み部 (Reading MotionData Part) に与える情報は

- 左脚屈伸（Left leg）に対して {Left leg, Start Angle}

- 右脚屈伸（Right leg）に対して {Left leg, Middle Angle}

である．モーションの合成によって両脚屈伸を行いたければ，モーションデータ読み込み部 (Reading MotionData Part) に与える情報を

- 左脚屈伸（Left leg）に対して {Left leg, Start Angle}
- 右脚屈伸（Right leg）に対して {Left leg, Start Angle}

とすればよい．

4.2　動作実験 1

本節では，図 4.1 に示すアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームに対して追加した，図 4.2 に示すモーションの N 倍速実行機能 (Set Speed Function) の効果確認として，実行結果と効果の考察について述べる．実験に用いたロボットは 2 脚ロボットであり，第 3 章の動作実験で使用したものと同じである．制御 CPU は株式会社ルネサステクノロジ製の SH7145F である．本実験で使用するモーションデータは，右脚屈伸と左脚屈伸の 2 つである．図 4.3(a) に示したように，左脚屈伸，右脚屈伸ともに始点角度は脚を伸ばした状態の関節角度，中間点角度は屈伸において脚を最大に曲げた状態の関節角度，終点角度は始点角度と同じである．

4.2.1　N 倍速実行機能の実行結果

N 倍速実行機能が実現可能であることを右脚屈伸のモーションデータを使用し確認する．まず，1 倍速でのモーションの実行結果を図 4.4 に示す．図 4.4 は約 0.5 秒おきにキャプチャしたものであり，矢印は動きの方向を示している．(a) のスタート位置から (b)(c) と脚を曲げながら上げていき中間点角度に到達，(d)(e) と脚を伸ばしながら下ろしている様子が分かる．同様に (f)(g) と脚を曲げながら上げていき，(h)(i) と脚を伸ばしながら下ろしている様子が分かる．図 4.4 に示す右脚屈伸の様子から動作の周期が約 2.0 秒であることが分かる．以下では，モーションの N 倍速実行機能によって図 4.4 のモーションが 2 倍速で実行されることを確認する．モーションの N 倍速実行機能により 2 倍速にした右脚屈伸を図 4.5 に示す．図 4.5 は図 4.4 と同様に約 0.5 秒おきにキャプチャしたものである．図中の矢印は動きの方向を示している．図 4.4 と

図 4.5 を比較すると，図 4.4 の (a) のスタート位置から (b)〜(e) にかけて右脚屈伸が一度行われているのに対して，図 4.5 の (a) のスタート位置から (b) で中間点角度に到達，(c) で脚を下ろし屈伸が行われている．同様に (d) 中間点角度に到達，(e) で脚を下ろし屈伸が行われている．このことから，動作の周期が約 1.0 秒であり，図 4.5 に示す右脚屈伸は，図 4.4 に示す右脚屈伸の 2 倍の速さで実行されているいえる．これにより，モーションの N 倍速実行が可能であることを確認できた．

4.2.2 N 倍速実行機能の効果の考察

N 倍速実行機能を実現したことによる効果について考察する．図 4.1 に示すアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームと，N 倍速実行機能を有する図 4.2 に基づいた多関節ロボットのサーボモータ制御プラットフォームにおいて，モーションの速さのみが異なる例として 1 倍速〜N 倍速までに必要となるメモリ領域を考える．1 つのモーションデータの作成において，始点角度，中間点角度 m 箇所，終点角度とした場合を例に考える．ただし，m は自然数である．図 4.1 に示すアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームでは，表 4.1 に示すように速さごとに $2+m$ 箇所分のメモリ領域が必要となるため，1 倍速〜N 倍速では $N(2+m)$ 箇所分のメモリ領域が必要であった．一方で，モーションの N 倍速実行機能を取り入れることにより，1 つのモーションデータで 1 倍速〜N 倍速まで実行可能であるため，表 4.1 に示すように $2+m$ 箇所分のメモリ領域のみでよい．このように，N 倍速実行機能を取り入れることで，1 つのモーションデータを保持するのみで速さの異なるモーションを実現可能であり，図 4.1 に示すアーキテクチャと比較するとメモリ領域を $\frac{1}{N}$ に抑えることができる．つまり，システムとしてのメモリ効率を向上しながらも，基本となるモーションを保持するのみで多くのモーションを実行可能であるといえる．また，保証されたモーションからの変更点はモーションの速さのみであるため，モーションを保証するための評価は速さのみに着目した評価でよい．これらの効果を，図 4.1 の基本構造を変更せずに実現できたため，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性は高いといえる．

表 4.1: モーションデータ数の比較

Speed	Conventional method	Proposal method
single	$2+m$	
double	$2+m$	
triple	$2+m$	$2+m$
N	$2+m$	
total	N$(2+m)$	$2+m$

4.3　動作実験2

本節では，図4.1に示すアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームに対して，図4.2に示すモーションデータの読み込み部 (Reading MotionData Part) 内での機能追加によるモーション間の同期機能が実現可能であることを確認する．実験に用いたロボットおよび制御 CPU は4.2節と同じである．また，実験に用いるモーションデータは，図 4.3(a) に示した右脚屈伸と左脚屈伸のモーションデータである．右脚屈伸は図 4.4 と同じものであり，左脚屈伸は右脚屈伸と同じ関節角度情報をモーションデータとして使用している．右脚屈伸および左脚屈伸のモーション合成による両脚屈伸モーションを図 4.6 に示す．図中の矢印は，左右の脚の動きの方向を示している．(a) のスタート位置から (b)(c) へと両脚を曲げながら上げていき，(d)(e) と両脚を伸ばしながら下ろしている様子が分かる．同様に (f)(g) と両脚を上げていき，(h)(i) と両脚を下ろしている様子が分かる．本実験では，モーション間の同期において，左脚屈伸を基準にし，図 4.3(b) のように左脚屈伸の中間点角度到達時と右脚屈伸の開始タイミングを同期させることによって足踏み動作が実現可能であることを確認する．

4.3.1　モーション間の同期機能の実行結果

図 4.7 にモーション間の同期機能を用いた足踏み動作の実行結果を示す．図中の矢印は，左右の脚の動きの方向を示している．図 4.7 の (a) スタート位置から (b)(c) にかけて左脚屈伸が実行され中間点角度に到達する．その間，右脚屈伸は行われていない．そして，(c) において左脚屈伸が中間点角度到達したことにより，(d)(e) のように右脚屈伸が開始され脚をあげていることが分かる．このとき，左脚屈伸は継続され脚を下

ろしていることが分かる．(f) から (i) では，継続的に足踏み動作が行われていることがわかる．これにより，モーション間の同期機能が実現可能であることを確認できた．

4.3.2 モーション間の同期機能の効果の考察

モーション間の同期機能を実現したことによる効果について考察する．図 4.1 に示すアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームでは，同じ動きの組み合わせであるにもかかわらず，開始タイミングが異なるという理由で新たなモーションデータが必要である．図 4.2 のようにモーション間の同期機能を取り入れることにより，前項のような 2 脚ロボットの例では，左右の片脚屈伸，両脚屈伸のみではなく，足踏み動作を実現できる．また，左脚屈伸と右脚屈伸のモーションデータ作成において，始点角度と終点角度の間を担う中間点角度を多く取ることにより，モーション間の同期をとることが可能な点が増える．そのため，足踏み動作だけでも多くの種類を実行することが可能である．足踏み動作を実現するにあたって，左右の片脚屈伸において，それぞれ始点角度，中間点角度 m 箇所，終点角度が必要な場合でのメモリ領域について考える．ただし，m は自然数であり，図 4.3 と同様に始点角度と終点角度は同じ角度として考える．このとき，1 種類の足踏み動作には，左右の片脚屈伸に必要な角度情報の数であるため，$2(2+m)$ 箇所分のメモリ領域が必要である．また，左脚から開始する足踏み動作においての右脚屈伸の開始タイミングは，左脚屈伸の中間点角度の数だけ種類があるため m 種類，同様に右脚から開始する足踏み動作も m 種類である．そのため，足踏み動作は $2m$ 種類存在する．図 4.1 に示すアーキテクチャに基づいた多関節ロボットのサーボモータ制御プラットフォームでは，$2m$ 種類の足踏み動作を実現するために必要となるモーションデータは $2m$ 個であり，角度情報として $4m(2+m)$ 箇所分のメモリ領域が必要であった．一方で，モーション間の同期機能を追加したことにより，動きの同期をとるための情報をモーションの合成時に付加するだけでよいため，$2m$ 種類の足踏み動作を実現するために必要となるモーションデータの数は左右の片脚屈伸の 2 個であり，角度情報として必要とするメモリ領域は $2(2+m)$ 箇所分でよい．つまり，システムとしてのメモリ効率を向上しながらも，基本となるモーションを保持するのみで多くのモーションを実行可能であるといえる．また，保証されたモーションからの変更点はモーション間の同期のみであるため，モーションを保証するための評価はモーションの同期のみに着目した評価でよい．

ここで，4脚ロボットのように制御対象が増えた場合においての効果を考察する．図 4.8(a) に4脚ロボットのモーションデータの概略図を示す．これは，右前脚のモーションデータ（モーション名：Right foreleg），左前脚のモーションデータ（モーション名：Left foreleg），右後ろ脚のモーションデータ（モーション名：Right hind leg），左後ろ脚のモーションデータ（モーション名：Left hind leg）である．それぞれ，始点角度として地面に脚をつけた状態の関節角度，中間点角度1として脚を上げながら前に出した状態の関節角度，中間点角度2として地面に脚をつけた状態の関節角度，終点角度として脚を上げながら後ろに出した状態の関節角度をモーションのキーとして作成したものである．ただし，図 4.8(a) は説明を行いやすくするために用意したものであり，始点角度，中間点角度1，中間点角度2，終点角度は，異なる制御対象である右前脚，左前脚，右後ろ脚，左後ろ脚のそれぞれの状態における角度情報を抽象的に示しているものである．角度情報として同一の表現が用いられているが，制御対象が異なるため，それぞれの角度は異なる．また，実際にモーションを実行したい場合は，ロボットを構成する関節の数や実現したいモーションにより，モーションのキーとなる点の関節角度の数は異なる．モーション間の同期機能を用いることで，図 4.8(a) に示す4つのモーションデータのみで，トロット歩容，クロール歩容，ペース歩容，ギャロップ歩容のモーションを実現可能である．例えば，図 4.8(b) に示すように，右前脚のモーション（Right foreleg）を基準にし，右前脚モーション（Right foreleg）の始点角度の開始タイミングと左後脚モーション（Left hind leg）の開始タイミング，右前脚のモーション（Right foreleg）の中間点角度に到達時と左前脚モーション（Legt foreleg）および右後脚モーション（Right hind leg）の開始タイミングを同期させることにより，トロット歩容が実現可能であり，

- 右前脚モーション（Right foreleg）に対して {Right foreleg, Start Angle}
- 左前脚モーション（Legt foreleg）に対して {Right foreleg, Middle Angle2}
- 右後脚モーション（Right hind leg）に対して {Right foreleg, Middle Angle2}
- 左後脚モーション（Left hind leg）に対して {Right foreleg, Start Angle}

という情報を与えればよい．同様に，合成に用いるモーションにおいて基準となる脚のモーションと，その他の脚のモーションの開始タイミングを同期させることにより，クロール歩容，ペース歩容，ギャロップ歩容のモーションを実現することが可能であ

る．また，各モーションデータにおいて始点角度と終点角度の間を担う中間点角度を多く取ることによりモーション間の同期をとることが可能な点が増えるため，図 4.9 に示すように 1 つの歩容に対して多くの種類を実行することが可能である．ただし，各歩容が成り立つ各脚の着地タイミングを考慮しモーション間の同期をとる必要があるため，それぞれの脚の中間点角度の取り方には注意する必要がある．このように，制御対象が増えた場合においては，メモリ効率を向上や，モーションを保証するための評価はモーションの同期のみに着目した評価でよいという点は上述の例よりも効果が高い．また，基本となるモーションを保持するのみで多くのモーションを実行可能であり，モーションを保証するための評価はモーションの同期のみに着目した評価でよいという効果を，図 4.1 の基本構造を変更せずに実現できたため，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性は高いといえる．

4.4 まとめ

本章では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性について述べるために，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャに基づいて試作した多関節ロボットのサーボモータ制御プラットフォームに対して，モーションの N倍速実行機能とモーション間の同期をとる機能を追加した．モーションの N倍速実行機能とモーション間の同期をとる機能は，既存のモーションデータを再利用することにより実現しており，その効果は以下の通りである．

(1) モーションの N倍速実行機能を追加したことにより，保証されたモーションを再利用するのみで，速さの異なるモーションの実行が可能
(2) モーション間の同期機能を追加したことにより，保証されたモーションを再利用するのみで，足踏み動作のような組み合わせ動作の実行が可能
(3) 保証されたモーションを再利用するのみであるため，モーションデータとして必要になるメモリ容量の削減につながる

これらのことは，基本となるモーションを保持するのみで多くのモーションを実行可能であることを示している．加えて，モーションを保証するための評価は保証されたモーションからの変更点に着目した評価でよい．また，機能追加に際して，図 4.1 に

示した提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの基本構造は変更していない．そのため，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは拡張性が高いといえる．

4.5　第4章で使用する図

本章で使用する図を次頁より示す．

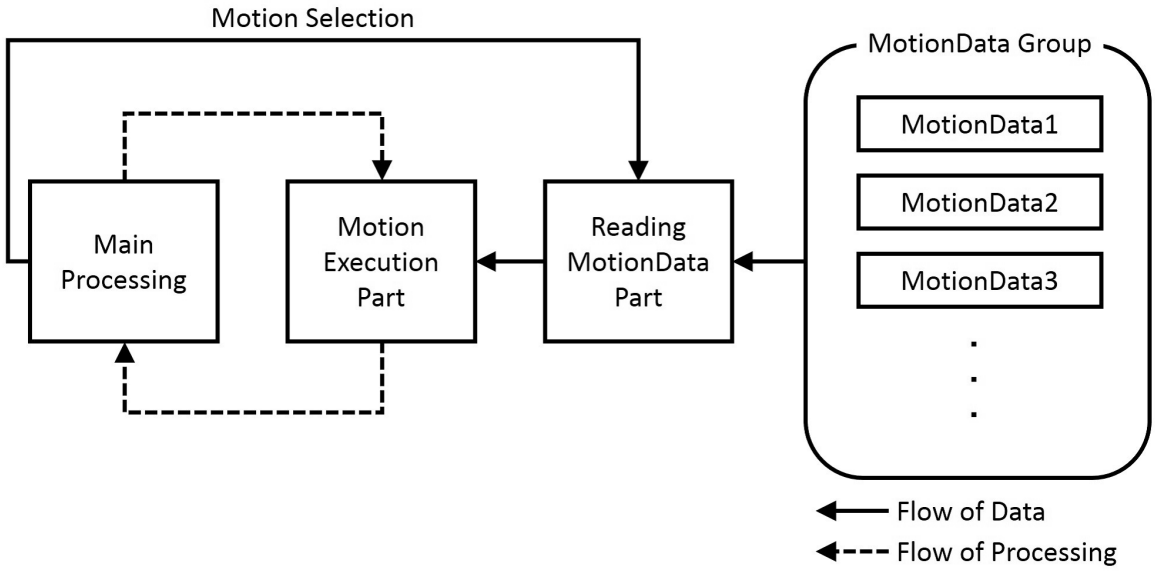


図 4.1: 多関節ロボットのモーション実行用ソフトウェアアーキテクチャ

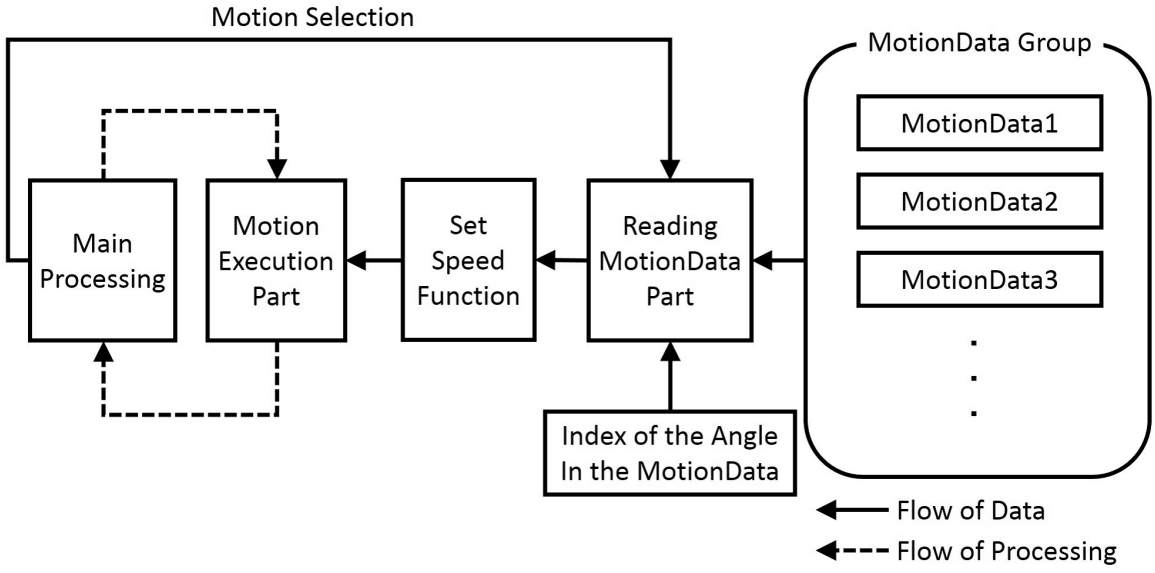
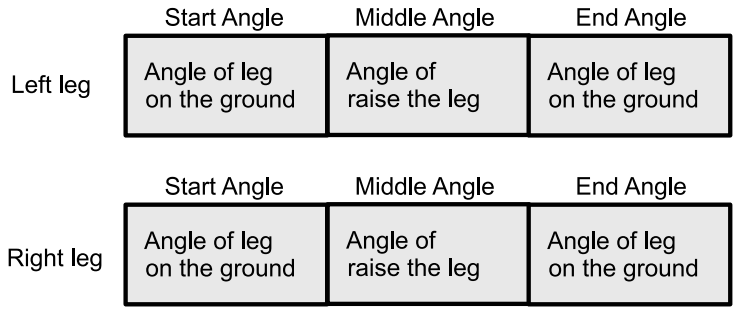
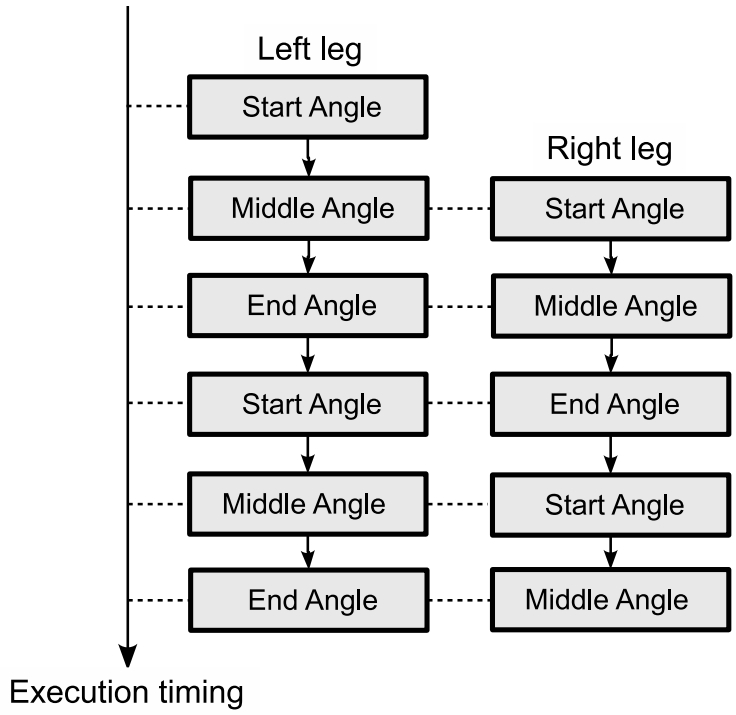


図 4.2: 多関節ロボットのモーション実行用ソフトウェアアーキテクチャの機能拡張

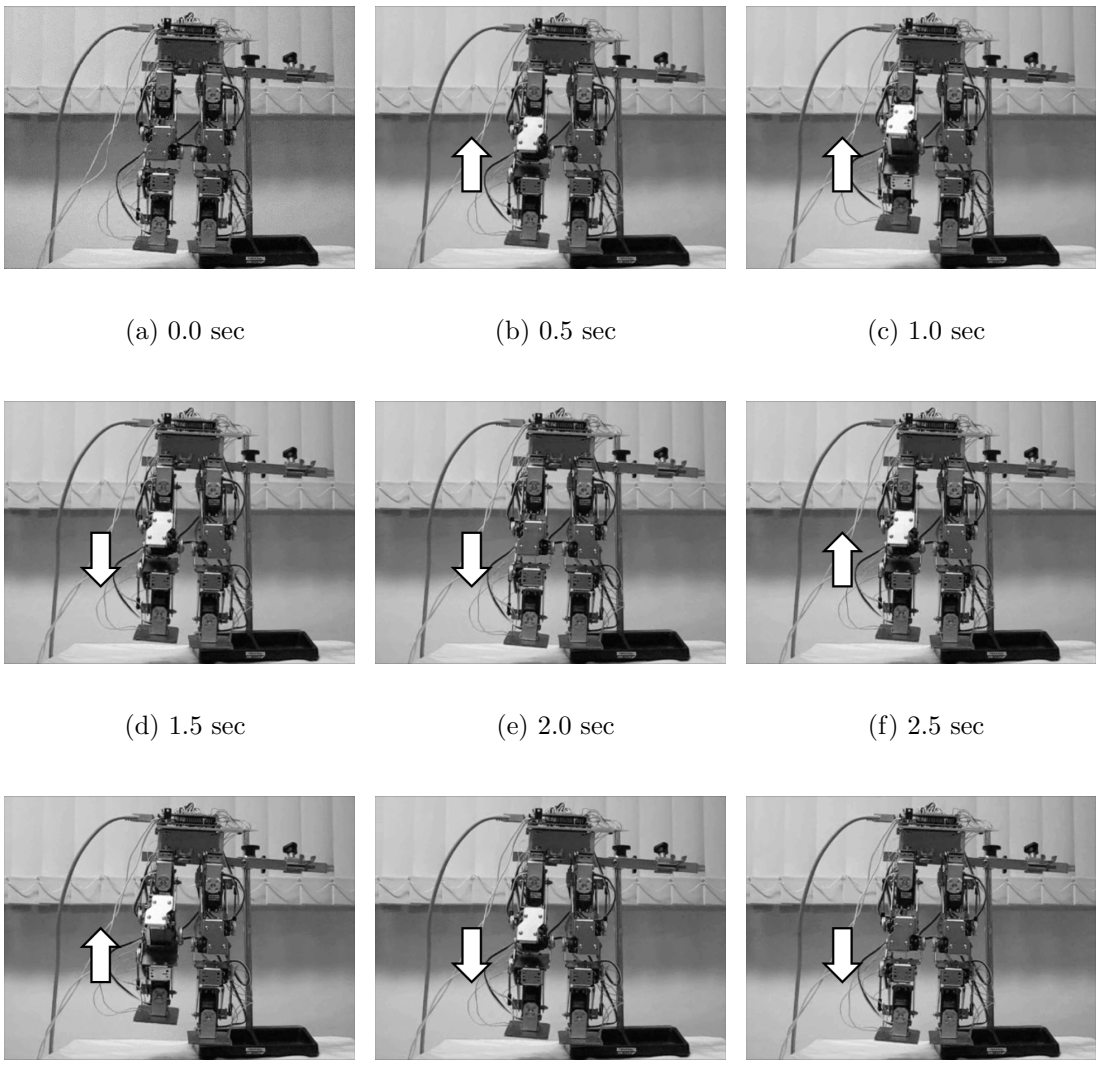


(a) モーションデータ



(b) 足踏み動作での角度情報の実行タイミング

図 4.3: 2脚ロボットのモーションデータと実行タイミングの概念図

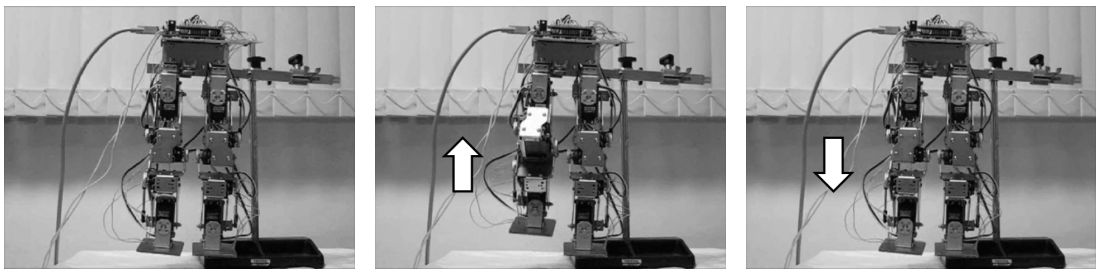


(a) 0.0 sec (b) 0.5 sec (c) 1.0 sec

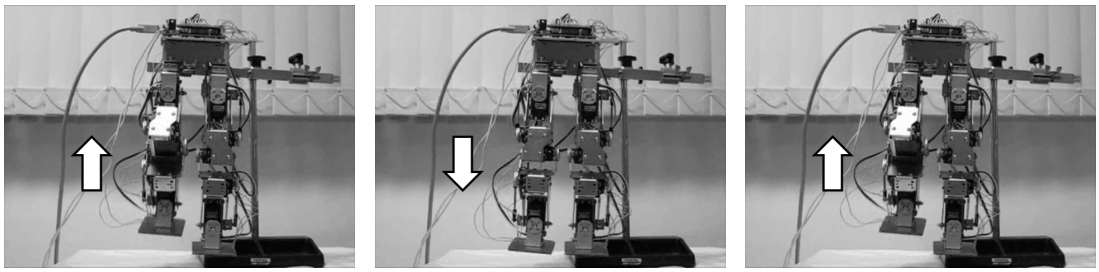
(d) 1.5 sec (e) 2.0 sec (f) 2.5 sec

(g) 3.0 sec (h) 3.5 sec (i) 4.0 sec

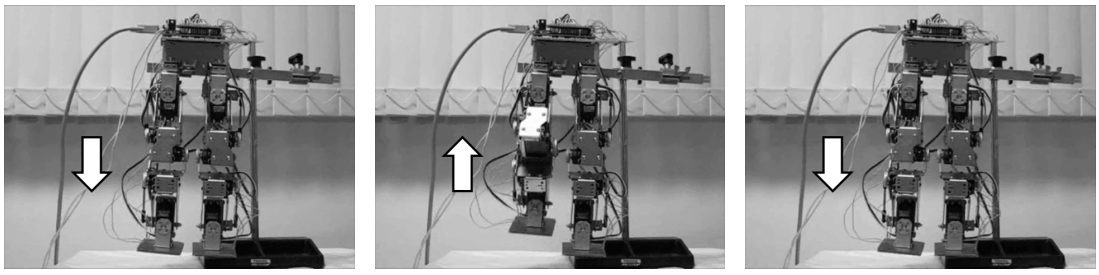
図 4.4: 右脚屈伸の様子



(a) 0.0 sec　(b) 0.5 sec　(c) 1.0 sec

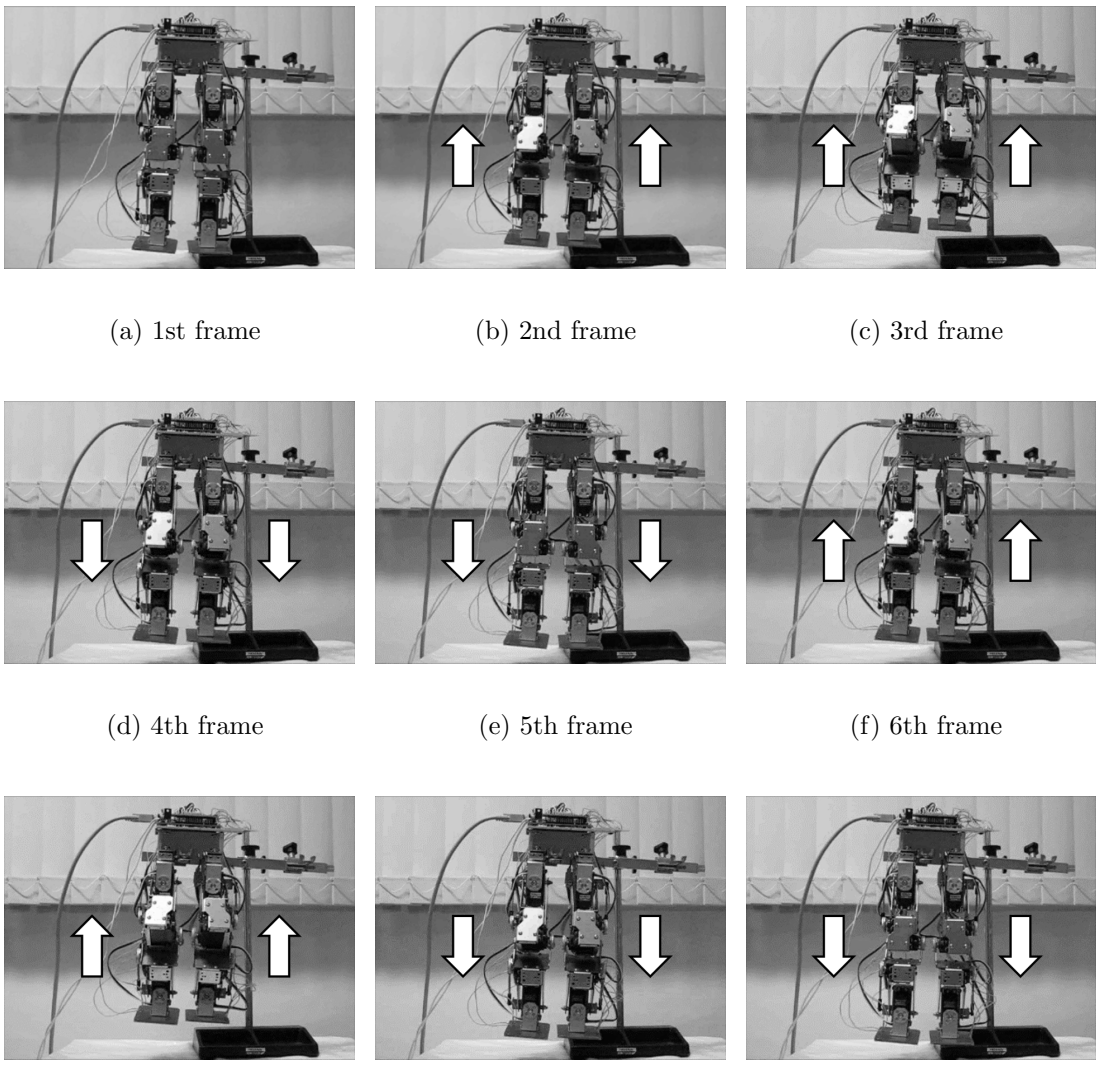


(d) 1.5 sec　(e) 2.0 sec　(f) 2.5 sec



(g) 3.0 sec　(h) 3.5 sec　(i) 4.0 sec

図 4.5: 2倍速での右脚屈伸の実行結果

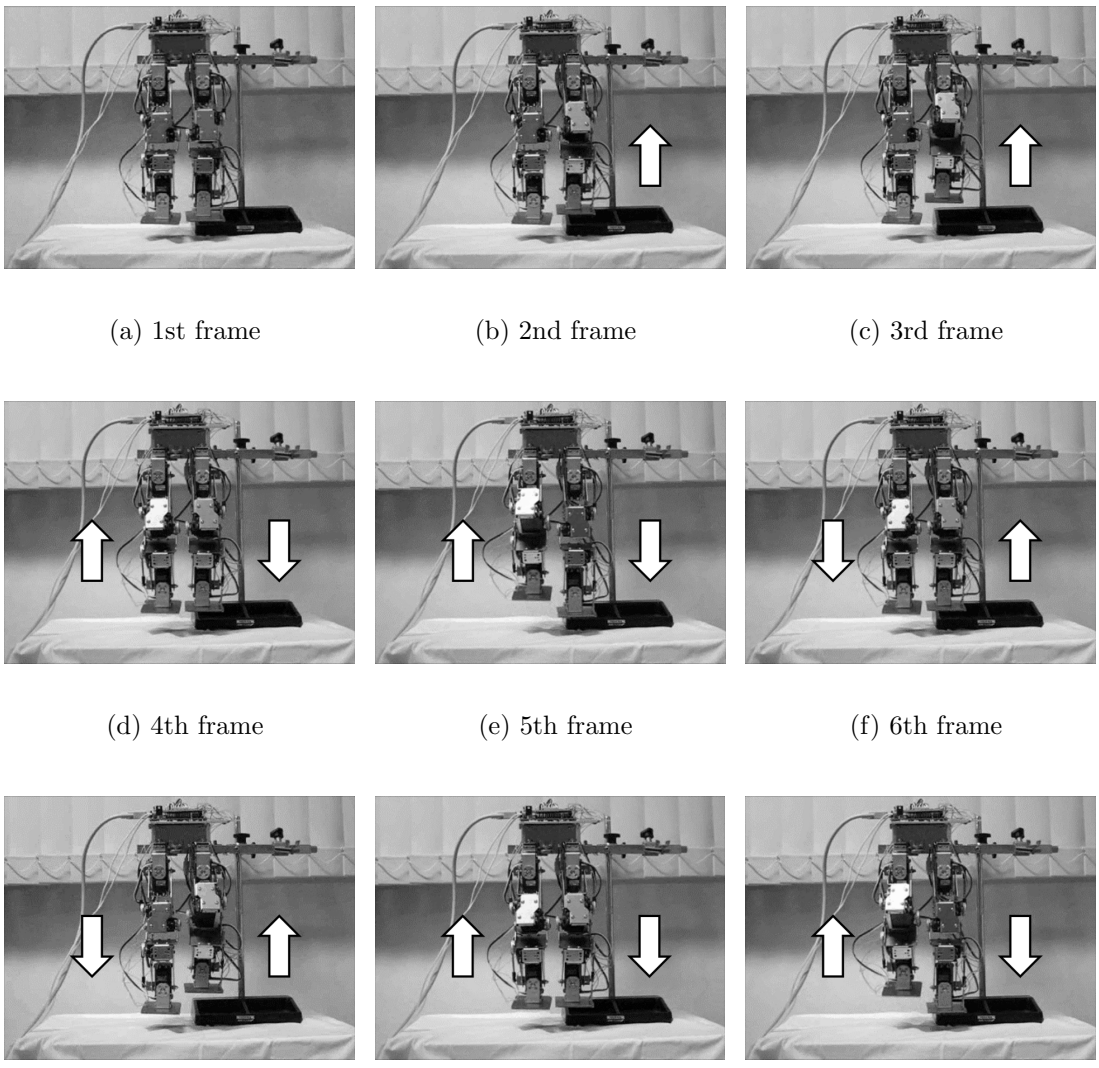


(a) 1st frame (b) 2nd frame (c) 3rd frame

(d) 4th frame (e) 5th frame (f) 6th frame

(g) 7th frame (h) 8th frame (i) 9th frame

図 4.6: 右脚屈伸と左脚屈伸の合成による両足屈伸の様子

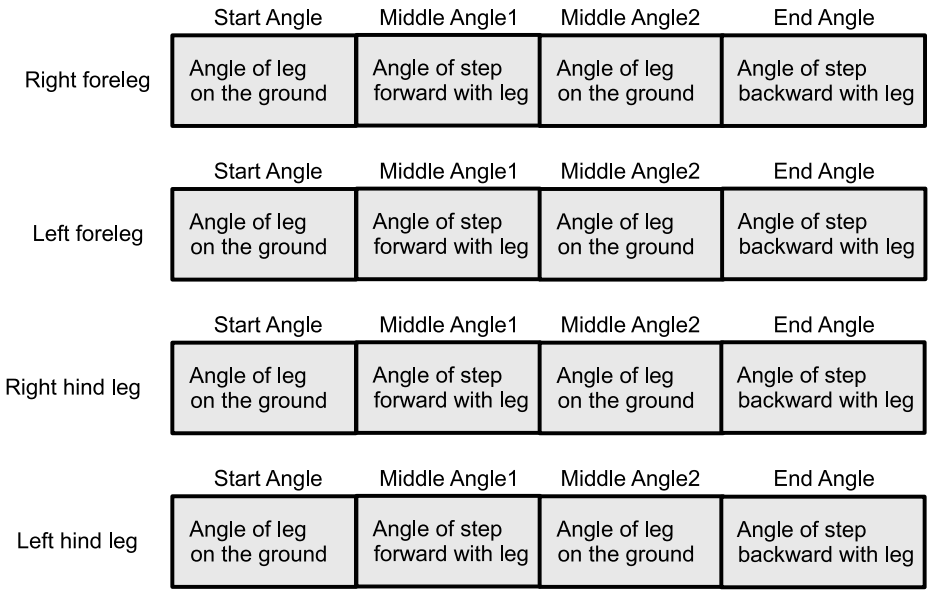


(a) 1st frame　(b) 2nd frame　(c) 3rd frame

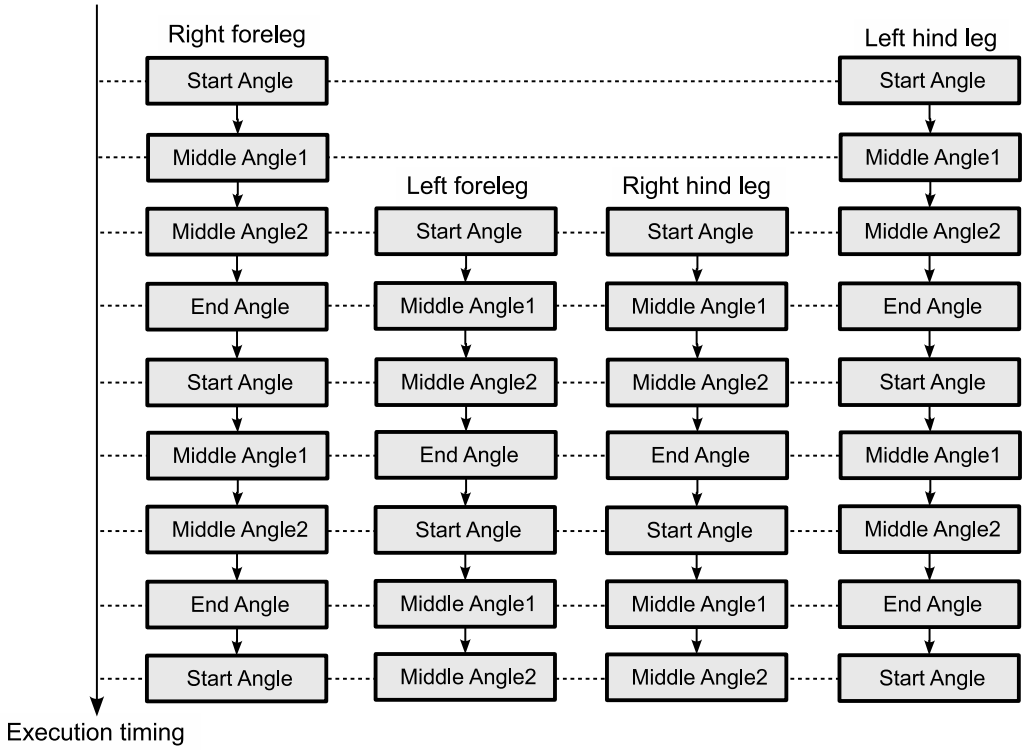
(d) 4th frame　(e) 5th frame　(f) 6th frame

(g) 7th frame　(h) 8th frame　(i) 9th frame

図 4.7: モーションの同期機能による足踏み動作の実行結果



(a) モーションデータの例



(b) トロット歩容での角度情報の実行タイミング

図 4.8: 4 脚ロボットのモーションデータと実行タイミングの概念図

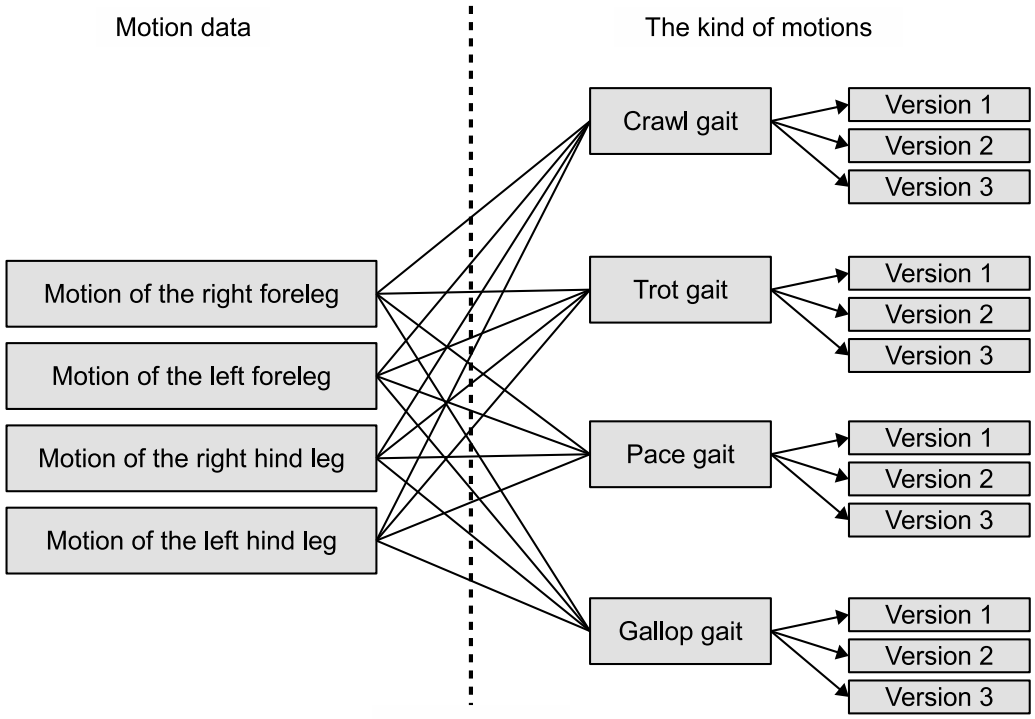


図 4.9: モーションの同期機能による効果

第5章

結論

本章では，本論文のまとめと今後の展望について述べる．

5.1　まとめ

本論文では，多関節ロボットの多機種展開に向けての問題点の対応策として，メモリ・ベースト制御による多関節ロボットのモーション実行用ソフトウェアアーキテクチャを提案した．多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，多関節ロボットのモーション実行用ソフトウェアの設計指針となるアクチュエータ制御システムの基本構造である．提案するモーション実行用ソフトウェアアーキテクチャは，保証されたモーションを実行するために必要な要素と，保証されたモーションの使用方法を考慮する要素を組み合わせた構造であり，保証されたモーションを実行するために事前に作成されたモーションデータを選択・実行するメモリ・ベースト制御を用いている．そのため，多関節ロボットの量産において必要になる試作ロボットでのモーション評価の繰り返しによるモーションの保証が可能となる．

第2章では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャについて述べた．具体的には，保証されたモーションの実行に必要な要素について述べ，多関節ロボットのモーション実行用ソフトウェアアーキテクチャの提案を行った．また，提案するモーション実行用ソフトウェアアーキテクチャを用いたモーション実行用ソフトウェア開発においての利点であるモーションの評価の繰り返しが行いやすいことについて，モーションの変更点に着目したモーションの評価によってモーションの保証が可能であることを述べ，モーションの評価によって保証されたモーションに対しての信頼性の確保について述べた．

第3章では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャを構成する要素について述べた．具体的には，保証されたモーションの実行に必要な要素であるモーションデータ，モーション実行機能，モーションデータの選択機能に加え，モーションデータの使用方法を考慮できる機能としてのモーションデータの読み込み機能について述べた．また，下位レベル制御であるモーションの実行（アクチュエータ制御）においてのセンサフィードバックの必要性について述べ，モーション実行に関するセンサフィーバックを行える仕組みについて述べた．さらに，提案するアーキテクチャを用いて試作したモーション実行システムにおいて動作実験を行い，モーションの実行が可能であること，センサフィードバック処理が可能であること，モーションデータの読み込み機能の効果について確認を行った．これによって，以下のことを示した．

(1) 多関節ロボットの機種や形態を問わず，同一のソフトウェアにおいてモーションデータの変更のみによるモーションの実行が可能であり，モーションの評価をモーションの変更点に着目して行うことができることから，モーションの評価を繰り返し行いやすい．また，共通のソフトウェア構造を用いてモーションの実行を可能にしているため，モーションの保証に対しての信頼性が確保される．
(2) モーションデータを組み合わせることで，実行できるモーション数を増やすことが可能である．
(3) メモリ・ベースト制御による保証されたモーションの実行が可能でありながら，センサフィードバック処理が可能である．

第4章では，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの拡張性について述べるため，メモリ・ベースト制御の問題点の対応策として第3章で試作したモーション実行システムを基本システムとして機能拡張を行った．追加した機能は，モーションの速さを変更する機能，モーション間の同期をとる機能であり，動作実験によってその効果を確認した．これによって，基本となるモーションを保持するのみで多くのモーションを実行可能であることを示した．加えて，モーションを保証するためのモーションの評価は，保証されたモーションからの変更点に着目した評価でよいことを述べた．また，機能追加に際して，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャの基本構造に変更はないため，提案する多関節ロボットのモーション実行用ソフトウェアアーキテクチャは拡張性が高いといえる．

以上より，多関節ロボットのモーション実行用ソフトウェアの設計指針となるアクチュエータ制御システムの基本構造である多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，多関節ロボットが多機種に展開され量産される際に，モーションを保証するためのモーションの評価においてモーションの変更点に着目した評価を可能にする．これは，量産に成功し人間の生活に密着している自動車と同様の評価手段を，多関節ロボットでも用いることが可能であることを意味する．そのため，多関節ロボットのモーション実行用ソフトウェアアーキテクチャは，多関節ロボットが多機種に展開され量産される場合においてモーションの評価を繰り返し行いやすくし，共通のソフトウェア構造を用いることによってモーションの保証に対しての信頼性確保を可能にするのみではなく，量産に成功し人間の生活に密着している自動車と同様の評価手段を用いることが可能である．このことから，多関節ロボットが多機種に展開され量産される際のモーション実行用ソフトウェア開発やモーション評価手段の土台になるといえる．また，モーションの変更点に着目した評価が可能であることから，モーションの評価項目の明確化につながる．これによって，評価に要する労力を削減することが可能である．その結果，より評価に注力することができ，モーションの安全性の確保や新しい機種への展開に貢献できるといえる．

5.2 今後の展望

ロボットは多くの技術の集まりである．ロボットの技術は，古くから開発されてきた数多くの技術要素を組み合わせているため，技術の共有には工夫が必要である．技術の共有を比較的容易にする工夫として，技術要素そのものや技術要素の組み合わせ方法，基本構造などの概念の提唱やルール化が挙げられる．技術要素そのものや技術要素の組み合わせ方法，基本構造などの概念の提唱やルール化を行うことで，技術要素は再利用可能な形で纏め上げていくことが可能であり，技術の共有に役立つ．技術要素を再利用可能な形で纏め上げたものとして画像処理技術ライブラリの OpenCV があるが，OpenCV の普及によって様々な人たちが画像処理に携わることが可能になった．大きな枠組みで捉えると，第 2 章で紹介した RT ミドルウェアや ROS も同じことがいえる．ルール化され再利用可能な形で纏め上げていくことで技術の共有が可能となり，その技術の評価にも役立つことから，評価された技術は基盤技術として扱われる．

本論文で提案した多関節ロボットのモーション実行用ソフトウェアアーキテクチャ

は，多関節ロボットのモーション実行用ソフトウェアの設計指針となるアクチュエータ制御システムの基本構造である．言い換えれば，多関節ロボットの技術要素のうちアクチュエータ制御を行うための一つの概念およびルールである．そのため，アクチュエータ制御の技術の共有や評価に役立つのみではなく，比較的容易に多関節ロボットのモーション実行システムの構築し，多関節ロボットのモーションを視認することが可能である．そのため，構築したモーション実行システムを用いた技術の共有の一つとして，教育目的の利用が可能であると考えている．たとえば，座学によって学習する機会の多い順運動学や逆運動学などについて，実際のロボットもしくは仮想空間でのロボットにおいて確認することが可能である．モーションが実行されるという分かりやすい状況を作ることが可能なことを利用するならば，プログラミングの結果を分かりやすく視認できるためプログラミング教育にも役立つ．そこで，今後は本論文で提案したアーキテクチャを用いたシステムによる教育やその他技術共有を行っていくことで，用途によるルールを抽出することについて取り組みたいと考えている．また，本論文では，提案するアーキテクチャを用いてモーション実行用ソフトウェアを開発した場合の利点として，モーションの変更点に着目したモーションの評価が可能であると述べた．しかしながら，モーションを保証するための評価であるモーションの評価の項目，内容，手段などについては述べていない．そのため，今後はモーションの評価の項目，内容，手段などについても追究したいと考えている．

本論文で提案したアーキテクチャは，多関節ロボットのアクチュエータ制御システムの基本構造であり，技術要素のうちアクチュエータ制御を行うための一つの概念およびルールであるため，基盤技術になり得る．今後，本論文で提案した多関節ロボットのモーション実行用ソフトウェアアーキテクチャが多関節ロボットの基盤技術として活用されることを望む．

謝辞

本学位論文は，中京大学大学院 情報科学研究科情報認知科学専攻 博士後期課程に入学してからの成果をまとめたものです．ここに，本研究を遂行するにあたり，私に携わってくださったすべての方々に，感謝の意を表します．

本学位論文の主査である中京大学大学院 情報科学研究科 青木公也教授，副査である中京大学大学院 情報科学研究科 沼田宗敏教授，橋本学教授，清水優准教授，平名計在准教授に感謝の意を表します．本学位論文の執筆にあたり，本学位論文の主査，副査を引き受けてくださったことのみでなく，本学位論文の内容に対して有益なご指導，ご助言をいただきました．先生方のご指導，ご助言がなければ，本学位論文の研究成果はなかったと思います．深く感謝いたします．

私は，2001 年 4 月に中京大学情報科学部に入学した後，2003 年度から 2005 年度の同大学大学院 情報科学研究科情報科学専攻 博士前期課程修了まで清水優准教授の研究室に在籍しました．社会人を経てから 2011 年 4 月に同大学大学院 同研究科情報認知科学専攻 博士後期課程に進学した後も，清水優准教授の研究室に在籍し，ロボットモーションに関する研究に従事しました．この長い期間において，ご指導，ご助言をいただいた清水優准教授には，言葉に表すことができないほどの深い感謝の思いがあります．博士という学位を取得するまで見届けてくださったこと，心より感謝いたします．また，修士論文の副査を引き受けてくださり，社会人から博士後期課程への進学について背中を押してくださった中京大学情報理工学部 嶋田晋 元教授，嶋田晋 元教授のもとで研究者の道を歩み始めた友人の早瀬光浩氏に深く感謝いたします．さらに，私との打ち合わせの時間をご用意してくださり，研究者としてのあるべき姿をご指導くださった橋本学教授，博士論文執筆にあたり私の学位論文のあるべき姿をご指摘くださった平名計在准教授に深く感謝いたします．そして，日ごろから私に携わってくださった沼田宗敏教授，青木公也教授に深く感謝いたします．沼田宗敏教授は，日ごろから私にお声をかけてくださり，そのたびに前に進む力をいただきました．深く

感謝いたします．青木公也教授は，博士前期課程進学についての相談や修士論文の副査を引き受けてくださり，博士後期課程進学後においては本学位論文の主査を引き受けてくださったことのみではなく，ひとりの友人として相談や雑談にお付き合いいただきました．深く感謝いたします．

最後に，本学位論文をまとめるにあたり，私に携わってくださった中京大学工学部各研究室の卒業生や大学院生，学部生の皆様に感謝いたします．また，教鞭を執る機会を下さった各学校の関係者の方々，私を先生と慕ってくれたすべての教え子たちに感謝いたします．そして，応援してくれた友人たち，私の研究活動を見守り続けてくれた両親にお礼を述べます．

参 考 文 献

[1] 独立行政法人 新エネルギー・産業技術総合開発機構 : NEDO ロボット白書 2014 社会を変えようとするとき、そこにロボット技術がある！ 第 1 章 (2014).

[2] George A. Bekey:自律ロボット概論（松田晃一・細部博史訳）, 毎日コミュニケーションズ　(2007).

[3] 青木猛，堀内淳史，鈴木達也，大熊繁：階層ファジィルールと学習オートマトンを用いた自律移動ロボットの障害物回避動作計画，日本ロボット学会誌， Vol. 14, No. 5， pp. 685–693 (1996).

[4] 近藤浩一，木村文彦：迷路法に基づく自由空間算出法を用いた障害物回避動作計画，日本ロボット学会誌， Vol. 5， No. 4， pp. 263–271 (1987).

[5] 近藤浩一 : 複数探索戦略を用いた自由空間算出による障害物回避動作計画，日本ロボット学会誌， Vol. 7， No. 4， pp. 352–362 (1989).

[6] 内田雅文，井出英人，横山修一 : 動的環境における障害物領域予測法，電気学会論文誌 C， Vol. 117-C， No. 2， pp. 198–204 (1997).

[7] 小方博之，新井民夫，太田順 : 時変環境でユーザ仕様を考慮した移動ロボットの軌道計画法，日本ロボット学会誌， Vol. 12， No. 6， pp. 905–910 (1994).

[8] 新井民夫，太田順 : 仮想的なインピーダンスを用いた複数移動ロボット系の動作計画，日本ロボット学会誌， Vol. 11， No. 7， pp. 1039–1046 (1993).

[9] 太田順，新井民夫，倉林大輔 : 作業の性質を考慮したロボット群の動作計画，日本ロボット学会誌，Vol. 13， No. 4， pp. 517–524 (1995).

[10] 倉林大輔，太田順，新井民夫，吉田英一 : 掃引作業における移動ロボット群の動作計画，日本ロボット学会誌， Vol. 16， No. 2， pp. 181–188 (1998).

[11] JIS　B　0134:2015　ロボット及びロボティックデバイス－用語, http://kikakurui.com/b0/B0134-2015-01.html

[12] 長谷川勉：自由空間分類表現法によるマニピュレータの衝突回避動作の計画，計測自動制御学会論文集， Vol. 22， No. 6， pp. 616–622 (1986).

[13] 長谷川勉，寺崎肇：中間ゴールを用いた分割探索による衝突回避動作の計画，計測自動制御学会論文集， Vol. 23， No. 8， pp. 842–848 (1987).

[14] 近藤浩一：ヒューリスティックなグラフ探索にもとづく自由空間算出法による障害物回避動作計画，日本ロボット学会誌， Vol. 6， No. 6， pp. 489–498 (1988).

[15] 丁全鋼，湯浅秀男，伊藤正美：関節空間における障害物回避パス探索の一方法，計測自動制御学会論文集， Vol. 26， No. 2， pp. 219–224 (1990).

[16] 井上雄紀，吉村俊秀，北村新三：大局的経路目標とヒューリスティックなグラフ探索を用いたマニピュレータの障害物回避アルゴリズム，計測自動制御学会論文集，Vol. 27， No.8， pp. 922–928 (1991).

[17] 長谷川勉，音田弘，松井俊浩：作業環境の空間構造解析に基づくマニピュレータの障害物回避動作計画，計測自動制御学会論文集， Vol. 27， No. 1， pp. 122–128 (1991).

[18] 登尾啓史，渡辺元彦，藤井剛：マニピュレータの連続的な障害物回避動作を生成する実際的アルゴリズム，計測自動制御学会論文集，Vol. 26，No. 12，pp. 1435–1442 (1990).

[19] 柴田昌明，村上俊之，大西公平：歪めコンフィグレーション空間に基づくマニピュレータのための障害物回避軌道の高速計算法，電気学会論文誌 D， Vol. 115-D，No. 12， pp. 1476–1483 (1995).

[20] 尾崎弘明，林長軍：コンプレックス法による動力学を考慮したマニピュレータの障害物回避軌道生成，日本ロボット学会誌， Vol. 15， No. 1， pp. 139–144 (1997).

[21] 田中良幸，辻敏夫，金子真：人間の上肢運動を模擬したロボットの軌道生成法，日本ロボット学会誌， Vol. 18， No. 5， pp. 699–705 (2000).

[22] 樹野淳也，梅木嘉道，石田祐樹，小林尚登 : 帯電ワイヤを用いた多自由度マニピュレータの障害物回避軌道の生成，電気学会論文誌 C，Vol. 125，No. 2，pp. 308–313 (2005).

[23] 田中敏幸，川口洋史，佐藤力：非線形計画法に基づく軌道計画のためのスプライン法，計測自動制御学会論文集，Vol. 28，No. 11，pp. 1321–1324 (1992).

[24] 尾崎弘明，丘華：移動障害回避のためのマニピュレータ軌道の局所修正法，計測自動制御学会論文集，Vol. 29，No. 1，pp. 122–124 (1993).

[25] 辻敏夫，Pietro G. Morasso，重橋薫，金子真 : 収束時間を調整可能な人工ポテンシャル法によるマニピュレータの動作計画，日本ロボット学会誌，Vol. 13，No. 2，pp. 285–290 (1995).

[26] 佐竹利文，林朗弘，鬼塚昭一，鈴木裕 : セル集団の自己組織化を適用したロボットアームの連続的な姿勢変化の生成，精密工学会誌，Vol. 62，No. 10，pp. 1415–1419 (1996).

[27] 岩村誠人，山本元司，毛利彰：非ホロノミック Caplygin System の近似最適軌道計画，日本ロボット学会誌，Vol. 17，No. 5，pp. 742–749 (1999).

[28] 寺崎肇，長谷川勉，高橋裕信：平行 2 指ハンドによる多面体物体の運搬作業のための把握動作計画，日本ロボット学会誌，Vol. 10，No. 2，pp. 273–282 (1992).

[29] 寺崎肇，長谷川勉：平行 2 指ハンドによる滑らし操作を利用した知的物体操作のための動作計画，日本ロボット学会誌，Vol. 12，No. 7，pp. 1056–1065 (1994).

[30] 寺崎肇，長谷川勉：指先に回転機構を有する平行 2 指ハンドによる知的物体操作のための動作計画，日本ロボット学会誌，Vol. 13，No. 7，pp. 1044–1052 (1995).

[31] 河原崎徳之，長谷川勉，西原主計：障害物環境下における多指ハンド・アームの把握計画，日本ロボット学会誌，Vol. 17，No. 3，pp. 449–456 (1999).

[32] 八島真人 : 動的多指可操作度に基づくサブゴールの優先的選択によるマニピュレーション計画法，計測自動制御学会論文集，Vol. 42，No. 4，pp. 452–460 (2006).

[33] 毛利彰，平野剛，山本元司：2 台のマニピュレータの協調動作経路計画，計測自動制御学会論文集， Vol. 34， No. 8， pp. 935–940 (1998)．

[34] 寺田英嗣，輻形和幸 : 風呂敷包み作業用マルチロボットシステムの運動計画法，精密工学会誌, Vol. 76， No. 5， pp. 546–551 (2010)．

[35] Richard P. Paul, Bruce Shimano:Kinematic Control Equations for Simple Manipulators, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 11, Issue 6, pp. 449–455 (1981).

[36] C. S. G. Lee and M. Ziegler:A Geometric Approach in Solving the Inverse Kinematics of PUMA Robots, IEEE Transactions on Aerospace and Electronic Systems, Vol. 20, Issue 6, pp. 695–706 (1984).

[37] J. M. Hollerbach, K. Suh:Redundancy Resolution of Manipulators through Torque Optimization, IEEE Journal on Robotics and Automation, Vol.3, No.4, pp.308—316 (1987).

[38] T. shimizu and H. Kawasaki:An Analysis for Inverse Kinematics of Robot Manipulators using Grobner basis, Journal of Robotics and Mechatronics Vol. 9, No. 5, pp. 324–331 (1997).

[39] L. G. Herrera-Bendezu, E. Mu, J. T. Cain:Symbolic Computation of Robot Manipulator Kinematics, Proc. IEEE Robotics and Automation, pp. 993–998 (1988).

[40] 遠山 茂樹, 波多江 茂樹:数式処理によるロボットの逆運動学問題の解法 (第 1 報), 精密工学会誌, Vol. 58, No. 1, pp. 151–156 (1992).

[41] 中村 仁彦, 花房 秀郎:関節形ロボットアームの特異点低感度運動分解, 計測自動制御学会論文集，Vol.20, No.5, pp.453—459 (1984).

[42] 米田 完:脚移動, 日本ロボット学会誌, Vol. 16, No. 7, pp. 897–901 (1998).

[43] 梶田 秀司:ゼロモーメントポイント (ZMP) と歩行制御, 日本ロボット学会誌, Vol. 20, No. 3, pp. 229–232 (2002).

[44] 比留川 博久, 加賀美聡:ヒューマノイドの知能-個としての知能-　, 日本ロボット学会誌, Vol. 20, No. 5, pp. 478–481 (2002).

[45] 池浦良淳，中里　央，猪岡光：人間の舞踊動作に基づくダンシングロボットの動作計画，日本ロボット学会誌， Vol. 15， No. 6，pp. 927–933 (1997).

[46] 長崎高巳，梶田秀司，横井一仁，金子健二，比留川博久，谷江和雄:ヒューマノイドのための走行パターンの生成，日本ロボット学会誌， Vol. 21， No. 8， pp. 902–908 (2003).

[47] 大須 賀公一:ロボットの制御, 日本ロボット学会誌, Vol. 16, No. 7, pp. 882–885 (1998).

[48] P.-B. Wieber, and C. Chevallereau:Online adaptation of reference trajectories for control of walking systems, INRIA Research Report, no. 5298 (2004).

[49] S. Schaal and C. G. Atkeson:Robot Juggling:An Implementation of Memory-based Learning, IEEE Control Systems Magazine, Vol. 14, Issue 1, pp. 57–71 (1994).

[50] Darrin C. Bentivegna, Christopher G. Atkeson and Gordon Cheng:A Framework for Learning from Observation Using Primitives, Journal of Robotics Society of Japan, Vol. 22, No. 2, pp. 28–33 (2004).

[51] S. Miyakoshi and G. Cheng:Utilizing Physical Relationships for Biped Walking Control:a preliminary study in identifying key essential properties for the two suppuort phases, Proc. of CLAWAR, pp. 543–550 (2003).

[52] 宮腰 清一:メモリ・ベースト運動制御による 2 足歩行の制御, 日本ロボット学会誌, Vol. 24, No. 5, pp. 623–631 (2006).

[53] 北垣 高成, 末廣 尚士, 神徳 徹雄, 平井 成興, 谷江 和雄:RT ミドルウェア技術基盤の研究開発について-ロボット機能発現のために必要な要素技術開発-　, 第 8 回ロボティクスシンポジア予稿集, pp. 487–492 (2003).

[54] 神徳 徹雄, 北垣 高成, 安藤 慶昭, 尹 祐根, 末廣 尚士:RT ミドルウェアのソフトウェア開発支援機能の検討, 第 9 回ロボティクスシンポジア予稿集, pp. 282–287 (2004).

[55] 安藤 慶昭, 末廣 尚士, 北垣 高成, 神徳 徹雄, 尹 祐根:RT 要素のモジュール化および RT コンポーネントの実装, 第 9 回ロボティクスシンポジア予稿集, pp. 288–293 (2004).

[56] 原 功:RT ミドルウェアによるロボットシステム構築, 日本ロボット学会誌, Vol. 28, No. 5, pp. 562–563 (2010).

[57] 松本 吉央:RT ミドルウェアによるロボットアーキテクチャ 移動ロボットシステム, 日本ロボット学会誌, Vol. 28, No. 5, pp. 564–565 (2010).

[58] 松坂 要佐:RT ミドルウェアによるロボットアーキテクチャ コミュニケーションシステム, 日本ロボット学会誌, Vol. 28, No. 5, pp. 566–567 (2010).

[59] 末廣 尚士:RT ミドルウェアによるロボットアーキテクチャ マニピュレーションシステム, 日本ロボット学会誌, Vol. 28, No. 5, pp. 568–569 (2010).

[60] 橋本 秀紀:遠隔微細操作システムに関する研究, 日本ロボット学会誌, Vol. 25, No. 2, pp. 209 (2007).

[61] 安藤 慶昭, 中岡 慎一郎, 神徳 徹雄:分散コンポーネント型ロボットシミュレータ・アーキテクチャ—RT コンポーネントを用いた実機と可換な制御ソフトウェア開発機能—, 日本ロボット学会誌, Vol. 26, No. 5, pp. 407–410 (2008).

[62] 金広 文男:RT ミドルウェアと OpenHRP3 によるロボットシミュレーション, 日本ロボット学会誌, Vol. 28, No. 5, pp. 556–561 (2010).

[63] 神徳 徹雄:RT システムインテグレーション, 計測と制御, Vol. 44, No. 11, pp. 761–764 (2005).

[64] 安藤 慶昭:初心者のための RT ミドルウェア入門—OpenRTM-aist-1.0 とその使い方—, 日本ロボット学会誌, Vol. 28, No. 5, pp. 550–555 (2010).

[65] 佐藤 知正, 岡野 克弥:RT ミドルウェアと知能モジュール構築プロジェクト, 日本ロボット学会誌, Vol. 28, NO. 5, pp. 546–549 (2010).

[66] 成田 雅彦, 島村 真巳子, 日浦 亮太, 山口 亨:ロボットサービスイニシアチブ (RSi) の活動を通して実現したロボットサービス共通プラットフォーム仕様, 日本ロボット学会誌, Vol. 26, No. 7, pp. 785–793 (2008).

[67] 菅野 重樹:RT ビジネスの戦略, 日本ロボット学会誌, Vol. 24, No.3, pp. 278–283 (2006).

[68] 小笠原 哲也:RT ミドルウェアによる再利用性向上とビジネス展開, 日本ロボット学会誌, Vol. 28, No. 5, pp. 574–576 (2010).

[69] 末廣 尚士:ロボット関連技術の国際標準策定活動:現状と展望, 日本ロボット学会誌, Vol. 29, No. 4, pp. 318–320 (2011).

[70] 安藤 慶昭:OMG における Robotic Technology Component(RTC) および関連仕様の標準化動向, 日本ロボット学会誌, Vol. 29, No. 4, pp. 333–336 (2011).

[71] 橋本 秀紀, 新妻 実保子, 佐々木 毅:空間知能化―インテリジェント・スペース, 日本ロボット学会誌, Vol. 23, No. 6, pp. 674–677 (2005).

[72] 大場 光太郎, 大原 賢一:ユビキタス・ロボティクス, 日本ロボット学会誌, Vol. 25, No. 4, pp. 505–508 (2007).

[73] 大場 光太郎:環境と作業構造のユニバーサルデザイン, 日本ロボット学会誌, Vol. 26, NO. 5, pp. 431–435 (2008).

[74] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler and A. Y. Ng, :ROS:an open-source Robot Operating System, ICRA Workshop on Open Source Software (2009).

[75] 岡田 慧:ROS（ロボット・オペレーティング・システム）, 日本ロボット学会誌, Vol. 30, No. 9, pp. 830–835 (2012).

[76] 岡田 慧:ROS, 5 年経って, 日本ロボット学会誌, Vol. 35, No. 4, pp. 270–273 (2017).

[77] 詳説 ROS ロボットプログラミング―導入から SLAM、Gazebo、MoveIt まで―

[78] 原 祥尭:ROS を用いた自律走行, 日本ロボット学会誌, Vol. 35, No. 4, pp. 286–290 (2017).

[79] 木内 裕介, 小島 弘義, 橋本 達矢, 小野 敬之, 小山 貴之, 橋本 健二, 高西 淳夫:三菱重工における ROS/Gazebo を活用したロボット, 日本ロボット学会誌, Vol. 35, No. 4, pp. 276–279 (2017).

[80] 渡辺 敦志:ROS を用いた石油プラント点検ロボット開発—ROS Ecosystem 活用と独自実装の選択—, 日本ロボット学会誌, Vol. 35, No. 4, pp. 291–294 (2017).

[81] 出村 公成, 出村 賢聖:RoboCup@Home における ROS の利用, 日本ロボット学会誌, Vol.35, No.4, pp. 295–298 (2017).

[82] 寺田 耕志, 村瀬 和都, 山本 貴史:生活支援ロボット Human Support Robot(HSR) における ROS 活用, 日本ロボット学会誌, Vol. 35, No. 4, pp. 260–283 (2017).

[83] 宮越 喜浩:デンソーロボットの ROS インタフェース, 日本ロボット学会誌, Vol. 35, No. 4, pp. 284–285 (2017).

[84] 倉爪 亮, ピョ ユンソク, 辻 徳生, 川村 晃宏:情報構造化環境プラットフォーム ROS - TMS と Big Sensor Box の提案, 日本ロボット学会誌, Vol. 35, No. 4, pp. 346–357 (2017).

[85] 山内 悠嗣, 藤吉 弘亘, 大日方 五郎:ROS をベースにしたロボット工学教育, 日本ロボット学会誌, Vol. 35, No.4, pp. 299–302 (2017).

[86] マルコ チャシン, 木川田 亘, リモ ピラット:ROS を使いやすく—Robotics System Toolbolx による MATLAB・Simulink と ROS の統合—, 日本ロボット学会誌, Vol. 35, No. 4, pp. 303–306 (2017).

[87] Min Lin Chan and Paul Hvass:ROS-Industrial:Expanding Industrial Application Capabilities, Journal of the Robotics Society of Japan, Vol. 35, No. 4, pp. 307–310 (2017).

研究業績

学術論文

1. 加藤央昌, 清水優: モーションデータの再利用性向上のためのロボットモーション実行基盤の機能拡張, **精密工学会誌**, Vol. 83, NO. 5, pp. 460–467 , 2017.
2. 加藤央昌, 石原裕平, 清水優, 橋本学: ロボットモーションプランニングの自動化に向けてのロボットモーション実行基盤の開発, **精密工学会誌** Vol. 80, NO. 1, pp. 99–106, 2014.

国際会議

1. Hiroaki Kato, Tomohiro Tsuruta, Yuhei Ishihara, Masaru Shimizu, Manabu Hashimoto: Development of Robot Motion Performance Platform for Auto Generation of Robot Motion Planning, *SICE Annual Conference2012(SICE2012)*, pp. 1685–1690, 2012.

その他

第１著者

1. 加藤央昌， 清水優， 橋本学: ロボットモーションを利用したC言語学習の提案—初学者の学習意欲を向上させる学習方法—, 平成27年度電気・電子・情報関係学会東海支部連合大会予稿集, pp. A3-1, 2015.
2. 加藤央昌， 清水優， 橋本学: ロボットモーション実行基盤のロボット教育への活用の提案, 第32回日本ロボット学会学術講演会予稿集, pp.RSJ2014AC1Q1-08,

2014.

3. 加藤央昌, 石原裕平, 清水優, 橋本学: ロボットの腕脚自動選択システムの提案—選択マッチングパラメータの検討—, 第 17 回知能メカトロニクスワークショップ IMEC2012 予稿集, pp. M3-3, 2012.
4. 加藤央昌, 石原裕平, 清水優, 橋本学: ロボットモーションプランニングの自動化に向けてのロボットモーション実行基盤の開発, 第 35 回人工知能学会 AI チャレンジ研究会予稿集, pp. 13–18, 2012.
5. 加藤央昌, 鶴田智寛, 石原裕平, 清水優, 橋本学: ロボットモーションプランニングの自動化に向けてのロボットモーション実行基盤の拡張, 第 54 回自動制御連合講演会予稿集, pp. 849–854, 2011.
6. 加藤央昌, 鶴田智寛, 石原裕平, 清水優, 橋本学: ロボットモーションプランニングの自動化に向けてのロボットモーション実行基盤の開発, 2011 年度精密工学会秋季大会学術講演会予稿集, pp. 960–961, 2011.
7. 加藤央昌, 鶴田智寛, 石原裕平, 清水優, 橋本学: ロボットモーションプランニングの自動化に向けてのセンサフィードバックを考慮したロボットモーション実行基盤, 第 29 回日本ロボット学会学術講演予稿集, pp. RSJ2011AC2J1-1, 2011.
8. 加藤央昌, 日浦一彰, 信原卓弥, 清水優: 基本動作パターンとセンサフィードバックによるモーションコントロールシステムの評価, 平成 18 年度電気関係学会東海支部連合大会予稿集, pp. O-195, 2006.

第 1 著者以外

1. 岡田正悟, 加藤央昌, 清水優: レスキューロボット訓練用マップにおける障害物の難易度調査, 平成 26 年度電気・電子・情報関係学会東海支部連合大会予稿集, pp. P2-3, 2014.
2. 堂本尭大, 加藤央昌, 清水優: 実在する建造物データを用いたロボット操縦訓練環境の検証, 平成 26 年度電気・電子・情報関係学会東海支部連合大会予稿集, pp. P2-4, 2014.
3. 石原裕平, 加藤央昌, 清水優: 時間逆行機能を考慮した物理シミュレータの実現, 平成 24 年度電気関係学会東海支部連合大会予稿集, pp. B3-1, 2012.
4. 鶴田智寛, 加藤央昌, 石原裕平, 清水優: CAD モデルデータの物理シミュレータ内

オブジェクトへの自動変換手法の提案, 平成 23 年度電気関係学会東海支部大会予稿集, pp. P5-1, 2011.

5. 若林勲, 遠藤直弥, 佐々木健翔, 早瀬光浩, 加藤央昌, 清水優: 食事支援ロボットのための形状モデルを用いた皿上の料理位置推測, 情報処理学会第 73 回全国大会予稿集 (pp.), pp. 1T-4, 2011.
6. 清水優, 日浦一彰, 加藤央昌, 信原卓弥: 4 足レスキューロボットのためのモータ負荷情報を用いた障害物対応脚モーション制御手法の提案, ロボティクス・メカトロニクス講演会 2007 予稿集, pp. 2P1-M01, 2007.
7. 清水優, 日浦一彰, 加藤央昌, 信原卓弥: 赤外線画像を用いた被災者探索のための画像処理手法の提案, 電子情報通信学会 2007 総合大会予稿集, pp. D-12-153, 2007.
8. 信原卓弥, 日浦 一彰, 加藤央昌, 清水優, 伊藤誠: 屋内探索を目的とした連結型飛行船群の提案と開発, 第 12 回ロボティックスシンポジア予稿集, pp. 448-453, 2007.
9. 信原卓弥, 日浦一彰, 加藤央昌, 清水優, 伊藤誠: 屋内探索を目的とした協調型飛行船群の提案と開発, 平成 18 年度電気関係学会東海支部連合大会予稿集, pp. O-193, 2006.