

Title	制御プログラム構造の違いによる実ロボット動作特性の評価
Author(s)	榎本, 俊数
Citation	
Issue Date	2000-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/629
Rights	
Description	Supervisor: 櫻井 彰人, 知識科学研究科, 修士

目 次

1	はじめに	1
1.1	背景と目的	1
1.2	本論文の構成	3
2	実験機材	4
2.1	MindStorms	4
2.1.1	R C X	4
2.1.2	タッチセンサと入力ポート	5
2.1.3	光センサ	6
2.1.4	出力ポート	7
2.1.5	モータ	7
2.1.6	I R タワー	8
2.2	加工品	8
2.2.1	ペンライト	8
2.2.2	接続ケーブルと接続ピン	9
3	3 方式の比較評価	10

10	3.1 比較対象とする 3 つの制御方式
10	3.1.1 集中制御
11	3.1.2 分散制御（通信有り）
11	3.1.3 分散制御（通信無し）
12	3.2 実験準備
12	3.2.1 ロボットに与える目標
13	3.2.2 ロボット製作
15	3.2.3 移動ロボットの機能
18	3.3 位置認識システムの可能性
18	3.3.1 実験方法
20	3.3.2 結果
26	3.4 考察
26	3.4.1 位置認識システム
26	3.4.2 失敗原因
4	4 制御プログラム構造の違いによる実ロボット動作特性の評価

28	
4.1	行動ベースアプローチ
28	
4.2	実験準備
29	
4.2.1	ロボットに与える目標
29	
4.2.2	ロボットの製作
30	
4.2.3	実験場
34	
4.2.4	実験条件
35	
4.3	ロボットの状況別行動
37	
4.3.1	状態と動作の記述
37	
4.3.2	状況 1 : 移動ロボット近辺に的無し (_N)
38	
4.3.3	状況 2 : 移動ロボット近辺に的有り (N)
39	
4.3.4	的ロボットの動作
40	
4.4	比較対象とする 4 つの構造
41	
4.4.1	1 タスク型
42	
4.4.2	2 タスク型
42	
4.4.3	複数タスク型

	43
4.4.4	OnlyTouch 型
	45
4.5	結果と考察
46	
4.4.1	動作特性
46	
4.4.2	統計処理によるデータの検証
	50
4.6	今後の課題
53	
4.6.1	問題点
	53
4.6.2	制約の緩和
53	
4.6.3	学習
53	
5	おわりに
54	
謝辞	
参考文献	

目 次

図 2.1	R C X	
	5	
図 2.2	タッチセンサと光センサ	6
図 2.3	Motor Power の出力波形	7
図 2.4	モータ	
	8	
図 2.5	I R タワー	8
図 2.6	ライトとの接続状態	9

図 3.1	移動ロボットの構成（上から）	13
図 3.2	移動ロボットの構成（横から）	14
図 3.3	I R 信号による敵と味方の識別	16
図 3.4	垂直距離	19
図 3.5	MindStorms 付属のシート	19
図 3.6	カラーパレット	20
図 4.1	subsumption architecture	29
図 4.2	移動ロボットの構成（上から）	30
図 4.3	移動ロボットの構成（横から）	31
図 4.4	的ロボットの構成（上から）	31
図 4.5	的ロボットの構成（横から）	32
図 4.6	I R 信号の伝達	33
図 4.7	実験場	34
図 4.8	移動ロボットの行動選択方式	36
図 4.9	ロボットの基本概念	

41

図 4.10 ダウンロード用インターフェイス

41

図 4.11 1 タスク型のプログラム構造

42

図 4.12 2 タスク型

43

図 4.13 1 タスク型と 2 タスク型の探索タスク構造

44

図 4.14 複数タスク型の探索タスク構造

44

図 4.15 複数タスク型

45

図 4.16 OnlyTouch 型

46

図 4.17 接触回数分布 (1 タスク型)

46

図 4.18 接触回数分布 (2 タスク型)

46

図 4.19 接触回数分布 (複数タスク型)

47

図 4.20 接触回数分布 (OnlyTouch 型)

47

図 4.21 接触効率の比較

48

図 4.22 実験達成回数の比較

49

図 4.23 タスク達成率の比較

49

表 目 次

表 2.1	RCX	5
表 2.2	入力ポートの動作モード	6
表 2.3	ペンライトの仕様	9
表 3.1	I R 信号リスト	
	16	
表 3.2	センサ別特性 1	
	21	

表 3.3 センサ別特性 2

22

表 3.4 センサ別特性 3

23

表 3.5 センサ別特性 4

24

表 3.6 色による光センサ値 1

25

表 3.7 色による光センサ値 2

25

表 3.8 色による光センサ値 3

25

表 4.1 一元分散分析表

51

第 1 章

はじめに

1.1 背景と目的

多くのロボット研究でシミュレーションが行われてきた。しかし現実世界には仮想世界とは違い様々なノイズや制約が存在しており、実ロボットはその影響を受ける可能性が非常に高い。このシミュレーション[8][10][12]と実ロボット実験[16][17][18][19]の結果の差は大きな問題になっており、この差に関する研究[2]やシミュレーションと実実験を両方行う研究[9][11][15]もある。したがって現実世界で動くロボットの研究には実際に実ロボットを使う必要があると考える。本研究では作成した制御プログラムを実ロボットに実装し研究を行う。

単独のロボットで解決できない問題を複数のロボットで解決するための研究[3][5][13][14]が行われてきた。複数のロボットの制御方法には大別して集中制御と分散制御の2つがあり、分散制御の中には通信を行う場合とそうでない場合がある。

集中制御は、ロボットが取得した全ての情報を一度中央に集め、そこで大局的に情報を処理して判断を行い、その結果を全てのロボットに伝えて制御を行う方式である。

分散制御は中央での一括したタスク処理を行わず、タスクを複数のロボットに分散して制御を行う方式である。そのため各ロボットの行動決定や動作は集中制御に比べて高速であり実時間性に優れている。

また分散制御の中には明示的に通信を行うものと、行わないものが有る。多くのロボット研究でロボットは通信機能を有しており、個々のロボットはセンシングにより得た情報を互いに通信しあっている。しかし、実環境での動作を考えた場合、

通信には様々な意味で大きなコストが掛かるので必ずしも実用的ではない。

3 つの制御方式の比較評価

今まで集中制御、分散制御（通信有り）、分散制御（通信無し）の 3 つの制御方式個別には様々な研究が行われてきたが、複数の実ロボットが動的な環境で動作する状況を考えた時にどの方式が優れているのか比較評価した研究は少ない。そこで本研究では複数の実ロボットによる競技により上述した 3 方式を比較し評価することにした。

しかし実験の準備段階であるロボットの製作過程で集中制御と分散制御（通信有り）のための機能を有するロボットがハード的制約より製作不可能であることがわかった。つまり 3 つの制御方式の比較は出来なかった。

制御プログラム構造の違いによる動作特性の比較評価

前述した 3 方式には既に説明したように一長一短があり一概にどれが優れているとは言えない。動的環境で動作する実ロボットを考えた場合、通信はコストが掛かるので必ずしも有効ではなく、また大局的な情報の取得も容易ではない。そこで本研究では通信コストが掛からず実環境での複数ロボットの動作に適していると考えられる分散制御（通信無し）に注目した。実ロボットへ実装する制御アーキテクチャとしては行動ベースアプローチ[1]を採用した。

分散制御（通信無し）によりロボットを制御する場合、ロボットの行動特性は制御プログラム構造に大きく依存することが経験的に分かっているが、どんな構造が適しているのか具体的な事は分かっていない。そこで本研究では自律移動ロボットに適した制御プログラム構造を明らかにするため、制御プログラムを実ロボットに実装し評価することを目的とする。

1.2 本論文の構成

本論文では 1 章で研究の概要を説明し 2 章で実験機材の説明を行う。3 章では初期の研究について実験方法を記した後、その失敗原因について考察する。4 章では後期の研究つまり本論文の主なテーマである「制御プログラム構造の違いによる動作特性の評価」について、実験方法や結果、考察、今後の問題について述べる。最後に 5 章で本研究の結論と意義を述べて終わりとする。なお本文中で本研究とは後期の研究を意味する。

第 2 章

実験機材

2.1 MindStorms

通常、実ロボット製作には多くの時間と費用がかかる。そこで今回はロボット製作が簡単で繰り返し部品の組み直しが可能な LEGO 社と MIT (マサチューセッツ工科大学) との提携により生まれた MindStorms[6][7][20][21]という製品を使用してロボットを製作した。MindStorms は部品であるブロックや、タッチセンサ、光センサを組み合わせることにより様々な機構を実現できる物理的自由度の高い製品である。また中央制御装置に当たる RCX は RIS という MindStorms 固有のビジュアル言語を使用して 3 つの入力ポートと 3 つの出力ポートを制御することが出来る。他の制御言語として ActiveX を通してプログラムを組み込む Visual Basic や Java (Beans API) 等があり本研究では Visual Basic6.0 を使用して制御プログラムを作成した。

MindStorms 1 パッケージには 727 個の LEGO ブロックと赤外線トランスミッター、ケーブル、光センサ、タッチセンサ、モータ、ギア、マニュアル、RIS のソフトウェア CD-ROM 等が含まれている。なお再実験を行いやすいようにロボット 1 体は MindStorms 1 パッケージで製作できるように配慮した。

以降に MindStorms の詳細を示す。

2.1.1 RCX

RCX は MindStorms でロボットを製作する時に制御装置となる部品であり、1 パ

ッケージに 1 つ入っている。図 2.1 から分かるように前部に I R 通信ポート、上部に操作ボタンと液晶パネルそれに 3 つの入力ポートと 3 つの出力ポートがある。後部にはアダプタ用電源ジャックが付いている。表 2.1 に RCX の仕様を記載する。

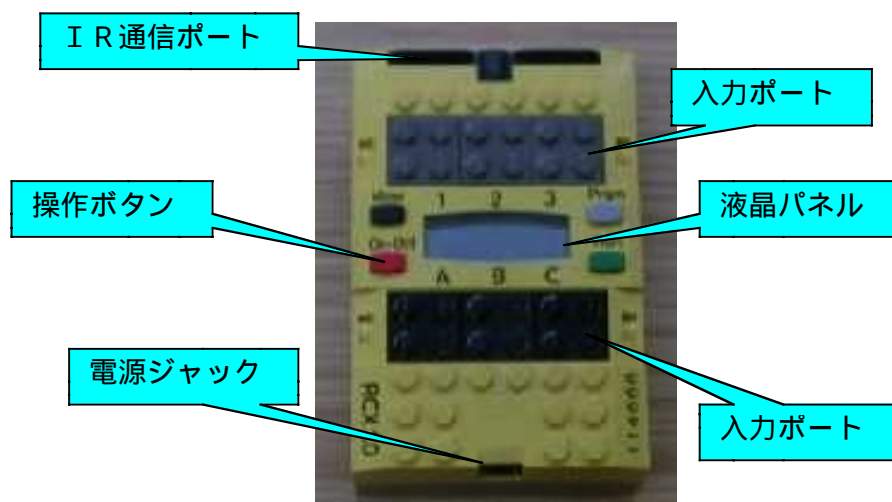


図 2.1 R C X

表 2.1 R C X

C P U	日立製 H8/3292 8bitCPU を搭載 16KByteROM を内蔵し firmware ダウンロード用ドライバが入っている
メモリ	32KbyteRAM には firmware とユーザープログラムを格納可能
電源	単 3 型電池 × 6 他に 9 ~ 12V / φ 2.5 の AC アダプタも使用可能

2.1.2 タッチセンサと入力ポート

タッチセンサはゲームコントローラのボタンのような構造をしている。動作も非常に簡単でタッチセンサが押されると、2 つの端子間の抵抗値が無限大から約 500 Ω に変化する。つまりタッチセンサが押されると、導電性のあるゴムがへこんでケース側の 2 つの金属端子に接触するため、端子間の抵抗が導電ゴムの抵抗になる。この入力ポートでの抵抗値の判定は次のようにして行う。

入力ポート内部回路の構成から入力ポートに何もつないでいない時やタッチセンサを繋いでいてもボタンを押していない時は、抵抗値は無限大となり、インプットポートには5Vがかかったままになる。入力ポートにはCPU内蔵のA/Dコンバータがあり0~5Vの電圧を10bitの分解能で読み取る。つまり0Vの時は0、5Vの時は1023という数値が返されCPUで演算に使われる。図2.2にタッチセンサと光センサを示す。

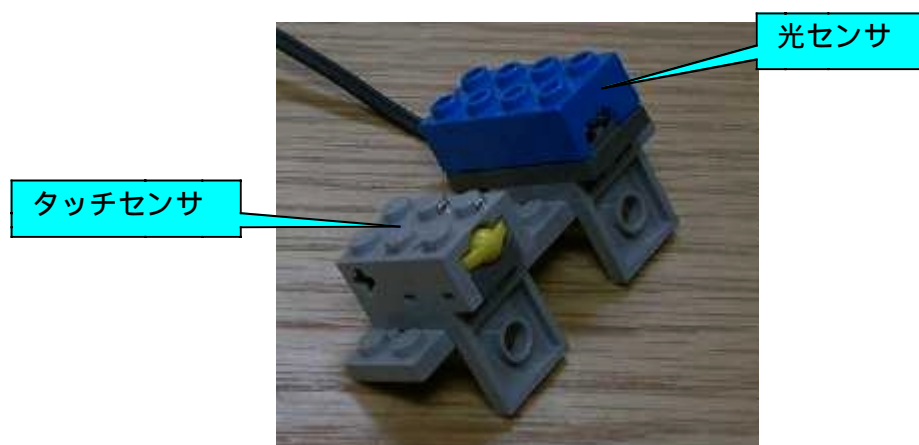


図 2.2 タッチセンサと光センサ

2.1.3 光センサ

光センサはタッチセンサと違いICやLEDを内蔵しているので、これらの素子に電力を供給しながら電圧も測定しなくてはならない。通常、この動作を行うために

表 2.2 入力ポートの動作モード

	タッチセンサモード	光センサモード
概要	抵抗値により電圧変化を読み取るモード	電力供給フェーズと電圧測定フェーズを交互に繰り返すモード
電力の供給	無し	有り
状況の表現	放したとき[0]/押した時[1]	<黒>0~100<白> 実際には20~70程度で変化

RCX の測定値の計算法	RCX の測定値は RAW value が 450 以下なら 1、565 以上で 0 と判定	RCX の測定値 = $164 - \text{RAW} / 7$ (ただし 0 ~ 100 の間で変化)
--------------	--	---

は電力供給線、グランド線、電圧測定線の最低 3 本も配線が必要になるが、RCX では電力を供給する「電力供給フェーズ」と電圧を測定する「電圧測定フェーズ」を分割して高速に切り替えることで、同様の動作を 2 本の配線で実現している。入力ポートの動作モードを表 2.2 に示す。

2.1.4 出力ポート

出力ポートは入力ポートに比べて簡単な動作になっている。RCX では Motor Power を 8 段階に調整でき、この制御を PWM 制御方式でコントロールしている。

PWM は「Pulse Width Modulation」の略で日本語ではパルス幅変調となる。これは ON 状態のパルス幅と OFF 状態のパルス幅の比を変えることによって電力をコントロールする方法である。図 2.3 に Motor Power が 2 の場合と 4 の場合の出力波形を示す。ここでは縦軸が電圧、横軸が時間を表している。

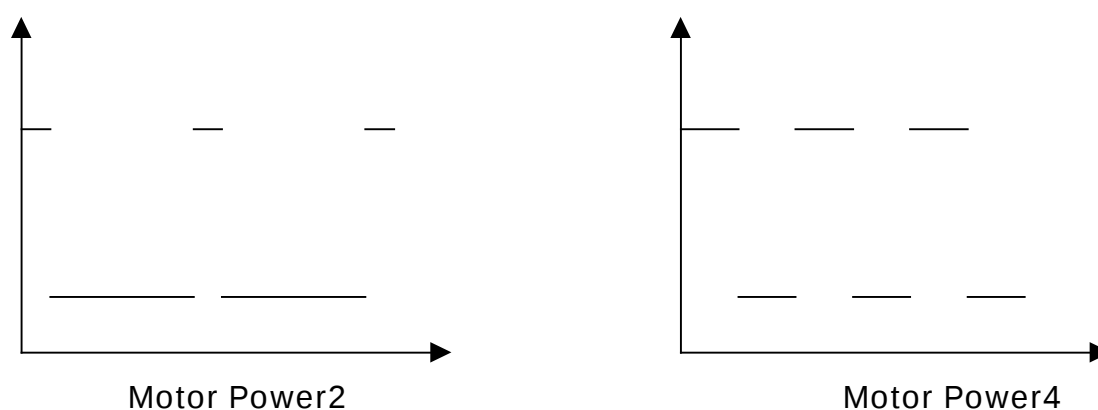


図 2.3 Motor Power の出力波形

2.1.5 モータ

図 2.4 のようなモータを使用している。Motor Power8 の場合無負荷時に約 4 mA、手で押さえて完全に回転を止めた状態だと約 350mA の電流を消費する。Motor

Power が 1 ～ 7 の時は 8 の時の $1/8 \sim 7/8$ の電流が流れる。



図 2.4 モータ

2.1.6 I R タワー

I R タワーは P C と RCX の通信に使用する装置で、通信は 38Kbit に変調された赤外線で行われる。通信速度は 2400bps で電源として 9V の 6P 型電池を一本使用する。図 2.5 に I R タワーを示す。



図 2.5 I R タワー

2.2 加工品

2.2.1 ペンライト

誘導用ライトとしては表 2.3 の製品を改造して使用した。

表 2.3 ペンライトの仕様

メーカー	富士電気化学株式会社
製品名	アルミニライト
使用電球	2.5V 0.3A MG5 球（クリプトンガス封入）
使用電池	単 4 型電池 × 2
電池寿命	約 2 時間 / アルカリ乾電池（連続使用）
寸法	φ18 × 136mm
重量	約 31g（電池別）

2.2.2 接続ケーブルと接続ピン

接続ケーブルには以下のようなより線コードを用いた。

φ12mm × 10 芯 VF8-80 色（赤）

接続ピンとしてはVクリップ（ゴールド）を用いた。

図 2.6 にライトとの接続状態を示す。



図 2.6 ライトとの接続状態

第 3 章

3 つの制御方式の比較評価

3.1 比較対象とする 3 つの制御方式

ここでは複数の実ロボットを対象とした競技により集中制御、分散制御（通信有り）、分散制御（通信無し）の 3 つの制御方式を比較評価する。

複数の実ロボットが存在する時の制御方法には大別して集中制御と分散制御がある。更に分散制御の中には通信を行う場合とそうでない場合があり合計 3 つの制御方式がある。

3.1.1 集中制御

集中制御は、ロボットが取得した全ての情報を一度中央に集め、そこで大局的に情報を処理して判断を行い、その結果を全てのロボットに伝えて制御を行う方式である。この方式は全ての情報を中央に集中的に集めて大局的な情報処理により行動を決定するので、比較的正確な判断が可能である。しかし全ての情報が集まる中央の処理速度が遅いとそれがボトルネックとなりシステム全体の速度が遅くなってしまふ。更に中央が故障してしまうと全てのロボットはシステムとして活動を続行できなくなるという欠点があり、システムとして堅牢さに欠ける。また実環境での動作を考えた場合、環境の大局的な情報の取得とその処理は必ずしも容易なことではない。

3.1.2 分散制御（通信有り）

分散制御は中央での一括したタスク処理を行わずタスクを複数のロボットに分散する制御方式である。そのため各ロボットの行動決定や動作は中央制御に比べて高速であり実時間性に優れている。またたとえロボットのうち 1 つが故障してもシステムとして活動不能に陥ることはなくシステムとして堅牢である。しかし大局的な情報処理からの判断を行えないため判断の正確さに欠ける。

3.1.3 分散制御（通信無し）

この方式も中央での一括したタスク処理を行わずタスクを複数のロボットに分散して制御する。そのため利点として高速な判断やそれに基づく動作やシステムの堅牢さ、欠点として判断の不確かさ等、先ほど述べた分散制御（通信有り）の特性を持ち合わせている。この方式は前述した「分散制御（通信有り）」とは違い個々のロボットは全く通信を行わないので、システムの拡張や記述が容易である。しかし個々のロボットは通信を行わないため環境を認識する情報は自らがセンシングによって得た情報しかなく、結果として自らの周りの小さな領域情報からしか判断を行えない。つまり大局的判断に基づいた動作が不可能なのである。しかし、実環境での動作を考えた場合、通信には様々な意味で大きなコストが掛かるので必ずしも有効とはいえない。

そこで初期の研究として、この 3 つの制御方式を複数の実ロボットを対象とした動作実験により評価を行うことにした。

3.2 実験準備

3.2.1 ロボットに与える目標

目標タスクとして複数ロボットによる相互2度押し問題を与え集中制御、分散制御（通信有り）、分散制御（通信無し）の3つの制御方式を評価する。

この問題はそれぞれ同じ制御方式を実装したロボット6体1チームを2チーム作り、各ロボットが2つずつ持っているスイッチを互いに押し合い、どちらかのチームが残り1体になるまで動作を続けるというものである。ロボットは探索、回避、追跡の3つの行動を組み合わせで競技を行う。組み合わせとして集中制御 v s 分散制御（通信有り）、分散制御（通信有り） v s 分散制御（通信無し）、集中制御 v s 分散制御（通信無し）の3パターンがある。この実験は以下に示すルールに従って行い、終了条件を満足するまでロボットは動作を続ける。

単体のルール

- 相手チームにより（以下、敵）二つのスイッチを正しい順序で押されたロボットは活動を停止する
- スwitchを押す順序は決まっており先に後スイッチを押し、次に一定時間（15秒）内に前スイッチを押し
- スwitchを押した時は必ずIR信号を発信し、それにより敵味方の識別を行う。（なお識別信号は表に従う）
- 活動停止になったロボットは競技が終了するまで活動を再開しない

全体のルール（終了条件）

- どちらかのチームのロボットが残り1体になる。
- 制限時間（10分）になる

なお、終了条件で制限時間を設けたのはどちらも敵を1体まで減らせなかった場合でも実験を終了させるためである。

この競技では味方チームのロボット（以下、味方）相手チームのロボットに付いている2つのスイッチを押すことを求められ、その際にもどちらのチームのロボットが押したのかIR信号により確認している。これには以下に示す2つの理由がある。

- 1) 偶然にロボットのスイッチが押されてロボットが活動停止してしまうのを防ぐため。偶然に1つのスイッチが押される事があっても、その後一定時間の間にもう1つのスイッチが押される事は確率的に低いと考えたからである。つまりスイッチを2つにする事によって偶然性により競技が左右されにくいようにしたのである。
- 2) 味方によってスイッチが押された時にその行動を無効にするため。競技中は複数の敵と味方が入り乱れて動作しているので、偶然仲間によってスイッチが押されてしまうことがあり、この誤った味方の攻撃により活動停止してしまうのを防ぐため。

3.2.2 ロボット製作

移動ロボットとして下図に示す物を製作した。

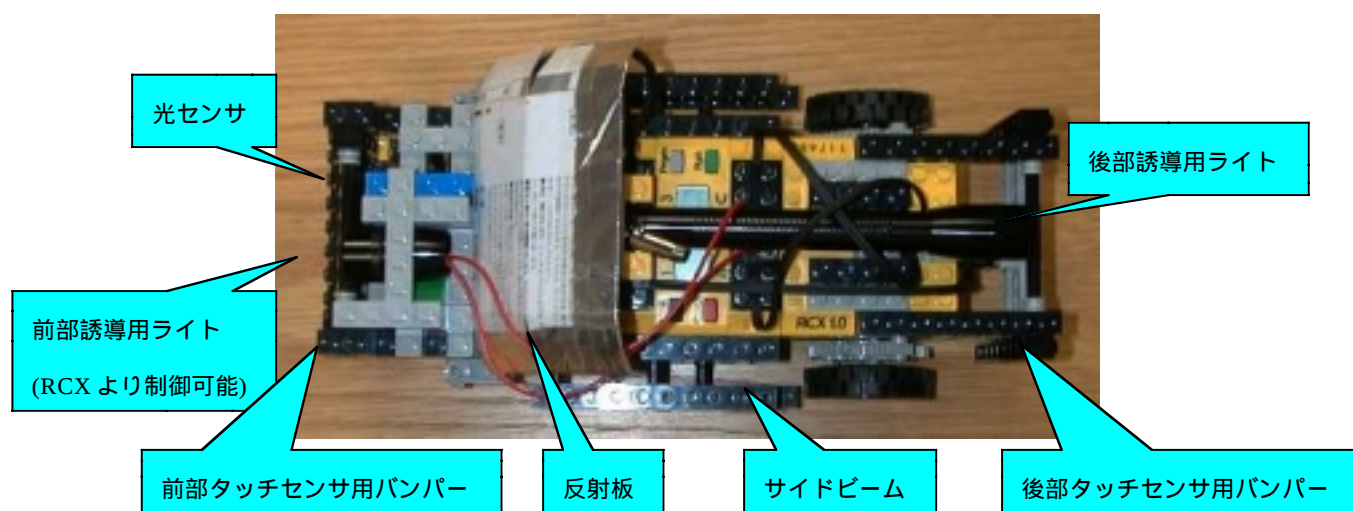


図 3.1 移動ロボットの構成（上から）

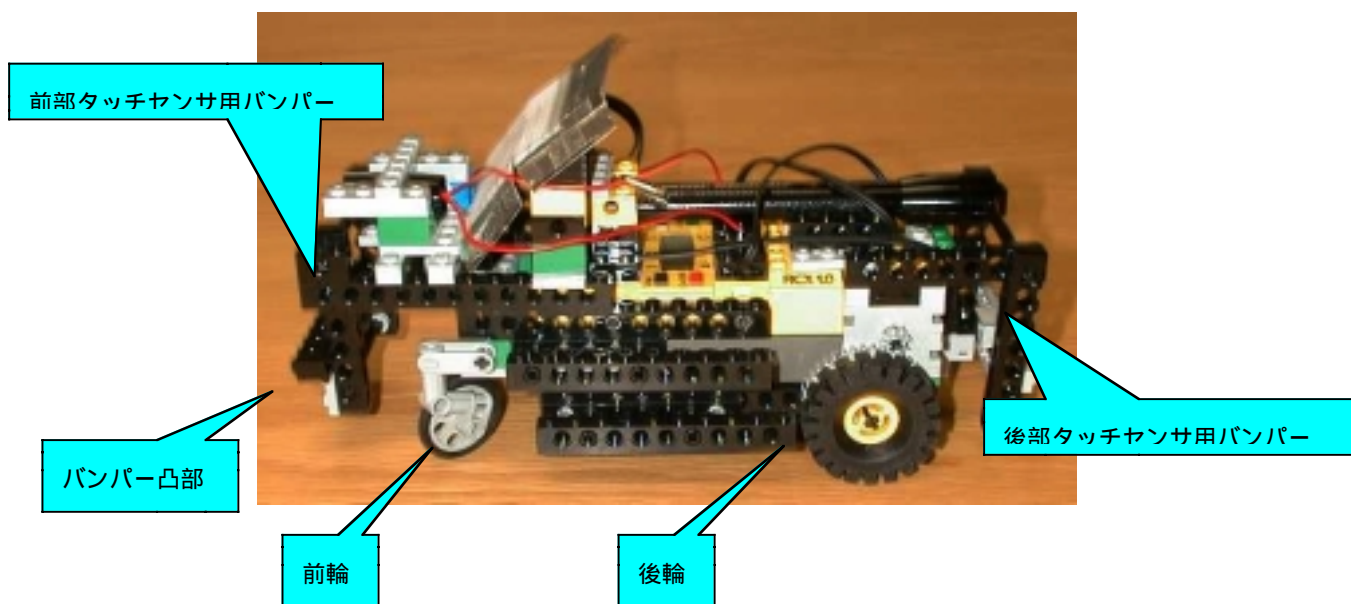


図 3.2 移動ロボットの構成（横から）

（１）機体製作時の注意点

今回、ロボットを作成するにあたって以下の点に注意した。

○機体の大きさ、形、強度

機体が旋回しやすいように全長をなるべく短くした。

動作時に車輪などで機体同士が引っ掛からないように側面に引っ掛かり防止ビームを設けた。

動作や衝突に絶えうる機体の物理的強度を確保するために、ブロックの接合強度に頼らない設計にした。

○タッチセンサ関連

ロボット同士が正確にタッチセンサを動作させやすいようにバンパーに凸部を設けた。また不必要な時センサが刺激されないようにセンサの動作部に間隙を設けた。

また故障率を下げるためにバンパーの押し戻しをタッチセンサのゴムに頼らないようにした。なおここでいうバンパーとは物理的力を間接的にタッチセンサへ伝える装置のことである。

○ I R センサ関連

ロボットの背後から I R 信号を受信するために機体上部に反射板を装備した。またある程度広い角度からも I R 信号を受け取れるようにするため反射板を曲面にした。送信側は機体上部の反射板に I R 信号を送信しやすいようにするため、センサの付いている機体前部を若干上方に傾けた。

○ ライト関連

誘導用ライトを R C X から制御するためペンライトを分解して R C X の出力ポートに接続した。出力ポートへの接続線の取り外しが容易になるように C 字型の取り付け金具を製作した。

3.2.3 移動ロボットの機能

(1) 3 つの制御方式の共通機能

それぞれの制御方式のロボットに共通して必要な機能は I R による通信機能である。

競技中、各ロボットは相手チームのロボット（以下、敵）の後部に付いているライトを目標にして敵に接近する。

ロボットの目標は相手チームのロボットのスイッチを押すことであるが、実験中は敵と味方が入り乱れて動作しているので、間違って味方にボタンを押されてしまう場合がある。そこで押したのが味方なのか敵なのか判断する必要がある。その判断のためにロボットには下図のような反射板を装備し I R（赤外線）信号により敵味方の識別を可能にした。

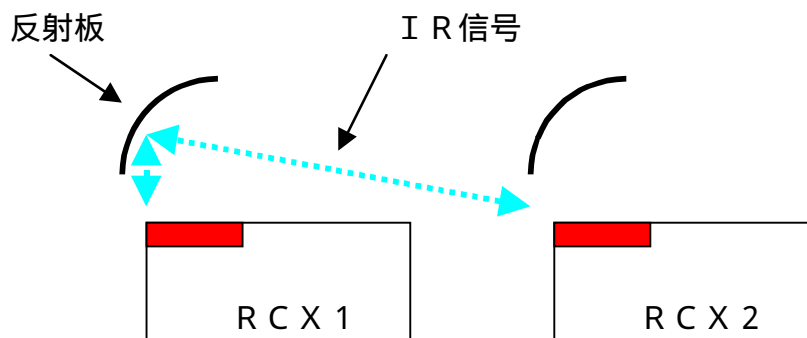


図 3.3 I R 信号による敵と味方の識別

図 3.3 で使用する I R 信号は以下の表 3.1 に定めたものを使用した。

表 3.1 I R 信号リスト

チーム	後スイッチ	前スイッチ	攻撃開始	悲鳴	活動停止
A	1	6	1 1	1 6	2 1
B	2	7	1 2	1 7	2 2

チーム：それぞれのチーム名（比較は常に 2 チームで行う）

後スイッチ：後のタッチセンサが押されたときに発信

前スイッチ：前のタッチセンサが押されたときに発信

攻撃開始：攻撃開始時に発信

悲鳴：敵の前部で後スイッチを押されたときに発信（この信号を受けて敵は攻撃を開始する）

活動停止：2 つのスイッチを押され活動停止時に発信

図 3.3 では RCX2 が攻撃側、RCX1 が被攻撃側である、この時二つのロボットが同じ A チームの場合は（説明）RCX1 は動作を続ける。しかし RCX 1 が A チームで RCX 2 が B チームの場合は、RCX1 は攻撃を受けたことになり悲鳴を発する。

(2) 集中制御

集中制御方式はロボットが取得した全ての情報を一度中央に集め、そこで大局的に情報を処理して判断を行い、その結果を全てのロボットに伝えて制御を行う方式である。流れとしては下記の様になる。

- ステップ 1 大局的判断に役立つ情報（以下、大局的情報）の取得
 （自分と相手の位置、状態、方向等）
- ステップ 2 取得した情報を中央へ送信
- ステップ 3 情報を元に環境をモデル化
- ステップ 4 モデルを元にし動作プランを立案
- ステップ 5 タスク実行
- ステップ 6 モータ制御

しかし MindStorms のハード的制約によりステップ 1 で必要な「大局的情報の取得」が不可能なことが、ロボット製作段階で判明した。MindStorms はタッチセンサ、光センサ、I R センサを装備しているが、それらのセンサ情報はモデル化や行動プランの立案が可能なほど高精度な物ではない。

またステップ 2 の情報の送信についても唯一の通信装備である I R センサが全方位通信を行えないため不可能だった。以上の理由からこの制御方式は実装出来なかった。

(3) 分散制御（通信有り）

分散制御は中央での一括したタスク処理を行わずタスクを複数のロボットに分散して制御する方式である。この方式の場合も集中制御程ではないが、ある程度大局的な判断に役立つ情報の取得が必要となる。

またロボット間で通信を行う時にや中央制御の場合と同じく全方位通信が必要になる。しかし MindStorms の能力では集中制御で述べた理由により、大局的な情報の取得と全方位通信が不可能だったので、この方式もロボットへの実装は出来なかった。

(4) 分散制御 (通信無し)

この制御方式は明示的な通信を必要とせず、ロボットは自分の周りのセンサ情報だけを頼りに行動する。制御アーキテクチャとして行動ベースアプローチを採用しているため、大局的情報からの環境のモデル化や行動プランの立案が不必要であり、通信を行わないので全方位通信も必要無い。よってこの方式だけは実装可能であった。

しかし 1 つの制御方式だけで比較による評価は出来ない。

3.3 位置認識システムの可能性

先ほど述べたように集中制御と分散制御 (通信有り) が実装不可能な要因は全方位通信と位置認識に有った。そこでこの要因の一つである位置認識システム実現のために光センサの色による位置認識手法を提案しその実装のために実験を行った。

3.3.1 実験方法

この手法は移動ロボットが動作する床に色を塗り、その床の色によって自分の位置を光センサにより認識するものである。たとえば図 3.5 のパレットの上でロボットを走らせたとき、光センサが 11 番の色と認識したらロボットは 11 番の上にいることになる。

この実験は以下の手順で行う。

(1) センサ別特性の測定

図 3.5 のような MindStorms 付属のシートと 10 個の光センサを用意し、各々の光センサの値を同じ白、黒、緑の 3 色で測定する。蛍光灯のついた研究室デスク上で図 3.4 のようにシートとの距離約 8 mm で測定。

(2) 色による光センサ値の測定

次に図 3.6 のようなカラーパレット用意し、各色を光センサの内蔵 LED で照らした場合の反射光強度を測定する。暗所でカラーパレットとの垂直距離約 8 mm で測定。なおこの時 No.10 の光センサを使用した。

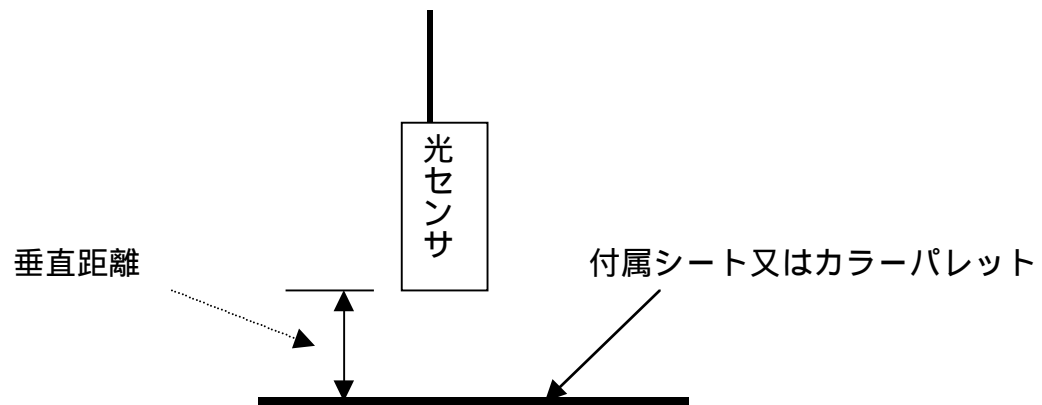


図 3.4 垂直距離

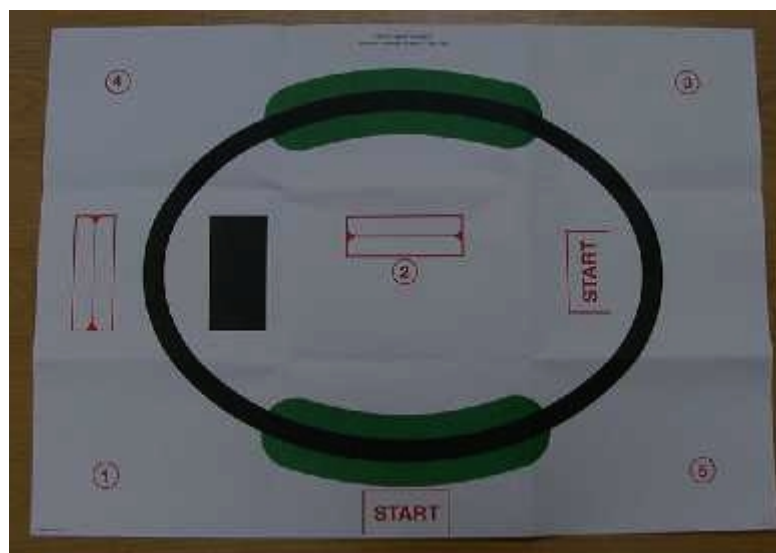


図 3.5 MindStorms 付属のシート

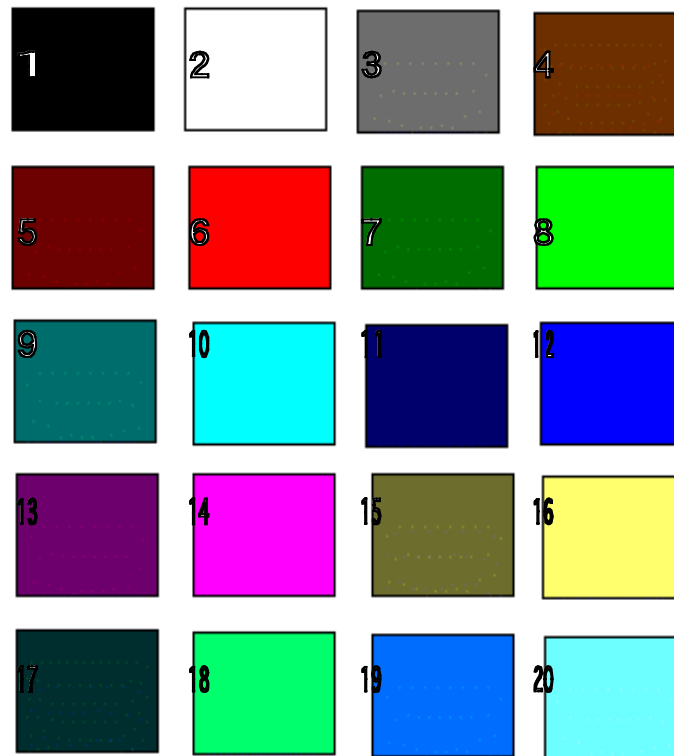


図 3.6 カラーパレット

3.3.2 結果

MindStorms による位置認識システム構築のために光センサの色による位置認識手法を提案し、そのための実験を行った。結果を以下に示す。

(1) センサ別特性の測定

表 3.2～3.5 で横は色の種類（シートの 3 色）を、縦は光センサ値を 10bit で測定したときの値（0～1023 までの値を返し光が強くなるほど値は小さくなる）を示す。最下行は平均値である。

表 3.2 センサ別特性 1

センサNo.1			センサNo.2			センサNo.3		
白	黒	緑	白	黒	緑	白	黒	緑
716	823	801	705	813	793	699	806	788
716	832	805	707	814	792	696	803	798
714	829	808	703	818	795	698	801	785
713	827	805	708	820	794	697	806	786
719	834	805	707	823	791	698	797	787
715	829	802	704	822	795	695	801	785
719	830	804	707	817	795	698	799	785
723	827	804	709	819	794	697	801	786
712	827	807	703	816	793	696	803	787
717	821	800	706	819	793	698	802	788
714	825	800	705	818	797	696	804	785
715	828	802	707	819	796	697	804	785
720	817	807	705	814	791	698	802	784
716	830	805	704	816	794	698	802	788
714	828	808	707	823	797	695	802	788
716	829	799	707	819	795	695	803	788
717	829	805	707	819	795	699	803	791
717	829	799	706	821	796	699	801	786
719	824	805	711	824	794	697	803	787
716.421	827.263	803.736	706.210	818.631	794.210	697.157	802.263	787.210
1	2	8	5	6	5	9	2	5

表 3.3 センサ別特性 2

センサNo.4			センサNo.5			センサNo.6		
白	黒	緑	白	黒	緑	白	黒	緑
699	813	791	691	801	784	692	805	787
702	815	792	693	801	787	691	807	787
702	816	795	695	806	785	697	805	790
702	812	794	692	804	788	697	804	790
701	815	796	695	804	787	694	807	787
702	818	791	693	806	789	697	804	788
702	810	795	695	804	789	699	806	790
700	810	796	693	805	787	693	806	788
703	815	796	695	805	789	695	808	790
702	814	794	695	808	789	697	805	790
703	814	797	698	808	791	697	808	787
702	814	795	695	808	785	696	810	789
701	813	798	698	808	790	696	807	790
703	818	793	698	808	793	698	806	787
704	818	798	696	809	791	700	806	792
705	816	798	698	809	792	697	809	790
704	815	796	697	808	785	698	811	790
703	819	798	694	810	791	697	807	790
707	819	796	695	808	791	695	810	786
702.473	814.947	795.210	695.052	806.315	788.578	696.105	806.894	788.842
7	4	5	6	8	9	3	7	1

表 3.4 センサ別特性 3

センサNo.7			センサNo.8			センサNo.9		
白	黒	緑	白	黒	緑	白	黒	緑
726	832	815	678	785	766	713	820	801
726	836	818	679	788	767	715	819	803
732	837	819	681	788	767	717	819	805
729	836	818	679	789	767	716	823	803
732	838	817	678	786	766	717	820	804
729	838	815	681	789	769	714	822	803
732	839	818	678	788	768	715	823	805
730	837	819	680	788	772	718	824	805
735	842	821	680	790	771	719	823	808
735	840	821	679	787	769	716	822	805
731	838	821	682	789	771	718	825	806
734	839	824	682	788	771	717	825	809
734	840	821	683	790	772	718	822	809
732	840	821	683	789	772	717	826	807
734	840	823	683	788	771	718	823	807
732	838	823	682	789	771	716	824	804
732	842	824	683	789	771	717	825	807
735	841	823	682	790	770	717	826	808
732	842	823	680	790	774	716	826	806
731.684	838.684	820.210	680.684	788.421	769.736	716.526	823	805.526
2	2	5	2	1	8	3		3

表 3.5 センサ別特性 4

センサNo.10		
白	黒	緑
683	799	770
686	798	776
684	800	774
687	801	774
688	802	775
687	799	776
687	801	776
687	799	777
685	801	778
688	802	779
689	804	780
690	806	779
686	804	782
690	803	777
687	803	780
690	803	778
687	803	779
690	805	779
689	804	780
687.368	801.947	777.315
4	4	8

(2) 色による光センサ値測定

表 3.6 ~ 3.8 に No.10 の光センサでカラーパレットを使いセンサ値を測定した。

表 3.6 色による光センサ値 1

黒 (1)	白 (2)	灰 (3)	茶(4)	茶(5)	橙(6)	黄緑(7)	緑 (8)
782	683	722	718	728	687	717	738
787	685	726	723	726	690	717	736
785	685	722	721	730	693	718	739
785	687	724	720	728	691	719	742
788	685	727	718	728	691	719	741
786	685	724	721	730	688	718	739
788	684	726	719	728	692	718	739
786	684	722	721	731	690	716	742
785	690	726	722	729	690	717	738
787	691	724	723	728	689	716	744
785.777	685.333	724.333	720.333	728.666	690.222	717.666	739.333
8	3	3	3	7	2	7	3

表 3.7 色による光センサ値 2

青緑 (9)	青緑 (10)	群青 (11)	紫 (12)	紫(13)	桃 (14)	黄緑(15)	黄(16)
745	739	759	751	716	689	724	688
744	737	759	756	717	689	724	691
746	740	758	755	717	687	723	688
746	741	757	753	715	690	722	689
748	740	757	751	714	687	720	689
745	739	754	752	714	687	722	689
746	738	756	757	715	689	723	688
746	739	755	751	714	693	726	688
746	737	755	751	719	689	722	690
745	740	756	754	714	689	724	688
745.777	738.888	756.666	753	715.666	688.888	722.888	688.888
8	9	7		7	9	9	9

表 3.8 色による光センサ値 3

深緑(17)	緑(18)	青(19)	水色 (20)	青紫(21)	赤(22)	橙(23)	黄(24)
751	747	757	721	730	689	598	685
754	742	757	724	730	689	693	685
756	745	755	724	728	689	688	685
756	741	755	725	727	691	690	689

755	745	755	723	728	689	685	687
754	744	757	724	725	693	687	689
755	745	757	727	729	689	685	689
751	741	756	722	729	688	688	689
752	744	755	725	727	692	688	688
755	743	753	723	728	691	688	687
753.777	743.777	756	723.888	728.111	689.888	678	687.333
8	8		9	1	9		3

3.4 考察

3.4.1 位置認識システム

(1) センサ別特性の測定

表 3.2 ~ 3.5 から分かるように各光センサで平均値はまちまちで一定ではない。しかし色間の値差は白と黒で約 110、黒と緑で約 20、白と緑で約 90 とどの光センサでもほぼ等しくなっている。これから色の変化によるセンサ値の変化割合がどの光センサでも一定である事をが分かった。また同じ色を測定した時、センサの違いによる測定値の差は最大値で約 50 であり、同じセンサで同じ色を測定したときの測定値の変動幅は約 5 である事も分かった。ここで得られた情報はロボットに光センサを実装する時、光センサ値の微調整に使用できる。

(2) 色による光センサ値の測定

表 3.6 ~ 3.8 から分かるように白と赤のように光センサによる値が非常に近い場合、光センサでは色の判別が出来ない。そこで正確な判別のために色同士の値が 15 ~ 20 以上離れているものを採用することにした。ここでは黒 (1) 白 (2) 緑 (8) 郡青 (11) 紫 (13) を採用した。

しかし採用できた色は 5 パターンだけでありとても位置認識システムに使用できる数ではない。

このことから MindStorms への位置認識システムの実装を断念した。

3.4.2 失敗原因

今回、集中制御、分散制御 (通信有り) 分散制御 (通信無し) の 3 つの制御方式を競技により比較評価を行おうとした。しかし使用した実験材料の制約により集中制御と分散制御の制御方式についてはロボットへの実装が出来なかった。

理由は MindStorms の入力ポートと出力ポートの数が少ない事、センサの種類と特性に問題がある事であった。

集中制御や分散制御 (通信有り) でロボットは中央もしくは他のロボットと通信する必要がある。移動するロボットの通信ではロボットの向きが時間的に動的に変

化するので通信手段としては全方位に対して送受信出来るものが望ましい。しかし唯一の通信手段である I R 通信はある特定の方向にしか通信できず、常に通信の必要がある集中制御や、ある程度頻繁な通信が必要な分散制御（通信有り）には適さなかった。

また集中制御と分散制御（通信有り）でロボットは大局的な判断に役立つ情報（例えば自分と相手の位置情報など）を取得する必要がある。今回製作したロボットには 2 つのタッチセンサと一つの光センサ、そして I R センサが付いている。しかしこれらの 4 つのセンサだけでは位置情報のような大局的判断に役立つ情報を取得するのは不可能だった。

結局、全方位通信と位置認識がネックとなって集中制御と分散制御（通信有り）をロボットに実装出来なかった。

第 4 章

制御プログラム構造の違いによる実ロボット動作特性の評価

4.1 行動ベースアプローチ

本研究では実験機材として扱うロボットのアーキテクチャに行動規範型アプローチである行動ベースアプローチ[1]を採用する。

行動ベースアプローチは遂行する行動ごとにタスクを分割し、その複数のタスクを並列に実行させるアーキテクチャである。複数のタスクを単純に並列に実行するだけではタスク間相互の矛盾や干渉が発生してシステムとして上手く動作しない。行動ベースアプローチではそれを解消するため図 4.1 のように各タスクに優先度を設けて、その優先度に従って行動を決定する方法を採用している。この簡潔な行動選択方式が素早さと、システム構築の容易さ、システムの堅牢さを生み出している。また行動ベースアプローチで構築したロボットは、環境のモデルを内部に持つ必要が無く、プランニングも行わない。

しかし、固定した優先度により行動を決定するため、モジュールの構成の仕方によっては、例えば袋小路に入り込んで抜け出せなくなるなどデッドロックに陥ることもある。これはある状態に対して一意に行動を決定するため起こる現象であり行動ベースアプローチの欠点でもある。

図 4.1 に行動ベースアプローチの概念図を示す。

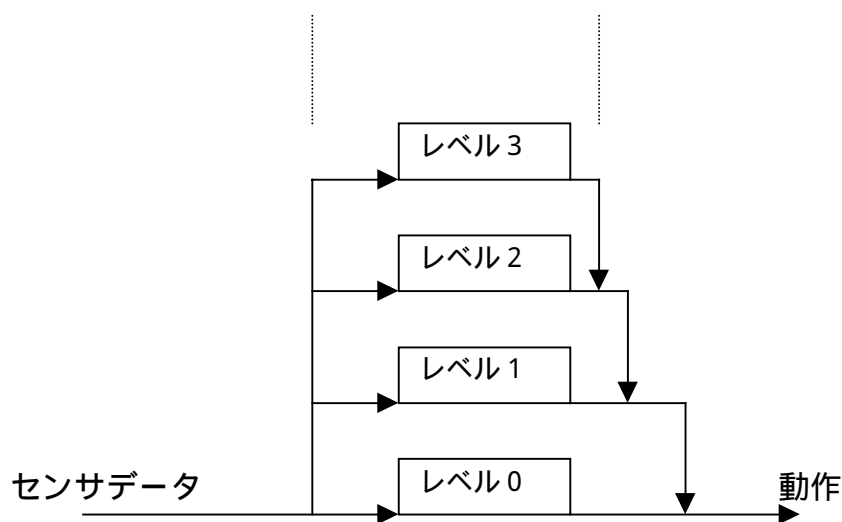


図 4.1 行動ベースアプローチ

4.2 実験準備

4.2.1 ロボットに与える目標

目標タスクとして複数ロボットによる的 2 度押し問題を与え前述した 1 タスク型、2 タスク型、複数タスク型、OnlyTouch 型の 4 つの制御プログラムについて動作特性の評価を行う。

この問題は複数の移動ロボットと移動ロボットによって攻撃される的ロボット（以下、的）により行われるものである。同じ制御プログラムを実装したロボット 6 体を 1 チームに編成し、そのチームで的の前後に付いた 2 つのスイッチを正しい順序で押すのが目標タスクである。的には押すべきスイッチの上にライトが点灯するようになっており、このライトを頼りに移動ロボットは的の位置を認識する。この問題は以下のルールに従って行い、終了条件を満たすまで続ける。

単体のルール

- 的は移動ロボットにより先に後スイッチ、次に一定時間（15 秒）以内に前スイッチに接触されれば活動を停止する。なお順序が逆の場合はこの限りで無い。
- 移動ロボットはスイッチ（タッチセンサ）が動作した場合 I R 信号を発信する。
- 的はスイッチが押され、かつ移動ロボットの I R 信号を受信した場合のみ正しく接触されたと判断する。

全体のルール（終了条件）

- 的が活動停止する
- 制限時間（10 分）になる。

なお、終了条件で制限時間を設けたのは的の活動を停止できなかった場合でも実験を終了させるためである。

4.2.2 ロボットの製作

移動ロボットと的ロボットとして下図に示すものを製作した。

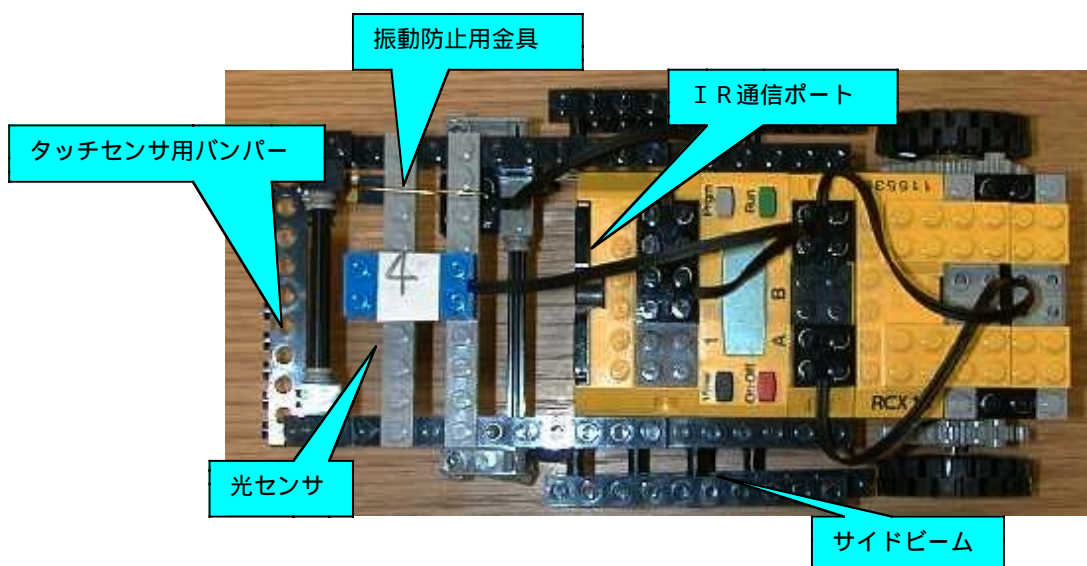


図 4.2 移動ロボットの構成（上から）

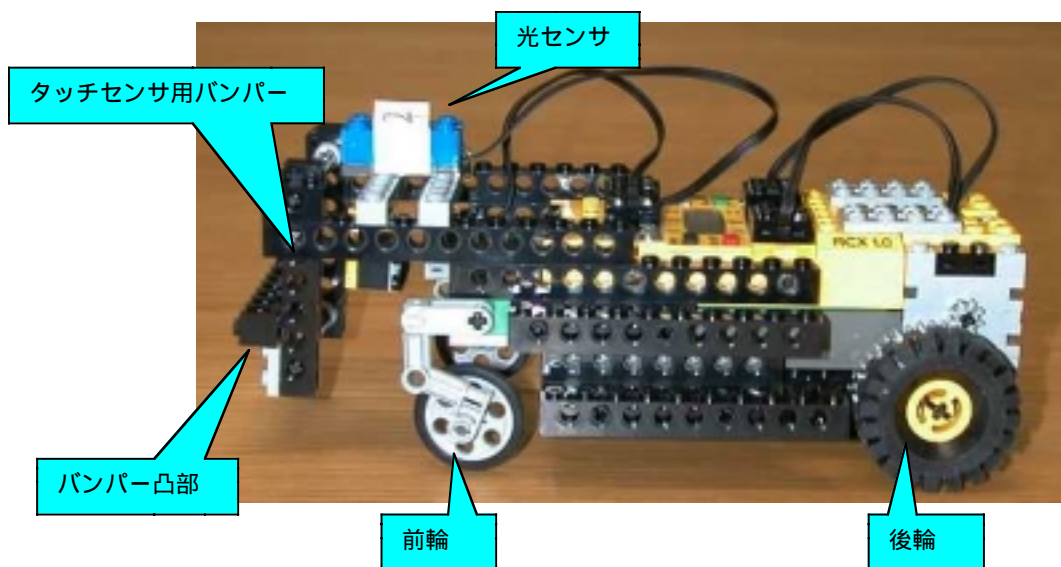


図 4.3 移動ロボットの構成（横から）

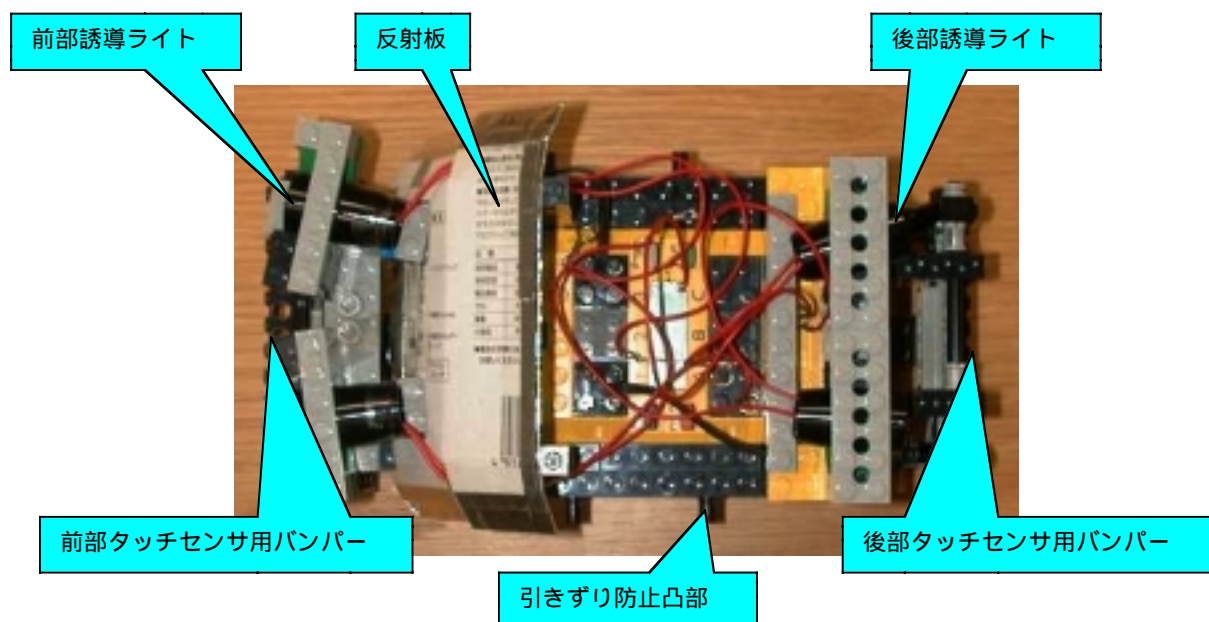


図 4.4 的ロボットの構成（上から）

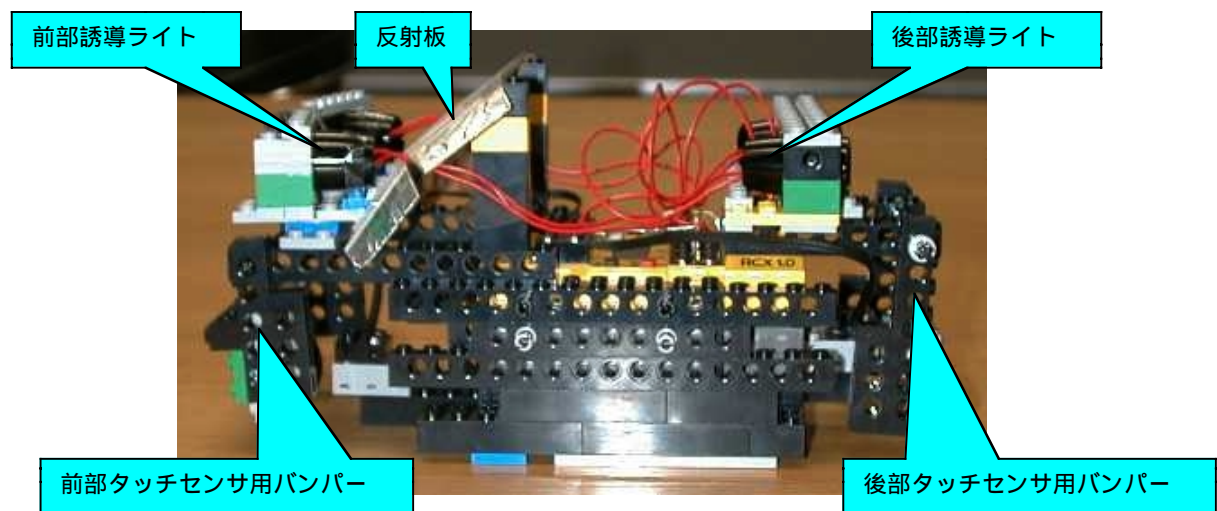


図 4.5 的ロボットの構成（横から）

図 4.4～4.5 を見ても分かるように的自体には移動能力は無い。しかし実験中、移動ロボットに押されたり引きずられたりして移動する事はある。

（１）機体製作時の注意点

今回、ロボットを作成するにあたって以下の点に注意した。

○機体の大きさ、形、強度

機体が旋回しやすいように全長をなるべく短くした。

動作時に車輪などで機体同士が引っ掛からないように側面に引っ掛かり防止ビームを設けた。

移動ロボットに引っ掛けられる事によって的が長時間引きずられないように、移動ロボットのタッチセンサが動作する高さのに的に引きずり防止凸部を設けた。

動作や衝突に絶えうる機体の物理的強度を確保するために、ブロックの接合強度に頼らない設計にした。

○タッチセンサ関連

移動ロボットが正確に的ロボットのタッチセンサを動作させやすいようにバンパ

ーに凸部を設けた。また不必要な時センサが刺激されないようにセンサの動作部に間隙を設けた。また故障率を下げるためにバンパーの押し戻しをタッチセンサのゴムに頼らないようにした。

移動ロボットのバンパーが動作時に振動により誤動作するため、バンパーに振動防止用金具を取り付けた。なおここでいうバンパーとは物理的力を間接的にタッチセンサへ伝える装置のことである。

○ I R センサ関連

ロボットの背後から I R 信号を受信するために機体上部に反射板を装備した。またある程度広い角度からも I R 信号を受け取れるようにするため反射板を曲面にした。的が移動ロボットの I R 信号を受信しやすいように、的の車高を移動ロボットより低くした。なお I R 信号の伝達は図 4.6 のようにして行われる。

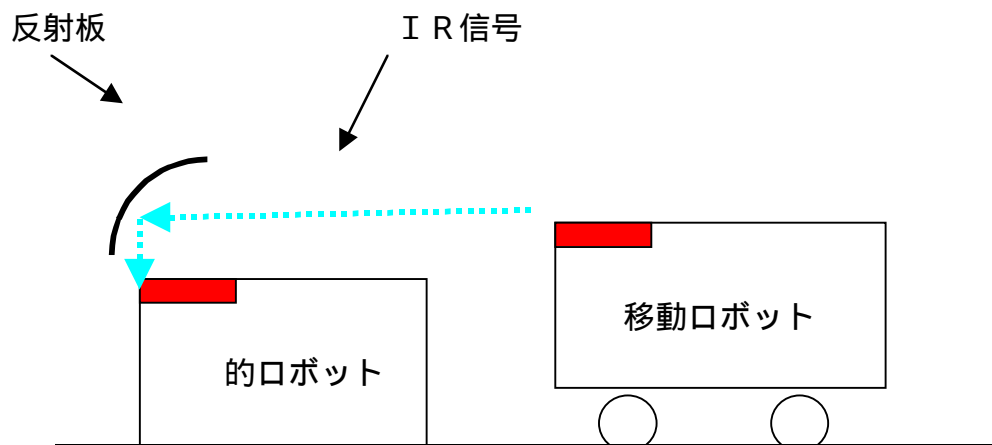


図 4.6 I R 信号の伝達

○ ライト関連

誘導用ライトを R C X から制御するためペンライトを分解して R C X の出力ポートに接続した。ペンライトを直接 R C X の出力ポートに接続へ接続すると規格の関係で電球が破損してしまう。そこで電球を 2 個と 1.5Ω の抵抗を直列に接続して破損を防いだ。またライトを前後 2 個ずつ取り付けた事によつて的のライトによる誘導

角度が広がった。出力ポートへの接続線の取り外しが容易になるようにC字型の取り付け金具を製作した。

4.2.3 実験場

実験は図 4.7 のような 1 辺 95cm の正六角形の実験場で行った。的を実験場の中心に置き移動ロボットは周囲の 6 箇所に配置した。移動ロボットと的の向きとして移動ロボットでは 4 パターン、的では 6 パターンの方向を用意し、そのパターンの中から Excel のランダム関数を使用してランダムに方向を決定した。詳細は付録にあり。

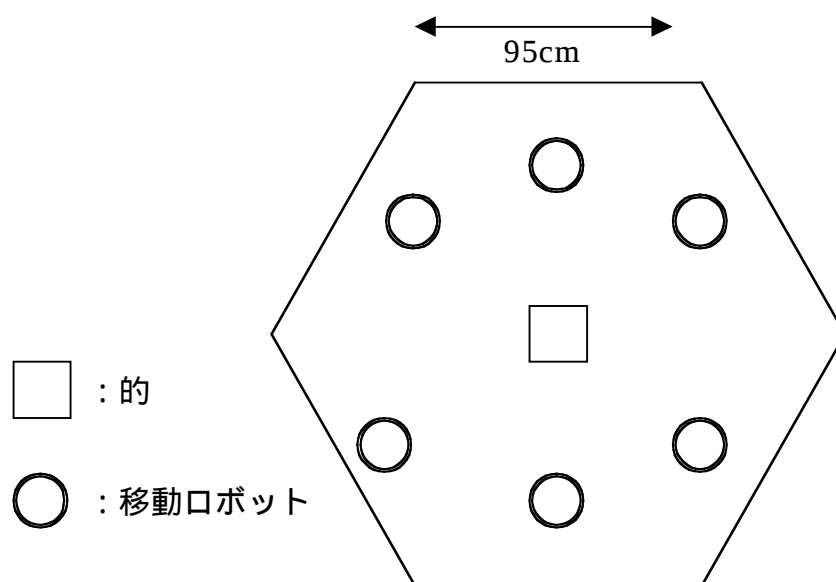


図 4.7 実験場

この問題では的の 2 箇所のスイッチを押す必要があり的が端ではどちらか一方が押せなくなる可能性が高い。また誘導用のライトが実験場の壁に接しておりかつライトが壁の方を向いていると移動ロボットが追跡できなくなってしまう。これを防ぐために的を実験場の中心に置いた。

実験時は全てのロボットに同時に競技開始命令を送る必要があるため、I R 信号を利用した自作リモコンによって問題を解決した。

4.2.4 実験条件

(1) 仮定

通信を行わず、かつ局所的な情報を基に移動ロボットが行動するためには、環境に対しての仮定をたて、それに基づいて作成した制御プログラムを有している必要がある。そこで「環境には的以外の光源は存在しない」という仮定を環境に対して設定した。そして実験環境もこの仮定に合うように設定した。この仮定は移動ロボットが近くに的があるか否かを判定するのに使われる。

(2) 状況

複数移動ロボットによる的2度押し問題における動的環境に対応するようにロボットの制御プログラムを記述する必要がある。そのためにロボットが環境の中で自分の置かれた状況を認識する必要がある。その認識方法として環境に対するパラメータを設定し、そのパラメータとして状況を認識する方法が考えられる。ここでは「移動ロボット近辺の的の有無」を環境のパラメータ N として用いる。先ほどの仮定から実験環境には的以外に光源を有する物体は無い。そのため移動ロボットは光センサに閾値以上の光が入ってきた時は近くに的があると判断できる。

このパラメータは2値を取るので、実験時の動的環境を論理記号 N と $_N$ で記述できる。 N は近くに的がある事を意味し、 $_N$ は近くに的が無い事を意味している。

このパラメータ N の真偽判定は以下のようにして行う。

ロボットは内部時計により光を認識した瞬間から現在までの時間経過を測定している。この経過時間がある一定時間内の時に追跡有効時間内であるという。 N の真偽はこの**追跡有効時間**によって決定され、時間内であれば N は真、時間外であれば N は偽と判定する。

以上の理由から環境には2つの状況が存在するとする。

〔	状況 1 : N	移動ロボットの近くに的が存在する。
	状況 2 : $_N$	移動ロボットの近くに的が存在しない。

(3) 行動選択

本研究で使用したロボットは全て図 4.8 のような行動選択方式に基づいて動いている。この行動選択方式ではロボットは環境からのセンサ入力に基づき、状況認識装置が今置かれている移動ロボットの状況を判断し、適切な動作集合を決定する。

動的環境に適応するためにロボットは自分自身で環境を認識し、適切な動作集合（行動）を選択しなければならない。そのためロボットの状況認識装置はNの真偽を決定するために常にセンサから入力データを監視している。

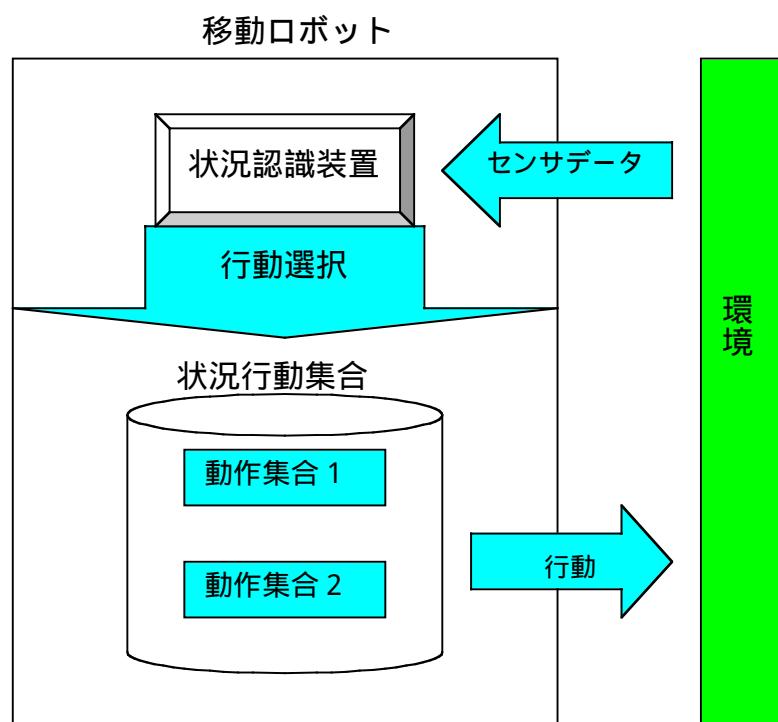


図 4.8 移動ロボットの行動選択方式

4.3 ロボットの状況別行動

4.3.1 状態と動作の記述

移動ロボットは環境として2つの状況を認識する。また本研究ではロボットのアーキテクチャとして行動ベースアプローチを採用している。行動ベースアプローチでは行動がタスクごとに分かれているので個々の状況に対して、行動を記述していく必要がある。

行動は「IF 状態 THEN 行動」という IF-THEN ルールで記述する。ここで状態はセンサによってダイレクトに決定される事象であり、行動は動作の集合である。動作とはモータに対するコマンドである。移動ロボットと的について以下に具体的な状態と動作を記述する。

(1) 移動ロボット

状態

have-light : 光センサに反応有り。つまり前方に的が有る。

no-light : 光センサに反応無し。つまり前方に的は無い。

have-touch : タッチセンサに反応有り。つまり前方に障害物有り。

no-touch : タッチセンサに反応無し。つまり前方に障害物無し。

receive-end : 活動停止信号受信

動作

go-straight : 直進。

stop : 停止。

go-back : バック。

turn-right/left : 右旋回、左旋回。

change-direction : 180 ° 旋回して方向転換。

wait : 少し待機。

transmit-attack-message : 攻撃信号を送信

end-all : 活動を停止する

(2) 的ロボット

状態

have-front-touch：前部タッチセンサに反応有り。

no-front-touch：前部タッチセンサに反応無し。

have-back-touch：後部タッチセンサに反応有り。

no-back-touch：後部タッチセンサに反応無し。

receive-attack-message：攻撃信号を受信

receive-end：活動停止信号を受信

in-attack-time：攻撃待ち受け時間内。後部スイッチに接触されてから 15 秒以内。

out-attack-time：攻撃待ち受け時間外。後部スイッチに接触されてから 15 秒以上経過している。

動作

on-front-light：前部ライトを点灯

off-front-light：前部ライトを消灯

on-back-light：後部ライトを点灯

off-back-light：後部ライトを消灯

transmit-end：活動停止信号を送信

end-all：活動を停止する

以上の状態と動作を IF-THEN ルールの条件部と結論部として、これ以降にそれぞれに適した動作集合を記述していく。対象となる状態や動作が複数の場合は論理積 { } で記述し、時間的に順序のあるものには { → } で記述した。なお同一動作集合の中で動作の競合が起きた場合は「より新しい動作命令を優先して実行する」という規則によって競合を解消した。

4.3.2 状況 1：移動ロボット近辺に的無し (_N)

移動ロボットが状況 1 の時に取りうる動作集合 (D 1 ~ D 4) について記述する。

- D 1 IF no-light no-touch
THEN 探索 (go-straight、stop、go-back、turn-right/left、wait の組み
合わせて構成される一連の行動パターン)
(光センサとタッチセンサ両方に反応が無いならば、探索を行う。)
- D 2 IF no-light have-touch
THEN go-back → wait → change-direction
(光センサに反応が無く、タッチセンサに反応があるならば、少しバックし
た後に 180 ° 方向転換する。)
- D 3 IF have-light no-touch
THEN go-straight
(光センサに反応があり、タッチセンサに反応がなければ、直進する。)
- D 4 IF have-light have-touch
THEN go-back → wait
(光センサに反応があり、タッチセンサに反応があるならば、少しバック
する。)

4.3.3 状況 2：移動ロボット近辺に的有り (N)

移動ロボットが状況 2 の時に取りうる動作集合 (D 5 ~ D 8) について記述する。

- D 5 IF no-light no-touch
THEN turn-left
(光センサとタッチセンサ両方に反応が無いならば、左旋回する。)
- D 6 IF no-light have-touch
THEN go-back → wait
(光センサに反応が無く、タッチセンサに反応があるならば、少しバックす
る。)
- D 7 IF have-light no-touch
THEN go-straight
(光センサに反応があり、タッチセンサに反応がなければ、直進する。)

D 8 IF have-light have-touch

THEN go-back → wait

(光センサに反応があり、タッチセンサに反応があるならば、少しバックする。)

4.3.4 的ロボットの動作

的ロボットでは条件として攻撃信号が受信 (receive-attack-message) されない場合、結論は全て後部ライトを点灯させ前部ライトを消灯 (off-front-light on-back-light) となる。そのため以下には攻撃信号が受信された場合のみを示す。

D 9 IF out-attack-time no-back-touch

THEN off-front-light on-back-light

(攻撃有効時間外で後部タッチセンサに反応が無ければ、後部ライトを点灯させ前部ライトを消灯する。)

D 10 IF out-attack-time have-back-touch

THEN on-front-light off-back-light

(攻撃有効時間外で後部タッチセンサが押されている場合、前部ライトを点灯させ後部ライトを消灯する。)

D 11 IF in-attack-time no-front-touch

THEN on-front-light off-back-light

(攻撃有効時間内で全部タッチセンサが押されてなければ、前部ライトを点灯させ後部ライトを消灯する。)

D 12 IF in-attack-time have-front-touch

THEN transmit-end → end-ll

(活動停止信号を送信した後、活動を停止する)

なお、ここでは結論を導くための必要最低限の条件のみを記述し、結論に影響を与えない状態については省略した。例えばD 9に no-front-touch 等の前部タッチセンサの状態を追加したとする。だが「攻撃有効時間外で後部タッチセンサに反応が無い」場合は前部タッチセンサがどんな状態であろうと「後部ライトを点灯させ前部ライトを消灯する」という結論は変わらない。以上の理由から分かり易くするために、結論に影響を与えない状態については省略した。

4.4 比較対象とする4つの構造

本研究では行動ベースアプローチを用いてロボットの制御プログラムを構築する。行動ベースアプローチを採用したロボットは複数の行動タスクを並列に実行しながら動作する。動作させるロボットは図 4.9 に示すように探索、回避、追跡の3つの基本行動で構成されており、その構成の仕方によってそれぞれ異なった動作特性がある。特に標的を発見する動作の基本となる探索と追跡の関係は重要である。本研究ではこの部分の構造に注目し、構造の異なる4種類の制御プログラムを用意して競技により評価する。またロボットへ作成した制御プログラムをダウンロードするために図 4.10 に示すようなインターフェイスを作成した。

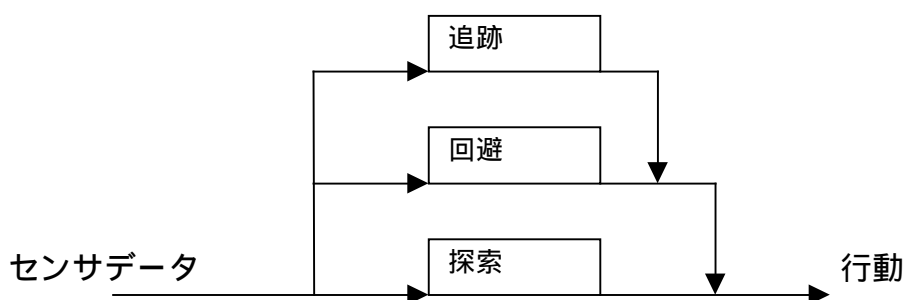


図 4.9 ロボットの基本概念



図 4.10 ダウンロード用インターフェイス

4.4.1 1 タスク型

この方式では探索と追跡の行動をプログラム構造上で一つのタスクの中に入れた。この場合探索と追跡が同じタスク内に入っているのでタスクの優先度によって行動を切り替えることは出来ない。そこで行動の切り替えはランダムな時間間隔ごとに組み込まれた IF 文による条件分岐で行った。またこの探索タスクは左旋回、右旋回、直進など具体的な行動に基づいて記述されており、その動作順序（左旋回→直進→右旋回→停止等）は解析により最適だと考えられるパターンを人間が用意し、それに基づいて設計されている。

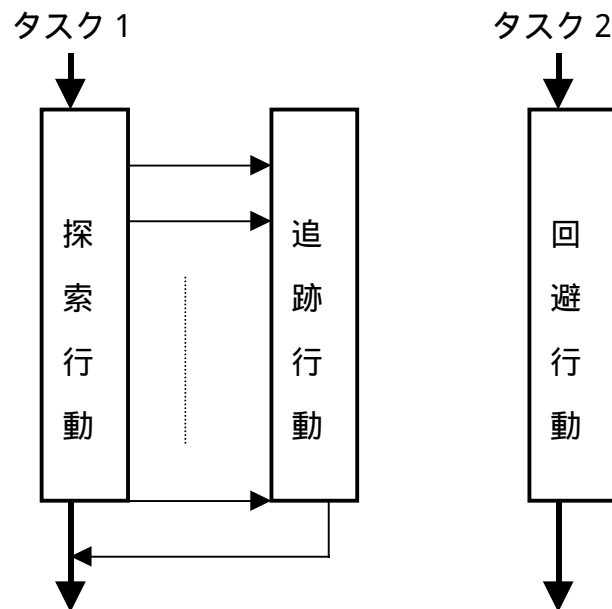


図 4.11 1 タスク型のプログラム構造

4.4.2 2 タスク型

この方式では探索と追跡の行動を 2 つのタスクに分けた。この場合は図 4.12 のように探索、追跡、回避がそれぞれ独立のタスクで構成されており、行動タスクはタスク間の優先度によって決定される。この 2 タスク型の探索タスクも左旋回、右旋回、直進など具体的な行動に基づいて記述されており、その動作順序（左旋回→直

進→右旋回→停止等）は解析により最適だと考えられるパターンを人間が用意し、それに基づいて設計されている。

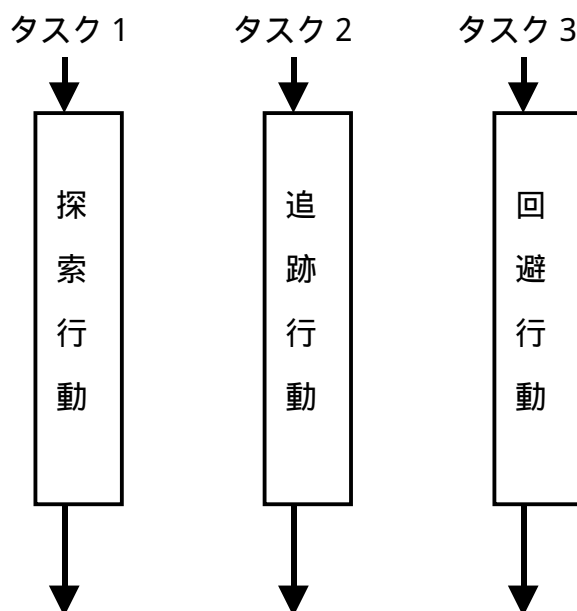


図 4.12 2 タスク型

4.4.3 複数タスク型

このタスクでは探索行動を複数のタスクで構成し、追跡行動を単独のタスクで構成した。1 タスク型と 2 タスク型の探索行動は図 4.13 のように右旋回、左旋回、直進など具体的動作を直列に並べた形で構造になっている。

一方この複数タスク型では図 4.14 のように探索行動を 2 つのモータへのコマンド On/Off、正転、逆転など一段抽象度の低い概念で記述し、それらを直列に並べた構造になっている。探索行動は 6 つのタスクで構成され、追跡と回避行動のタスクと合わせて全部で 8 つのタスクで構成されている。この探索行動のタスクの組み方は「一度命令により動作を始めたモータは、次に命令があるまで同じ動作を続ける」という RCX の制御特性を利用したもので、個々のモータへの命令をランダムなタイミングごとに繰り返すことによってランダムな動きをさせることが出来る。この 6 つの探索タスク間では優先度は設定されておらず、タスクの優先度は 6 つとも同じであ

る。なお複数タスク型での実際の探索行動は6つのタスクで構成されるが、図 4.14では説明を簡単にするため4つのタスクで構成した。

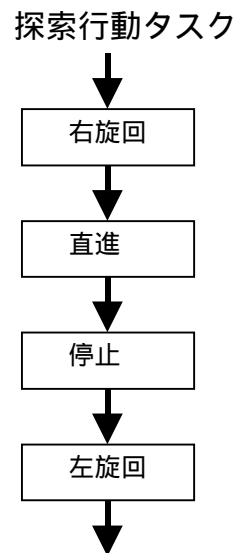


図 4.13 1 タスク型と 2 タスク型の探索タスク構造

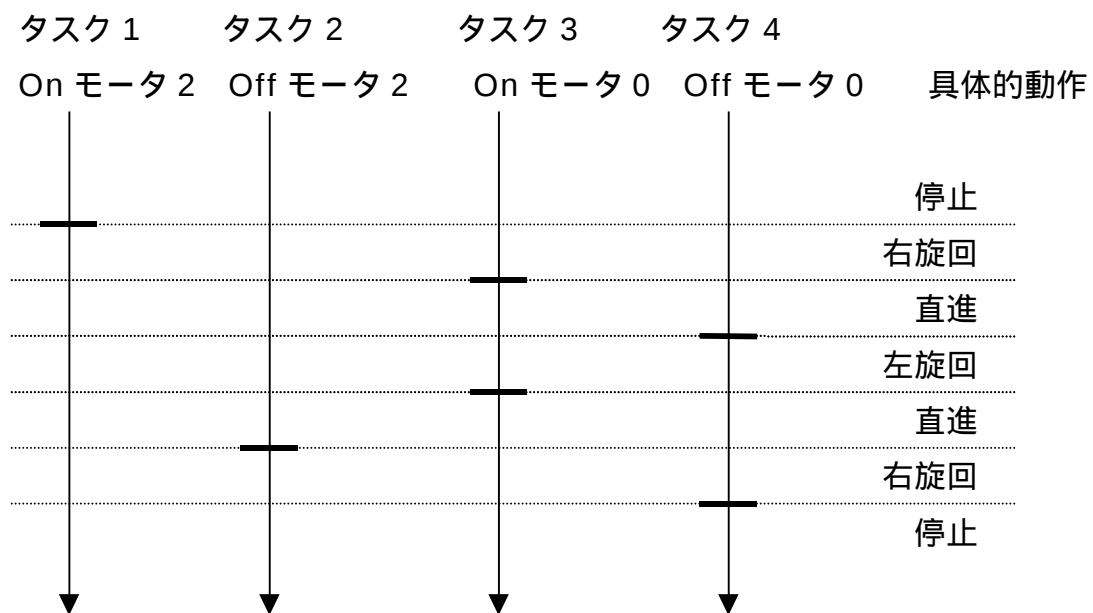


図 4.14 複数タスク型の探索タスク構造

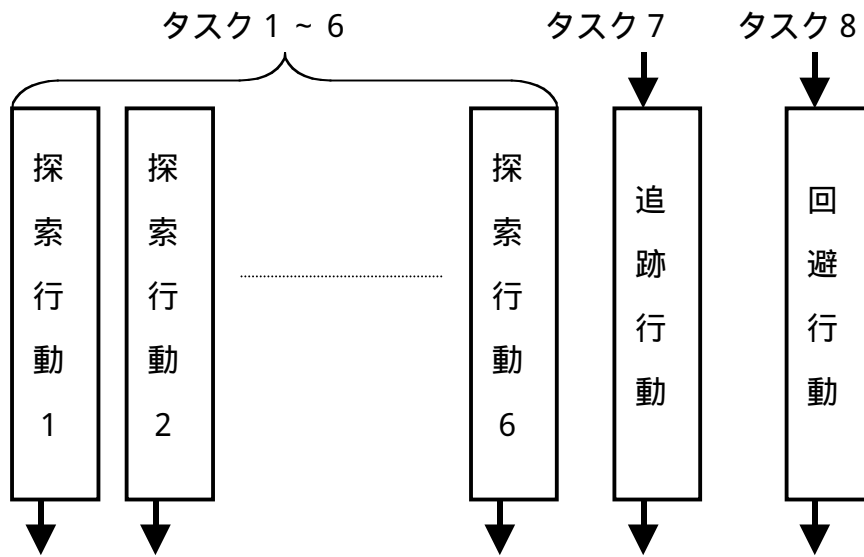


図 4.15 複数タスク型

4.4.4 OnlyTouch 型

上述した 3 つの型の制御プログラムはどれも光センサによる追跡行動のタスクを有している。しかし光センサによる追跡タスク無しで探索と回避だけを行った場合でも目標を達成できるかもしれない。そこで光センサによる追跡タスクの有効性を検証するために、追跡タスクをはずしたロボットによる実験を行った。

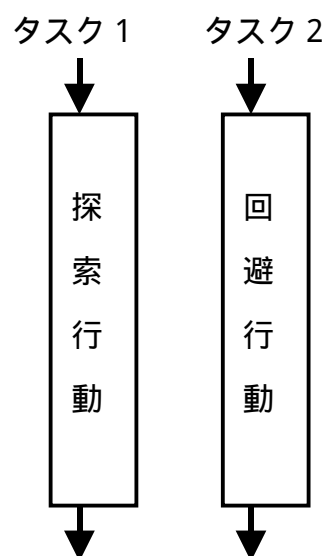


図 4.16 OnlyTouch 型

4.5 結果と考察

4.4.1 動作特性

ここでは前述した4種類の構造の制御プログラムをロボットに実装し、実験を行った結果を述べる。実験で測定は4タイプそれぞれで20回ずつ行った。

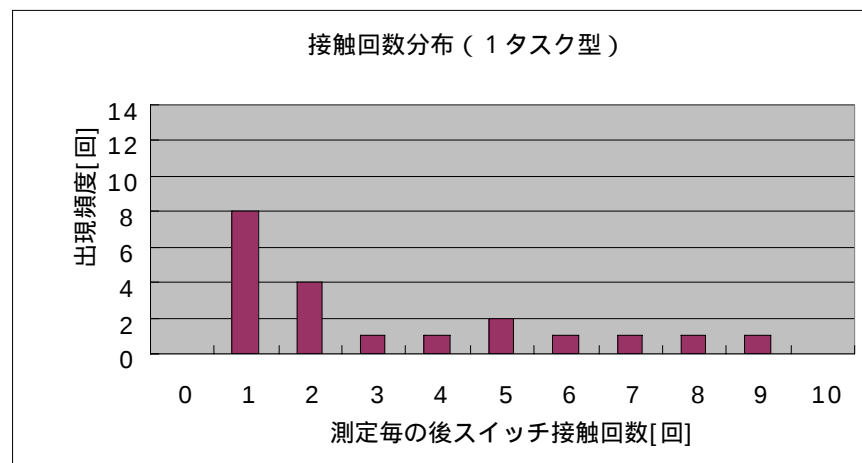


図 4.17 接触回数分布 (1タスク型)

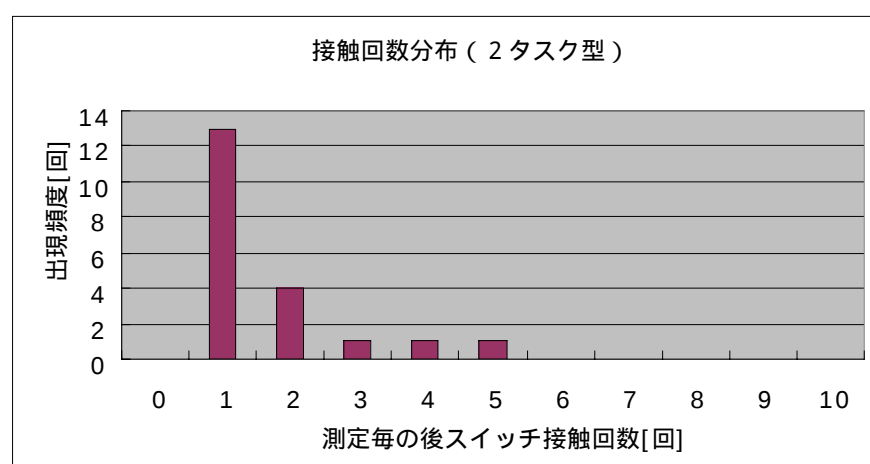


図 4.18 接触回数分布 (2タスク型)

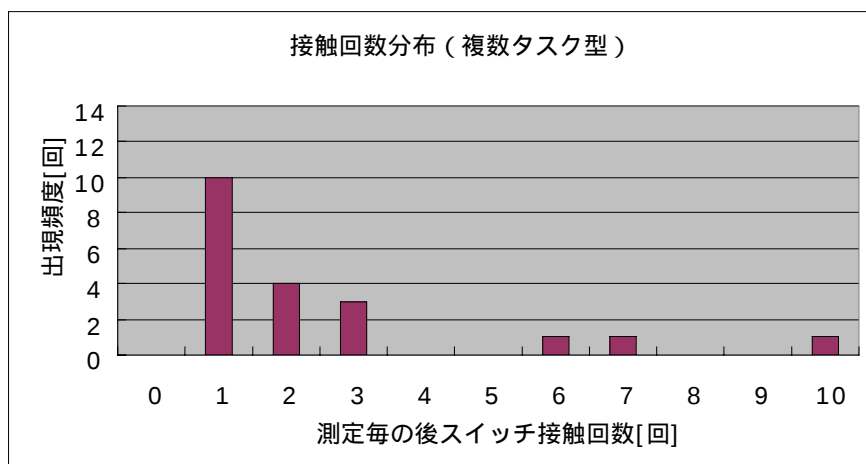


図 4.19 接触回数分布（複数タスク型）

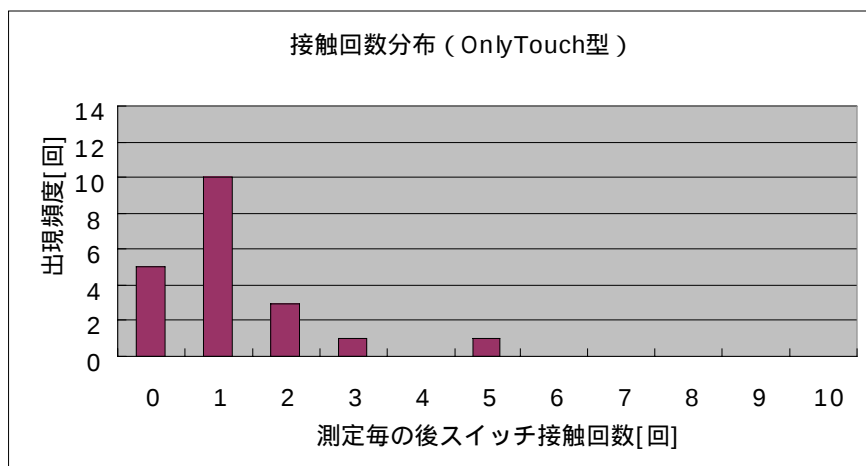


図 4.20 接触回数分布（OnlyTouch 型）

図 4.17～4.20 は 4 つの型でそれぞれ 20 回ずつ測定を行った結果である。このグラフで横軸は 1 回の測定中でも接触回数を表し、縦軸はその出現頻度を表している。そのため出現頻度の合計は常に測定回数である 20 となる。接触回数とは 1 回の測定で終了条件を満たすまでの間、もしくは制限時間になるまでの間に後スイッチに接触した回数のことである。

この図 4.17～4.20 を見ると、どの型でも測定毎の接触回数は 1 回がもっとも多い。これには以下の 2 つの理由が考えられる

- 1) 単にロボットの性能が低くてスイッチに接触出来ない。
- 2) タスク達成率（後スイッチに接触した後、一定時間内に前スイッチに接触する確率）が高い、つまり性能が良いので 1 回で十分。

上記の 2 つの理由は互いに矛盾したものであり、このことから接触回数だけでは全体の正しい比較評価が出来ない。

ただ光センサによる追跡行動を採用していない OnlyTouch 型だけは 0 回（測定実験時に一度もセンサに接触できなかった）という分布が存在しており、これから光センサによる追跡行動が有効なことだけは分かった。

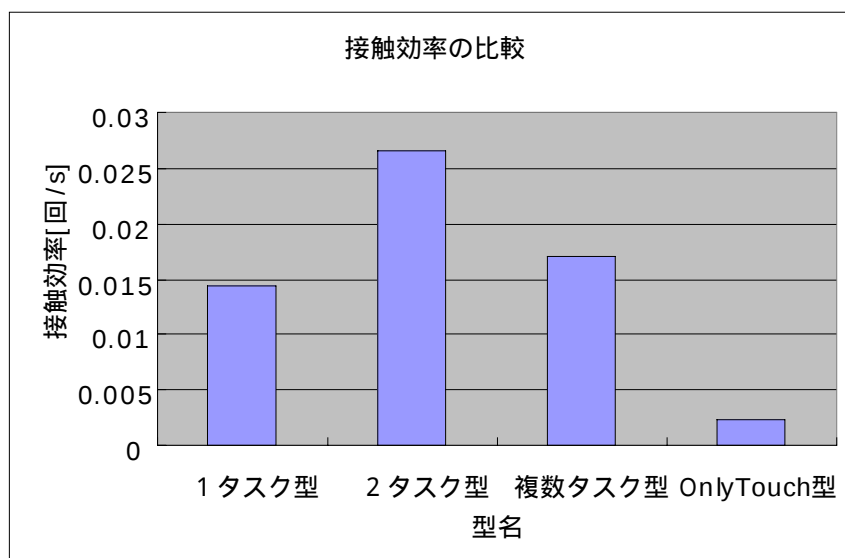


図 4.21 接触効率の比較

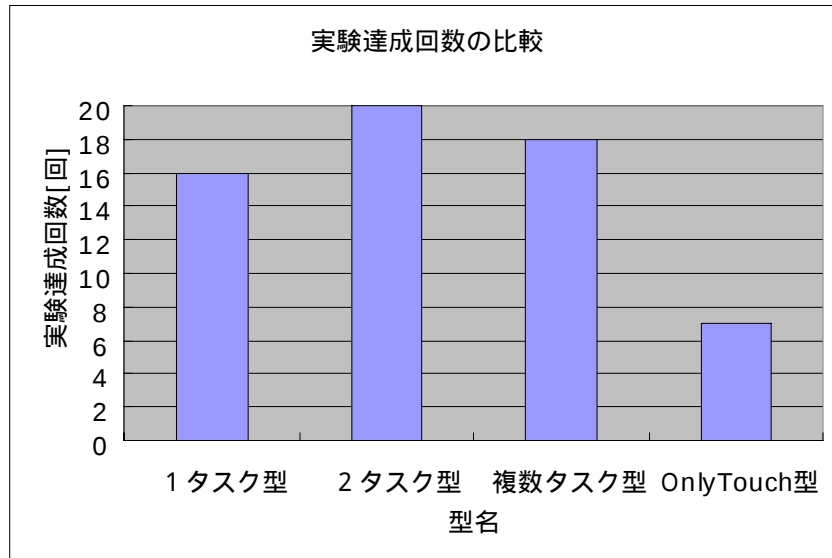


図 4.22 実験達成回数の比較

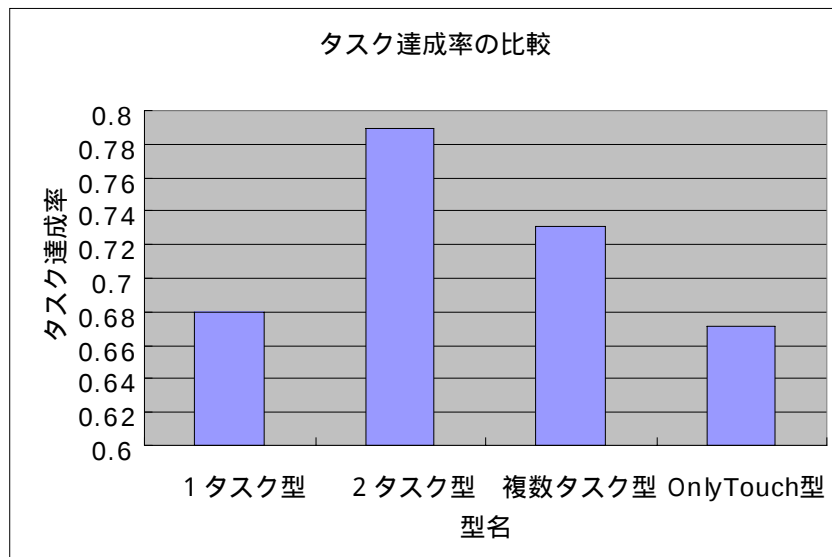


図 4.23 タスク達成率の比較

図 4.21 ~ 4.23 に接触効率、実験達成回数、タスク達成率をそれぞれの型で比較したグラムを示す。用語の意味を以下に示す。

接触効率：単位時間（1 秒）あたりに接触した回数。

実験達成確立：各 20 回の測定実験中にタスクを達成できた回数。

タスク達成率：後スイッチに接触した後、一定時間内に前スイッチに接触する確率。

この 3 つのグラフでは縦軸の値が大きいほどロボットの性能は良い。図 4.21 ~ 4.23 の 3 つのグラフいずれにおいても以下のような性能の順位付けが出来る。

No.1 2 タスク型：探索行動は人間が状況を分析して設計し、常にセンサによる環境の監視を行っている。

No.2 複数タスク型：探索行動はランダムに行い、常にセンサによる環境の監視を行っている。

No.3 1 タスク型：探索行動は人間が状況を分析して設計し、センサによる環境の監視は断続的に行っている。

No.4 OnlyTouch 型：探索行動は人間が状況を分析して設計し、常にタッチセンサのみによる環境の監視を行っているが、光センサによる追跡行動は行わない。

4.4.2 統計処理によるデータの検証

次にこのデータの妥当性を検証するために統計処理を行う。

データには 2 つのパラメータ、接触回数と動作時間が有る。しかし接触回数では一意にプログラム構造の性能は測れない。例えば接触回数が多いということは一見優れているように見えるが、逆に後スイッチに何回も接触しながらタスクを完全に達成出来ていない可能性も残るからだ。

動作時間は短ければ優秀で、長ければ悪いと簡潔に判断出来るので、ここでは統計処理の対象としてロボットの動作時間を採用した。なお方法として一元配置分散分析を用いた。結果を以下の表に示す。

表 4.1 一元分散分析表

要因	平方和	自由度	平均平方	F 比
郡間	2578053	3	859350.9	29.73361
郡内	2196527	76	28901.67	
全体	4774580	79		

この表 4.1 から 4 種類の制御プログラムには有意な差がある事が分かった。次にどの組み合わせに有意な差があるのか確認するため多重比較を行った。ここではその中の LSD 法を用いた。

LSD 法：有意水準（LSD 法では、必ず 5%）をクリアする最小の幅を計算する。

その幅が、条件の差よりも大きければ有意であるし、小さければ有意でないとする。

計算により $LSD = 106.98$ となった。

これを基にそれぞれの制御プログラムで有意さがあるか調べた。結果を以下に示す。

1 タスク型 - 2 タスク型 = 155.75	> LSD	有意差が有る
1 タスク型 - 複数タスク型 = 71.3	< LSD	有意差が無い
1 タスク型 - OnlyTouch 型 = 276.75	> LSD	有意差が有る
2 タスク型 - 複数タスク型 = 102.45	< LSD	有意差が無い
2 タスク型 - OnlyTouch 型 = 432.5	> LSD	有意差が有る
複数タスク型 - OnlyTouch 型 = 348.05	> LSD	有意差が有る

次にこの結果を踏まえて考察を行う。

考察 1 OnlyTouch 型は他のどのタイプよりも成績が悪い。これは追跡行動の有無に起因する。

考察 2 2 タスク型と複数タスク型を比較すると、どのグラフでも 2 タスク型の方が成績が良い。この成績の違いは探索行動の設計方法の違いに起因している。しかし統計処理の結果、2 タスク型と複数タスク型の間には有意な差が無い。そのためこの考察から導かれる結論には意味が無い。

考察 3 1 タスク型と 2 タスク型を比較すると、どのグラフでも 2 タスク型の方が成績が良い。この 2 つの型の成績差は、環境の監視方法の違いに起因している。

以上の理由から本研究のように動的環境で動作する移動ロボットには以下の 2 つの事が考察結果として言える。

考察結果 1 光センサによる追跡行動は有効である。

考察結果 2 タスク数の増加による制御部への負荷増大を考慮しても、絶えずセンサで環境を監視することは有効である。

4.6 今後の課題

4.6.1 問題点

今回、複数移動ロボットによる的 2 度押し問題を使用して制御プログラム構造の違いによる実ロボットの動作特性について評価した。

この研究でロボットはあらゆる状態に有的に対してタスクを達成できるわけではない。以下にタスクを達成できない例を示す。

- ・ ロボットが的を発見する動作は偶然的要素に頼っている部分が多い。よって長時間探索を行っても移動ロボットが的を発見できない場合がある。これは環境が広くなればなるほど深刻な問題となる。
- ・ 的が壁際にありライトが点灯している押すべきスイッチが壁側に有る場合はタスクを達成出来ない。

4.6.2 制約の緩和

今回、ロボット製作のため実験材料として LEGO 社の MindStorms を使用した。しかし MindStorms の制御部である RCX には 3 つの入力ポートと出力ポートしかないため、多くのセンサやアクチュエータを使用することが出来ない。付属のセンサもタッチセンサと光センサだけであり、自律移動ロボットとしては非常にきつい制約を持っている。本研究ではその制約条件下で目的タスクを達成出来るロボットを作成し実験を行った。しかし、この制約ゆえに更に複雑な目標タスクを持った実験には MindStorms は不向きであると考える。複雑な目標タスクを設定する必要がある場合は、この制約を緩和しロボットに多くの機能を追加（センサ種やセンサの増設）する必要がある。特により大局的情報の取得が必要な場合には、センサ種とセンサ数を増やす必要がある。

4.6.3 学習

本研究では複数移動ロボットによる的 2 度押し問題で想定される動的環境に対応

出来る移動ロボットを製作した。この移動ロボットの設計は設計者が動作環境で起こりうる全ての状態を予測し、それに対して適応できる動作集合を記述するというトップダウン的な設計方法を取っている。よってあらかじめ設計者が予期していなかった状況にロボットが陥るとロボットは目標タスクを達成できない可能性も有る。つまり本研究で製作したロボットは時間的に環境が変化するという意味での動的環境には適応できるが「エージェントにとっての動的環境」[4]には適応出来ない。ここでエージェントにとっての動的環境とは設計者が予め記述した動作集合では適応出来ないような事が起こる環境、言い返ればロボットの持つ環境モデルでは予測不可能な現象が起こる環境のことを示す。

この問題に対する解決手段としてロボットに学習機能を持たせることが考えられる。本研究でも動作の選択、競合の解消、動作集合のバリエーション変更、優先度の付け直し等、様々な曲面での学習が考えられる。学習方法としても遺伝的アルゴリズム（GA）やニューラルネット、強化学習など様々な学習方法がある。しかし適応度と実時間性を兼ね備えた学習方法はまだ確立されていないように思われる。移動ロボットでの高効率で最適な学習方法の開発は今後の課題であろう。

4.6.4 平均値の差の原因

制御プログラムの組み合わせ比較によっては平均値に差が有りながら、統計的有意差が見られないものがあつた。例えば2タスク型と複数タスク型の組み合わせがそれにあたる。しかし平均値に差があるということは、もしかしたら何らかの有意差があるのかもしれない。今後、この差の原因について詳しい検討が望まれる。

第 5 章

おわりに

本研究で複数移動ロボットによる的 2 度押し問題のために行動ベースアプローチを用いた移動ロボットの制御プログラムを作成した。「移動ロボット近辺の的の有無」は制御プログラム内で環境の状況を表すパラメータとして使用されている。そしてロボットがそれぞれの状況に対応できるように動作集合を設計した。ロボットはセンサにより環境を監視することで現在置かれた状況を認識し、それに適した動作集合を選択していく。

次にここで作成した制御プログラムを 6 台の実ロボットに実装した。この動的環境に対応出来る自律移動ロボットの製作には、MindStroms という製品を使用した。また実装の際に 4 種類の構造を持つ制御プログラムを用意し、実験によりそれらの動作特性を比較評価した。実験結果を多重比較したところ、1 タスク型と複数タスク型、2 タスク型と複数タスク型以外の全ての組み合わせに有意な差がある事が分かった。これから次の考察結果が導かれた。

< 考察結果 > タスク数の増加による制御部への負荷増大を考慮しても、絶えずセンサで環境を監視することは有効である。

つまり自律型移動ロボットの制御プログラムとしては、全ての行動を独自のタスクに割り当て、並列に動作させる構造が適している。

最後にこの研究の将来に対する意義について考えてみる。現在、様々な移動ロボットの研究が行われており、近い将来移動ロボットが消費者向け生活用品として販売される日が来ると思われる。こうした販売用ロボットとしては性能とコストの面でトレードオフがあり、この 2 つのバランスをとることは重要である。

本研究では MindStroms という非常にきつい制約を持った機材を使用して動的環

境に適応する自律型移動ロボットを製作した。これにより低い能力しか持たないハードでも実環境に耐えるロボットを製作出来る事を示した。つまりコンシューマレベルの機材を使用したこの研究は低コスト移動ロボット製作への 1 つの回答を示したとも言える。

謝辞

本研究を行うにあたり、非常に熱心に御指導していただきました櫻井彰人教授、荒木修助手に心から厚くお礼申し上げます。またゼミなどで熱心な討論、アドバイスをくださいました櫻井研究室の同輩、後輩の方々に感謝いたします。

異なる研究室にも関わらず研究の初期段階で熱心に相談に乗ってくださった橋本敬助教授、林幸雄助教授に深く感謝いたします。

同じく異なる研究室にも関わらず討論、指摘などをしてくださった林研究室、橋本研究室の方々に感謝いたします。中でも多くの率直なアドバイスを下さった黄林春氏に心から感謝いたします。

参考文献

- [1] Rodney A Brooks : "A Robust Layered Control System For A Mobile Robot" IEEE Transaction on Robotics and Automation、vol.2、No.1、pp14-23,1986
- [2] 岡田浩之、伊藤修、萩原由香里、仁木和久、大森隆司 : "ネットワーク分散型移動ロボットシステム" 日本ロボット学会誌、Vol.17、No.6、pp896-904、1999
- [3] 服部文夫、坂間保雄、森原一郎 : "エージェント通信"、p19、オーム社、1998
- [4] 山田誠二 : "適応エージェント"、p42、共立出版、1997
- [5] 山田誠二、斎藤淳也 : "マルチロボットによる箱押しのための明示的通信を用いない適応的選択行為"、日本ロボット学会誌 Vol.17、No.6、pp818-827、1999
- [6] Jin Sato : "MindStorms 情報局"、<http://www.mi-ra-i.com/JinSato/MindStorms/>、2000年2月現在
- [7] Jin Sato、白川裕記、牧瀬哲郎、倉林大輔、衛藤仁郎、古川剛 : "MindStorms パーフェクトガイド"、株式会社 翔泳社、1999
- [8] 内部英治、浅田稔、細田耕 : "他のエージェントの行動理解 - サッカーロボットにおける強化学習のマルチエージェント環境への適用に向けて -"、第13回ロボット学会学術講演会予稿集、1995。
- [9] 内部英治、浅田稔、細田耕 : "マルチエージェント環境における部分空間同定法を用いたエージェントの判別と行動理解 - 分散協調視覚システムにおける対象の行動理解法 -"、情報処理学会研究報告(97-CVIM-103)(コンピュータビジョンとイメージメディア研究会)、pp.51-58、1997。
- [10] 内部英治、中村理輝、浅田稔 : "共進化による協調 - ロボカップにおける協調

- 行動の獲得-“，第12回人工知能学会全国大会予稿集，pp.96-97，1998．
- [11] 倉橋大輔、新井民夫、岩瀬寛司、大田順：“地図誤差に動的に対応する移動ロボットの掃引作業計画”、日本ロボット学会誌、Vol.17、No.5、pp677-684、1999.
 - [12] 及川一美、土谷武士：“行動規範型自律移動ロボットの世界像獲得およびナビゲーション手法”、日本ロボット学会誌、Vol.16、No.1、pp65-73、1998
 - [13] 藤田豊己、木村浩：“密な協調を行うマルチロボットのためのオブジェクト指向システムの構成法”、日本ロボット学会誌、Vol.16、No.7、pp927-933、1998
 - [14] 小菅一弘、大住智宏、千葉晋彦：“単一物体を操る複数移動ロボットの分散協調制御”、日本ロボット学会誌、Vol.16、No.1、pp87-95、1998
 - [15] 吉村裕司、太田順、井上康介、平野智一、倉橋大輔、新井民夫：“群ロボットによる多数物体の繰り返し搬送計画”、日本ロボット学会誌、Vol.16、No.4、pp499-507、1998
 - [16] 水野裕識、川元志英、興梠正克、村岡洋一：“フィードバック制御、フィードフォワード制御の適用によるサッカー動作の生成手法”、日本ロボット学会誌、No.16、No.6、pp807-815、1998
 - [17] 中村恭之、浅田稔：“画像運動情報に基づく単眼視覚移動ロボットの行動獲得”、第9回人工知能学会全国大会論文集，1995
 - [18] 永谷圭司、油田信一：“タスクオリエンテッドアプローチによる自律移動マニピュレータの研究”、日本ロボット学会誌、Vol.17、No.6、pp865-875、1999
 - [19] 港隆史、浅田稔：“環境の変動に適応する移動ロボットの行動獲得”、第16回日本ロボット学会学術講演会予稿集，vol.1、pp399-400，1998
 - [20] The LEGO Group：“Controlling LEGO Programmable Bricks Technical Reference”、LEGO Corporation、1998
 - [21] The LEGO Group：“MindStorms ROBOTICS INVENTION SYSTEM USERS GUIDE, CONSTRUCTOPEDIA, 日本語マニュアル”、LEGO Corporation、1998