LOGOによる簡易ロボット制御システムの作成

坂倉, 守 / 平尾, 亜佐子 / SAKAKURA, Mamoru / HIRAO, Asako

(出版者 / Publisher)
法政大学計算センター
(雑誌名 / Journal or Publication Title)
Bulletin of Computer Center, Hosei University / 法政大学計算センター研究報告
(巻 / Volume)
1
(開始ページ / Start Page)
25
(終了ページ / End Page)
38
(発行年 / Year)
1987-08-01
(URL)
https://doi.org/10.15002/00024615

LOGOによる簡易ロボット制御システムの作成

平尾亜佐子*・坂倉　守

法政大学工学部電気工学科計測制御専攻†

計算機言語LOGOの一般的によく知られた特徴の一つであるタートル・グラフィック機能を，外部に接続した簡易ロボットで実現するための基本的なシステムを作ってみた。また，簡易ロボットに光センサとタッチ・センサを付加することにより，ロボットの移動に支障が生じた場合の処理も一応考慮できるようになっていて，しかも，光の反射の著しく異なる面があれば，反射率のいい面でのみロボットを移動させるとか，CRT上のタートルの動きを，そのままロボットに実行させることにより，計算機入門者用の初級教育にも利用できると考えられる。

1. 序論

近年，教育分野でのパソコンの導入が活発化してきた。そのさきがけとして“LOGO”という言語が開発され，よく知られるようになってきた。LOGOの特徴の1つに，タートル・グラフィック機能がある。そこで，このタートル・グラフィック機能を応用し，タートルと同じ動作をする簡易ロボットシステムを作成した。ロボットシステムは，二次元平面内でタートルと同じ動作をするだけではなく，数種類の制御命令の判別と，人が描いた線上の走行も可能である。本システムを用いることによって，学習意欲がわき，ロボットに親近感が持たれたら幸いである。

2. ハードウェア・システム

2.1 システムの構成

図1にハードウエアの構成を示した。

NECのPC-9801VM2とロボットのインターフェイス用ICとして8255を用いた。ロボットは嘉穂無線のMV-9511WAOの車輪部だけを使用し，走行中，物体に接触した時の為のタッチセンサと人が描いた線上を走行する為の光センサを取り付けた。本章での入出力はPC-9801を中心とする。つまり，パソコンから外部へ出て行く信号を出力，パソコンの内部に入って来る信号を入力とする。

2.2 各ブロックの説明

2-2-1　インターフェイス用IC回路

図2(a)(b)にインターフェイス用IC回路を示す。PC-9801の拡張I/Oとしてボードを作る場合，最初にI/Oアドレスを決めなければならない。PC-9801VM2では，ユーザ用にDOH-DFHが開放されているが，マウスインターフェイス用にD1Hを使用している。したがって，安全のためD8Hから使用した。

8255には，入出力を制御するための8ビットのポートが3個と，これらのポートを制御するためのコントロールレジスタが2個ある。3個のポートは，ポートA，B，Cといわれる。このうち，ポートCだけは少し特殊で，コントロールレジスタにより1ビットごとに操作ができ，その他の時は8255の内部で自動的に操作される。レジスタは，MSBが0か1かによって働きが違う。今回は，MSBを1とし，ポートAを出力，ポートBを入力にしてポートCは使用しなかった。また，ポートAは，モータ駆動回路に，ポートBは，センサ回路に接続されている。

2-2-2　モータ駆動回路

図3(a)(b)にモータ駆動回路を示す。この回路は，回路図の

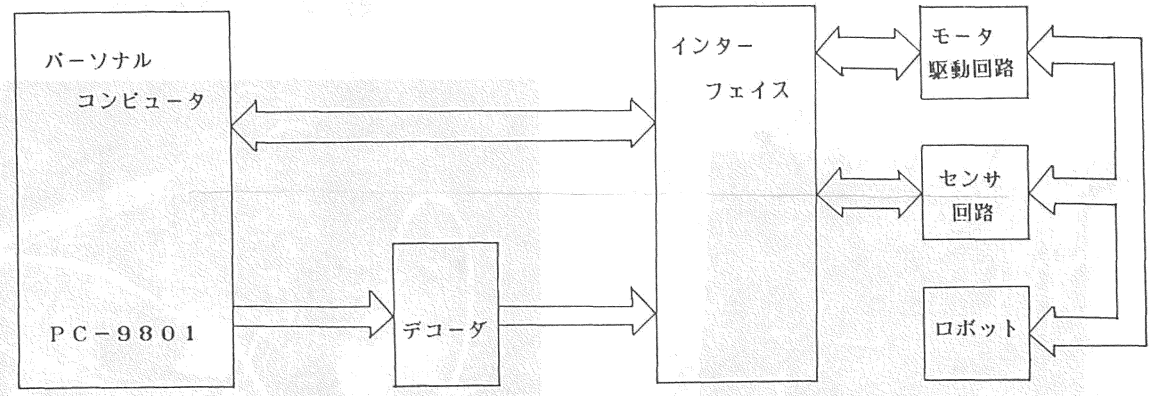


図1　a)　ハードウエアの構成

† 〒184 小金井市梶野町3-7-2

* 現・トヨタ自動車工業株式会社

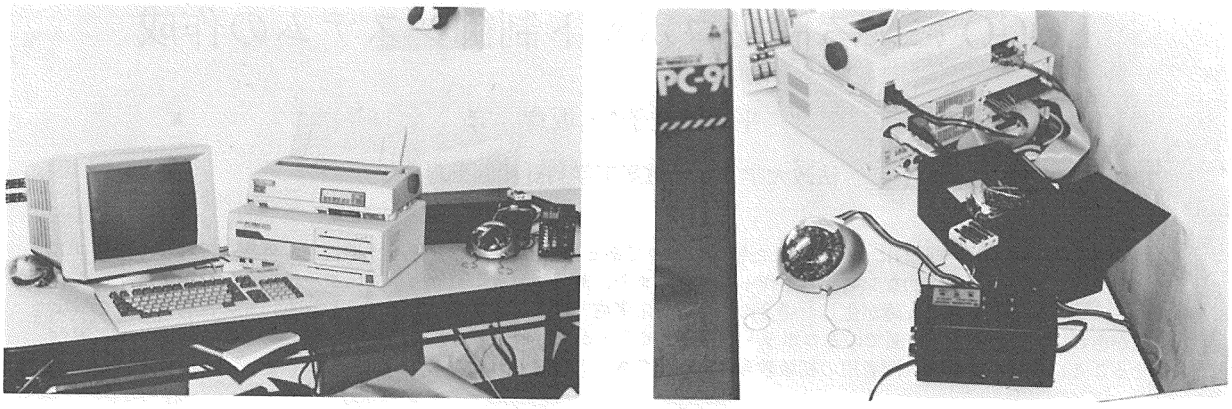


図1　b)　本システムの外観

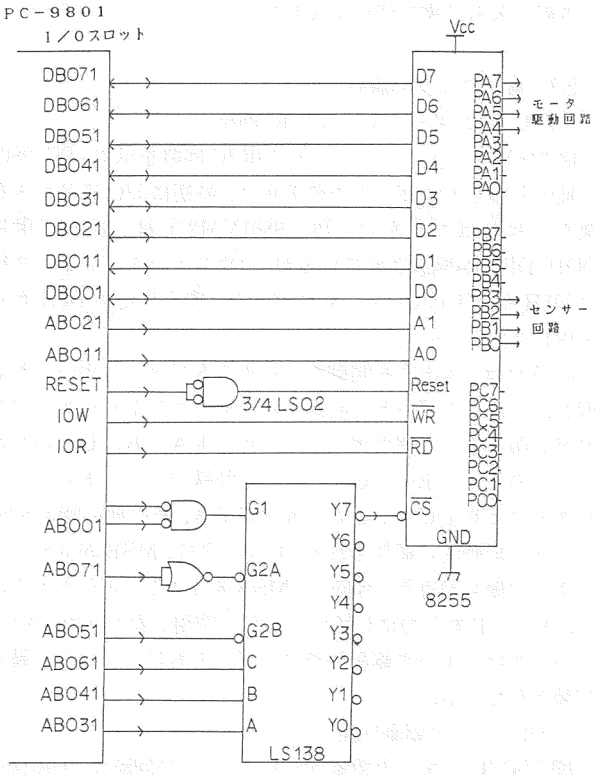


図2　a)　インターフェイス用 IC 回路図

左下のリレーで右車輪のモータの ON/OFF を，左上のリレーで車輪の正転，逆転制御を行っている。同様に，右下のリレーで左車輪のモータの ON/OFF を，右上のリレーで車輪の正転，逆転制御を行っている。この回路は，8255 のポート A と，ロボットのモータに接続される。ポート A からの信号が，“H”のとき，モータは ON または正回転し，“L”のとき，モータは OFF または逆回転する。この回路で車輪を止めるとき，電流を止めただけでは，モータが惰性で動く。モータは，発電機の役割を持っているので，モータの端子をショートさせて，運動エネルギーを電流が流れることによっておこる熱エネルギーに変換し，消費させてモータを止めるようにしてある。

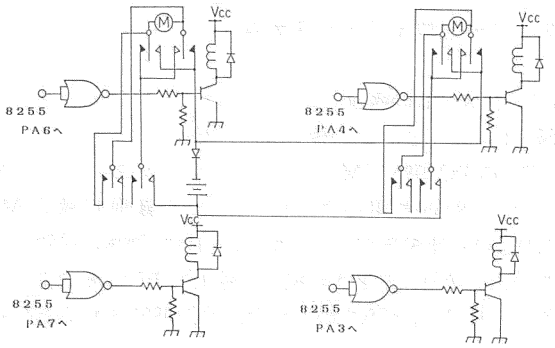


図3　a)　モータ駆動回路図

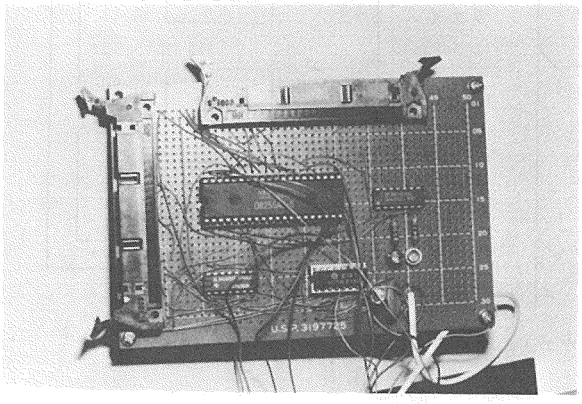


図2　b)　8255 周辺の外観

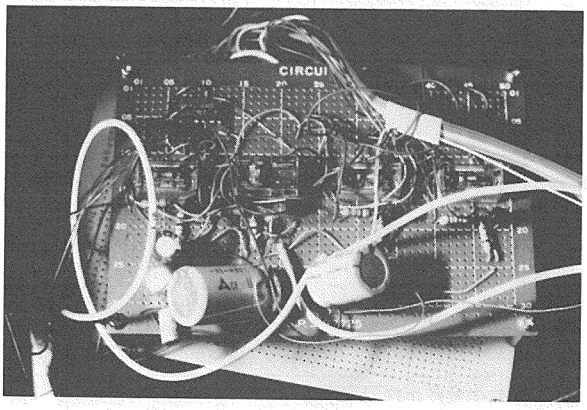


図3　b)　モータ駆動回路の外観

2-2-3　センサ回路

図4(a)，(b)にタッチセンサ回路，光センサ回路を示す。センサ回路は，2個ずつ製作した。タッチセンサ回路では，物体がタッチセンサに当るとスイッチが入り，“H”の信号をインターフェイス用IC8255のポートBに送る。
光センサは，光の量を検知するcdsを用い，暗くなると“H”，明るくなると“L”の信号をインターフェイス用IC8255のポートBに送る。

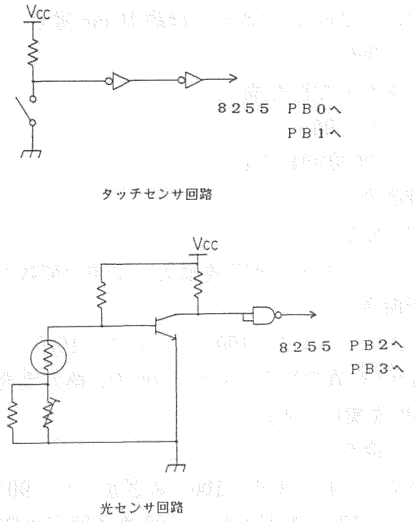


図4　a)　センサ回路図

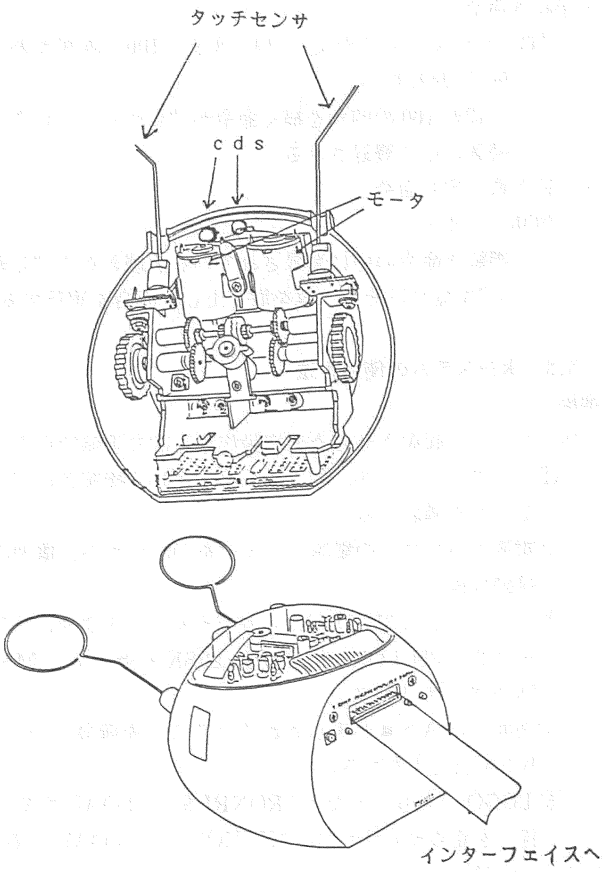


図5　a)　ロボットの構造

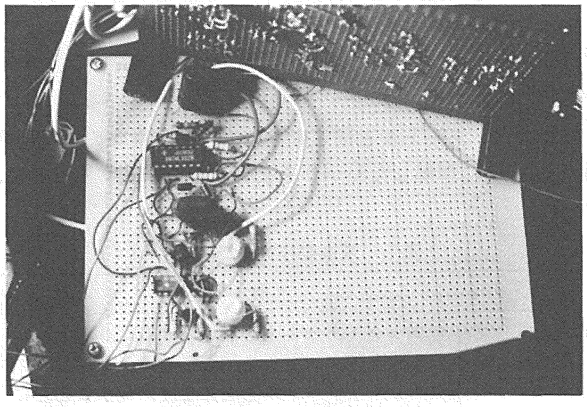
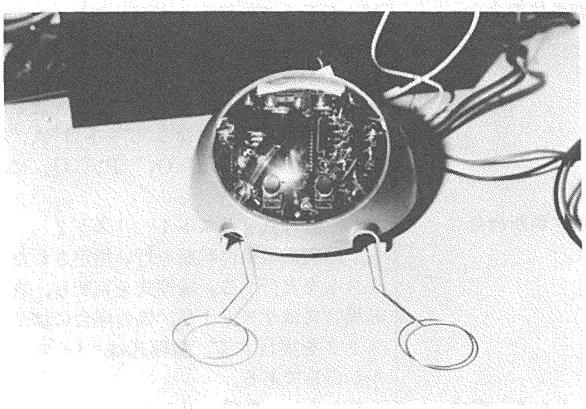


図4　b)　センサ回路の外観

2-2-4　ロボット

図5にロボットの構造を示す。ロボットは，嘉穂無線のMV-9511WAOのメカ部分を用いた。このメカを用いた理由は，LOGOのタートルと同じ動作が可能であるという点と，左右の駆動軸をクラウンギアで連結し，同期を取ることにより，アナログ2モータの欠点であった非直進性が一挙に解決され，直進性を持っているという点である。このようなメカのロボットにタッチセンサ，及び光センサを取り付け，インターフェイス部とケーブルでつないだ。



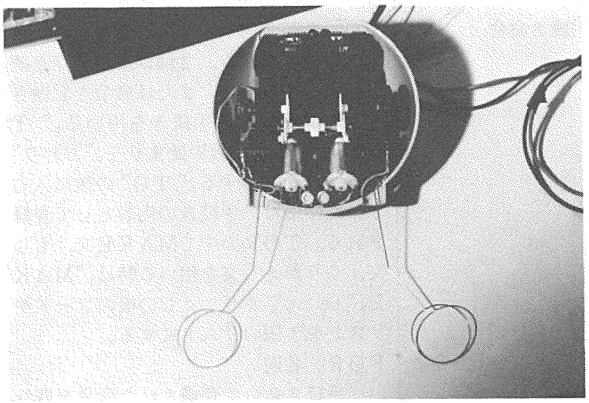


図5　b)　ロボットの外観

3. ソフトウェア・システム

3-1 本章の目的

本章では，タートルと同じ動作の命令と数種類の制御命令等のLOGOに準拠したわかりやすい言語を作成すると共に，その言語を用いて前章で作成したロボットシステムを作動させるソフトウェアをLOGOの上に構築した。

表1 基本動作命令の種類

コマンド	機　能
ま　え	ロボット・タートルを前進させる。
うしろ	ロボット・タートルを後退させる。
うせつ	ロボット・タートルを右に1度回転するごとに1歩ずつ前進させる。
させつ	ロボット・タートルを左に1度回転するごとに1歩ずつ前進させる。
みぎまわり	ロボット・タートルを右に回転させる。
ひだりまわり	ロボット・タートルを右に1度回転するごとに1歩ずつ後退させる。
うしろうせつ	ロボット・タートルを右に1度回転するごとに1歩ずつ後退させる。
うしろさせつ	ロボット・タートルを左に1度回転するごとに1歩ずつ後退させる。
みぎいどう	ロボット・タートルを右に移動させる。
ひだりいどう	ロボット・タートルを左に移動させる。

表2 制御命令の種類

命　令	書　式・機　能
変数処理命令	MAKE　ワード　オブジェクト 指定したワードにオブジェクトを値として設定する。この命令によってワードはオブジェクトを値として持つ変数として機能するようになる。ワードは"A," K," L," M," N," O," P," Q，に限られている。
条件判断命令	もし　論理式　リスト1　リスト2 論理式の条件判断を行い指定されたリストを実行する。論理式を判断し，真の場合にはリスト1を，偽の場合にはリスト2を実行する。論理式は＝(イコール）のみである。
くりかえし命令	くりかえし　数値　リスト 指定された回数だけリストを実行する。
手続き命令	TO　名前　命令・・・　命令　おわり 手続きを作る。2個以上の動作を続けて命令する場合，または命令を記憶させたい場合にこの手続きを用いる。"TO"で手続きの定義の始まりを,"おわり"で終わりを宣言する。"TO"の後に入力された命令は，手続きの内容として登録される。TO文の中でMAKE文，もし文，くりかえし文を用いる時は,"MAKE," もし," くりかえしの後のワードやリストを1個のリストにする。
手続き取り出し命令	FOR　名前 手続き命令で登録された命令を取り出す。

3.2 ロボット言語の種類

ロボットの基本動作を**表1**に，制御命令を**表2**に示す。各々の命令の使い方を例題を挙げて説明する。

・基本動作命令

まえ　100

タートル及びロボットが100歩前進する。面画上では，約2 cm，ロボットは約10 cm進む。

うせつ　360

右まわりで円を描く。

みぎまわり　90

右に90度回転する。

・変数処理命令

MAKE A 3

Aというワードが3を値として持つ変数となる。

・条件判断命令

もし　A＝3〔まえ　100〕〔うしろ　100〕

A＝3が真の場合，まえ　100を，偽の場合　うしろ100を実行する。

・くりかえし命令

くりかえし　4〔まえ　100　みぎまわり　90〕

まえ　100　みぎまわり　90を4回ずつ実行する。その結果，1辺の長さが100の四角が描ける。

・手続き命令

TO　しかく　くりかえし〔4〔まえ　100　みぎまわり　90〕〕おわり

1辺が100の四角を描く命令が“しかく”という手続きの中に登録される。

・手続き取り出し命令

FOR　しかく

手続き命令の中に登録されている手続きから“しかく"を見つけその内容を取り出し，命令を実行する。

3.3 本システムの使用方法

準備

本システムを起動させる為の準備作業を次の手順で行う。

①インターフェイス，パソコン等が正しく接続されていることを確認する。

②電源とパソコンの電源スイッチをONにする。電池の接続は後で行う。

③ドライブ1に3D-LOGOオリジナルディスクを，ドライブ2に3D-LOGO　WORK-DISKをセットしMS-DOSを立ち上げる。

④画面上にA＞■が表示されていることを確認したら，B：A↵と入力する。

⑤LOGOが立ち上ったら“RONRI 9”をLOADする。線上を走らせる場合は，“HOKOU 7”をLOADする。

システムの起動

次の手順でロボット，タートルを動かす。

① HAJIME ↵と入力する。
②画面上に“命令を入れて下さい。”と出てきたら，ロボットの電池を入れる。（ロボットを動かしたくない時は，電池を入れない。）
③各種の命令を入れる。１つの命令ごとにリターンキーを押す。
④リターンキーを押すと，画面上で文字が消え，タートルが現われ命令どおり（手続命令を除く）の動作をする。このとき，ロボットも同時に命令どおりの動作をする。(基本動作のみタートルが動き終ってからロボットが動き出す)
⑤終了するときは“おわり”と入力する。システムが終了する。

終了

システムを終了し，画面上に OK が出たら“. SYSTEM ↵”と入力する。

3.4　ソフトウェアの流れ

ロボット言語は 3D-LOGO で書かれてあり，24 個のプロシジャーから成り立っている。このプロシジャーの流れを図 6 に示し，その目的を説明する。

HAJIME

このプロシジャーからプログラムが始まる。出力ポートを初期化する。

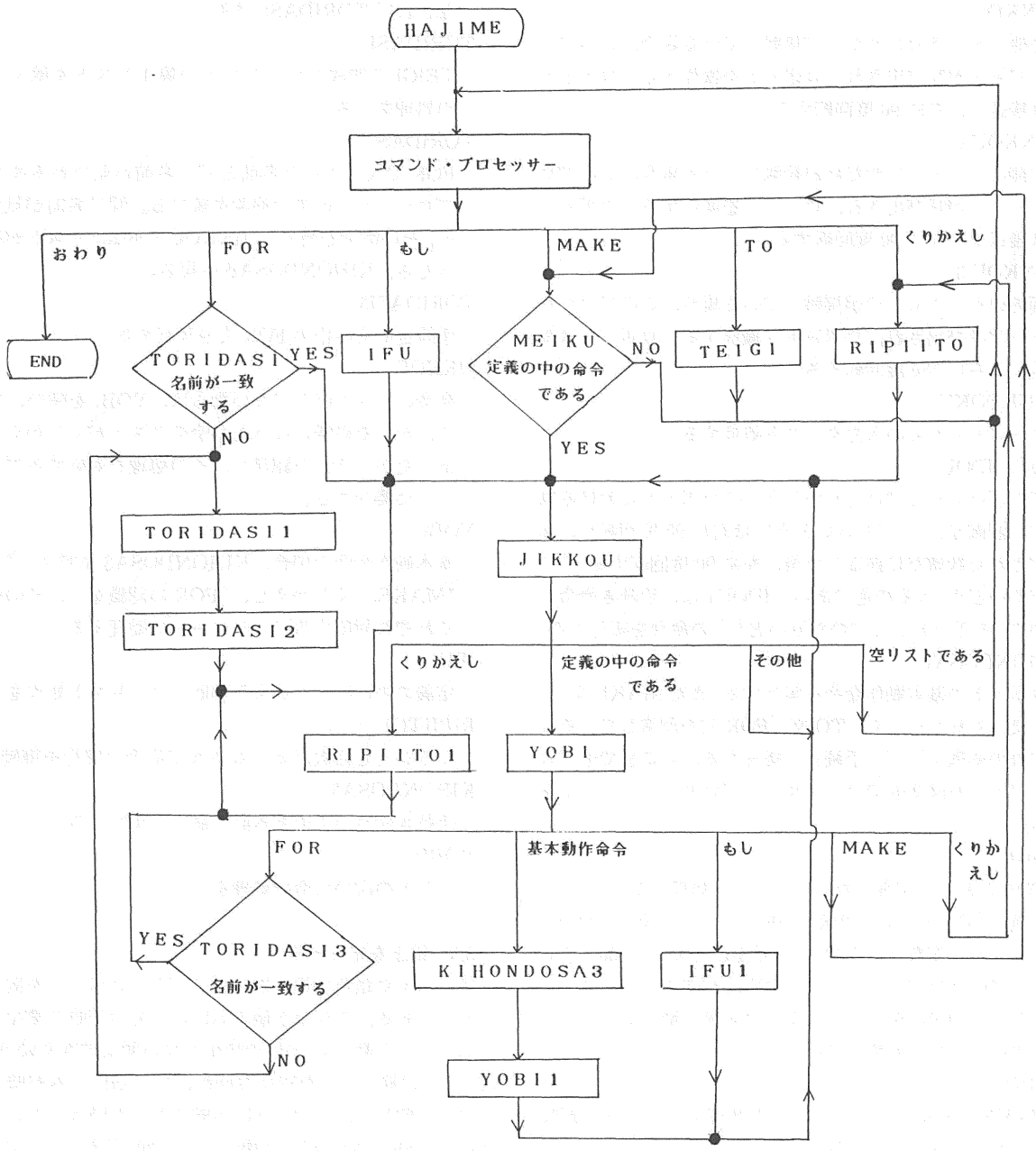


図 6　ソフトウエアの流れ

KIHONDOSA0
HENKA を呼び，ロボットと物体との接触がないかどうかを調べた後，画面上に，“命令を入れて下さい。”というプリント文を出し，ユーザの入力を待つ。NYURYOKU で入力された命令の基本動作命令のみをタートルで動かした後，KIHONDOSA 1 を呼ぶ。

HENKA
ロボットが物体に接触しているかどうかをタッチセンサからの入力で調べる。ロボットが物体に接触している場合，接触情報の区分により，HENKOU 1，HENKOU 2，HENKOU 3 のいずれかのプロシジャーでロボットの操作を行う。物体に接触していない場合と，操作を終えた場合は，KIHONDOSA0 に戻る。

HENKOU 1
左側のタッチセンサだけが接触している場合，このプロシジャーが呼び出され，ロボットを操作する。ロボットは後退し，右に 90 度回転する。

HENKOU 2
右側のタッチセンサだけが接触している場合，このプロシジャーが呼び出され，ロボットを操作する。ロボットは後退し，左に 90 度回転する。

HENKOU 3
両方のタッチセンサが接触している場合，このプロシジャーが呼び出され，ロボットを操作する。ロボットは後退し，右に 180 度回転する。

NYURYOKU
キーボードからの入力をリスト処理する。

IDOU，IDOU1
〝みぎいどう〞，〝ひだりいどう〞のロボットにおける動作の制御を行う。〝みぎいどう〞は右に 90 度回転し，指定された数値だけ前進した後，左に 90 度回転する。〝ひだりいどう〞はその逆である。IDOU1 は，手続き命令の中の〝みぎいどう〞，〝ひだりいどう〞の命令を実行する。

KIHONDOSA1
ロボットの基本動作命令を実行する。また MAKE 文，もし文，くりかえし文，TO 文，FOR 文を認識して，それぞれの処理を以下の手続きに委託する。ここまでのプロシジャーを図 2-6 では，コマンドプロセッサーとしてある。

TEIGI
TO～おわりで定義したものをリスト処理する。
例　TO　ぜんご　まえ　100　うしろ　100　おわり
グローバル変数：D の中に〔〔ぜんご〔まえ　100　うしろ　100　おわり〕〕〕と入り次の定義が作られるとぜんごのリストの後に付く。このように定義が増えていくと：D の中のリストも増えていく。

MEIKU
MAKE で始まった命令をリスト処理する。ワードが指定されているワードに等しいならばそのワードの中にオブジェクトを入れる。

IFU
もしで始まった命令のリスト処理をする。
例　もし　A＝3　〔まえ　100〕〔うしろ　100〕
〝A が指定されているワードに等しいならば，そのワードの内容と〞 3 とを比較し，等しいならば〔まえ　100〕違っていれば〔うしろ　100〕をもって JIKKOU を呼ぶ。

RIPIITO
くりかえしで始まった命令のリスト処理と，くりかえし回数から 1 を減算して，くりかえし回数を減らしていく。

TORIDASI
FOR で入力された名前と同じ名前を TEIGI で作成したリストの中から検索する。このプロシジャーでは，TEIGI で作成した 1 番目のリストを検索する。同じ名前が発見されるとその命令を持って JIKKOU を呼ぶ。発見されなければ TORIDASII を呼ぶ。

TORIDASII
TEIGI で作成されたリストの第 1 リストを除くリストの処理をする。

TORIDASI2
FOR で入力された名前と同じ名前が見つかるまでこのプロシジャーの中で検索を続ける。同じ名前が見つかると，その命令を持って JIKKOU を呼ぶ。リストが空になったら，KIHONDOSAϕ へ戻る。

TORIDASI3
手続き命令の中の FOR 文を実行する。

JIKKOU
命令のリストが空でないならば，YOBI を呼び，空であり，かつその空になった命令のリストがくりかえし文であったかどうかを認識し，その処理を対応するプロシジャーに委託する。

YOBI
基本動作命令の場合，KIHONDOSA3 を呼ぶ。〝もし，〝MAKE,〞くりかえし，〝FOR の認識をし，その処理をそれぞれ対応するプロシジャーに委託する。

IFU1
定義の中に出てくる条件判断命令のリスト処理をする。

RIPIITO1
くりかえし回数が ϕ になるまで命令の実行を継続る。

KIHONDOSA3
手続き命令の中の基本動作命令を実行する。

YOBI1
リストの次の命令の処理をする。

3.5　線上を走らせる

黒い机上に銀色で線を描き，その上にロボットを置いて線上を走らせる。これは 2 個の cds が，光の明暗の変化を調べてロボットを動かす。cds の変化がない間は前進を続け，左の cds のみが暗になった時は右回転，右の cds のみが暗になった時は左回転し，左右の cds が暗になった時は，右に 180 度回転し，cds の変化がない場合，左に 360 回転し，さらに cds の変化がない場合は右に 180 度回転し元の位置に戻り動作を終える。ここで，暗になった時というのは，センサが線上か

らずれ，光の反射が暗（机が黒い為）になったということである。また，走行中物体に接触した場合は，動作を終える。

4. 結　　言

ロボット制御システムで，ロボットとインターフェイスを接続してあるケーブルが太く，ロボットが回転しずらいという点と，ロボットだけが乾電池で動いており，乾電池の消耗度によって，ロボットの速度に多少の違いが出てくるという点に問題がある。しかし，それほど大きな問題ではないと思われる。また，ロボット言語として，絵を描く為に役立つ命令を主に作成したが，この他に，ペンアップやペンダウン(絵を描いたり描かなかったりさせる)，ペンの配色や描いた絵を消す等のより楽しくなる命令や手続きの表示，削除等，命令の拡大が可能である。

今後，以上のようにハード，ソフト両面でのさまざまなアイデアによって，改良を加えれば，楽しいシステムになると思われる。

プログラム・リスト

```
TO HAJIME
 GRAPHIC OFF
 OUTPORT 222 131
 OUTPORT 216 0
 MAKE "D []
 MAKE "B3 []
 MAKE "X 0
 MAKE "X1 []
 MAKE "X2 []
 MAKE "X3 []
 MAKE "Y 0
 MAKE "Y1 0
 MAKE "X9 []
 KIHONDOSA0 :X :Y
END

TO KIHONDOSA0 :X :Y
 CT
 HENKA
 PR [命令を入れて下さい。]
 NYURYOKU
 GRAPHIC ON
 MAKE "U 0
 IF EQUAL? :X "まえ [WHILE :U < :Y [FD 1 MAKE "U :U + 1]]
 IF EQUAL? :X "うしろ [WHILE :U < :Y [BK 1 MAKE "U :U + 1]]
 IF EQUAL? :X "うせつ [WHILE :U < :Y [FD 1 RT 1 MAKE "U :U + 1]]
 IF EQUAL? :X "させつ [WHILE :U < :Y [FD 1 LT 1 MAKE "U :U + 1]]
 IF EQUAL? :X "みぎまわり [WHILE :U < :Y [RT 1 MAKE "U :U + 1]]
 IF EQUAL? :X "ひだりまわり [WHILE :U < :Y [LT 1 MAKE "U :U + 1]]
 IF EQUAL? :X "うしろうせつ [WHILE :U < :Y [BK 1 RT 1 MAKE "U :U + 1]]
 IF EQUAL? :X "うしろさせつ [WHILE :U < :Y [BK 1 LT 1 MAKE "U :U + 1]]
 IF EQUAL? :X "みぎいどう [WHILE :U < :Y [MR 1 MAKE "U :U + 1]]
 IF EQUAL? :X "ひだりいどう [WHILE :U < :Y [ML 1 MAKE "U :U + 1]]
 IF EQUAL? :X "おわり [GRAPHIC OFF]
 KIHONDOSA1 :X :Y
END

TO HENKA
 MAKE "I 0
 MAKE "I INPORT 218
 IF EQUAL? :I 1 [HENKOU1]
 IF EQUAL? :I 2 [HENKOU2]
 IF EQUAL? :I 3 [HENKOU3]
 IF EQUAL? :I 5 [HENKOU1]
 IF EQUAL? :I 6 [HENKOU2]
 IF EQUAL? :I 7 [HENKOU3]
 IF EQUAL? :I 9 [HENKOU1]
 IF EQUAL? :I 10 [HENKOU2]
 IF EQUAL? :I 11 [HENKOU3]
 IF EQUAL? :I 13 [HENKOU1]
 IF EQUAL? :I 14 [HENKOU2]
 IF EQUAL? :I 15 [HENKOU3]
END
```

```
TO HENKOU1
 GRAPHIC ON
 PR [ﾛﾎﾞｯﾄ を動かします。]
 MAKE "R 0
 MAKE "S 0
 MAKE "T 0
 MAKE "U 0
 WHILE :R < 100 [BK 1 MAKE "R :R + 1] WHILE :S < 100 [OUTPORT 216 170 MAKE "S :S
 + 1] WHILE :T < 90 [RT 1 MAKE "T :T + 1] WHILE :U < 90 [OUTPORT 216 186 MAKE "U
 :U + 1]
END

TO HENKOU2
 GRAPHIC ON
 PR [ﾛﾎﾞｯﾄを動かします。]
 MAKE "R 0
 MAKE "S 0
 MAKE "T 0
 MAKE "U 0
 WHILE :R < 100 [BK 1 MAKE "R :R + 1] WHILE :S < 100 [OUTPORT 216 170 MAKE "S :S
 + 1] WHILE :T < 90 [LT 1 MAKE "T :T + 1] WHILE :U < 90 [OUTPORT 216 234 MAKE "U
 :U + 1]
END

TO HENKOU3
 GRAPHIC ON
 PR [ﾛﾎﾞｯﾄを動かします。]
 MAKE "R 0
 MAKE "S 0
 MAKE "T 0
 MAKE "U 0
 WHILE :R < 100 [BK 1 MAKE "R :R + 1] WHILE :S < 100 [OUTPORT 216 170 MAKE "S :S
 + 1] WHILE :T < 180 [RT 1 MAKE "T :T + 1] WHILE :U < 180 [OUTPORT 216 186 MAKE
"U :U + 1]
END

TO NYURYOKU
 MAKE "DATA REQUEST
 MAKE "X FIRST :DATA
 MAKE "YY BUTFIRST :DATA
 MAKE "Y LAST :DATA
END
```

```
TO KIHONDOSA1 :X :Y
 MAKE "R 0
 PR [ﾛﾎﾞｯﾄがｽﾀｰﾄします。]
 IF EQUAL? :X "まえ [OUTPORT 216 250 WHILE :R < :Y [MAKE "R :R + 1] OUTPORT 216
0 KIHONDOSA0 :X :Y]
 IF EQUAL? :X "うしろ [OUTPORT 216 170 WHILE :R < :Y [MAKE "R :R + 1] OUTPORT 21
6 0 KIHONDOSA0 :X :Y]
 IF EQUAL? :X "うせつ [OUTPORT 216 200 WHILE :R < :Y [MAKE "R :R + 1] OUTPORT 21
6 0 KIHONDOSA0 :X :Y]
 IF EQUAL? :X "させつ [OUTPORT 216 50 WHILE :R < :Y [MAKE "R :R + 1] OUTPORT 216
 0 KIHONDOSA0 :X :Y]
 IF EQUAL? :X "みぎまわり [OUTPORT 216 186 WHILE :R < :Y [MAKE "R :R + 1] OUTPOR
T 216 0 KIHONDOSA0 :X :Y]
 IF EQUAL? :X "ひだりまわり [OUTPORT 216 234 WHILE :R < :Y [MAKE "R :R + 1] OUTP
ORT 216 0 KIHONDOSA0 :X :Y]
 IF EQUAL? :X "うしろうせつ [OUTPORT 216 136 WHILE :R < :Y [MAKE "R :R + 1] OUTP
ORT 216 0 KIHONDOSA0 :X :Y]
 IF EQUAL? :X "うしろさせつ [OUTPORT 216 34 WHILE :R < :Y [MAKE "R :R + 1] OUTPO
RT 216 0 KIHONDOSA0 :X :Y]
 IF EQUAL? :X "みぎいどう [IDOU KIHONDOSA0 :X :Y]
 IF EQUAL? :X "ひだりいどう [IDOU KIHONDOSA0 :X :Y]
 IF EQUAL? :X "TO [TEIGI]
 IF EQUAL? :X "FOR [TORIDASI]
 IF EQUAL? :X "もし [IFU]
 IF EQUAL? :X "MAKE [MEIKU KIHONDOSA0 :X :Y]
 IF EQUAL? :X "くりかえし [RIPIITO]
 IF EQUAL? :X "おわり [OUTPORT 216 0 PR [終わります] TOPLEVEL]
END

TO TEIGI
 MAKE "B []
 MAKE "B1 []
 MAKE "B2 []
 MAKE "C []
 MAKE "B BUTFIRST :DATA
 MAKE "B1 FIRST :B
 MAKE "B2 BUTFIRST :B
 MAKE "C LIST :B1 :B2
 MAKE "D LPUT :C :D
 KIHONDOSA0 :X :Y
END

TO MEIKU
 MAKE "E FIRST :YY
 MAKE "E1 LAST :YY
 IF EQUAL? :E "A [MAKE "A :E1]
 IF EQUAL? :E "K [MAKE "K :E1]
 IF EQUAL? :E "L [MAKE "L :E1]
 IF EQUAL? :E "M [MAKE "M :E1]
 IF EQUAL? :E "N [MAKE "N :E1]
 IF EQUAL? :E "O [MAKE "O :E1]
 IF EQUAL? :E "P [MAKE "P :E1]
 IF EQUAL? :E "Q [MAKE "Q :E1]
 IF EQUAL? :X "FOR [MAKE "X3 BUTFIRST :X2 JIKKOU]
END
```

```
TO IFU
 MAKE "F FIRST :YY
 MAKE "F1 BUTFIRST :YY
 MAKE "F2 FIRST :F1
 MAKE "F3 LAST :F1
 MAKE "F4 FIRST :F
 MAKE "F5 LAST :F
 IF EQUAL? :F4 "A [MAKE "XX :A]
 IF EQUAL? :F4 "K [MAKE "XX :K]
 IF EQUAL? :F4 "L [MAKE "XX :L]
 IF EQUAL? :F4 "M [MAKE "XX :M]
 IF EQUAL? :F4 "N [MAKE "XX :N]
 IF EQUAL? :F4 "O [MAKE "XX :O]
 IF EQUAL? :F4 "P [MAKE "XX :P]
 IF EQUAL? :F4 "Q [MAKE "XX :Q]
 IF EQUAL? :XX :F5 [MAKE "X3 :F2 JIKKOU] [MAKE "X3 :F3 JIKKOU]
END

TO RIPIITO
 MAKE "J FIRST :YY
 MAKE "J1 BUTFIRST :YY
 MAKE "J :J - 1
 MAKE "X3 FIRST :J1
 JIKKOU
END

TO TORIDASI
 MAKE "B4 []
 MAKE "B5 []
 MAKE "B6 []
 MAKE "B7 []
 MAKE "B3 BUTFIRST :DATA
 MAKE "B7 FIRST :B3
 MAKE "B4 FIRST :D
 MAKE "B5 FIRST :B4
 MAKE "B6 BUTFIRST :B4
 MAKE "X3 FIRST :B6
 IF NOT EMPTY? :D [IF EQUAL? :B7 :B5 [JIKKOU] [TORIDASI1]] [KIHONDOSA0 :X :Y]
END

TO TORIDASI1
 MAKE "D1 BUTFIRST :D
 TORIDASI2
END

TO TORIDASI2
 MAKE "B4 FIRST :D1
 MAKE "B5 FIRST :B4
 MAKE "B6 BUTFIRST :B4
 MAKE "X3 FIRST :B6
 IF NOT EMPTY? :D1 [IF EQUAL? :B7 :B5 [JIKKOU] [MAKE "D1 BUTFIRST :D1 TORIDASI2]
] [KIHONDOSA0 :X :Y]
END

TO JIKKOU
 IF NOT EQUAL? :X "FOR [MAKE "X9 :X]
 IF NOT EMPTY? :X3 [YOBI] [IF EQUAL? :X9 "もし [MAKE "X3 :X8 MAKE "X9 [] JIKKOU]
 [IF EQUAL? :X9 "くりかえし [RIPIITO1] [KIHONDOSA0 :X :Y]]]
END
```

```
TO YOBI
 MAKE "Y2 0
 MAKE "X1 FIRST :X3
 MAKE "X2 BUTFIRST :X3
 IF EQUAL? :X1 "もし [MAKE "X8 BUTFIRST :X2 MAKE "Y2 FIRST :X2 MAKE "X9 :X1 IFU1
]
 IF EQUAL? :X1 "MAKE [MAKE "YY FIRST :X2 MEIKU]
 IF EQUAL? :X1 "くりかえし [MAKE "X8 BUTFIRST :X2 MAKE "YY FIRST :X2 MAKE "X9 :X
1 RIPIITO]
 MAKE "Y1 FIRST :X2
 KIHONDOSA3 :X1 :Y1
END

TO IFU1
 MAKE "X5 0
 MAKE "H FIRST :Y2
 MAKE "H1 BUTFIRST :Y2
 MAKE "H2 FIRST :H1
 MAKE "H3 LAST :H1
 MAKE "H4 FIRST :H
 MAKE "H5 LAST :H
 IF EQUAL? :H4 "A [MAKE "X5 :A]
 IF EQUAL? :H4 "K [MAKE "X5 :K]
 IF EQUAL? :H4 "L [MAKE "X5 :L]
 IF EQUAL? :H4 "M [MAKE "X5 :M]
 IF EQUAL? :H4 "N [MAKE "X5 :N]
 IF EQUAL? :H4 "O [MAKE "X5 :O]
 IF EQUAL? :H4 "P [MAKE "X5 :P]
 IF EQUAL? :H4 "Q [MAKE "X5 :Q]
 IF EQUAL? :X5 :H5 [MAKE "X3 :H2 JIKKOU] [MAKE "X3 :H3 JIKKOU]
END

TO RIPIITO1
 MAKE "X3 FIRST :J1
 IF EQUAL? :J "0 [IF EQUAL? :X "FOR [MAKE "X3 :X8 MAKE "X9 [] JIKKOU] [KIHONDOSA
0 :X :Y]] [MAKE "J :J - 1 JIKKOU]
END

TO KIHONDOSA3 :X1 :Y1
 CT
 PR [ﾛﾎﾞｯﾄがｽﾀｰﾄします]
 GRAPHIC ON
 MAKE "V 0
 MAKE "W 0
 IF EQUAL? :X1 "まえ [WHILE :V < :Y1 [FD 1 MAKE "V :V + 1] OUTPORT 216 250 WHILE
 :W < :Y1 [MAKE "W :W + 1] OUTPORT 216 0 YOBI1]
 IF EQUAL? :X1 "うしろ [WHILE :V < :Y1 [BK 1 MAKE "V :V + 1] OUTPORT 216 170 WHI
LE :W < :Y1 [MAKE "W :W + 1] OUTPORT 216 0 YOBI1]
 IF EQUAL? :X1 "うせつ [WHILE :V < :Y1 [FD 1 RT 1 MAKE "V :V + 1] OUTPORT 216 20
0 WHILE :W < :Y1 [MAKE "W :W + 1] OUTPORT 216 0 YOBI1]
 IF EQUAL? :X1 "させつ [WHILE :V < :Y1 [FD 1 LT 1 MAKE "V :V + 1] OUTPORT 216 50
 WHILE :W < :Y1 [MAKE "W :W + 1] OUTPORT 216 0 YOBI1]
 IF EQUAL? :X1 "みぎまわり [WHILE :V < :Y1 [RT 1 MAKE "V :V + 1] OUTPORT 216 186
 WHILE :W < :Y1 [MAKE "W :W + 1] OUTPORT 216 0 YOBI1]
 IF EQUAL? :X1 "ひだりまわり [WHILE :V < :Y1 [LT 1 MAKE "V :V + 1] OUTPORT 216 2
34 WHILE :W < :Y1 [MAKE "W :W + 1] OUTPORT 216 0 YOBI1]
 IF EQUAL? :X1 "うしろうせつ [WHILE :V < :Y1 [BK 1 RT 1 MAKE "V :V + 1] OUTPORT
216 136 WHILE :W < :Y1 [MAKE "W :W + 1] OUTPORT 216 0 YOBI1]
 IF EQUAL? :X1 "うしろさせつ [WHILE :V < :Y1 [BK 1 LT 1 MAKE "V :V + 1] OUTPORT
216 34 WHILE :W < :Y1 [MAKE "W :W + 1] OUTPORT 216 0 YOBI1]
 IF EQUAL? :X1 "みぎいどう [IDOU1]
```

```
 IF EQUAL? :X1 "ひだりいどう [IDOU1]
 IF EQUAL? :X1 "おわり [GRAPHIC OFF OUTPORT 216 0 YOBI1]
END

TO YOBI1
 MAKE "X3 BUTFIRST :X2
 JIKKOU
END

TO IDOU
 MAKE "R 0
 MAKE "S 0
 MAKE "T 0
 IF EQUAL? :X "みぎいどう [WHILE :S < 45 [OUTPORT 216 186 MAKE "S :S + 1] OUTPOR
T 216 250 WHILE :R < :Y [MAKE "R :R + 1] WHILE :T < 45 [OUTPORT 216 234 MAKE "T
:T + 1] OUTPORT 216 0]
 MAKE "R 0
 MAKE "S 0
 MAKE "T 0
 IF EQUAL? :X "ひだりいどう [WHILE :T < 45 [OUTPORT 216 234 MAKE "T :T + 1] OUTP
ORT 216 250 WHILE :R < :Y [MAKE "R :R + 1] WHILE :S < 45 [OUTPORT 216 186 MAKE "
S :S + 1] OUTPORT 216 0]
END

TO IDOU1
 MAKE "R 0
 MAKE "S 0
 MAKE "T 0
 MAKE "A1 0
 IF EQUAL? :X1 "みぎいどう [WHILE :A1 < :Y1 [MR 1 MAKE "A1 :A1 + 1] WHILE :S < 4
5 [OUTPORT 216 186 MAKE "S :S + 1] OUTPORT 216 250 WHILE :R < :Y1 [MAKE "R :R +
1] WHILE :T < 45 [OUTPORT 216 234 MAKE "T :T + 1] OUTPORT 216 0 YOBI1]
 MAKE "R 0
 MAKE "S 0
 MAKE "T 0
 MAKE "A1 0
 IF EQUAL? :X1 "ひだりいどう [WHILE :A1 < :Y1 [ML 1 MAKE "A1 :A1 + 1] WHILE :S <
 45 [OUTPORT 216 234 MAKE "S :S + 1] OUTPORT 216 250 WHILE :R < :Y1 [MAKE "R :R
+ 1] WHILE :T < 45 [OUTPORT 216 186 MAKE "T :T + 1] OUTPORT 216 0 YOBI1]
END
```

```
TO HAJIME                    (線上を走らせる)
 CT
 OUTPORT 222 131
 OUTPORT 216 0
 PR [ﾛﾎﾞｯﾄを線上に置いて下さい。]
 PR [ﾛﾎﾞｯﾄを動かしますか？ Y / N]
 MAKE "DATA REQUEST
 IF EQUAL? :DATA "[Y] [DOUSA]
 IF EQUAL? :DATA "[N] [PRINT "終わります。 TOPLEVEL]
END

TO DOUSA
 CT
 OUTPORT 216 0
 GRAPHIC ON
 MAKE "A 0
 MAKE "B 0
 MAKE "C 0
 MAKE "S 0
 MAKE "T 0
 MAKE "U 0
 MAKE "V 0
 MAKE "I INPORT 218
 WHILE :I = 0 [MAKE "A 1 OUTPORT 216 250 FD 1 MAKE "I INPORT 218]
 IF :A = 1 [OUTPORT 216 0 MAKE "R 0 OUTPORT 216 170 BK 1 WHILE :R < 5 [MAKE "R :
R + 1] OUTPORT 216 0]
 WHILE :I = 4 [MAKE "A 2 OUTPORT 216 186 RT 1 MAKE "B :B + 1 MAKE "I INPORT 218
IF :B > 53 [MAKE "I 16]]
 IF :A = 2 [OUTPORT 216 0 MAKE "R 0 OUTPORT 216 234 LT 1 WHILE :R < 5 [MAKE "R :
R + 1] OUTPORT 216 0]
 WHILE :I = 8 [MAKE "A 3 OUTPORT 216 234 LT 1 MAKE "C :C + 1 MAKE "I INPORT 218
IF :C > 53 [MAKE "I 16]]
 IF :A = 3 [OUTPORT 216 0 MAKE "R 0 OUTPORT 216 186 RT 1 WHILE :R < 5 [MAKE "R :
R + 1] OUTPORT 216 0]
 WHILE :I = 12 [MAKE "A 4 OUTPORT 216 186 RT 1 MAKE "S :S + 1 MAKE "I INPORT 218
 IF :S > 22 [OUTPORT 216 234 LT 1 WHILE :T < 22 [MAKE "T :T + 1] OUTPORT 216 0 W
HILE :I = 12 [OUTPORT 216 234 LT 1 MAKE "U :U + 1 MAKE "I INPORT 218 IF :U > 40
[OUTPORT 216 186 RT 1 WHILE :V < 78 [MAKE "V :V + 1] OUTPORT 216 0 MAKE "I 16]]]
]
 IF :A = 4 [OUTPORT 216 0 MAKE "R 0 OUTPORT 216 234 LT 1 WHILE :R < 5 [MAKE "R :
R + 1] OUTPORT 216 0]
 IF :I = 1 [MAKE "I 16 HAJIME]
 IF :I = 2 [MAKE "I 16 HAJIME]
 IF :I = 3 [MAKE "I 16 HAJIME]
 IF :I = 5 [MAKE "I 16 HAJIME]
 IF :I = 6 [MAKE "I 16 HAJIME]
 IF :I = 7 [MAKE "I 16 HAJIME]
 IF :I = 9 [MAKE "I 16 HAJIME]
 IF :I = 10 [MAKE "I 16 HAJIME]
 IF :I = 11 [MAKE "I 16 HAJIME]
 IF :I = 13 [MAKE "I 16 HAJIME]
 IF :I = 14 [MAKE "I 16 HAJIME]
 IF :I = 15 [MAKE "I 16 HAJIME]
 IF :I = 16 [HAJIME]
 IF :I < 16 [DOUSA]
END
```

参考文献

1) M.Lesser：“LOGO FOR MICROS”, Newnes Technical Books, 1985
2) 榊正憲・鈴木隆・波磨薫：“周辺装置の製作“, アスキー, 1983
3) “トランジスタ技術“, 1986
4) “PO 9801 VM ユーザーズマニュアル“, 日本電気
5) “3D-LOGO REFERENCE MANUAL”

キー・ワード

LOGO, タートル・グラフィックス, ロボットの制御, タッチ・センサ, 光センサ

Summary

Simple Robot Control System by Language LOGO

Asako HIRAO and Mamoru SAKAKURA

Department of Instrumentations, College of Engineering, Hosei University†

Turtle graphics are well known features of Language LOGO. Our plan is controlling simple Robot moving similar to the Turtle on CRT from LOGO programs. We added two pair of touch-sensors and photo-sensors for the detections of beeing in the way of the Robot. Using signals from these sensors, we can controle the Robot moving only on high photo-refrection way, or no defence way. The simple system of Robot is applicable to the education for novice of the computer.

Key Words

LOGO, Turtle Graphics, Robot Control, Touch Sensor, Photo Sensor.

† 3-7-2 Kajino-cho, Koganei-shi, Tokyo 184 JAPAN