卒業研究

ジョイスティックによる全方向移動車の制御

高知工科大学　知能機械システム工学科

１０１０１８８

野村　和寿

２００１年２月

目次

第一章　緒言　1

１．１ はじめに・・・・・・・・・・・・・・・・・・・・・・・1

１．２ 本研究の内容・・・・・・・・・・・・・・・・・・・・2

１．３ 本文の構成・・・・・・・・・・・・・・・・・・・・・3

第二章　全方向移動車について　4

２．１ 全方向移動車の定義・・・・・・・・・・・・・・・・4

２．２ 全方向移動車の利点・・・・・・・・・・・・・・・・5

第三章　メカニズム　7

３．１ メカニカル部の構成・・・・・・・・・・・・・・・・7

３．２ 移動時の動作・・・・・・・・・・・・・・・・・・・8

３．３ 本メカニズムの利点・・・・・・・・・・・・・・・・9

３．４ 実物・・・・・・・・・・・・・・・・・・・・・・・１０

第四章　制御システム　１３

４．１ 制御システムの構成と制御のながれ・・・・・・１３

４．２ 制御ボード・・・・・・・・・・・・・・・・・・１５

４．２．１　制御ボードの構成・・・・・・・・・１５

４．２．２　ＣＰＵボード・・・・・・・・・・・１６

４．３．３　ＡＤ－ＤＡボード・・・・・・・・・１７

４．２．３　制御ボードボックス・・・・・・・・１８

４．３　ジョイスティック・・・・・・・・・・・・・２０

４．３．１ ジョイスティックのメカニズム・・・２０

４．３．２ ジョイスティックの信号処理・・・・２１

４．３．３ スイッチ・・・・・・・・・・・・・・・２２

４．４　モーター・・・・・・・・・・・・・・・・・・２３

４．４．１ モーター制御の構成・・・・・・・・・２３

４．４．２ ドライバー・・・・・・・・・・・・・・２４

４．５　ブレーキ・・・・・・・・・・・・・・・・・・２６

４．６　角度センサー・・・・・・・・・・・・・・・・２８

４．６．１ 角度センサーのメカニズム・・・・・２８

４．６．２ 角度センサーの信号処理・・・・・・・２９

４．７　電源・・・・・・・・・・・・・・・・・・・・３１

第五章　プログラム　３４

５．１　プログラムのながれ・・・・・・・・・・・・・３４

５．２　プログラムソース・・・・・・・・・・・・・・３６

５．３　プログラムの解説・・・・・・・・・・・・・・４６

第六章　まとめ　４９

６．１　研究成果と考察・・・・・・・・・・・・・・・４９

６．２　今後の展開・・・・・・・・・・・・・・・・・５０

参考文献　５１

謝辞　５２

第一章　緒言

本文は移動ロボットのプラットホームとしての全方向移動車を，ジョイスティックによって制御した結果をまとめたものである．

１．１ はじめに

ロボットは人間にかわる労働手段として様々な研究，開発が行われてきた．その需要はおもに工場における生産，組み立て作業のオートメーション化に従事するロボットマニピュレータや，深海や宇宙空間など人が作業できない場所で働くロボットなどが求められ，基本的には人間がそこにいない，もしくは直接手などで関わらない場で作業することを前提に設計されてきた．しかし，近年においては，Ｓｏｎｙの“ AIBO ”のようなペット型ロボットや HONDA の“ Ｐ３ ”のような人間型ロボットなどアミューズメントとして人とふれあうロボ

ットが社会で注目を集めている．さらに研究段階では人間のケアをする介護ロボットなどの開発が行われ，掃除ロボットや家事ロボットのような人間の生活の場に密接したロボットの需要が増えてきている．このような場所は得てして狭く，煩雑であり，その床面もカーペット，フローリング，畳など様々なものがある．その中での移動するロボットはこの限られた空間の中で極力無駄がなく正確に動き，どのような床面にも対応する必要がある．このような条件に対応したロボットの移動手段の一つとして本研究の全方向移動車があげられる．

１．２ 本研究の内容

本研究は王によって応用が提案され，設計，作製された全方向移動車を筆者が動きを解析し，ジョイスティックによる制御システムを組み上げたものである．よって本文では制御システムの説明をメインにおいて，全方向移動車本体の設計における細かな形状，寸法などは省略し，大まかなメカニズムのみを記載する．

１．３　本文の構成

本節は本文の構成とその具体的な内容について説明する．第二章では全方向移動台車についての定義と，その利点について説明する．第三章では本研究における全方向移動車のメカニカル部の構成と移動における動作，他の全方向移動機構に対する相違点とその利点についてまとめ，実物を写真を交えて説明する.第四章では制御システムの構成と制御の流れ，そして各部品の制御に対する説明を記載．第五章では制御プログラムを解説．終わりに第六章において本研究の結果と今後の展開をまとめる．

第二章　全方向移動車について

本章では全方向移動車の定義と，一般的な車両に対してどのような利点が生ずるのかを論ずる．

２．１　全方向移動車の定義

本研究における全方向移動車とは具体的には床平面上での移動において以下の機能を持つ車両を指すこととする．

（１）車体の向きを変えることなく３６０度全ての方向に対して直進できる

（２）その場で回転することで３６０度全ての方向に車体を向けられる

２．２　全方向移動車の利点

前節において定義した機能を有することで，床平面上での移動において目標に対して移動する際に，車体を移動する方向にむき直す，目標に達したときに希望の方向に車体の向きを向け直す，といった無駄な動きを省くことができる．この移動は目標に対してきわめて直進的に行動がとれるので，制御操作もわかりやすく，容易になる．以下には自動車と全方向移動車の移動を比較した例を挙げる．

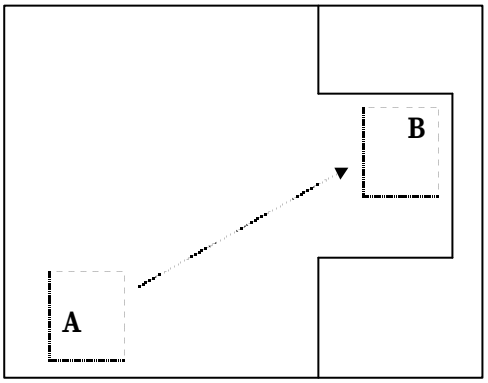


図 1

図１で図示する地形においてＡ地点からＢ地点へ自動車と全方向移動車の移動経路を比較する．点線で囲まれた四角形は各車両の大きさであり，Ｂ地点においては車体前面を図の上方に向かせることを条件とする．

自動車の場合図２の矢印で示したような経路となる．この動きは自動車の縦列駐車である．このような複雑な移動は操

縦者に技量を求める．また，車体の前後に移動するためのスペースが不可欠である．

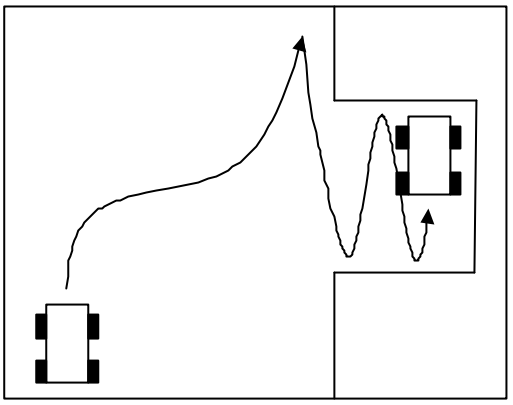


図２　自動車の場合

対して全方向移動車の場合図３の矢印で示すように，直進的かつ単純であり，移動するのに車体の幅以上のスペースを必要としていない．

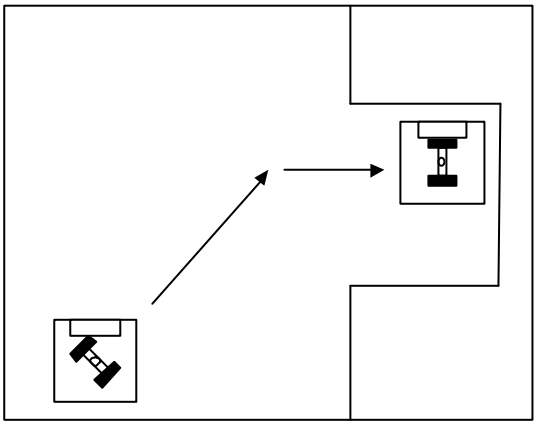


図３　全方向移動車の場合

以上はあくまで１例にすぎないが，狭く混雑な床面上での移動に対して全方向移動車が適していると言える．

第三章　メカニズム

本章では本研究における全方向移動車の機械構造と動きの検証，他で考案されている全方向移動車に対しての利点の比較を行う．そして実際に作製された全方向移動車を写真を交えて説明する．

３．１　メカニカル部の構成

本研究で制御された全方向移動台車の機構部分の構成は図４に示すようにフリーに動く四つのキャスターで支えられた台車の中央に，２つの独立して動く駆動輪を並列に配置した駆動輪部を，回転軸で連結させたものである．回転軸はブレーキによって台車と駆動輪部の固定，回転が任意にできる．図５は上からの台車，回転軸，駆動輪部を示している．

図４　全方向移動車イメージ

図５　台車，回転軸，駆動輪部

３．２　移動時の動作

車体の方向を維持したまま任意の方向に直進する場合，図６の様に，両駆動輪を異方向に駆動することで駆動輪部を回転させる（状態　１）．駆動輪部が台車に対して任意の方向をとったとき，回転軸を固定（状態　２）．両駆動輪を同方向に駆動することで直進する．（状態　３）

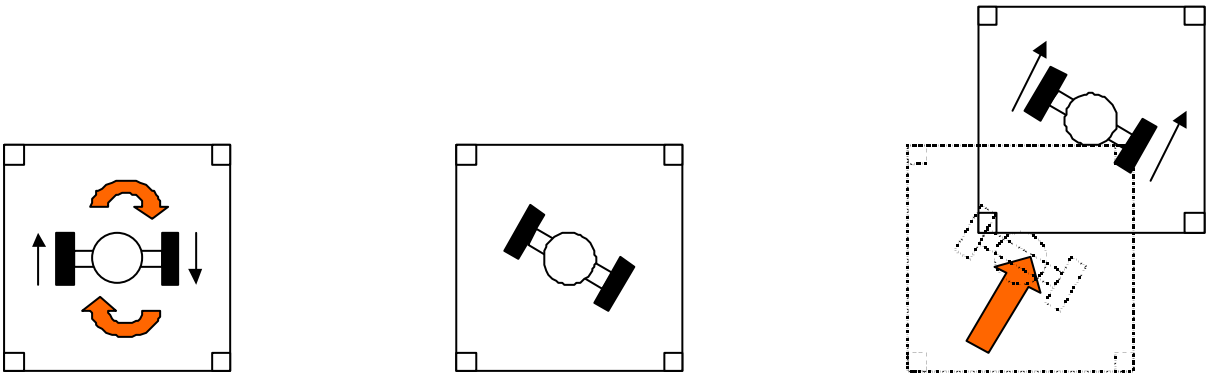


図６　６０度方向に直進するプロセス

また，回転軸を固定したまま両駆動輪を異方向に駆動することで，その場で回転し，車体を任意の方向に向かせることができる．

以上により本メカニズムは第二章で定義した全方向移動車の条件を満たしたといえる．

３．３　本メカニズムの利点

他の研究で開発されている全方向移動車の多くは，オムニホイールのといった軸方向に対する摩擦なくす特殊な車輪を用い，それを３つから４つをそれぞれ独立して駆動することによって全方向移動を実現している（図７参照）．

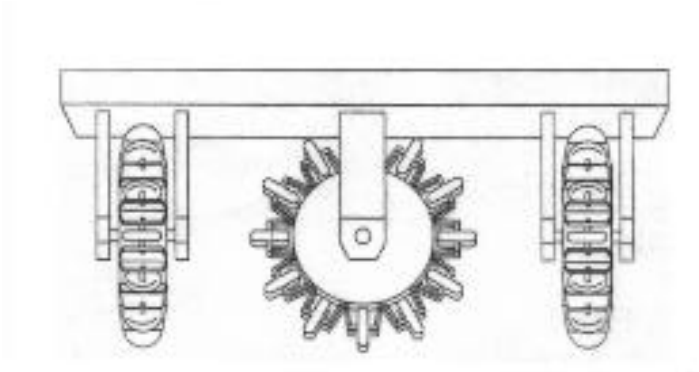


図７　オムニホイール型全方向移動システム

それに対して本全方向移動車のメカニズムの特徴は，第一に特殊な車輪を必要としていない．図７からもわかるようにオムニホイールはその複雑な形状から，駆動方向に対する床面への摩擦が弱く，積載量も制限されてしまう．また，その移動もスムーズとは言い難く移動できる床面も限られてしまう．本メカニズムは駆動輪が通常の車輪を使用するので床面の状態や積載量，室内外での使用と様々な使用条件に対応することができる．

第二に独立して駆動させる車輪が二つだけである．つまり，駆動に必要なモーターが二つですむということである．このことによりシステムの小型化，メカニズムの簡略化，費用の安価，車体重量の軽量化，電源の省エネなどさまざまな利点が生じている．

３．３　実物

本節では上記で述べたメカニズムを元に設計された実物を，メカニカル部のみを対象として写真を交えて説明する．

写真１　全方向移動車　上面

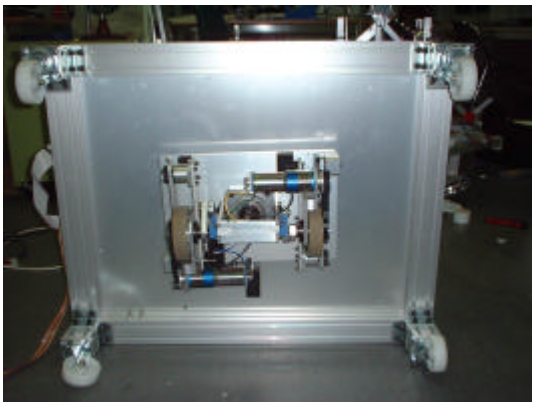


写真２　全方向移動車　下面

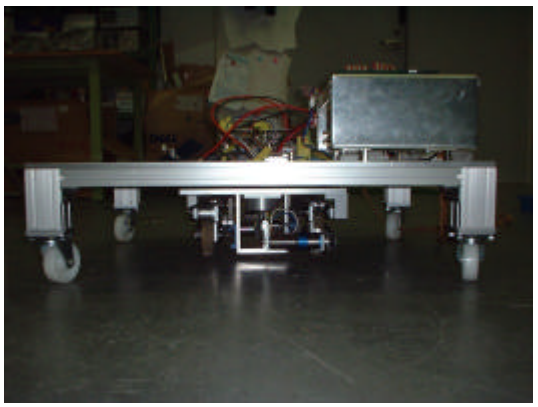


写真３　全方向移動車　側面

写真１は全方向移動車の上面部で，車体の上に乗っているのは第四章で説明する制御ボード，ドライバー，ジョイスティックである．写真２は全方向移動車の下面部である．本体の４すみにはフリーに動くキャスターが付いており，中央には２つのモーターと駆動輪を持つ駆動部がある．写真３は全方向移動車の側面部であり，４っつのキャスターと２つの駆動輪が床面に接地しているのがわかる．

台車部分の寸法は７８０×６５０×２１５（ｍｍ）である．

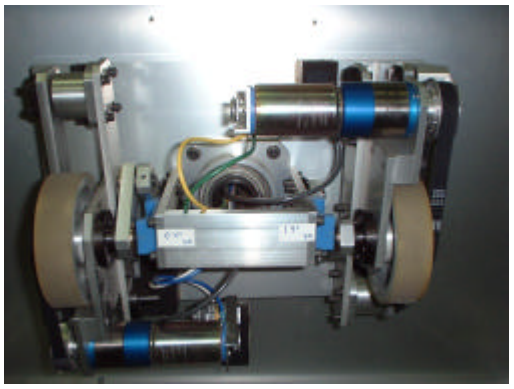


写真４　駆動輪部

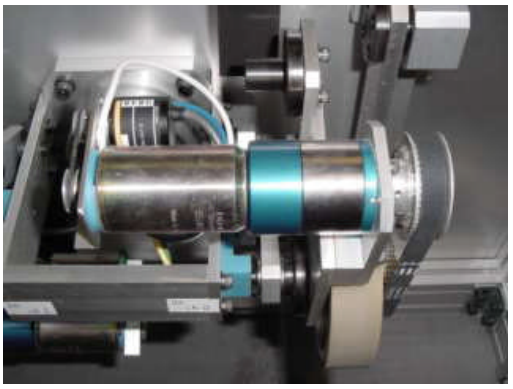


写真５　モーター

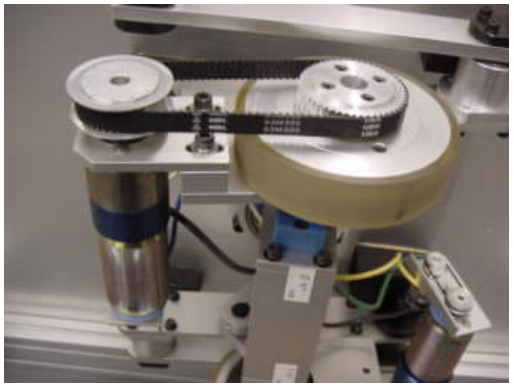


写真６　駆動輪と伝達ベルト

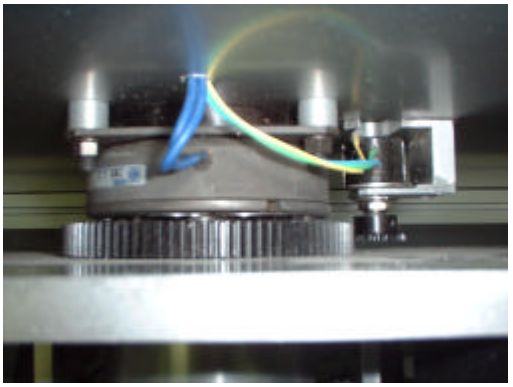


写真７　回転軸

写真４は駆動輪部，写真５はモーターである．モータから駆動輪への力の伝達は，写真６の様にベルトによって行われる．写真７は台車と駆動輪をつなぐ回転軸で，回転を固定するブレーキと角度を検出する角度センサーがついている．この２つの部品についても第四章で詳しく述べる．

第四章　制御システム

本章では制御システムの構成から制御のながれにはじまり，制御に関する各部の詳しい説明を記載する．写真８は実験段階での制御システムである．

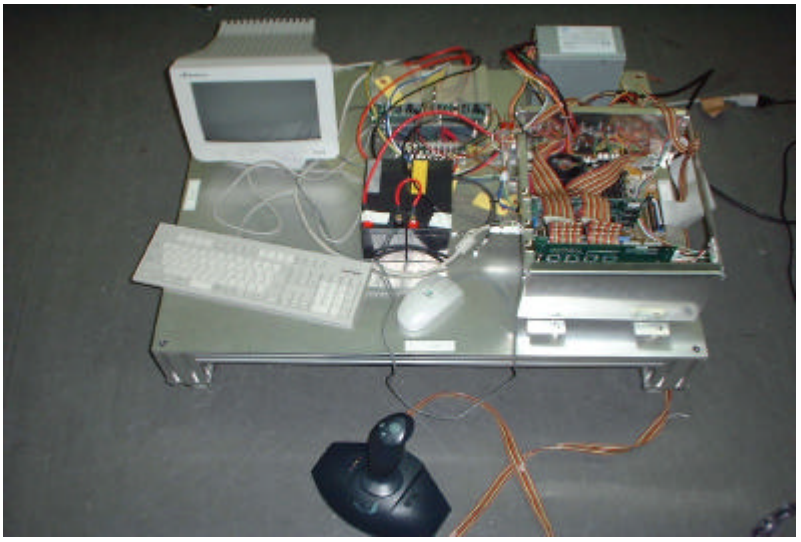


写真８　制御システム

４．１　制御システムの構成と制御のながれ

本研究の制御システムは制御ボード，ジョイスティック，モーター×２，ブレーキ，角度センサー，そして電源部によ

って構成されている．制御ボードは演算用ＣＰＵボードとアナログ信号の入出力用ＡＤ－ＤＡボードで構成されており，モーターはドライバーによって制御する．

図８は制御システムの構成図であり，矢印向きは入出力する信号の流れを表している．設置場所としてジョイスティック以外に特に指定されていないものは台車上部に置く．

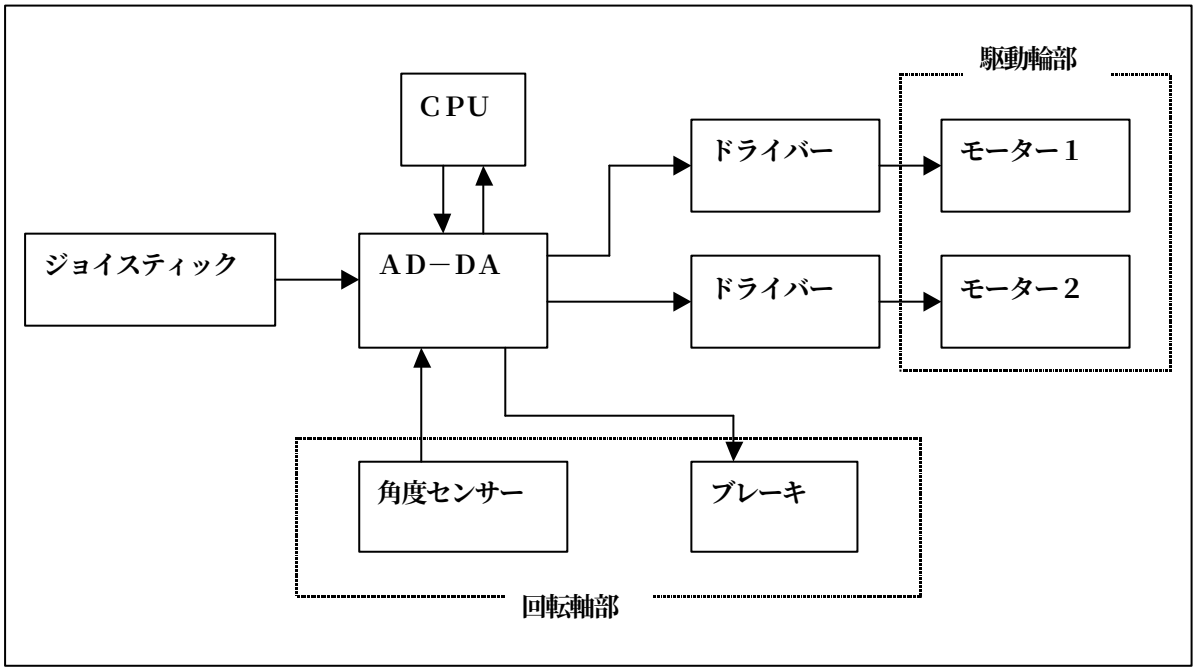


図８　制御システム構成図

制御のながれとしては，ジョイスティック，角度センサーからの信号をＣＰＵが受け取り，その状態に応じてドライバーとブレーキに信号を送る．ドライバーは受け取った信号に

応じてモーターを駆動．なお，ＣＰＵに対して入出力する信号は全てアナログ信号なので，間にＡＤ－ＤＡを通してデジタル化して処理する．

４．２　制御ボード

本研究においての制御ボードとは，単純に使用された電子基板を指すのではなく，ＣＰＵボード，ＡＤ－ＤＡボードを内包したコンピュータシステム全体のことを指す．

４．２．１　制御ボードの構成

制御ボードの構成はＣＰＵ，メモリを有するＣＰＵボードと拡張ボードであるグラフィックカード，ＬＡＮカード，ＡＤ－ＤＡボード，主記憶装置のハードディスク，入力機器のキーボード，マウス，グラフィックカードからの画像情報を表示するＣＲＴで構成される（図９参照）．

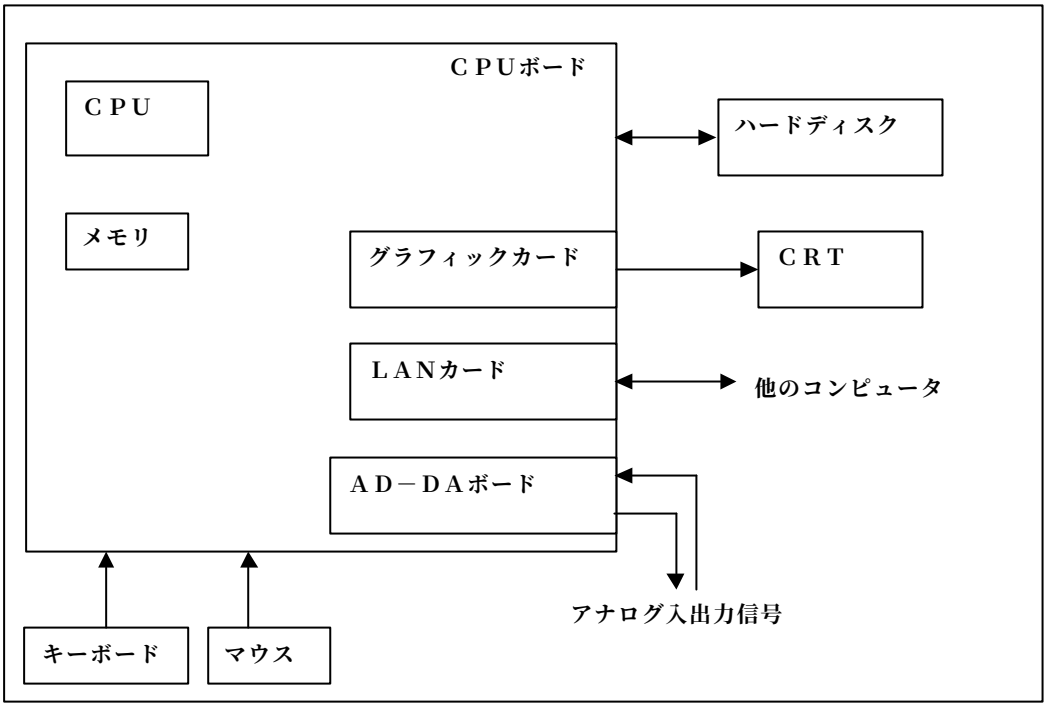


図９　制御ボードの構成図

４．２．２　ＣＰＵボード

ＣＰＵボードは産業用のものではなく，市販されている組み立てパソコン用のものを使用した．その意義は，処理速度が高速，ＩＳＡバス，ＰＣＩバスを有し拡張性が高い，安価の３点である．よって本研究のＣＰＵボードの構成は一般的なパソコンとなんら変わりは無い．主なスペックを表１で示す．

ＣＰＵ　Ｃｅｌｅｒｏｎ　７００ＭＨｚ
メインメモリ　１２８ＭＢ
ハードディスク　１０ＧＢ

表１

４．２．３　ＡＤ－ＤＡボード

ＡＤ－ＤＡボードには富士通ロボット用インターフェイスボード“ＲＩＦ－０１”を使用（写真９）．このボードはＩＳＡバス規格のハーフサイズ・ボードで，±１０ｖレンジのアナログ入力１６チャンネル，±１０ｖレンジのアナログ出力１６チャンネル有している．

写真９　ＲＩＦ－０１

それぞれのデータ長は１２ビットで，ＩＳＡバスの割り込み線はＩＲＱ９～ＩＲＱ１５まで選択ができる．この設定はジャンパピンで行い，本研究ではＩＲＱ１５を選択している．

４．２．４　制御ボードボックス

制御ボードを納めるボックスはアルミ材で作製．サイズは３５０×２８０×１６０（ｍｍ）でＣＰＵボードのサイズに合わせており，ハードディスクや電源はボックスの外に配置した．ボックス正面左側にはパワースイッチとＬＥＤを配置（写真１０）．

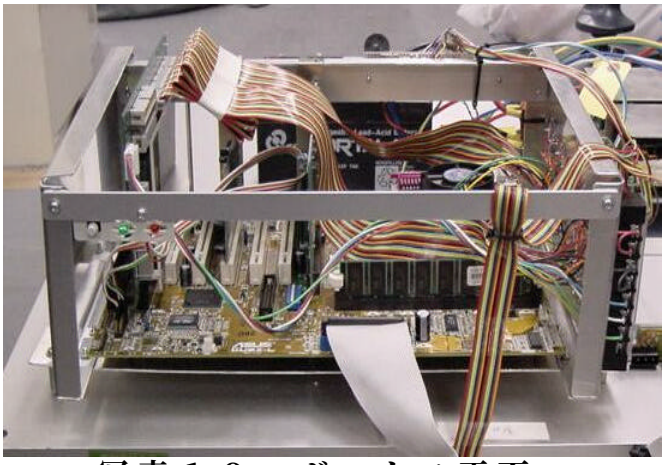


写真１０　ボックス正面

ボックス側面の右側にはセンサー，モーターなどのアナログ入出力機器の拡張を容易にするために，ＡＤ－ＤＡボードの入出力各１６チャンネルと電源＋５ｖ，±１２ｖにつないだ

コネクタを並べる（写真１１）.

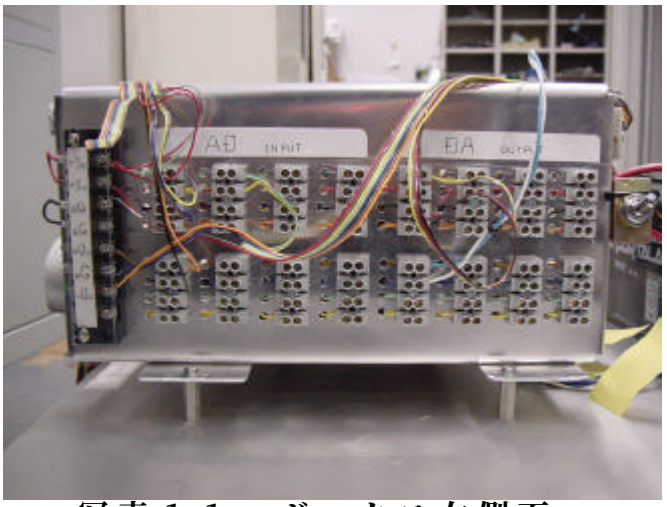


写真１１　ボックス右側面

また，背面にはキーボード，マウス，ＣＲＴの接続の他にドライバー，ブレーキ，角度センサーに接続するコネクタボードと２４ｖバッテリーのＯＮ／ＯＦＦスイッチを設置する（写真１２）.

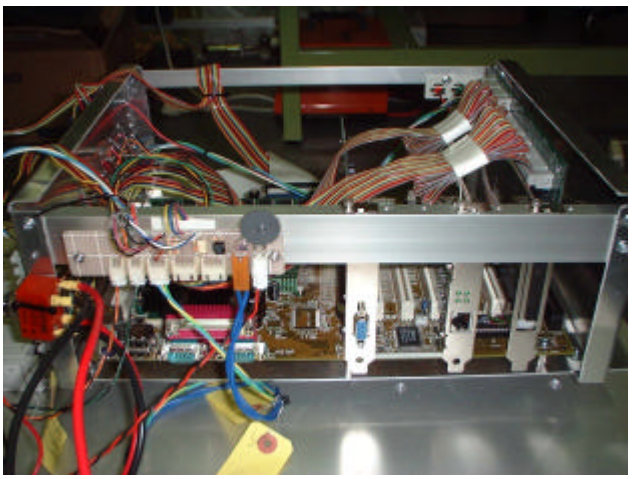


写真１２　ボックス背面

４．３　ジョイスティック

制御用のコントローラーであるジョイスティックは，市販のパソコンゲーム用のものを流用した（写真１３）.

写直１３　ジョイスティック

４．３．１ ジョイスティックのメカニズム

Ｘ－Ｙ平面の角度を指定する命令はスティックを任意の角度に倒して行う．そのメカニズムは図１０のように，スティックの動きに対してＸ，Ｙそれぞれ方向に連動するようにポテンションメーターを２つ配置する．各ポテンションメーターの信号を制御ボードのＡ／Ｄ入力チャンネル０，１に入力する.

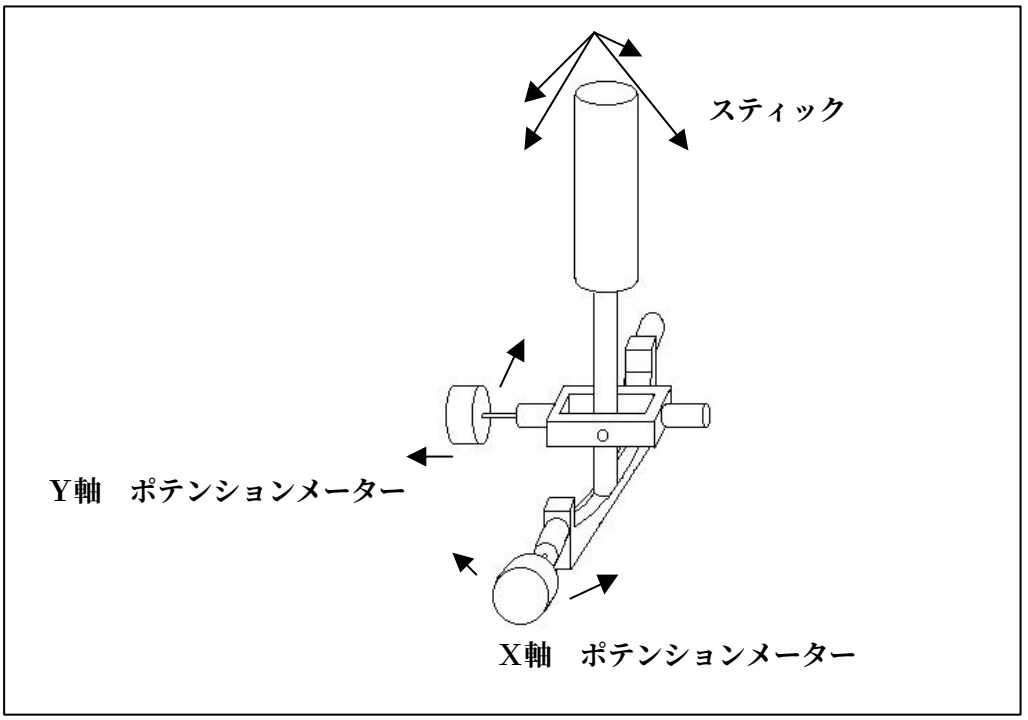


図１０　ジョイスティックのメカニズム

４．３．２　ジョイスティックの信号処理

制御ボードが受け取った信号はプログラムの中で,（１）式によって角度に変換する.

$$角度 = \frac{180}{p} \times \arctan(x, y) \qquad (1)$$

$$\left(x = \frac{x\text{軸の入力電圧}}{x\text{軸の原点時の電圧}},\ y = \frac{y\text{軸の入力電圧}}{y\text{軸の原点時の電圧}}\right)$$

さらに本研究ではこの値を±２２．５度の角度は誤差範囲として，４５度ずつの８方向（前後左右とその間の斜め４方向）に置き直している．これは人間が２０度や１５０度のような微妙な角度を認識して操作をするのは難しいと判断したからである．

４．３．３　スイッチ

また，図１２のようにこのジョイスティックには十字キー１つと４つのスイッチが搭載されている．そのうちの十字キーの左右をそれぞれＡ／Ｄ入力チャンネル２，３に接続し，その場での右回転，左回転の信号に割り当てている．その他のキーは現状では使用していない．

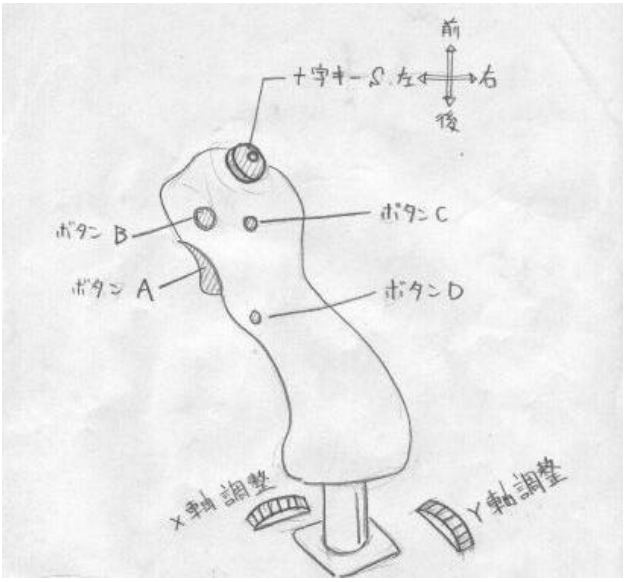


図１２　ジョイスティックのスイッチの配置図

４．４　モーター

駆動用のモーターは，ギア比４３：１のギアボックス付きの２４ｖＤＣモーターを使用．その制御はドライバーによって行われる．

４．４．１　モーター制御の構成

モーターに対する指令信号は制御ボードのＤ／Ａ出力チャンネル０，１より出力し，それぞれのモータに接続されたドライバーに信号を送る．ドライバーは指令された信号に応じて電源からモータへと電流を流す．図１３はドライバーによって制御されるモーターのイメージ図である．

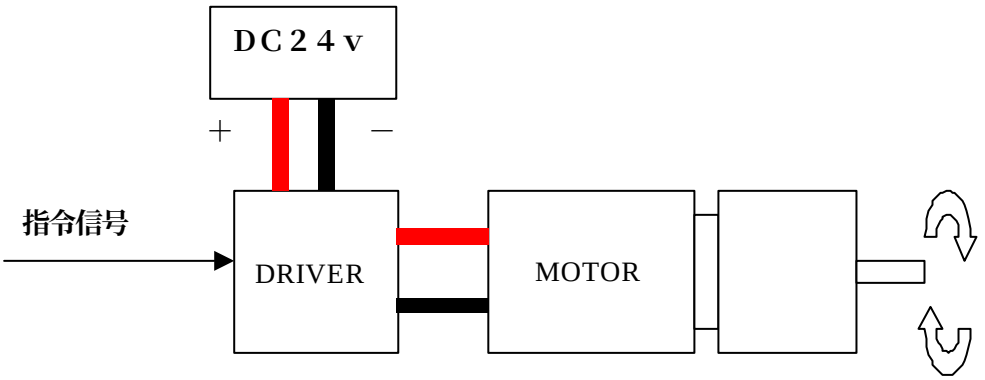


図１３　モーター制御のイメージ

４．４．２　ドライバー

モーターの制御用ドライバーには，東京工業大学の”ＴＩＴＥＣＨ　ＲＯＢＯＴ　ＤＲＩＶＥＲ”(写真１４参照)を使用した.

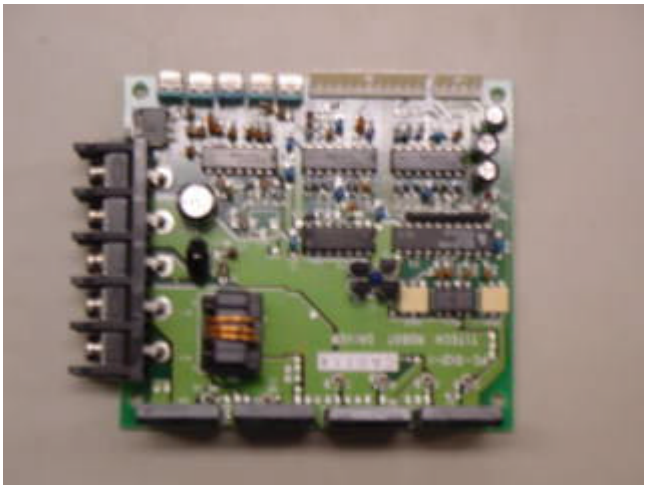


写真１４“ＴＩＴＥＣＨ　ＲＯＢＯＴ　ＤＲＩＶＥＲ”

“ＴＩＴＥＣＨ　ＲＯＢＯＴ　ＤＲＩＶＥＲ”には電流制御モード，速度制御モード，位置制御モードという３つの動作モードがある．本研究では指令信号によって速度を制御するために，速度制御モードを使用する.

速度制御モードでフィードバックに用いる回転速度情報は，疑似タコジェネレータ回路”電子ガバナ回路”によって演算する方法と，外部の機械的タコジェネレータによって測定す

る方法の２通りあるが，制御システムの簡略化のために，外部の速度計を必要としない疑似タコジェネレータ回路を使用する方法をとった．

以上の設定により，モーターは負荷がかかっても，指令信号の速度を保つよう制御される．

写真１５は放熱版をつけて本体に配置されたドライバーである．

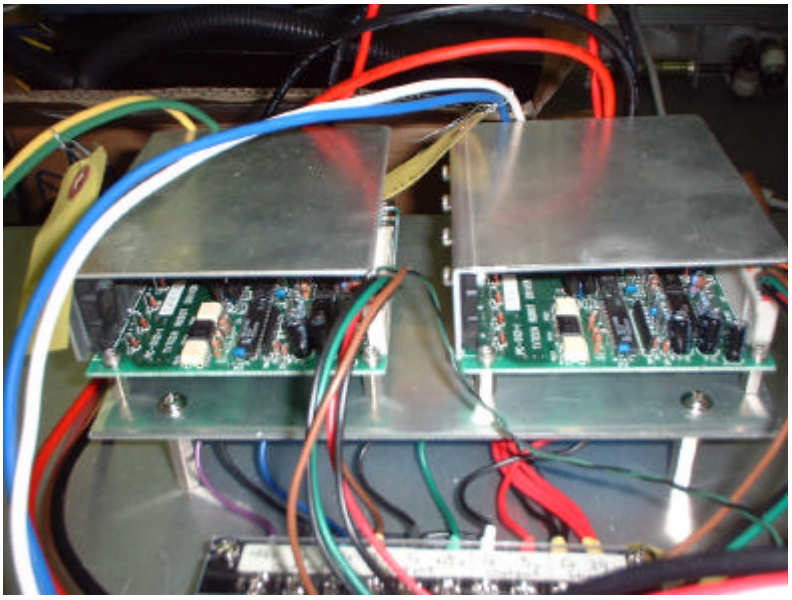
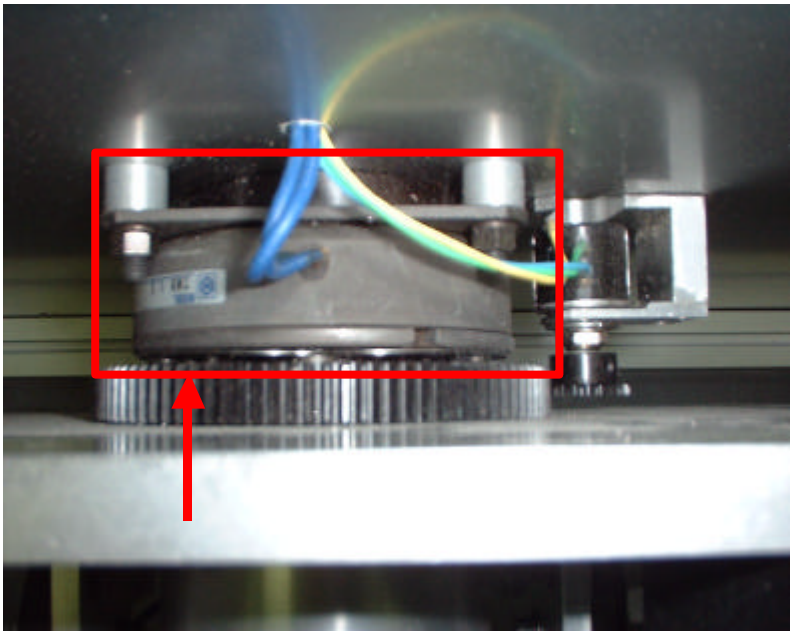


写真１５　配置されたドライバー

４．５　ブレーキ

回転軸の固定をするためのブレーキは小倉クラッチ社の“ＴＭＢ１．２”を使用（写真１６矢印）.



写直１６　ブレーキ

ＤＣ２４ｖ駆動で静摩擦トルクが１２Ｎｍである．制御ボードのＤ／Ａ出力チャンネル２からの信号を図１４に示す回路で増幅して駆動させる．回路は制御ボードボックス背面のコネクタボードの右側に設置（写真１７参照）.

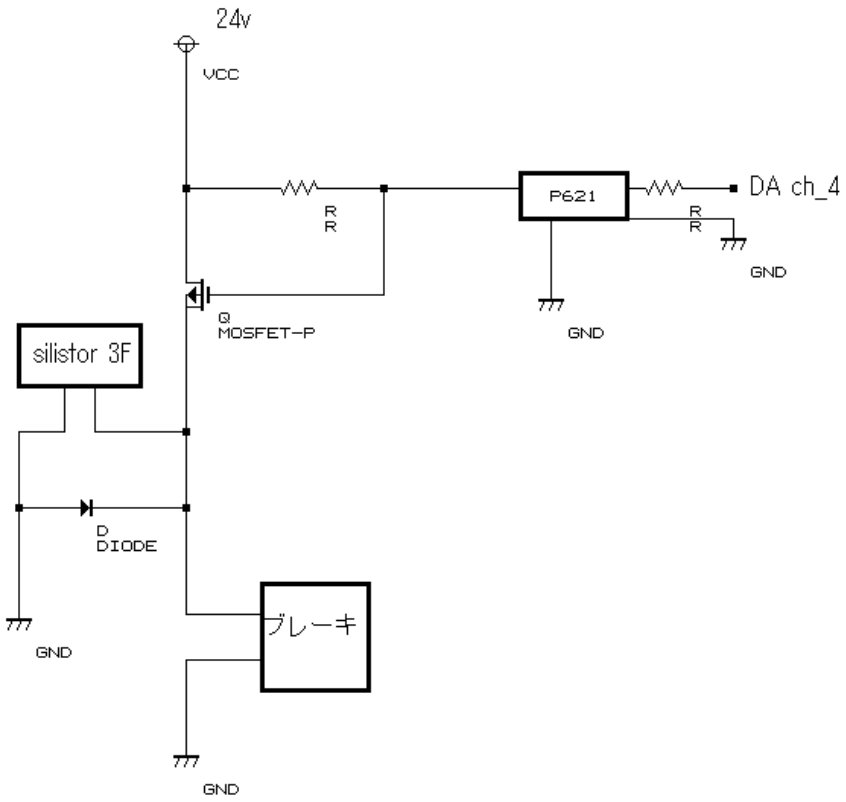
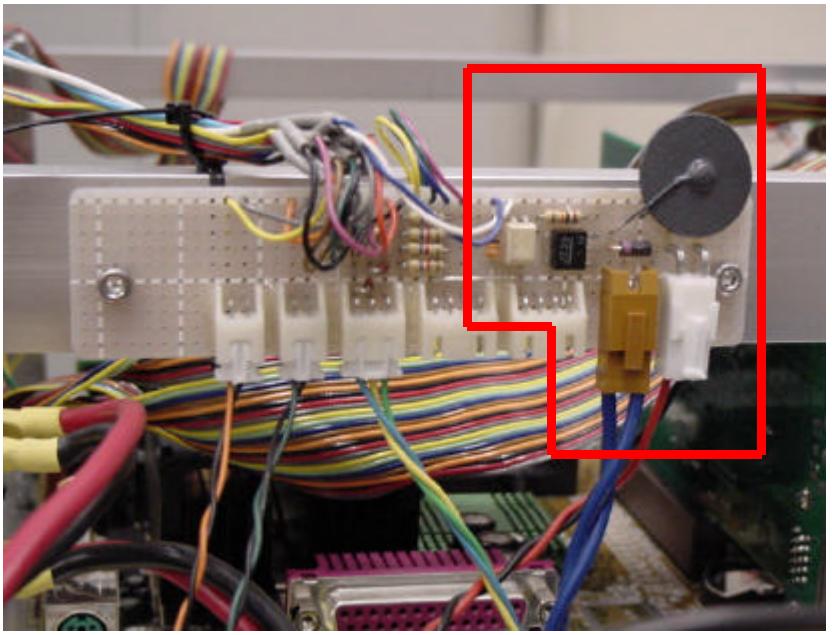


図１７　ブレーキ駆動回路図



写直１７　コネクタボード　ブレーキ駆動回路部

４．６　角度センサー

台車と駆動輪部との間の角度を測る角度センサーには，ポテンションメーターを使用（写真１８矢印）.

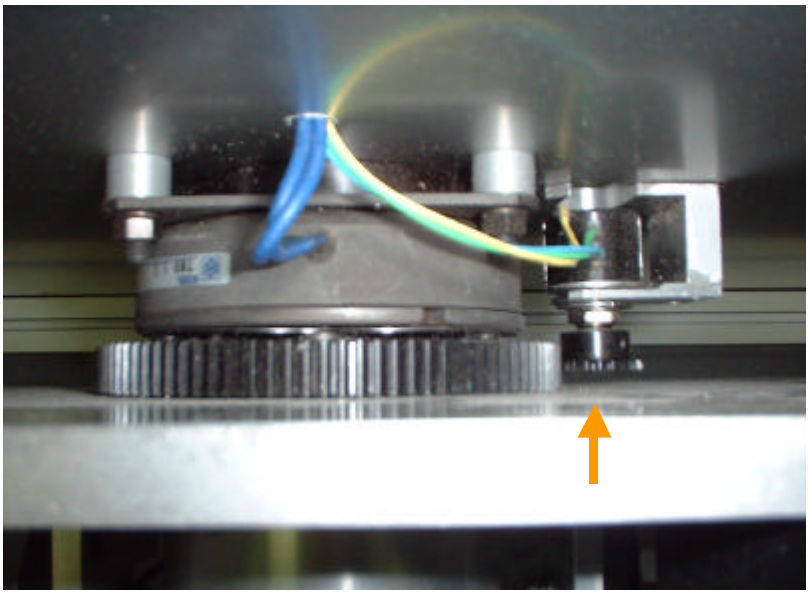


写真１８　角度センサー

４．６．１　角度センサーのメカニズム

ポテンションメーターとはつまみを回すことで抵抗値の変わる可変抵抗で，本研究では写真１８の様に回転軸の台車側に取り付けて，駆動輪部が回転するとギアによってつまみが回るようになっており，１２ｖの電圧を流したときの電圧の変化を制御ボードのＡＤ入力チャンネル４で読みとらせて，計算によって３６０度の角度に変換させている.

４．６．２　角度センサーの信号処理

表２は台車に対して駆動輪部を１５度ずつ回転させ，それぞれの角度に対して読みとった電圧の値を表にしたものである．

角度	電圧	電圧の差
180	2.71	
165	2.99	0.28
150	3.27	0.28
135	3.55	0.28
120	3.88	0.33
105	4.22	0.34
90	4.56	0.34
75	4.94	0.38
60	5.33	0.39
45	5.78	0.45
30	6.26	0.48
15	6.815	0.555
0	7.37	0.555

表２　角度に対する電圧の値

この表からは各角度間の電圧の差，つまり変化の度合いは一定ではなく，１８０度に近づくほどに大きくなっていくという特性がみられる．よって角度の算出を（１）式の放物線で近似した．

$$Y = A \times X^2 + B \times X + C \qquad (1)$$

$$(Y:\text{角度}\quad, X:\text{電圧})$$

定数 A，B，C は前後左右に対しての直進移動に重点をおいて，０度，９０度，１８０度の３点の電圧値から逆算して割り出した（表３）．表４は割り出した値によって算出した角度と理想の角度を示す．算出した角度と理想の角度の差は１．５度以内であり，誤差の範囲内であると判断した．

X	Y
2.71	180
4.56	90
7.37	0

A=	3.566561995
B=	−74.57755435
C=	355.9119844

表３　逆算した定数Ａ，Ｂ，Ｃの値

電圧	計算した角度	理想の角度
2.71	180	180
2.99	164.8105177	165
3.27	150.1802724	150
3.55	136.1092639	135
3.88	120.2435244	120
4.22	104.7094676	105
4.56	90	90
4.94	74.53581815	75
5.33	59.73572271	60
5.78	44.00684995	45
6.26	28.82149894	30
6.815	13.31215921	15
7.37	0	0

表４　算出した角度の値

４．６　電源

写真１９　ＤＣ１２ｖ，７Ａの鉛蓄電池

電源はＧＳ社のＤＣ１２ｖ，７Ａの鉛蓄電池（写真１９）２つを直列につないでＤＣ２４ｖの電源を作る．図１５に示すように，並列してＤＣ－ＤＣコンバーターを接続し，ＤＣ＋５ｖ，ＤＣ±１５ｖ，ＤＣ±１２ｖの電源を作り出す（写真２０　左から＋５ｖ，ＤＣ±１５ｖ，ＤＣ±１２ｖ）．それぞれの電源を電源配分用コネクタ（写真２１矢印）に接続．＋２４ｖをモータ駆動用として，＋５ｖと±１５ｖを制御用電源としてドライバーに供給．＋２４ｖをブレーキに供給している．また安全対策として＋２４ｖにはＯＮ／ＯＦＦスイッチを設置した

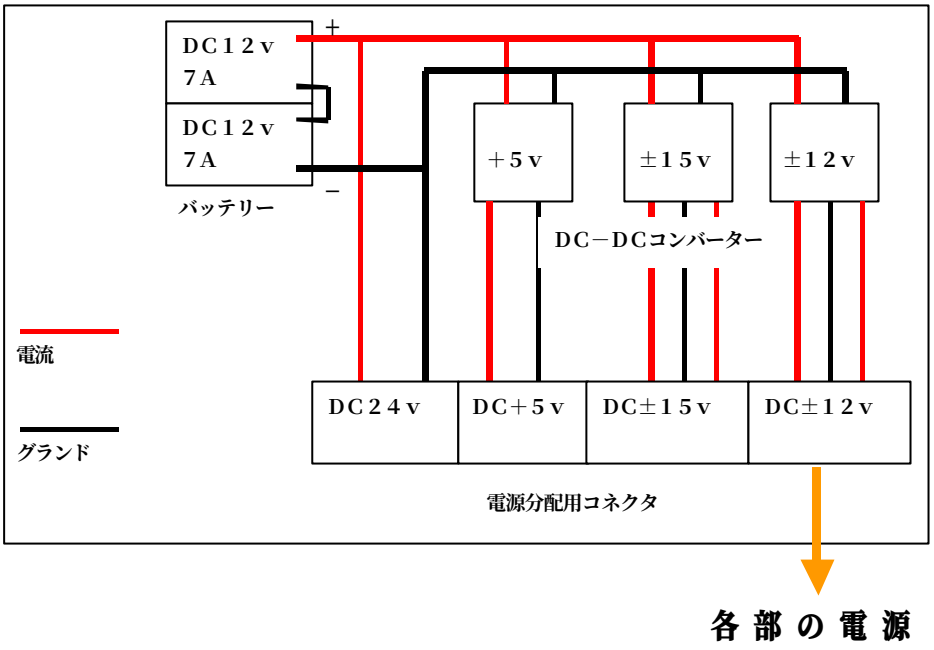


図１５　ＤＣ電源分配の配線図

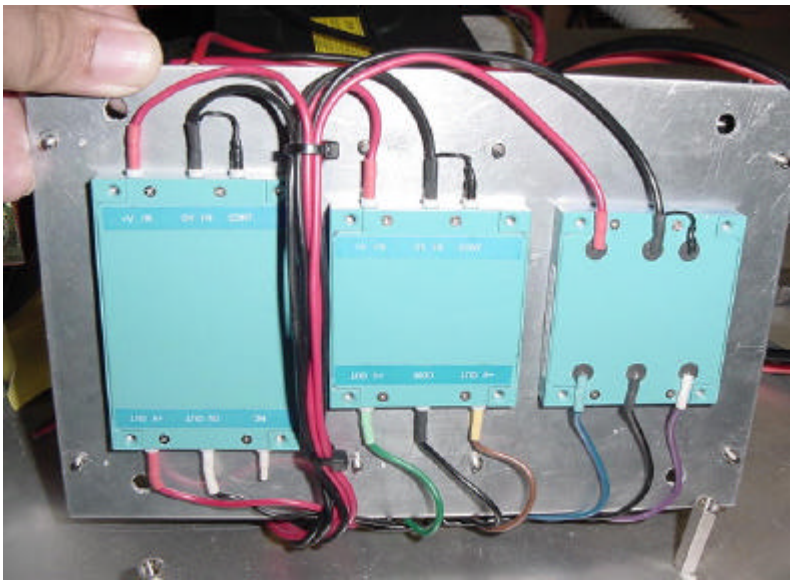


写真２０　ＤＣ－ＤＣコンバーター

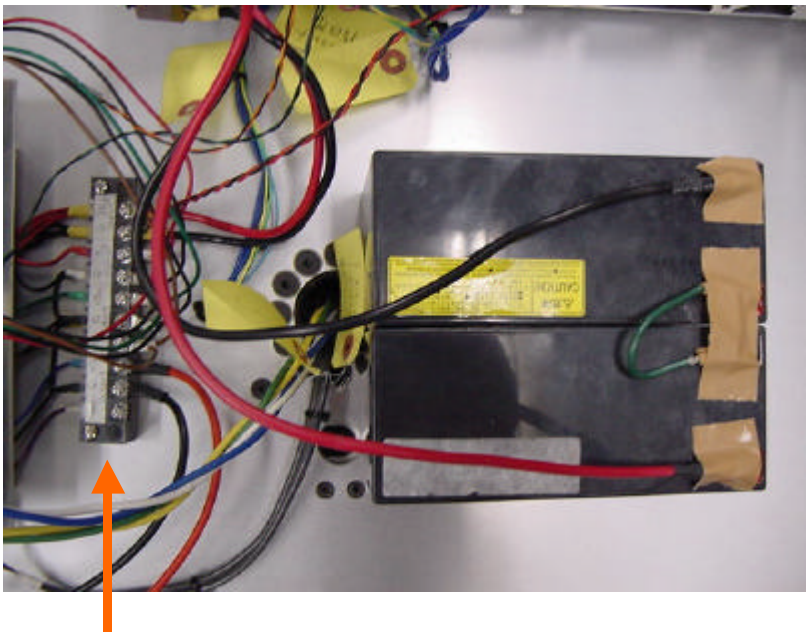


写真２１　電源分配用コネクタ

ただし，制御ボードはパソコン用の電源ボックスにつなぎＡＣ１００ｖの家庭用電源から供給している．ＣＲＴの電源も同様にＡＣ１００ｖの家庭用電源から供給している．ジョイスティック，角度センサーのポテンションメーターの電源は，制御ボードボックスに着けた電源コネクタから供給されている．

第五章　プログラム

本研究ではC言語を使用して制御プログラムを組んだ．本章ではプログラムの流れを提示し，ソースとその内容を解説する．

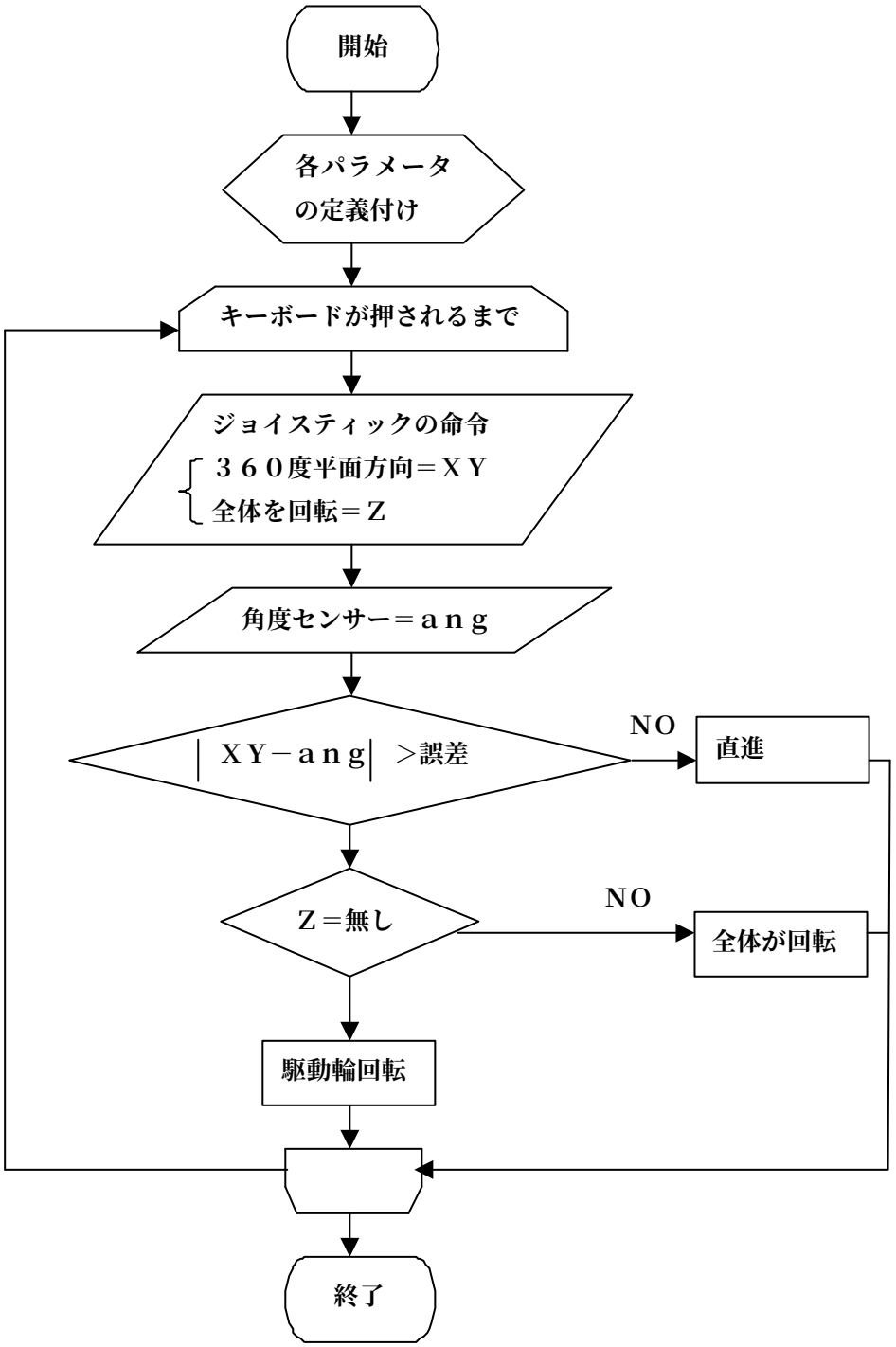
コンパイル環境　本研究制御ボード

コンパイラ　　ボーランド社　“Turbo C++ ４．２ｊ”

５．１　プログラムのながれ

本プログラムのながれを図１６に図示する．これはフローチャートではなく本プログラムによる制御の働きを大まかに理解する手助けとするものであるので，実際のプログラムに対して省略している部が多々有ることをここに前述しておく

図１６　プログラムのながれ図



５．２　プログラムソース

インクルードファイル　“N_RIF01.c”

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>
int da_data,i,volt,volt1;
unsigned int reg_data,baseadr1,baseadr2;
char erron,erroff;

int regst( unsigned baseadr)
{
reg_data=0xcf;
outpw(baseadr+4, reg_data);
erroff=(baseadr+4);
return(erroff);
}

int regst_outpw(unsigned baseadr ,unsigned int regst_no)
```

```
{
reg_data=regst_no;
outpw( baseadr, reg_data);
return 0;
}
double da_outpw(unsigned baseadr ,int port_no, double value )
{
if( (port_no >= 0)&&(port_no <= 15) ){
  da_data = (value+10.0) / 20.0 * 4096.0;
  if (da_data <     0) da_data =     0;
  if (da_data > 4095) da_data = 4095;
  outpw(baseadr+2, 0xf0+port_no);
  outpw(baseadr, (unsigned)da_data);
  erroff=(da_data);
  return(erroff);
}
else{
  erron=1;
  return(erron);
}
}
```

```
double ad_inpw( unsigned int baseadr ,unsigned int port_no )
{
double        volt;
int           ad_read_data;
unsigned int ad_write_data;
ad_write_data = port_no*8+5;
   if( (port_no >= 0)&&(port_no <= 15) ){
   outpw(baseadr+2, 0x70+port_no);      /*0..7:AD1, 8..F:AD2 */
  outpw(baseadr , port_no*8+5);
                                                /* SWCONV on, 2'comp. format */
 for(i=0;i<10; i++) ad_write_data=i;              /* wait */
 ad_read_data = (signed)inpw( baseadr );
   if ( ad_read_data & 0x0800 ) ad_read_data |= 0xf000;
 volt = (double)ad_read_data // 2048 * 10;
   return(volt);
   }
   else{
 erron=(1);
 return(erron);
}
}
```

Cファイル　“Nomura.c”

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <conio.h>
#include "c:\N_works\N_rif01.c"

#define base_adr 0x1a0//RIF01ボード　ベースアドレス

void     control_body(void);
double   xy_angle(int *);
double   z_angle(void);
double   post_s(void);
void     post_control(double, double, double);
void     motion_control(double, double, int);

/*各固定値の定義*/

double  X0, Y0; //ジョイスティック(x,y)零点値
static double delta_x=0.7, delta_y=0.7;
                    //ロバスト方向を実現するためのしきい値
static int K=30; //駆動輪部角度検出の修正回数
static double delta_alph=5.0;//delta_alph=駆動輪部姿勢誤差
static double U_alph=2.5;//U_alph=駆動輪回転時の電圧
static double P_alph=0.5;//P_alph=本体回転時の電圧
static double U_motion=1.0;//U_motion=駆動輪直進時の電圧
```

```
static double P_fix=5.0;//P_fix=駆動輪部姿勢固定の電圧
static int joy_x=0,joy_y=1,joy_z1=2,joy_z2=3; //AD　入力 ch
static int post=4;
static int motor1=0,motor2=1;//DA　出力 ch
static int fix=2;

void main(void)
{
printf("\n");
regst(base_adr);          //レジスタの設定
X0=ad_inpw(base_adr,joy_x); //零点測定
Y0=ad_inpw(base_adr,joy_y);
control_body();
da_outpw(base_adr,motor1,0.0);//終了プロセス
da_outpw(base_adr,motor2,0.0);
da_outpw(base_adr,fix,0.0);
exit(0);
}

void control_body(void)      //方向目標値の測定
{
int       i;
int       p=0;
double   xy_R,   z_R;
double   alph=0.0;

for(;!kbhit();){
xy_R = xy_angle(&p);
```

```
z_R  = z_angle();
alph = post_s();

if(xy_R!=1000 || z_R!=1000){
if(fabs(xy_R - alph) > delta_alph){
post_control(xy_R, alph, z_R);
}
else{
motion_control(xy_R, alph, p);
}
}
else{
da_outpw(base_adr,motor1,0.0);//各部停止
da_outpw(base_adr,motor2,0.0);
da_outpw(base_adr,fix,P_fix);
}

printf("xy_R=%f p=%d z_R=%f alph=%f \r",xy_R,p,z_R,alph);
}
}

void post_control(double xy_R, double alph, double z_R)//回転
{
if(z_R == 1000){
da_outpw(base_adr,fix,0.0);
if(xy_R >= alph){
da_outpw(base_adr,motor1,+U_alph);
da_outpw(base_adr,motor2,+U_alph);
```

```
}
else{
da_outpw(base_adr,motor1,-U_alph);
da_outpw(base_adr,motor2,-U_alph);
}
}
else{
da_outpw(base_adr,fix,P_fix);
if(z_R > 0){
da_outpw(base_adr,motor1,-P_alph);
da_outpw(base_adr,motor2,-P_alph);
}
else{
da_outpw(base_adr,motor1,+P_alph);
da_outpw(base_adr,motor2,+P_alph);
}
}
}

void motion_control(double xy_R, double alph, int p)  //直進
{
da_outpw(base_adr,fix,P_fix);
if(p==12){
da_outpw(base_adr,motor1,-U_motion);
da_outpw(base_adr,motor2,+U_motion);
}
else{
da_outpw(base_adr,motor1,+U_motion);
```

```
da_outpw(base_adr,motor2,-U_motion);
}
}

double xy_angle(int *p)          //(x y) 平面の全方向目標値
(0-360)を測定する
{
int       stop=1000;
double   ang,x,y;
double   r,r1,r2;

x=X0 - ad_inpw(base_adr,joy_x);
y=ad_inpw(base_adr,joy_y) - Y0;

if(fabs(x)<=delta_x && fabs(y)<=delta_y) return(stop);

ang = (180.0/3.1415926)*atan2(y, x);
if(ang<=-22.5) ang=ang + 360.0;
for(i=0;i<=7;i++){
r = 45*i;
r1= r - 22.5;
r2= r + 22.5;
if(ang>=r1 && ang<r2) ang=r;
}
if(ang>=180){
*p=34;
ang=ang - 180.0;
}
```

```
else{
*p=12;
}

return(ang);
}

double z_angle()
                //z 軸回転（本体と駆動輪部を固定して回転）を指定
                                //順時計回り:ad_ch → joy_z1
                                //逆時計回り:ad_ch → joy_z2
{
int       stop=1000;
doubleb ang,z1,z2;

z1=ad_inpw(base_adr,joy_z1);
z2=ad_inpw(base_adr,joy_z2);

if(z1>4.0 && z2<1.0){
ang = -30.0;
}
else if(z1<4.0 && z2>1.0){
ang = 30.0;
}
else{
return(stop);
}
return(ang);
```

```
}
double post_s()                    //駆動輪部の角度検出
//alph=a*Vo^2+b*Vo+c
//(180 度→2.71v 90 度→4.56v 0 度→7.37v)の二次式による近
//Vo＝測定した電圧値
{
int i;
double a=3.566561995,b=-74.57755435,c=355.9119844;
double alph,Vo;
for(i=0;i<K;i++){
Vo=ad_inpw(base_adr,post);
alph = alph + (a*Vo*Vo + b*Vo +c);
}
alph = alph/K;

return(alph);

}
```

５．３　プログラムの解説

Ｎ＿ＲＩＦ０１．ｃ

ＡＤ－ＤＡボードＲＩＦ－０１のドライバー．プログラムにインクルードすることで以下の関数が使える．

double da_outpw(unsigned baseadr ,int port_no, double value)

ベースアドレス，ポートナンバー，電圧を指定してＤＡポートから信号を出力．出力電圧範囲は±１０ｖである．

double da_outpw(unsigned baseadr ,int port_no, double value)

ベースアドレス，ポートナンバーを指定してＡＤポートからの信号を入力．入力電圧範囲は±１０ｖである．

ｎｏｍｕｒａ．ｃ

本プログラム

void　control_body(void)

本章で提示したプログラムの流れを組む関数.

double xy_angle(int *);

第四章ジョイスティックの節で提示した，ＸＹ平面方向への信号に対する角度の計算をする関数．指令がなければ１０００を返す．角度が１８０度以内ならＰ＝１２として角度を返し，１８０度を超えるとＰ＝３２として－１８０した角度を返す.

double z_angle(void);

第四章ジョイスティックの節で提示した,その場での右回転,左回転の信号を計算する関数．指令がなければ１０００を返す.

double post_s(void);

第四章角度センサーの節で提示した，台車に対する駆動輪の角度を計算する関数.

void　post_control(double, double, double);

ドライバーに指令を送り車体もしくは駆動輪を回転させる関数.関数 z_angle(void)の指令がある場合ブレーキを締めて回転．なければブレーキを解放する事で駆動輪部のみが回転する.

void　motion_control(double, double, int);

ドライバーに指令を送りブレーキを締めた状態で直進させる関数．関数 xy_angle(int *)のＰ＝１２の時正方向に直進し，Ｐ＝３２の時逆方向に直進.

第六章　まとめ

本文では，全方向移動車の一つの形を考察し，ジョイスティックによって制御する事を前提とした制御システムの作製をして，その構成をまとめてきた．本章では作製した全方向移動台車についての実験による研究成果の考察と，今後の課題などを述べる．

６．１　研究成果と考察

結果から述べると，本制御システムによって第二章で定義した動きを実現することには成功した．しかし，その実現性は高いとは言い難いものであった．

その問題の一つめとして長距離の直進が難しいという点が上げられる．本研究ではモーターの回転特性をドライバーによって調整ができるようになっているが，あくまで人の感覚によって調整されたものなので，その回転特性を同期させるのが難しく，結果として直進せずに緩やかにカーブを描くよ

うな移動をとった．この対処法として，調整によるモーターの同期には限界があると考え，外部の速度計などでフィードバックする事が考えられる．

二つめの問題として予想以上に積載用が低いという点である．何も乗せていない状態では，円滑に直進，回転といった行動がとれたのだが，５０ｋｇの荷物を載せたところ，その場で回転する動作をとらしたとき，回転軸のブレーキが滑って回転する事ができなかった．１００ｋｇの荷物を載せた場合は床面を駆動輪が滑り出し，直進行動すらとることができなかった．この問題はブレーキの力が足りないのと，駆動輪の床に対する摩擦が弱いことが原因だと考えられ，５０ｋｇ以上の積載量を求めるので有ればメカニカル部の部品に対する再設計が必要であると考える．

６．２　今後の展開

今後の展開としてはまず前節であげたような欠点を取り除くこと．次にさらなる研究課題として各種のセンサーを搭載し，自動化を図る．そしてこの全方向移動車の応用による製品の開発をすることを目標に挙げる．

参考文献

日向俊二,“すぐに役立つ！C／C＋＋プログラミング辞典”
翔泳社（2000）

謝辞

本文は，筆者の高知工科大学知能機械システム工学科卒業研究の内容をまとめたものです．

本研究を行うのに当たって，何の知識も持ち合わせていない筆者を２年間にわたって親身となってご指導をしていただき，ものを作る楽しさを教えていただいた高知工科大学知能機械システム工学科王碩玉助教授に深く御礼申し上げます．

また，日頃から研究において励まし，アドバイスをくださった同大学王碩玉知能ロボティクス研究室の種植健二氏，溝渕宣誠氏，二木亮輔氏，菅野正人氏，隅田由紀氏に感謝します．

そして，自身が忙しいにも関わらず，筆者のわがままな行動に時間を割いていただいた同大学４回生の浜口和洋氏，飯田一生氏に深く感謝いたします．

最後に，筆者の研究に対して理解を示し，支援してくれた両親に深く感謝をいたします．

２００１年２月