屋外での乗馬療法用馬ロボットのための
４足歩行機構の研究

氏名：　松浦　純
学籍番号：　１０２０１５１
知能機械システム工学科
知能ロボティクス研究室

目次

第１章　　序章

本論文は，健康増進を目的とする野外用乗馬ロボットについての研究をまとめたものである．そのため本章では，まず，健康増進について，次に，生馬，馬ロボットにおける乗馬医療について，最後に本研究の意義について説明する．

1.1　乗馬療法による健康増進

1.1.1　健康増進とは

健康の定義については，1946 年に WHO（世界保健機関）が,「健康とは単に病気でない，虚弱でないというのみならず，身体的，精神的そして社会的に完全に良好な状態を指す」[*注)] と提唱したことが国際的にも初めてのもの.

＊ 注）原文“Health is a state of complete physical, mental, and social well-being and not merely the absence of disease or infirmity.”

その後,「健康とは生きる目的ではなく毎日の生活の資源である」という考え方から,「主体的な健康の捉え方」が強調されるようになった．すなわち，健康を「心身ともに健やかであること」「仕事（勉強）ができること」「遊べること」など，人々がさまざまな生活上の出来事を通じて体現するものとして捉えられるようになった.

さらに，慢性疾患の増加などから,「完全に良好な健康」という概念は多くの人々にとって日常生活の中で現実的に受け入れられなった．そこで,「健康とは，その人の潜在能力を最大限に生かし，個人の望みを確認・実現しようとする状態をいい，それゆえ健康は，生きる目的ではなく，毎日の生活の資源である」という新しい健康観が生まれてきました.

健康増進については，1950 年代において，感染症の予防対策としての一次予防の中に位置づけられ，一般的抵抗力の強化や健康教育による感染機会の回避を意味していましたが，その後，米国の「Healthy People」で応用された際には，健康増進とは個人の生活習慣の改善を意味するようになりました.

そして，1980 年代に入って，個人の生活習慣の改善だけでなく，個人の生活習慣や健康等に影響している環境整備を合わせたものとして WHO のオタワ憲章として改めて提唱されました.

1.1.2　乗馬療法

1.1.2.1　乗馬療法とは

乗馬による健康増進は，乗馬療法と呼ばれている．乗馬療法の効果は，馬のリズミカルなスウィングと三次元の揺れが，人間の体と脳に一定の理学的効果をもたらし，脳幹が刺激され，筋肉の発達，血液の循環を助け，筋肉運動の整合や，姿勢，平衡感覚，移動感覚，各部の機能をも向上させ，健康全般を促進する．

また，騎乗時に受けるフィード・バックの影響は絶大である．右の手綱を引けば右に曲り，両方の手綱を引けば停止する．何もしなければ，何も起こらない．一つの行為が一つの反応をもたらす．こうした反応を経験することによって，乗り手は身体的・精神的満足感を覚える．

身体的効果の具体的な例としては以下の事柄が挙げられた．

① 健康と楽しみの獲得
乗馬を活発に行うことが一般的な健康維持（循環と呼吸）につながり，また，障害のために通常の活動が制限されている人でも楽しむことが出来る．

② 弱い筋肉の強化
サポートを受けて静止した馬の上に座ることから，馬の動きを自分でコントロールするまでの過程で弱い筋肉が強化される．

③ バランスと筋肉の協同の向上
乗馬・下馬の際に必要となる動きや，歩調と方向を変化させたりしながらバランスを維持する運動を行うことで，バランスと筋肉の協同が身につく．

④ 歩行の再教育
一定でなかったりアンバランスな歩行をする人は，骨盤を水平にし，正常な人の歩行パターンと類似している馬の歩行（正常で左右対称の歩き方）を体験することによって，自分の歩行パターンを向上させることができる．

⑤ 痙攣性の脳性麻痺の緩和
リズミカルな馬の動きと温かい馬のからだに直接触れることによって，

硬くなった筋肉がリラックスし，自由に動けるようになる．

⑥ 股関節の柔軟化
大腿筋が鍛えられ，股関節が柔軟になることで動作が機敏になり転倒が予防される．

知的障害を持った人や自尊心の低い人などへの精神的・社会的効用として次のような事柄が挙げられる．

① 社会性の向上
普段，問題行動を抱えている子供達でも，馬となら良い関係を持てることがよくあり，乗馬のレッスンなどを通してルールを守るようになる．

② 教育への応用
国が定めた適切なカリキュラムの中で，数字あわせなどを馬上で行ったりする．

③ 他者との関係性の向上
馬の世話をしたりすることで，コミュニケーションに対する自信を持つことが望める．

1.1.2.2 乗馬医療の歴史

健康増進として注目されている乗馬療法．では，だれが考え，どのように普及されてきたかなど，その歴史を述べる．

古代ギリシャ時代にはすでに障害を持った人が馬に乗るという試みがなされていた．障害者がいつから，どのようにして馬に乗るようになったのかまでははっきりしないが，ギリシャで発見された文献に「紀元前５世紀，戦争で傷ついた兵士を馬に乗せることで治療した」と記されている．馬は，人類のいかなる時代においても，単なる輸送や移動の手段としてのみならず，苦痛を緩和し障害を軽減する有効な手段として利用されている．

Riding for the Disabled＝「障害者のための乗馬」という言葉を最初に用いたのは，２０世紀初頭におけるＤアグネス・ハントとオリーブ・サンズという二人の英国人だった．ハントは１９０１年にオズウェストリー整形外科病院を創設．サンズは理学療法協会の会員で，第一次大戦で兵役に就き負傷．この経験から「戦線へ復帰することに不安を抱く負

傷兵たちも，乗馬によって自信を回復することができるのではないか」と考え，自分の馬を病院へ持ち込み兵士たちを乗せたのがきっかけだ．二人とも障害者乗馬が多くの人に希望をもたらすことを信じ，将来一般に認知されることをひたすら願っていた．彼らの信念が間違いなかったことは，今日までの多くの事例が証明している．

とりわけリズ・ハーテルの偉業は特筆に値する．彼女は両足麻痺というポリオ障害を克服したのみならず，１９５２年のヘルシンキオリンピックの馬術競技へ出場．そしてドレッサージュで，みごと銀メダルを獲得したのだ．実はこのオリンピックで，初めて男女とも同じ条件で競技を行うことになった．ハーテルにとっては，その条件でさへなお健常者と同等ではなかったが，それでも彼女はすばらしい活躍を見せて世界中から賞賛された．

ハーテルの影響は絶大だった．ノルウェーの理学療法士で乗馬経験も豊富なエルスベス・ボッスカーも，ハーデルの快挙に障害者乗馬の可能性を見出した人物だった．彼女はさっそくコペンハーゲンの理学療法士ウルラ・ハーボスと共に自分たちの患者に乗馬を勧め，治療の一手段として活用し始めた．そして間もなく，障害者乗馬が多大な効果をもたらすことを痛感した．

ボッスカーと親交のあったノラ・ジャックスは，同様の試みをイギリスにも定着させたいと考えた．彼女はまず患者を自宅の裏庭で馬に乗せることから始め，後に『障害者乗馬信託』を設立．当時の主任インストラクターだったジョン・アンソニー・デービスは，今ではこの分野の世界的権威となっている．

英国ブリストルにあるウィンフォード整形外科病院の院長ステラ・セイウェルもこの分野の第一人者だ．ここは１９４８年に障害者乗馬の施設として公式な指定を受けた最初の病院だ．ジリアン・ピーコック博士は今なお障害者乗馬に取り組んでいる．彼女はＲＤＡ（障害者乗馬協会）の活動に携わりながら，一方ではフォーチュン乗馬治療センターの顧問委員会議長を務め，医療団体と障害者乗馬の現場との橋渡し的な役割を果たしている．このように障害者乗馬は多くの国々において熱意ある人々の尽力によって発展を遂げている．

１９６４年，イギリスで障害者乗馬の先駆者たちが一堂に会し，障害者乗馬顧問委員会が結成された．これが１９６９年になってＲＤＡ(障害者乗馬協会)へと発展する．

ＲＤＡは公認慈善団体の規約に基づいて，個々のセンター単位で活動している（この点はアメリカと異なる）．ＲＤＡの支部は国内に７２７，国外にも旧英国領を中心にオーストラリア，カナダ，ニュージーランドなど世界各地で展開している．本部は英国で，総裁をアン王女が務めている．

活動の特徴として，ＲＤＡが登録ボランティア団体であり，スタッフの確保からスポンサーシップに至るまで，ほとんど独自で運営している点にある．行政や医療や学校などと対等に，横の連携によって人材や情報の交流が成り立っている．さらに注目すべきは障害者乗馬活動が医療保険の対象となっていることだ．ＲＤＡ認定のインストラクターが指導する乗馬レッスンは保険が効くのだ．それはインストラクターの質を保つ一方で，障害者

乗馬の現場への医師や理学療法士の参画をも容易にするものと推察される．

ＲＤＡの活動はボランティアとして参加するヘルパーを抜きには語れない．センターが義務づける人員を確保するには，政府の援助だけでは不十分だからだ．言い替えれば，障害者乗馬は障害者に奉仕したいという気持ちを持ったボランティアによって成り立っていると言っても過言ではない．幸いなことにボランティア活動は近年ますます活発化し，多様化している．だが一方で乗馬を始めたいという障害者の数も増加しており，ボランティアの数はまだまだ足りないというのが現情のようだ．

米国の障害者乗馬はヨーロッパとは異なる形で発展した．概略を述べれば，治療的効果に重きを置くのではなく，障害者でも取り組めるレジャーあるいはスポーツといった色彩が強い．米国における障害者乗馬はすでに１９６０年代には存在していた．当時は，自分の馬と時間を提供しようという献身的な人たちが各個に活動していた．やがてそれらが統括され組織化されるにつれて，障害者乗馬も大きな発展を遂げることになる．

もっとも有名な障害者乗馬の団体はＮＡＲＨＡ（北米障害者乗馬協会）だ．ＮＡＲＨＡは１９６６年に，馬事新聞の編集者だったアレキサンダー・マッコーエンと，英国シグウェルセンター職員のジョン・Ａ・デービスの両名によって創設された．やがて米国のさまざまな団体が活動を支援するようになり，安全性の基準が作られ，メンバー相互の情報交換の場として有効に機能するようになる．ＮＡＲＨＡの具体的な活動内容としては，優秀なインストラクターの訓練および認定，安全と運営に関する独自の基準を満たした施設の認定，治療のために必要な馬の科学的な研究などが挙げられる．さらにＮＡＲＨＡは，１９８９年の時点で４６１ヶ所を数える活動センターとその管理者および事務員に対して，包括的な保険機構に入れるようにした．この措置は，米国において頻発し得る訴訟問題を考えると，特に重要なポイントだったと言えるだろう．

アメリカの障害者乗馬の歴史を語るとき，１９６７年にモーディー・ハンター・ウォーフェルが創設したＨＨＦＴＨ（障害者の楽しい乗馬の集い）に言及しないわけにはゆかない．その目的はいたって単純で，「障害者が乗馬を楽しむこと」だ．その手法も，技術論を重視するのではなく，馬や環境全体に目を向けている．治療効果はあくまで結果であり目的ではない．この団体には何人もの有名な人物が所属している．中でもアリゾナ州ツーソン在住のデーブ・トレックスラーは，両足を失っているにもかかわらずウェスタン乗馬の名手として，全米にその名を知られている．

ＮＡＳＣＰ（米国立脳性麻痺者スポーツ協会）は，全米脳性麻痺者協会の支部として１９７６年に設立され，脳性麻痺関連の障害者がスポーツをする場を提供している．そこでは障害の種類や程度によって競技者を分け，オリンピックを規範に，馬術を含めたさまざまな競技会を開催している．各競技会では，意欲ある障害者の参加が年々増加している．

ＮＡＳＣＰが肢体不自由者を対象とするのに対して，スペシャル・オリンピックスは知的障害者のためのスポーツ運営機関だ．現在は全米３６州で約４，０００人の障害者が練習に励んでいる．訓練を終えたインストラクターは現在７５０人．加えて世界１２ヶ国か

ら競技者が加わっている.ここではドレッサージの他にも,バレルレースやリレーなど様々な競技が行なわれている.それぞれ引き馬によるものと単独騎乗によるものと二通りあり,馬の種類や外観ではなく,競技者が自分の馬をどれだけアピールできるかが重要視される.さらにチームで行う団体種目もいくつかある.

１９７６年に設立されたデルタ・ソサイエティーは，人と動物と環境の関係についての教育を行っている国際的機関だ．馬の部門は広範な事業の一部分に過ぎないが，諸会議や出版物を見るかぎり，障害者乗馬が強い関心を集めている様がうかがえる．デルタでは，治療目的の乗馬に毎年２つの賞を贈っている．一つは模範的なプログラムに対して，もう一つは優れた馬に対してだ．さらにマッカロー記念賞というものがあって，人と人，あるいは人と動物の相互理解を深めるのに寄与した者を表彰している．最近では，１９８９年にコロラド州リトルトンのマリー・ウールバートンにこの賞が贈られた．彼女はベトナム戦争の退役軍人に乗馬を奨励するという先駆的な業績を残した人物だ.

もうひとつの全米的な組織がアメリカ４H青年クラブだ．これは１９００年代初頭に農家の青年を農事的・社会的活動に奉仕させるために組織されたものだ．４H青年クラブと障害者乗馬の関りは，１９７０年にミシガン州立大学の公開講座で行ったプログラムを，ミシガン州が率先して採用したことに始まる.現在はペンシルベニア州を筆頭に各地の４H青年クラブが，積極的にそのプログラムを取り入れている．ペンシルベニア障害者乗馬会議では，季刊紙を発行しながら全米の障害者乗馬に携わる人々のための教育セミナーやインストラクターの養成を行っている.

１９６４	障害者乗馬顧問委員会結成（イギリス）
１９６９	障害者乗馬協会（ＲＤＡ）設立（イギリス）
	北米障害者乗馬協会（ＮＡＲＨＡ）設立
１９７０	乗馬療法協会設立（ドイツ）
１９７２	ベルギー乗馬療法協会設立
１９７５	香港ＲＤＡ活動開始
１９７９	オーストラリアＲＤＡ活動開始
１９８２	シンガポールＲＤＡ活動開始
１９８４	**障害者乗馬の会設立**
１９８６	**日本身体障害者乗馬連盟設立**
１９９０	アジア障害者乗馬協会結成
１９９４	**日本障害者乗馬協会（ＪＲＡＤ）発足**
１９９５	**日本乗馬療法協会（ＮＲＴＡ）設立**
	さわやかポニークラブ設立
１９９８	**ＲＤＡ　Ｊａｐａｎ設立**

	日本乗馬療法インストラクター養成学校設立
１９９９	**全日本障害者乗馬協議会（ＡＮＴＲＡ）設立**

※太字は日本の動向

世界の障害者乗馬組織年表

1.1.3 馬ロボットによる乗馬療法

生馬と馬ロボットとの大きな違いは当たり前のことだが，生馬は生き物で，馬ロボットは物であるということだ．馬は元来おだやかで人なつっこく，しかも従順．正しく接すれば，人間を拒否したり攻撃したりはしない．反対に警戒心を持つと，思うように動いてくれない．これが大事な点である．生馬での乗馬医療として，知的障害者に対しての特殊教育がある．知的障害者は大型動物である馬に乗ったり，馬に話しかけたり，毛をとかしたりすることにより馬とコミュニケートが出来た事で自信がつき，家族以外の人と話さなかった子が，馬と一緒に歩いてくれたボランティアの人に話しかけるようになったりすることがある．では馬ロボットではどうかというと，こういった効果を出すためには現在，市販されているペットロボット以上の性能が必要であると思われるため，非常に難しいだろう．

1.1.3.1 室内用乗馬ロボット

上記したように馬ロボットでは生馬による乗馬療法での効果をだすのは困難であると思われる点がある．では，なぜ馬ロボットでの乗馬療法を研究するのか．

実際に生馬での乗馬療法を行うためには，いろいろな条件，制約がある．まず，馬を生育，維持するためのコストと場所が必要になってくる．次に，馬を調教し，乗馬の訓練を教える専用インストラクターが必要である．

これらの条件を克服し，生馬の乗馬療法にない効果を生み出したのが，木村哲彦教授によって提案され，松下電工が開発した室内用乗馬ロボット（図１）を利用して，王ら研究グループによりその有効性を示していている．実際に実験を行い多くの被験者が腹筋，背筋，歩幅などの健康増進につながった．

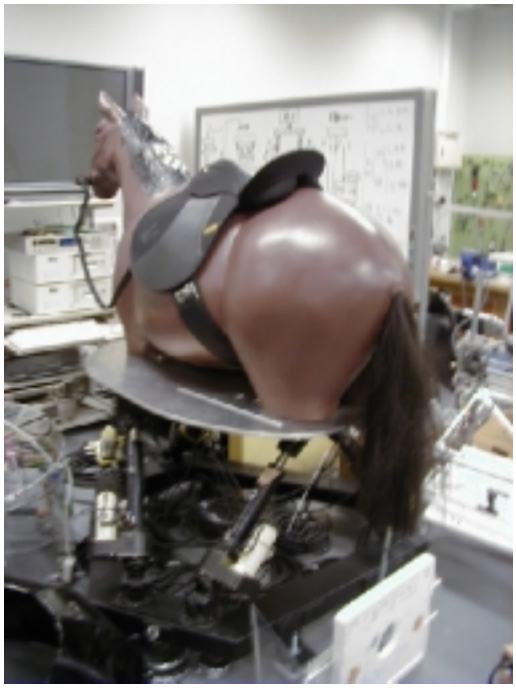


図１，室内用乗馬ロボット

これにより室内での乗馬療法が可能になり，馬を生育，維持できない狭い場所での医療が受けられるようになった．また，機械制御によって繰り返し同じ負荷を与えることができ，一人一人の健康状態，身体状況に応じた最適負荷を与えられる．また，患者のデータを蓄積し，医療解析することも可能であり，健康増進の実効果を定量的に評価できる．

1.1.3.2 屋外用４足歩行乗馬ロボット

室内での乗馬医療は固定された乗馬ロボットであるため, 本来の生馬による前に進む速度効果が失われ，騎乗時に受けるフィード・バック効果も失われている問題がある．なにより屋外の青空の下で馬に乗って走るという楽しさが室内では無い．実際，室内用乗馬ロボットでの被験者の多くから乗馬療法を野外でも行いたいと言う希望が出てきた．

1.2 研究の意義

上記したように馬ロボットによる乗馬療法は生馬での乗馬療法における様々な制限が克服できると思われ，そして乗馬療法は屋外で行うことで室内では出せない効果を期待できる．

よって本研究では屋外での馬ロボットによる乗馬療法ができるよう，馬の４足歩行機構について研究する．

第２章　　実験

2.1　　実験手法

実際に人が乗れるような馬ロボットを作成するまえに小さい実験用の馬ロボットを作る．アルミ版で胴体，足を作成し，足関節の動力にサーボモーターを使用する．ＰＩＣ（マイコン）でそのサーボを制御し，馬ロボットを歩かせる．

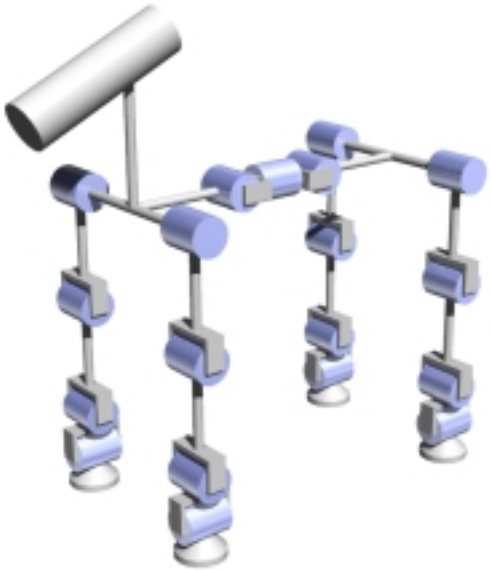


図 2-1

図 2-1 は当初のイメージ図である．この図では背骨に３自由度あり，各足に４つの自由度が設けられているが，まずは各足２つの自由度を持つ馬ロボットを製作してみる．

2.2　　メカシステム

2.2.1　　回路図

図 2-2 は本実験の回路図である．回路図ではサーボは１３個だがこのうち８個を使用して

実験する.

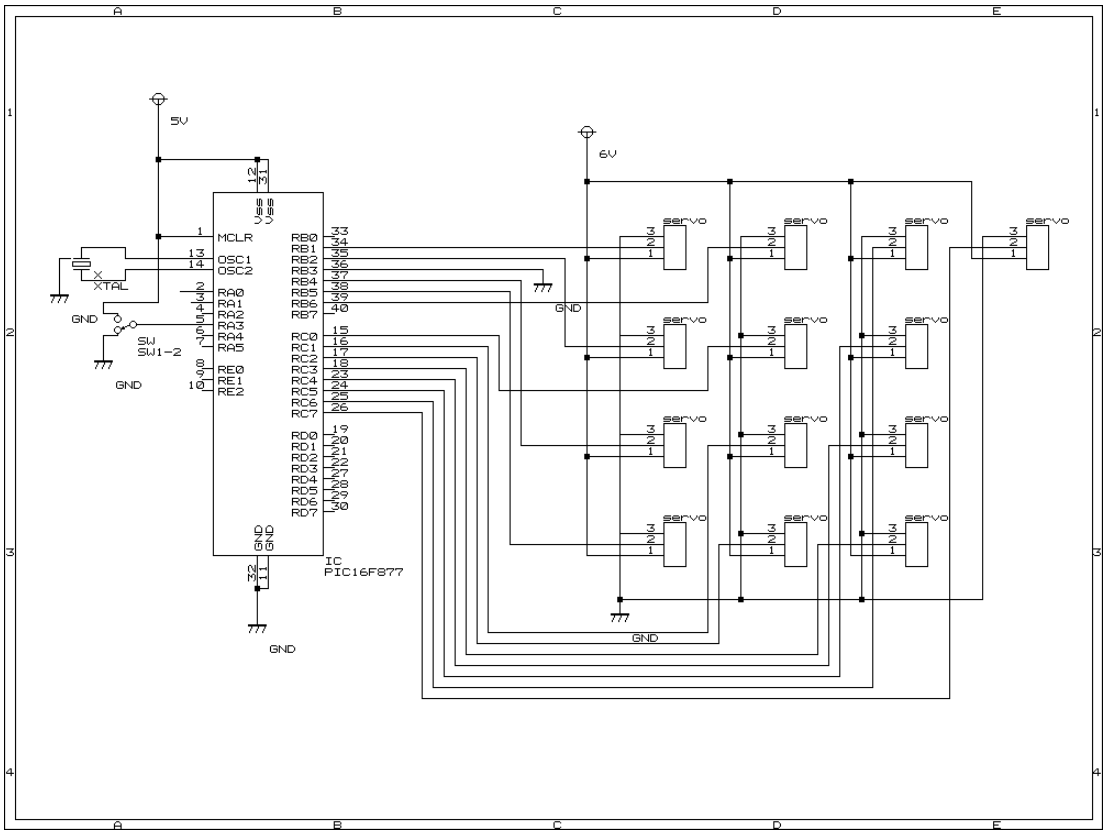


図 2-2

2.3.2 馬ロボットの外観

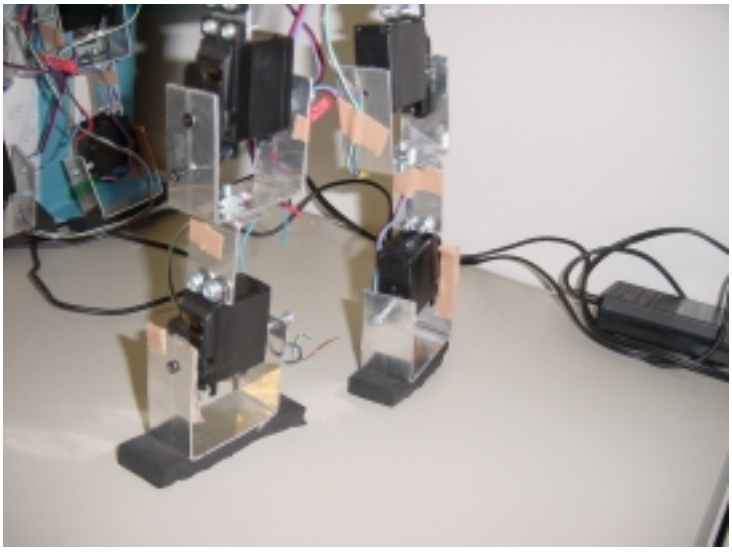


図 2-3

馬ロボットの関節は図 2-3 のようにサーボをアルミ板でつなげている．サーボの回転角度がそのまま関節の角度になる.

馬ロボット（サーボ 8 個）

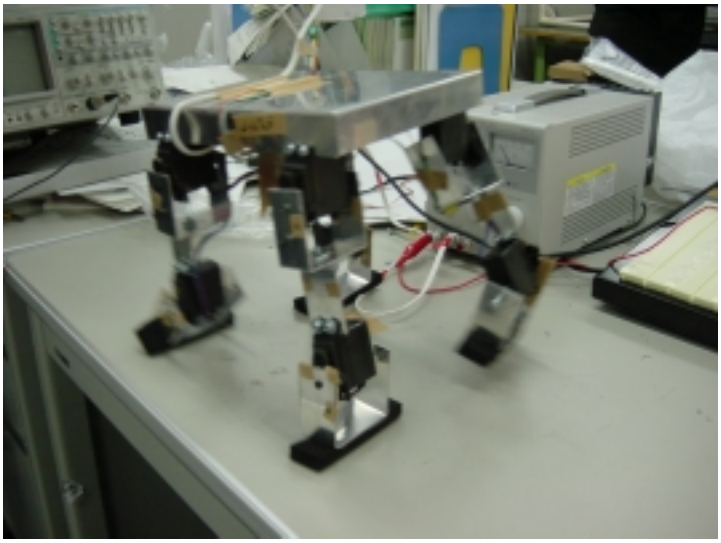


図 2-4

1 台目の馬ロボットはサーボが各脚に 2 つ，計 8 個になっている．胴体の部品はアルミ板を使用している．高さは２１ｃｍ，足の長さは１９ｃｍ，幅１５ｃｍ，胴の長さ，２０ｃｍである．

馬ロボット（サーボ 13 個）

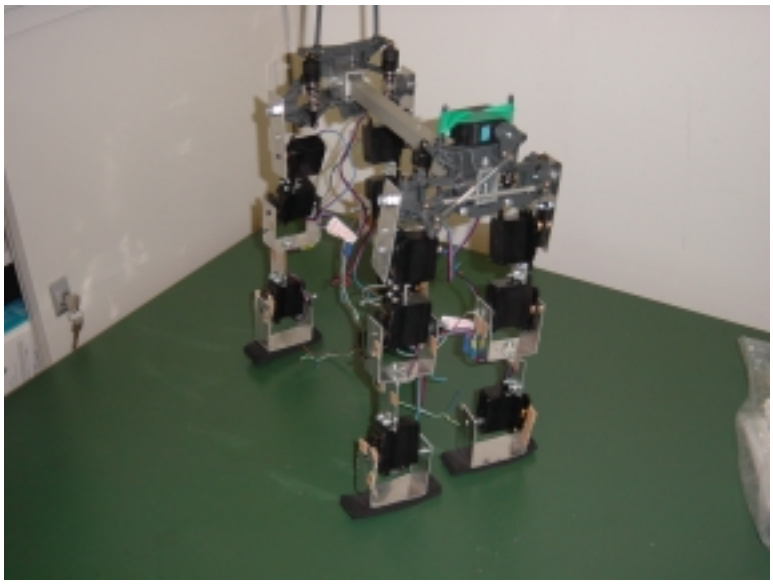


図 2-5

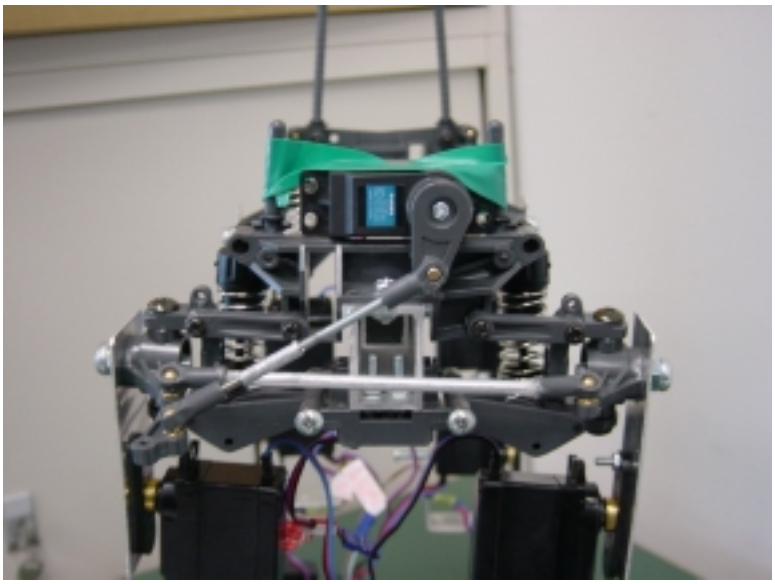


図 2-6

2 台目の馬ロボットは各脚に３つのサーボと，前脚を床平面に垂直な軸を回転させるための（車でいうステアリング）サーボが１つ（図 2-14），計１３個のサーボを使用している．そして，ショックを吸収するため，サスペンションを付けている．

このモデルは脚部は上記した１台目の馬ロボットの脚にアルミ板でサーボをもう１つずつ追加したものである．そしてボディ部はラジコンカーのサスペンションとステアリングの機構を使っている．

脚の関節が生馬と同じ３つになったことでより生馬に近い動きが期待できる．

2.3 使用システム

2.3.1 ハードウェア

2.3.1.1 サーボモーター

サーボモーターとはパルス信号を送ることによって回転軸の角度を決めることのできるモーターである．

サーボの内部では駆動軸にポテンショメータ（可変抵抗）が取り付けられており，ポテンショメータの角度に対応した幅のパルスが発生するようになっている．

外部から入力されたパルス幅とサーボ内部のパルス幅と異なっている場合，サーボ内部のモーターが動作し同じ二つのパルス幅が同じくなるまで駆動軸が回転する．このような方式にすることでサーボの駆動軸角度が外力によって変化した場合でも，サーボの駆動力の範囲内で元の角度に戻ろうとする力を発生させることができる．本実験では私の調べたラジコン用サーボの中で最もトルクの大きかった「Hyper ERG-VB」（サンワ）を使用す

る(図 2-2，表 2-1 参照)．ラジコン用サーボモーターは，DC モータ，減速機構（ギヤボックス)，サーボアンプ（モーターを駆動するための回路）が一体になっているため，取扱い易いアクチュエータである．またラジコン用サーボはラジコンホビーの分野で大量に利用されている為に価格が安く，ラジコンショップや模型店で購入できる．つまり，ロボット等を製作する際に大変便利な部品と言える．最近ではラジコン用サーボを使用したロボットの製作記事も多く見掛ける．

電源： RC サーボの電源はおおよそ+ 4.8 ～ 6 V の範囲のものが多く，この範囲内で供給する必要があるが，特に安定化する必要は無いと思われる．但し，サーボ内のモーターを動かすのに十分な電流を供給できないとサーボの仕様上の最大トルクが出せないことがあるので，ある程度電源容量に余裕をもたせておく必要がある（高トルクのサーボ程電流を必要とするので注意)．なお仕様電源電圧の範囲内であれば，電圧が高い方がトルクが高くなりレスポンスも良くなる．

制御信号： 制御信号のパルスの周期は 10～20[msec]が一般的で，各主要メーカーはほぼ同じである．この周期はサーボ内のモーターの駆動周期を決めるもので，多少周期が違っていても直接サーボの駆動軸角度には影響しないが，周期が短いほどトルクは高くなりレスポンスも良くなる．但し，あまり短い周期のパルスを入力すると RC サーボ内部の回路が信号に追従できなくなるので，限界がある．

Hyper ERG-VB

図 2-7

速度	0.10sec/60° （6V）
トルク	13kg ・cm（6V）
寸法	39.0×20.0× 37.4mm
重量	60g
配線 （Ｚコネクター）	赤：電源（４．８～６Ｖ） 黒：ＧＮＤ 青：制御信号
パルス周期	１０～２０ｍｓ

パルス幅	１．５ｍｓ（０°） ±０．５ｍｓ（±６０°）

表 2-1

2.3.1.2　ＰＩＣ

PIC（Peripheral Interface Controller）はその名前の由来どおり，コンピュータの周辺に接続される周辺機器との接続部分を制御するために開発された「マイクロコントローラ」と呼ばれる領域の IC である．つまり，それほど高機能，高速性は必要としないが，周辺機器を制御するのに便利な機能は内蔵しているといった，使用目的が比較的明確な範囲に限られているマイクロコンピュータの１種である．

PIC には，図 2-3 の写真のように目的に合わせて数多くの種類がある．

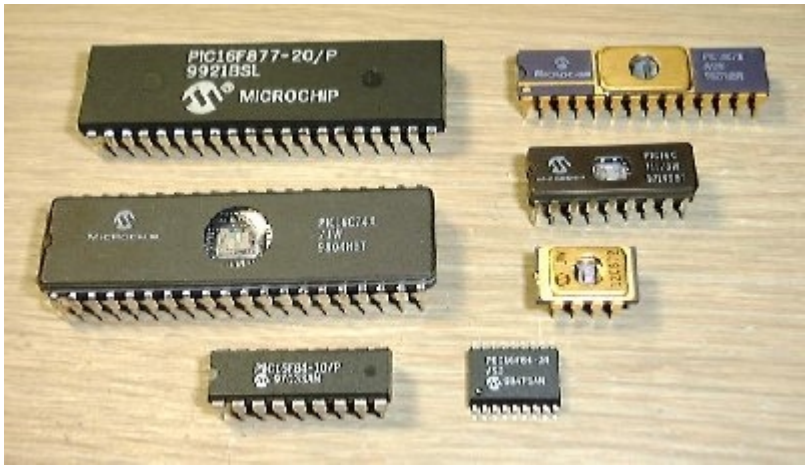


図 2-8

わずか８ピンの小さな IC の中に，マイクロコンピュータとしての機能が一通り納まっているものから，数多くの周辺機能を内蔵し，パッケージも 40 ピンや 64 ピンもある大型のタイプまで用意されている．さらにこれらのアーキテクチャが共通となっているため，大きな PIC は，小さな PIC の上位互換となっていて，同じプログラムで動かすことが出来る．

本実験では比較的安価で，高機能である「ＰＩＣ１６Ｆ８７７」を使用する(表 2-3，図 2-5 参照).

ＰＩＣ１６Ｆ８７＊シリーズの特徴

１．　ログラムメモリをフラッシュメモリ化

何回でもすぐ上書きで書き直しが出来，しかも価格も紫外線消去タイプの１／３近くの価格である．しかも，このプログラムメモリはプログラムで自分自身を書き換えることが出来るので，遠方からプログラムのバージョンアップデータのダウンロード

などが可能となります.

２．５V の低電圧プログラミングが可能

従来の１３V による書き込み方法に加え，コンフィギュレーション設定により，５V でプログラミングが可能となった.

但し，5V モードを選択すると，ポートの RB3 ピンが汎用入出力ピンとしては使えなくなり，プログラミング用の専用ピンとなってしまう.

３．A/D 変換の分解能が１０ビットにアップ

Ａ／Ｄ変換とは，Analog/Digital Conversion　つまりアナログ信号からディジタル信号への変換を意味する. 従来８ビット分解能であった A/D 変換が１０ビットにアップしたことで，１０２４段階の計測が可能になって，応用範囲が広くなった.

さらに Reference も，最高電圧を決める Vref+　に加え，最低電圧を決める Vref-が追加されたので，０V 以上のオフセットが可能となった.

４．ブラウンアウトリセット機能追加

電源電圧監視により電圧低下でリセットがかけられるようになった.

ＰＩＣ１６Ｆ８７＊シリーズの注意点

１．　コンフィギュレーションの低電圧プログラミングを OFF にする

従来と同じプログラミング方法で進める時には，コンフィギュレーションビットの中の LVP ビットを「0」に設定する．そうしないとポートのＲＢ３が入出力ピンとしてつかえなくなる.

ちなみに私はこれを知ったのがプログラムを書いた後なので,
設定しておらず，ＲＢ３は入出力ピンとして使用していない.

２．　A/D 変換が１０ビットになって A/D 変換の待ち時間が長くなった

チャンネルして後，A/D 変換を開始するまでのアクイジションタイムと，A/D 変換そのものに必要な変換時間いずれも長くなった. 表 2-2 は８ビット時と１０ビット時の A/D 変換の待ち時間である.

	８ビット	１０ビット
アクイジションタイム①	１２μsec	20μsec
A/D 変換時間②	15.2μsec	19.2μsec
①＋②	27.2μsec	39.2μsec

（注　クロックは 20MHz で最高速度の A/D 変換の時）

表 2-2

ＰＩＣ１６Ｆ８７７

開発元	米国マイクロチップテクノロジ社（Microchip Technology Co.）
ピン数	４０個
動作速度	２０MHｚ　クロック入力 ２００μｓ　命令サイクル
動作電圧範囲	２．０～５．５Ｖ

表 2-3

ピン配置

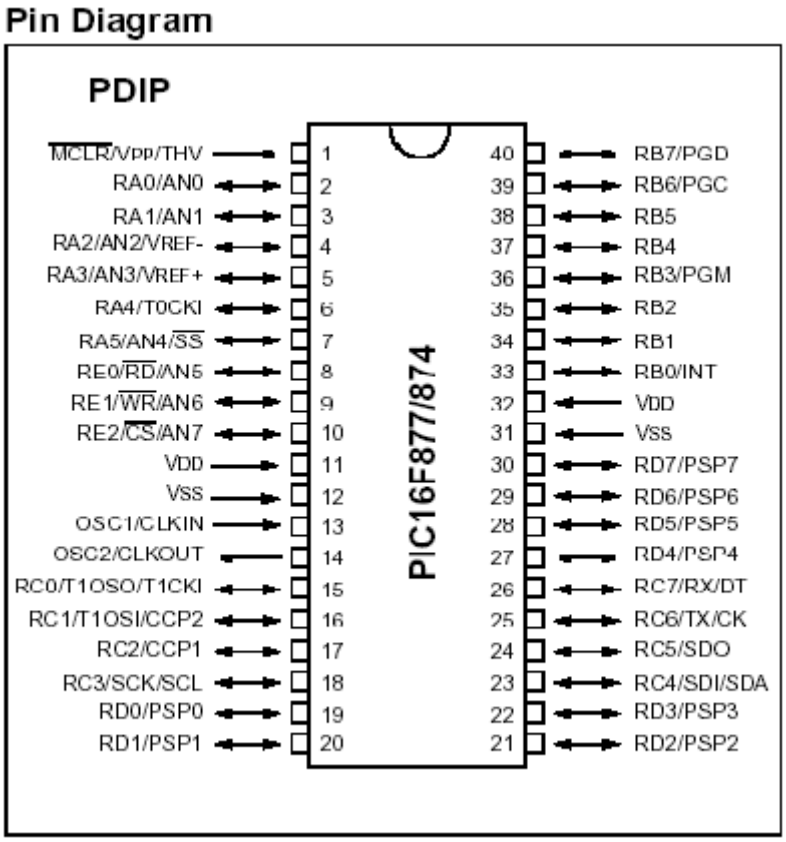


図 2-9

RA0-5 : 入出力ポートＡ

RB0-7 : 入出力ポートＢ

RC0-7 : 入出力ポートＣ

RD0-7 : 入出力ポートＤ

RE0-2 : 入出力ポートＥ

AN0-4 : アナログ入力

RX : USART 非同期受信ポート

TX : USART 非同期送信ポート

SCK : 同期シリアル入力

SCL : /SPI, I²C ﾓｰﾄﾞ出力

DT : 同期データ

CK : 同期クロック

SDO : SPI データ出力（SPI ﾓｰﾄﾞ）

SDI : SPI データ入力（SPI ﾓｰﾄﾞ）

SDA : データ入出力（I²C ﾓｰﾄﾞ）

CCP1,2 : ｷｬﾌﾟﾁｬ入力/比較出力/PWM 出力
OSC1/CLKIN : クロック入力
OSC2/CLKOUT : クロック出力
MCLR : ﾏｽﾀｸﾘｱ (Ｌﾚﾍﾞﾙでリセット)
Vpp : プログラム書き込み制御
THV : ﾃｽﾄﾓｰﾄﾞ制御
VREF+/- : 基準電圧
SS : ｽﾚｰﾌﾞ選択
T0CKI : タイマー0 ｸﾛｯｸ入力
T1OSO : タイマー1 発振器出力
T1OSI : タイマー1 発振器入力
T1CKI : タイマー1 ｸﾛｯｸ入力
PGD : ﾌﾟﾛｸﾞﾗﾐﾝｸﾞﾃﾞｰﾀ入力
PGC : ﾌﾟﾛｸﾞﾗﾐﾝｸﾞｸﾛｯｸ入力
PGM : プログラミング低電圧入力
INT : 外部割り込み入力
VDD : 電源
Vss : 接地

2.3.1.3 ＰＩＣライター

ＰＩＣライターとは作成したプログラムを PIC のメモリに書き込むために使用するもの．プログラムの内容を PIC に書くために，通常はパソコンとライターおよびライターソフトウェアを使う（図 2-6）．ライターソフトについては後述する．

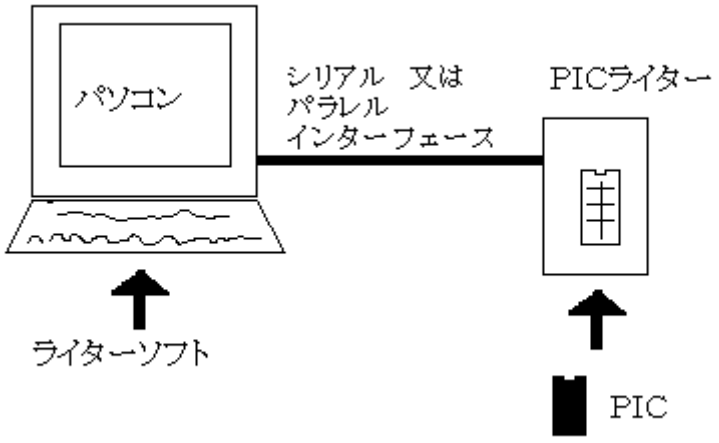


図 2-10

本実験では「ＡＫＩ－ＰＩＣプログラマキット」（秋月電子通商）を使用した.

2.3.1.4 オシロスコープ

ＰＩＣから出ているパルスが異常な場合いきなりサーボをつなぐとサーボが破損する恐れがある．サーボをつなぐ前にオシロスコープでパルスが正常なものかを確かめなければならない.

測定方法：まずオシロスコープの電源を入れる．すると図 2-7 のような画面が出てくる.

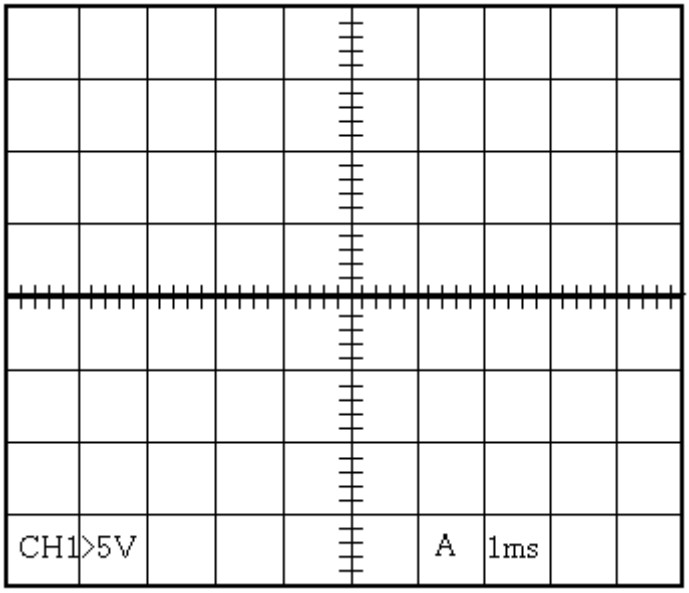


図 2-11

図の太い横線と文字は光っている．この線や文字がぼやけていたり，暗かったりしているときは，FOCUS ダイヤルと
INTENSITY ダイヤルを使って調整する．光が強すぎると焼きつきを起こす可能性があるので弱めに設定する．図の左下の「CH1」は測定端子をつないである方の CH を VERT MODE ダイヤルで選ぶ.「５V」は縦の１マス分が５V を表している.
この値は VARIABLE VOLTS/DIV ダイヤルで設定できる.
右下の「１ｍｓ」は横の１マス分が１msec を表している.
この値は SWEEP TIME/DIV ダイヤルで設定できる．この状態でパルスを測ってみて図 2-8 のように表示されたとする.

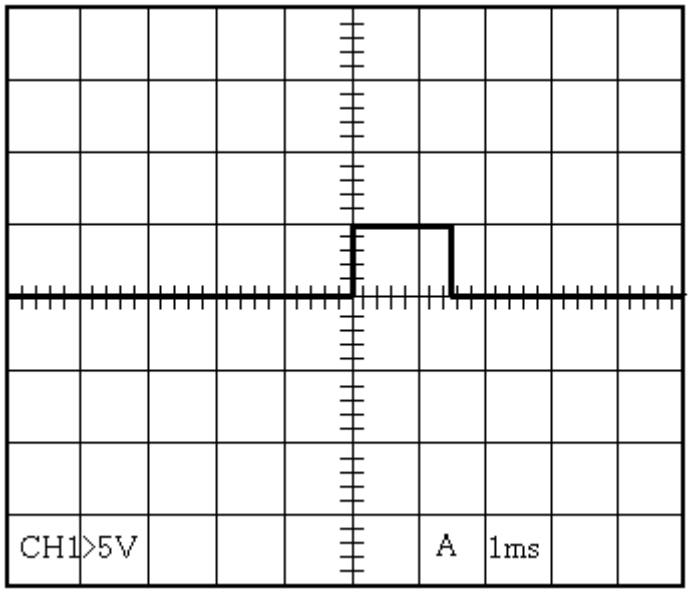


図 2-12

この図では５V の電圧が約１．５msec 出ていることがわかる．しかしこの表示ではパルスの周期がわからないので SWEEP TIME/DIV　ダイヤルを回し「１ｍｓ」を「５ｍｓ」にする．すると図 2-9 のような表示になる．

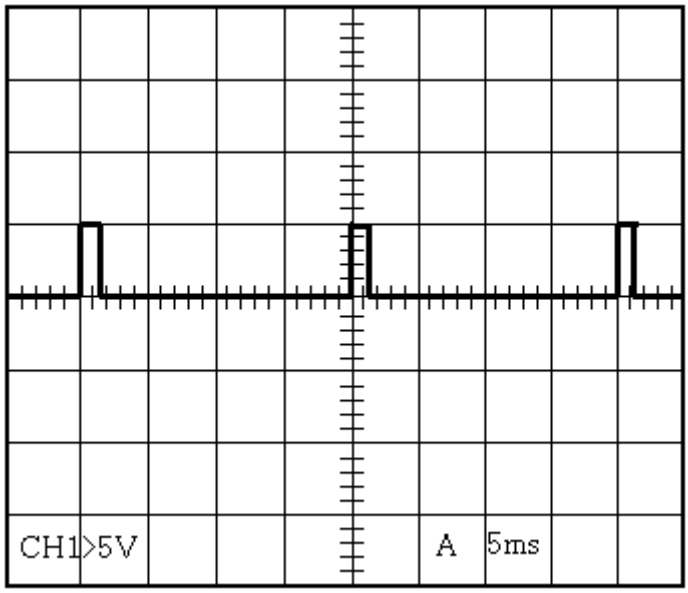


図 2-13

この図では４マス，つまり２０msec に一回パルスの出力があるのでこのパルスの周期は２０msec であることがわかった．

以上の測定でパルスが５V の出力でパルス幅が１．５msec，周期が２０msec であることがわかり，サーボの稼動範囲内であることが確認できたので，安心してサーボにパルスを送ることができる．

2.3.2　ソフトウェア

2.3.2.1　コンパイラ

プログラムをＰＩＣに書きこむためには，なんらかの言語で書かれたプログラムをコンパイルしてＨＥＸファイルを作らなければならない．本実験では「ＰＩＣ　Ｃ　コンパイラ」（ＣＣＳ社）を使用した．

ＰＩＣ　Ｃ　コンパイラの特徴

１．MPLAB との統合環境が構築できる

ｎソフトをインストールしたあと，簡単に MPLAB と統合できる．統合してしまうと MPLAB 環境が主となりコンパイラは表からは明確には見えなくなる．またデバッグが MPLAB 上でできるようになるので，非常に扱い易くなる．

しかし，この時のデバッグはアセンブラベースとなるので，コンパイラが生成するリストを使ってデバッグする．

２．豊富な組込み関数が用意されている

RS232 シリアル通信，A/D 変換，I2C，入出力ピン制御，CCP 制御など PIC の周辺機能を直接制御できる多くの組込み関数が用意されており，プログラミングが非常に楽にできる．

本実験で使用した組込み関数の詳細を後述する．

３．　ビットデータ定義も可能

データとして、１ビット、８ビット、１６ビットが扱える他に、３２ビットの浮動小数点数も扱うことが出来る。

４．　アセンブラ記述を挿入できる

C 言語のソースプログラムにインラインでアセンブル記述を挿入することが出来る．また，C 言語との変数渡しもサポートされており，自由に行き来ができる．

５．　標準入出力デバイスが用意されている

どの PIC にでもピンを自由に指定できる RS232 シリアル通信の標準入出力デバイスが使えるため，printf 関数で簡単にパソコンとの通信が実現できる．

６．　豊富な外部周辺デバイスのサポート

PIC の外部周辺デバイスとして接続される多くの IC や機器を制御するためのデバイスドライバ関数が提供されている．

液晶表示器，シリアル接続メモリ，キーパッドなどがある．

７． 変数エリアの有効利用

定数はプログラムメモリに格納し，一時変数は同じ変数エリアを共有するなど，少ない変数エリアを有効活用できる．

８． 割り込みサポート

割り込み処理に必要なレジスタ待避復旧や，フラグのリセットなど必要な処理は自動的に作り込まれるため，割り込み処理作成が非常に楽にできる．

本実験で使用した組込み関数

INPUT(pin)

解説：ＣＰＵの任意のピンからそのピンの状態（ハイ or ローレベル）を入力する．この値を入れる変数は SHORT INT 型にする．

使用例：SHORT INT S;

S=INPUT(PIM_A4)→ポートＡの 4 番ピンの状態をＳに入れる．Ｓは SHORT INT 型に．

OUTPUT_HIGH(pin)

解説：指定された出力ピンをハイレベル（Ｖｃｃ）にする．

使用例：OUTPUT_HIGH(PIN_C0)

→ポート C の 0 番ピンをハイレベルにする．

OUTPUT_LOW(pin)

解説：指定された出力ピンをローレベル（グランド）にする．

使用例：OUTPUT_LOW(PIN_C0)

→ポート C の 0 番ピンをローレベルにする．

#INCLUDE <FILENAME>

#INCLUDE "FILENAME"

解説：コンパイルの時指定したファイルを接続して実行する．<>の時は system ディレクトリを探し，""の時はカレントディレクトリを探します．この標準のインクルードファイルはあらかじめコンパイラをインストールした時に用意されており，指定するだけで標準

的なラベルを使うことが可能になります.（上例では，PIN_C0　など）
使用例：#INCLUDE <16F877.h>

#USE DELAY(CLOCK=speed)
解説：PIC の動作速度を指定する命令．speed にはクロックを Hz 単位で入れる．DELAY_MS と DELAY_US を使うためにはこの命令でコンパイラにＣＰＵのスピードを知らせなければならない.
使用例：#USE　DELAY(CLOCK=20000000)
→CPUのクロックが20MHz

DELAY_MS(time)
DELAY_CYCLES(count)
機能：msec 単位のディレイとＣＰＵの命令実行サイクル単位のディレイを発生させる関数．ＣＰＵの命令実行サイクル単位はＣＰＵクロックの４倍の周期になる．設定できる引数の値は定数であれば０から６５０２５まで，変数であれば０から２５６まで使える．DELAY_MS(time)を使用するときには先に #USE DELAY でクロックスピードを知らせなければならない.
使用例：#USE　DELAY(CLOCK=20000000)
→CPUのクロックが20MHz

DELAY_MS(2);　　→50msec のディレイ
DELAY_CYCLES(count);　→count の値のディレイ

#FUSES options
この命令はプログラムをPICへ書きこむときにFUSEオプションを設定するものだ. プログラムのコンパイルへの影響は無く, 単に出力するファイルにFUSE情報を付加するだけだ. このFUSE情報はPICライターの書きこみソフトでも設定できるようになっている. FUSE情報の内容は製品の種類によってビットが反転していたりしるので, 古いライターの場合はライターの書きこみソフトで設定するほうが良い.
使用例:#FUSE HS,NOPUROTECT,NOWDT
→発信回路　HS, コードプロテクトをしない, ウォッチドッグ使用しない

データ型定義について
CCS C Compiler では標準Ｃに対して下記の特殊制限などがある.

(1) 型定義指定は変数が使われる前に必要

(2) TYPEDEF　で型の名前を新たに定義することができる.

(3) CONST　が型定義の前に付加された時は，定数として扱われプログラム中で変更することは出来ない.

(4) 定数は初期化が必須で，プログラム中で変更は出来ない.
　　またポインタ指定も使えない.

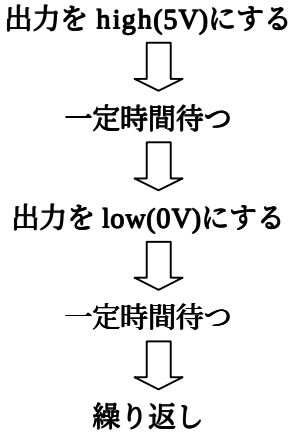
(5) SHORT　は特殊変数で，ビット操作の型定義.
　　SHORT の配列，ポインタ指定は使ない.

2.3.2.2　ＰＩＣライター

コンパイラで作成したＨＥＸファイルをＰＩＣに書きこむソフト．先述した「ＡＫＩ－ＰＩＣプログラマキット」に付属されているソフトを使用した.

2.4　制御方法

先述した通り，サーボモーターはパルス信号で制御され，そのパルスを，ＰＩＣを使って発生させるという手法をとる.
１つのプログラムで１つのパルスを作ることは非常に簡単で下記手順となる.



しかし，同時に多数のパルスを発生しなければならないので，工夫が必要に

なってくる．よく見かける方法としては下記のような手順がある．

a=パルス１のパルス幅
ｂ=パルス２のパルス幅
ｃ=パルス３のパルス幅
とする．

パルス１　出力を high にする
⇩
a　待つ
⇩
パルス１　出力を low にする
⇩
（x－a）　待つ
⇩
パルス２　出力を high にする
⇩
b 待つ
⇩
パルス２　出力を low にする
⇩
（x－b）　待つ

⇩
パルス３　出力を high にする
⇩
c 待つ
⇩
パルス３　出力を low にする
⇩
（x－c）　待つ
⇩
（λ－3x）　待つ
⇩
繰り返し

この手順により図 2-15 のようなパルスとなる.

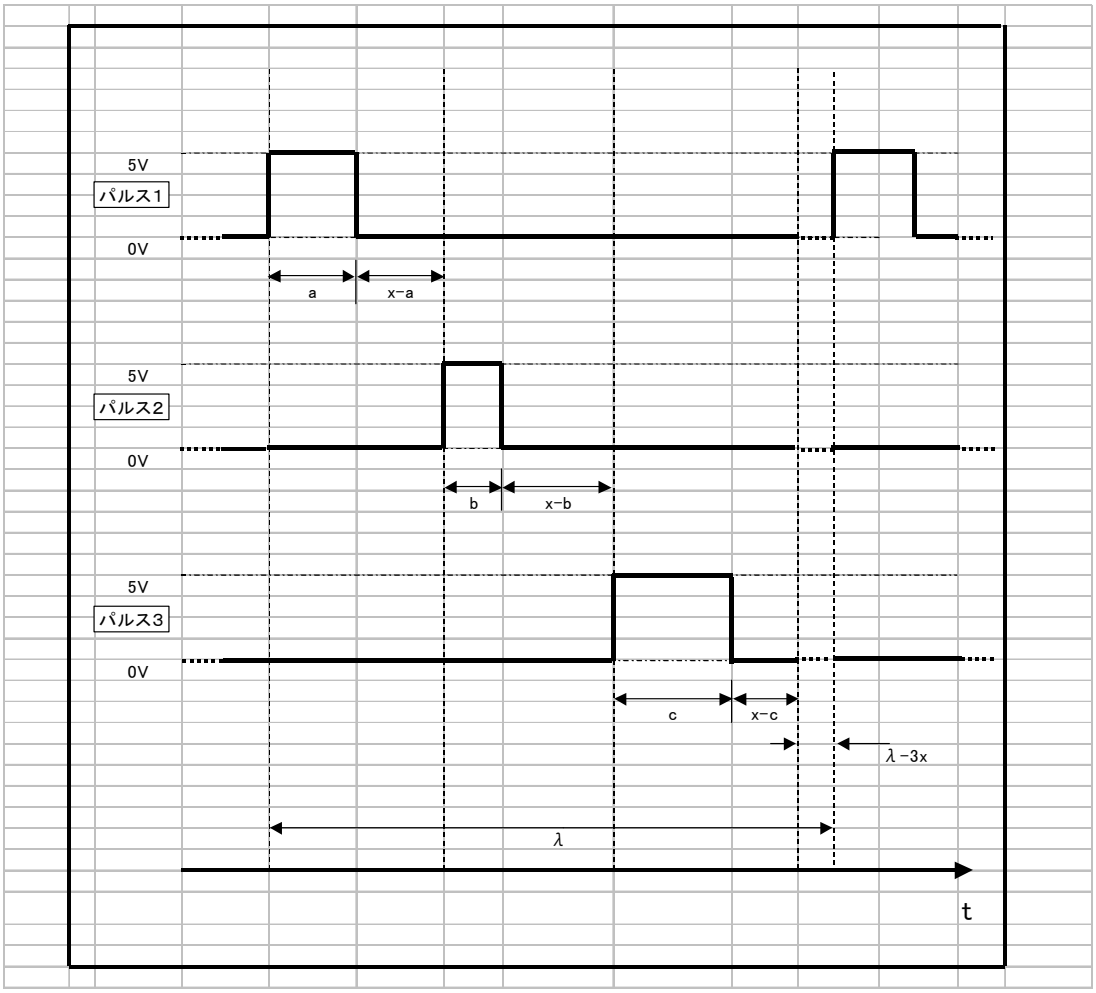


図 2-15

ｘは一つのパルスに与えられる領域で，サーボを制御するためのパルス幅は約１．０～２．０ms なので，ｘは２．０ms 以上必要となってくる．パルスの周期λは約２０．０ms でそれより大きくなるとサーボの動作が不安定になってしまう．よってこのプログラム手順では制御できるサーボの数が１０個以下と制限されてしまう．馬ロボットはより多くのサーボを使用するので，このプログラム手順では制御できない．　そこで下記のような手順を使用する．

パルス１　出力　high
パルス２　出力　high
パルス３　出力　high
⇩
b ms 待つ　：　b→最もパルス幅が短い
⇩
パルス２　出力　low
⇩
（a－b）ms 待つ　：　a→次にパルス幅が短い
⇩
パルス１　出力　low
⇩
（c－a）ms 待つ　：　c→最もパルス幅が長い
⇩
パルス３　出力　low
⇩
（λ－c）ms 待つ

この手順では図 2-16 のようになる．

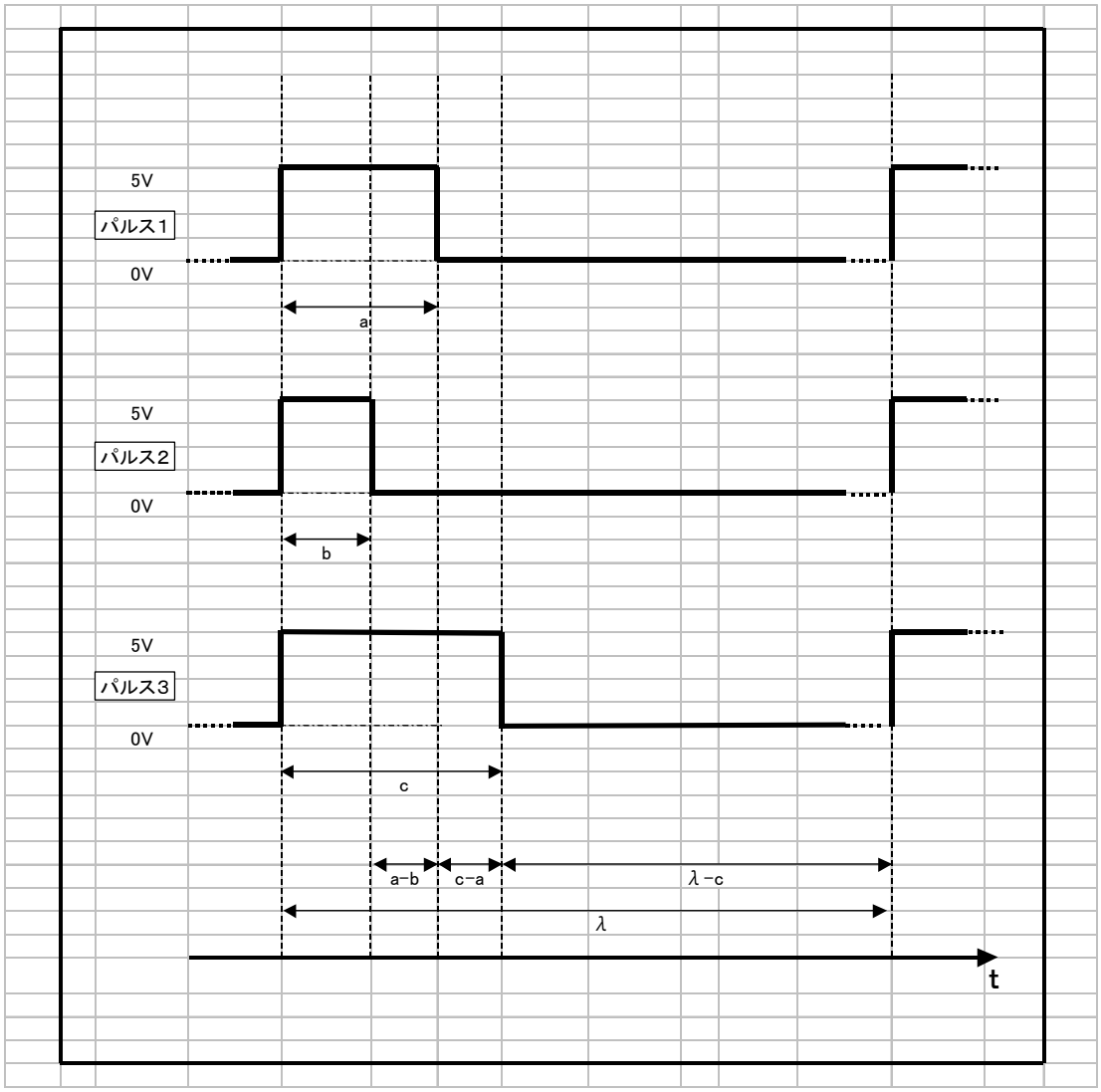
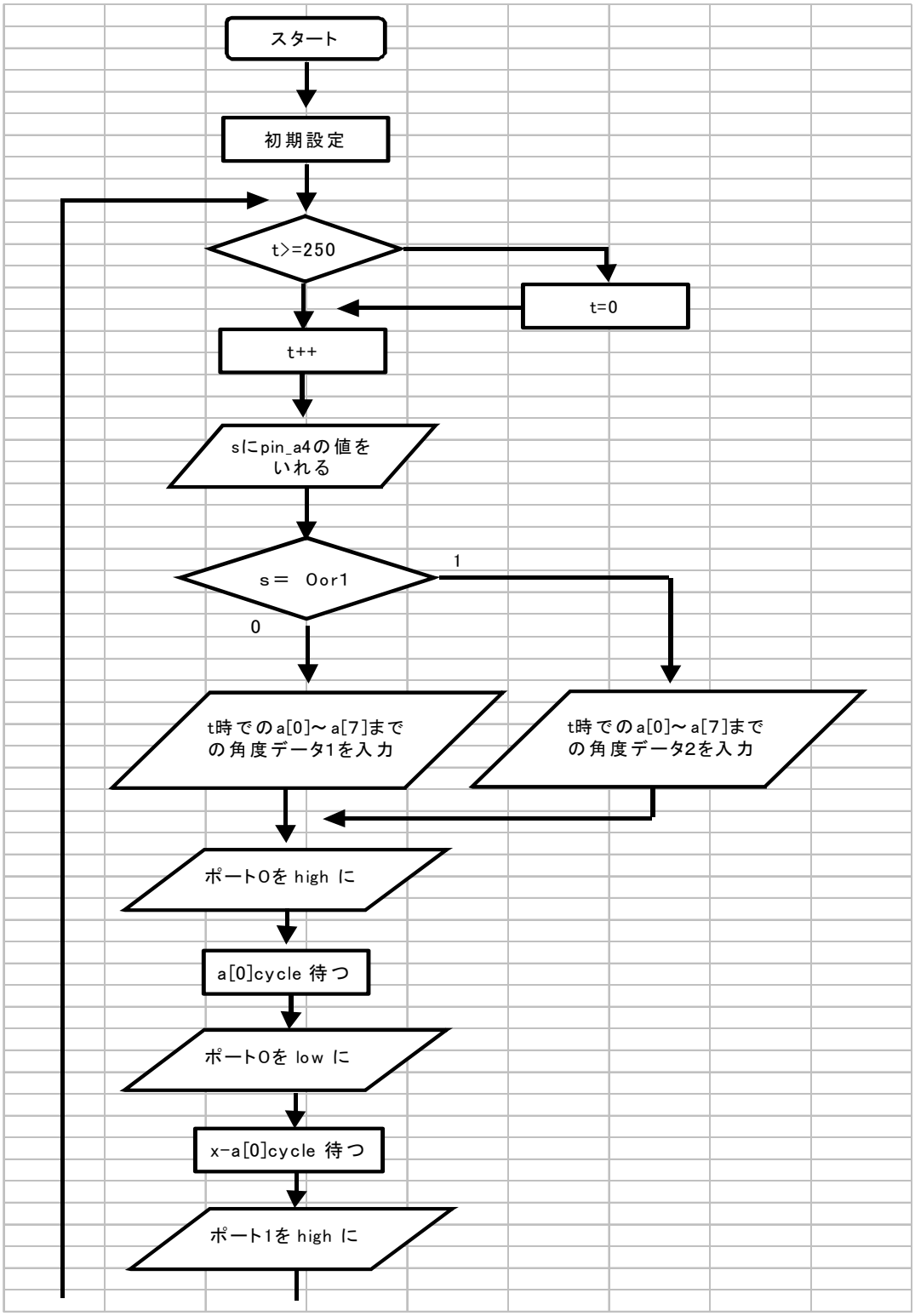


図 2-16

この手順で制御すると発生できるパルス数の制限をうけない.
ＰＩＣ１６Ｆ８７７は汎用ポートが３０個以上あるのでこれらのポートをすべて出力に設定すれば３０個以上のサーボを動かせると予想される.

2.4.1 プログラム（サーボ 8 個用）

2.4.1.1 プログラムの流れ


スタート
初期設定
t>=250
t=0
t++
sにpin_a4の値を
いれる
s＝ 0or1
1
0
t時でのa[0]～a[7]まで
の角度データ1を入力
t時でのa[0]～a[7]まで
の角度データ2を入力
ポート0を high に
a[0]cycle 待つ
ポート0を low に
x-a[0]cycle 待つ
ポート1を high に

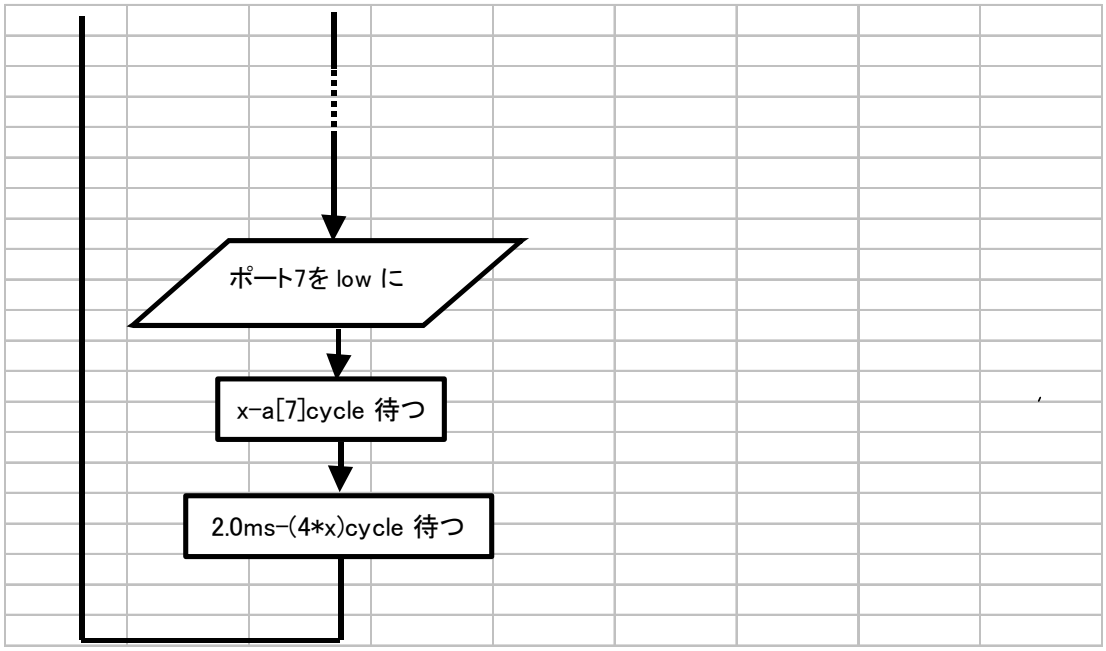


図 2-17

2.4.1.2　プログラムソース

```
#include <16F877.h>
#fuses HS,noprotect
#use delay(clock=20000000)

int i,low,cn,t,a[8],n;
short int s;

main(){

set_tris_a(0xf0);
#use fast_io(a)
while(1){
for(t=0;t<=4;t++){

s=input(pin_a4);
```

```
if(s==1){
if(t==1){cn=cn+1;}
if(cn>=30){cn=0;}
if(cn<=5){
            a[0]=150;
            a[1]=80;
            a[2]=130;
            a[3]=130;
            a[4]=130;
            a[5]=130;
            a[6]=140;
            a[7]=145;
            }
 if(cn>5){
   if(cn<=10){
               a[1]=110;
               a[2]=145;
               a[3]=150;
               a[4]=115;
               a[5]=120;
               a[6]=120;
               a[7]=180;}}
 if(cn>10){
   if(cn<=15){ a[0]=140;}}
 if(cn>15){
   if(cn<=20){
                  a[2]=110;
                  a[3]=180;
                  a[0]=130;
```

```
                    a[1]=130;
                    a[6]=130;
                    a[7]=130;
                     }}
 if(cn>20){
    if(cn<=25){
                    a[3]=150;
                    a[4]=140;
                    a[5]=90;
                    a[0]=120;
                    a[1]=110;
                    a[6]=145;
                    a[7]=150;
                    }}
 if(cn>25){
    if(cn<=30){f a[5]=120;}}

}

if(s==0){
if(t==1){cn=cn+1;}
if(cn>=12){cn=0;}
if(cn<=5){
              a[0]=135;
               a[1]=110;
               a[2]=125;
               a[3]=160;
               a[4]=135;
               a[5]=125;
               a[6]=125;
               a[7]=135;
```

```
            }
 if(cn>5){
    if(cn<=8){
                    a[0]=140;
                    a[2]=110;
                    a[4]=110;
                    a[6]=150;

                  }}
 if(cn>8){
    if(cn<=10){
                     a[1]=80;
                     a[3]=190;}}
 if(cn>10){
    if(cn<=12){
                      a[0]=135;
                      a[2]=125;
                      }}
 }



    output_high(PIN_b7);
    for(i=0;i<=a[0];i++){
        delay_cycles(48);}
    l=200-a[0];
    output_low(PIN_b7);
    for(i=0;i<=l;i++){
        delay_cycles(48);}

        output_high(PIN_b6);
```

```
for(i=0;i<=a[1];i++){
    delay_cycles(48);}
l=200-a[1];
output_low(PIN_b6);
for(i=0;i<=l;i++){
    delay_cycles(48);}

 output_high(PIN_b5);
for(i=0;i<=a[2];i++){
    delay_cycles(48);}
l=200-a[2];
output_low(PIN_b5);
for(i=0;i<=l;i++){
    delay_cycles(48);}

  output_high(PIN_b4);
for(i=0;i<=a[3];i++){
    delay_cycles(48);}
l=200-a[3];
output_low(PIN_b4);
for(i=0;i<=l;i++){
    delay_cycles(48);}

  output_high(PIN_a0);
for(i=0;i<=a[4];i++){
    delay_cycles(48);}
l=200-a[4];
output_low(PIN_a0);
for(i=0;i<=l;i++){
    delay_cycles(48);}
```

```
        output_high(PIN_b2);
    for(i=0;i<=a[5];i++){
        delay_cycles(48);}
    l=200-a[5];
    output_low(PIN_b2);
    for(i=0;i<=l;i++){
        delay_cycles(48);}

  output_high(PIN_b1);
    for(i=0;i<=a[6];i++){
        delay_cycles(48);}
    l=200-a[6];
    output_low(PIN_b1);
    for(i=0;i<=l;i++){
        delay_cycles(48);}

       output_high(PIN_a1);
    for(i=0;i<=a[7];i++){
        delay_cycles(48);}
    l=200-a[7];
    output_low(PIN_a1);
    for(i=0;i<=l;i++){
        delay_cycles(48);}

    for(i=0;i<=140;i++){
        delay_cycles(280);
        delay_cycles(200);}

   }
 }
}
```

2.4.1.3 プログラムの解説

```
output_high(PIN_b7);
for(i=0;i<=a[0];i++){
    delay_cycles(48);}
l=200-a[0];
output_low(PIN_b7);
for(i=0;i<=l;i++){
    delay_cycles(48);}
```

上記のプログラムは output_high(PIN_b7) でポートＢの 7 番ピン high にし，delay_cycles(48) を for 文で a[0]回繰り返すことで待ち時間を作る．そして output_high(PIN_b7) でｂ７ピンを low にし，このピンに与えられた時間が終わるまで待ち，次のピンの生成になる．

2.4.2 プログラム（サーボ 13 個用）

2.4.2.1 プログラムの流れ

MAIN　関数

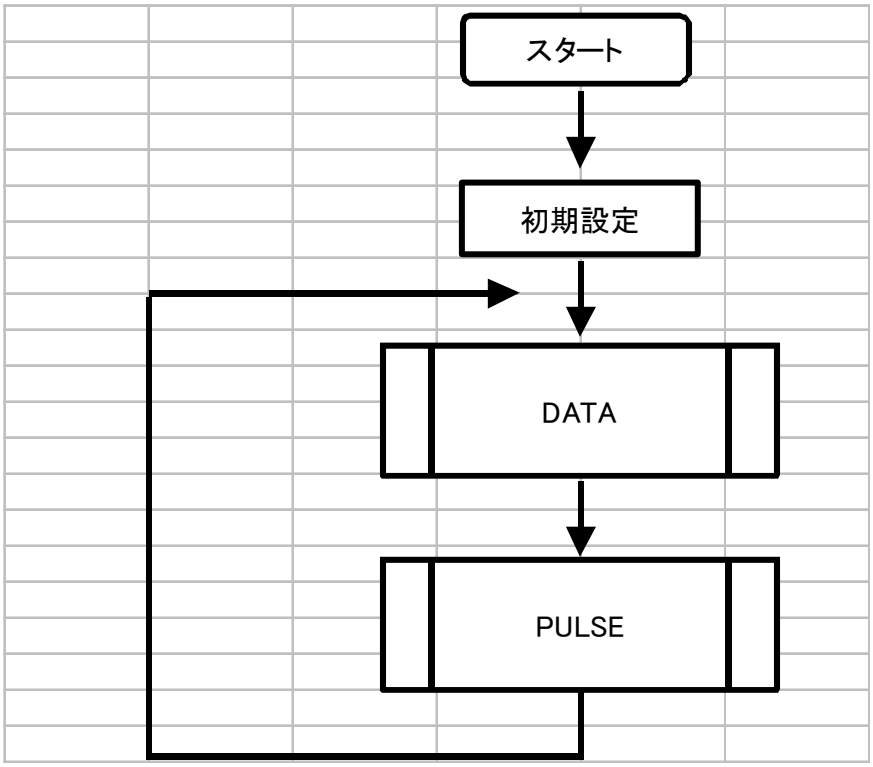


図 2-18-1

DATA　関数

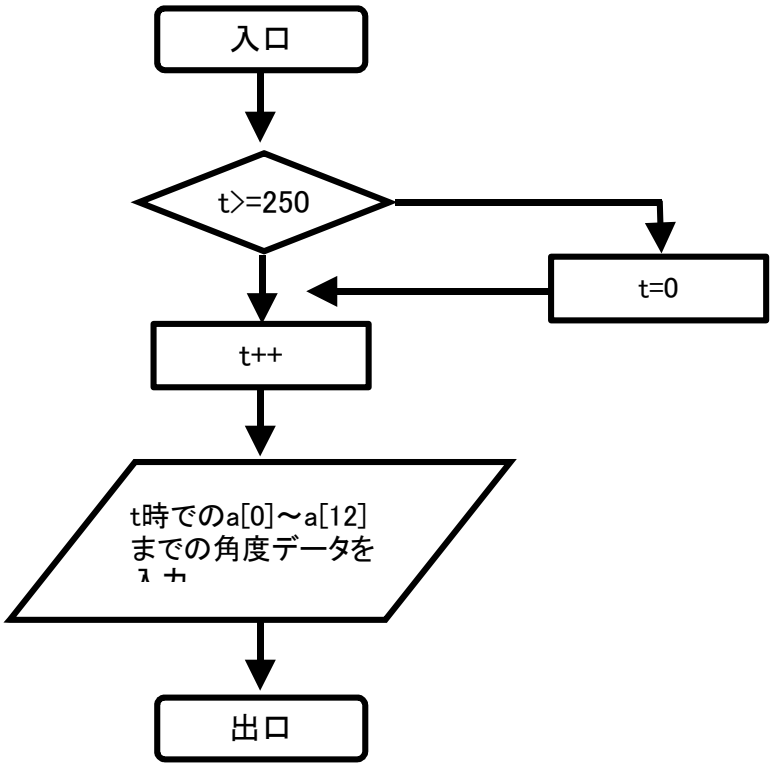


図 2-18-2

PULSE　関数

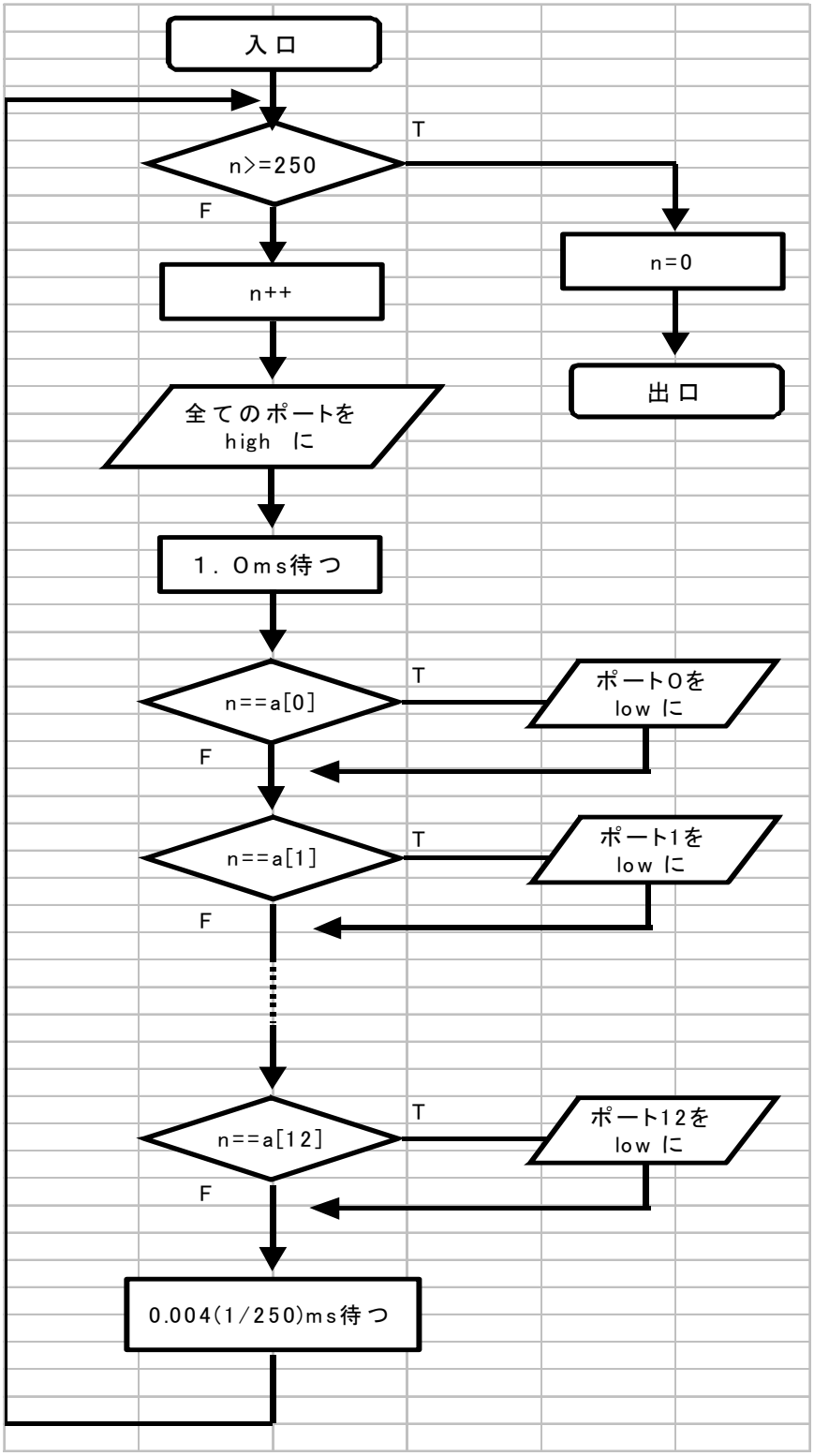


図 2-18-3

2.4.2.2　　プログラムソース

```
#include <16F877.h>
#fuses HS,noprotect,nowdt
#use delay(clock=20000000)

int i,n,a[13],b=0;

void date(int a[13],int b);
void parusu(int a[13]);

main()
{
int x[13],t;
while(1)
{
if(t>=250) t=0;
t++;

date(x,t);
parusu(x);

}
}


void date(int a[13],int b)
{

   a[0]=100;
   a[1]=130;
   a[2]=10;
   a[3]=30;
```

```
   a[4]=50;
   a[5]=250;
   a[6]=100;
   a[7]=200;
   a[8]=100;
   a[9]=130;
   a[10]=130;
   a[11]=130;
   a[12]=90;
   }

void parusu(int a[13])
{
    int n,i;

    output_high(PIN_c0);
    output_high(PIN_c1);
    output_high(PIN_c2);
    output_high(PIN_c3);
    output_high(PIN_c4);
    output_high(PIN_c5);
    output_high(PIN_c6);
    output_high(PIN_c7);
    output_high(PIN_b1);
    output_high(PIN_b2);
    output_high(PIN_b4);
    output_high(PIN_b5);
    output_high(PIN_b6);

    delay_ms(1);
```

```
    for(n=0;n<=250;n++){
    if(a[0]==n) output_low(PIN_c0);
    if(a[1]==n) output_low(PIN_c1);
    if(a[2]==n) output_low(PIN_c2);
    if(a[3]==n) output_low(PIN_c3);
    if(a[4]==n) output_low(PIN_c4);
    if(a[5]==n) output_low(PIN_c5);
    if(a[6]==n) output_low(PIN_c6);
    if(a[7]==n) output_low(PIN_c7);
    if(a[8]==n) output_low(PIN_b1);
    if(a[9]==n) output_low(PIN_b2);
    if(a[10]==n) output_low(PIN_b4);
    if(a[11]==n) output_low(PIN_b5);
    if(a[12]==n) output_low(PIN_b6);

    delay_cycles(24);
}
for(i=0;i<=200;i++){
delay_cycles(100);}
}
```

2.4.2.3 プログラム解説

サーボ 13 個用のプログラムはサーボの角度を決める DATA 関数がまだできていないが，サーボ 8 個用プログラムと同じように角度の値を入れればサーボは動く．現在は t によって変化しない適当な値を入れてあるが，全てのピンからのパルスの発生が確認できた．

```
    output_high(PIN_c0);
    output_high(PIN_c1);
                .
                .
    output_high(PIN_b5);
```

```
output_high(PIN_b6);
```

上記プログラムにより，使用する全てのピンをほぼ同時に　high にする．

```
for(n=0;n<=250;n++){
if(a[0]==n) output_low(PIN_c0);
if(a[1]==n) output_low(PIN_c1);
                .
                .
if(a[11]==n) output_low(PIN_b5);
if(a[12]==n) output_low(PIN_b6);

delay_cycles(24);
 }
```

上記のプログラムは　for 文で n を１ずつ増やしながら繰り返し a[]の値とｎが等しくなればそのピンを　low　にする．

2.5　実験結果

実験の結果 8 個のサーボを PIC ひとつで制御し, 馬ロボットを歩かせることができた. これにより馬ロボットで生馬の歩き方を再現させるためのベースができた.

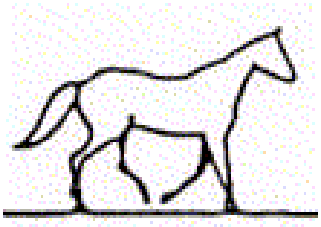
第３章　　考察

生馬の歩様（歩き方）には大きくわけて４種類あり，常歩（なみあし），速歩（そくあし），駈歩（かけあし），襲歩（しゅうほ）と分別されている．並歩から襲歩へと順に速さが増していく．本研究で作成した馬ロボットではこれらの歩様のうち常歩をすることができた．この章ではこれらの歩様を紹介し，生馬と馬ロボットの常歩を比較する．

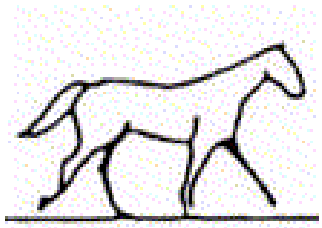
3.1　　歩様

3.1.1　　常歩

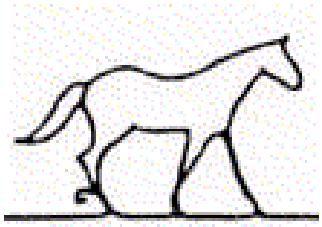
英語の「WALK」にあたる．人間が歩いている状態と同様の，ゆったりとしたスピードのない歩様．馬の肢運（あしはこび）は，左後肢，左前肢，右後肢，右前肢と，順に離地と着地を繰り返す．（図 3-1）リズムは，「１，２，３，４」の４拍になる．（図 3-2）もっとも基本となる歩様で，ゆったりと１肢ずつ肢運をするうちに，馬体の筋肉がほぐれ，運動のウォーミングアップになる．そのため，乗馬を行う時は必ず，この常歩を十分に行ってから次の運動を始めなければならない．人間がハイハイをしても全く同じような歩き方になると思われる．速さは１分間に１１０ｍほど．



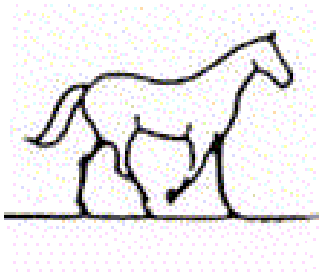
①左後肢を前に出す．左前肢が離地する．



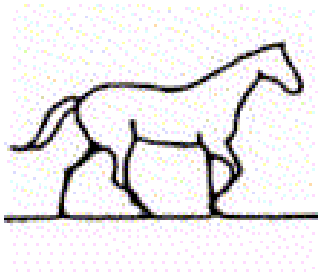
②左後肢が着地．左前肢が前に出る



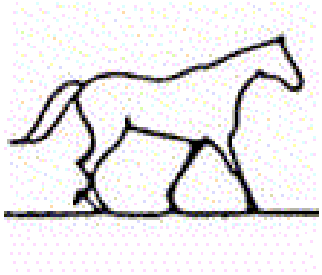
③左前肢が着地．右後肢が離地する．



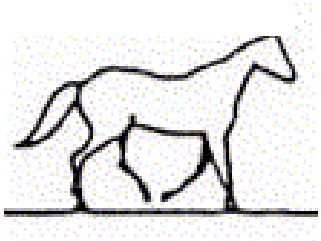
④右後肢が着地.
右前肢が離地



⑤右前肢が前に出る.

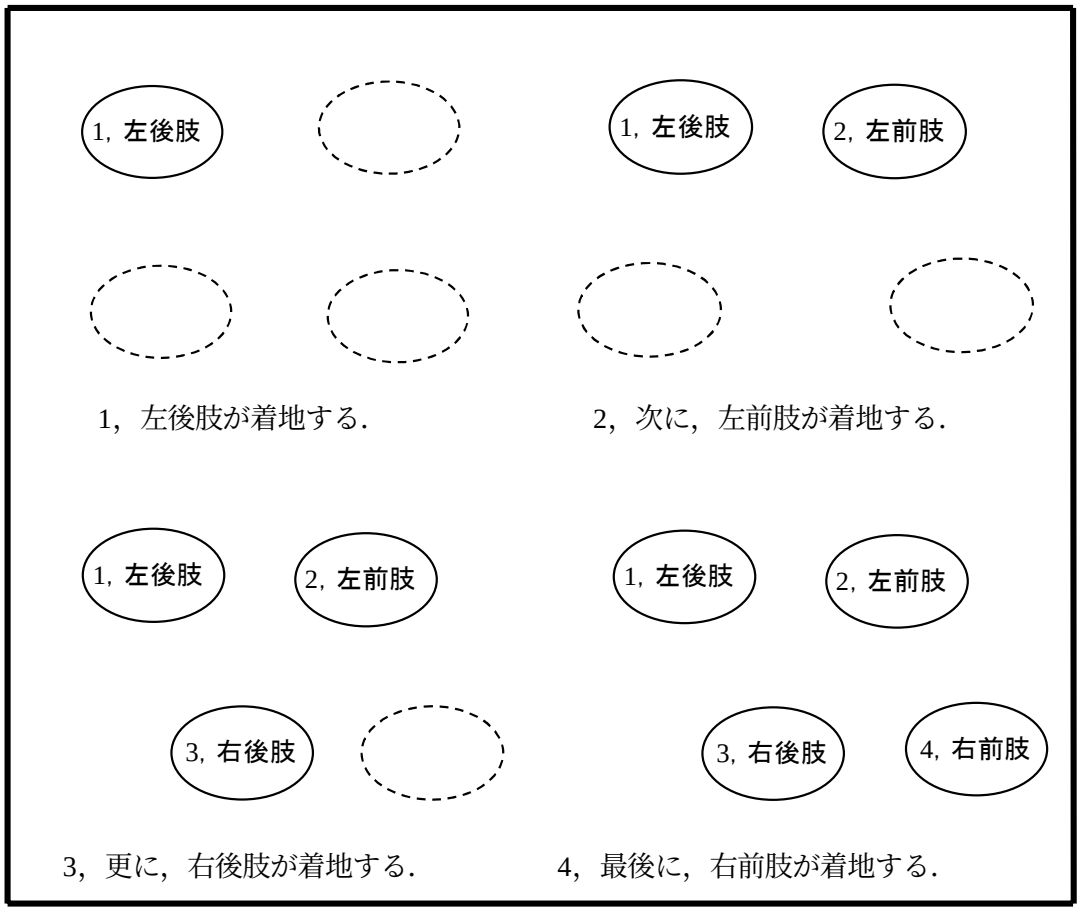


⑥左後肢が離地する.



⑦左後肢が前に出る.
左前肢が離地する.

図 3-1　常足



1，左後肢が着地する.　　2，次に，左前肢が着地する.

3，更に，右後肢が着地する.　　4，最後に，右前肢が着地する.

図 3-2　常歩の足あと

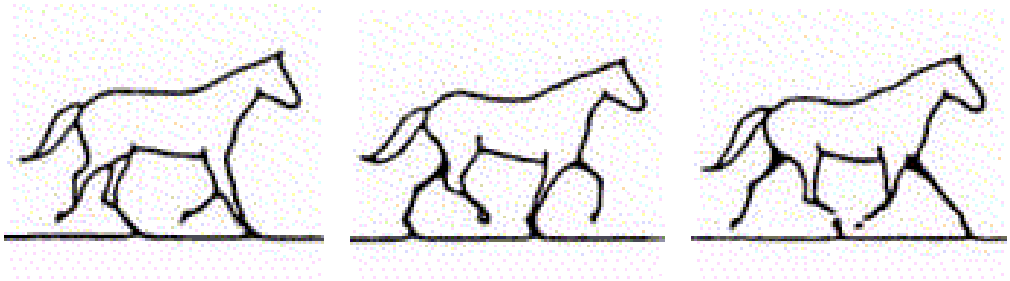
3.1.2 速歩

英語で「TROT（トロット）」と言う．この歩様はさらに斜対歩（しゃたいほ）と側対歩（そくたいほ）に分けることが出来る．速さは１分間に２２０mほど.

3.1.2.1 斜対歩

斜対歩というのは対角線の脚をペアにして行う歩様で，右前と左後（左前と右後）を同時に離地，着地させる.

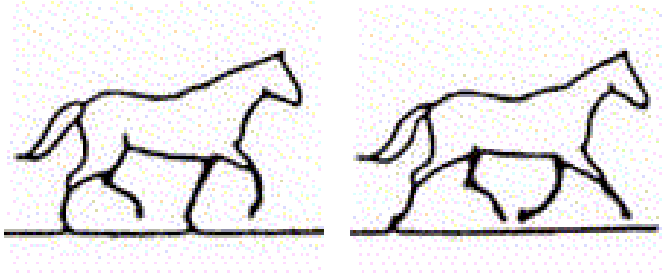
ほとんどの馬は斜対歩で速歩を踏むと思われる．また，速歩は２拍のリズムを刻むが，騎乗者が一拍毎に腰を挙げて，馬の背中が上下動するのを和らげる乗り方を軽速歩という.



①右後肢と左前肢が離地する.

②右後肢と左前肢が前に出る.

③右後肢と左前肢が着地する.
左後肢と右前肢が離地する.



④左後肢と右前肢が前に出る.

⑤左後肢と右前肢が着地する.

図 3-3 斜対歩

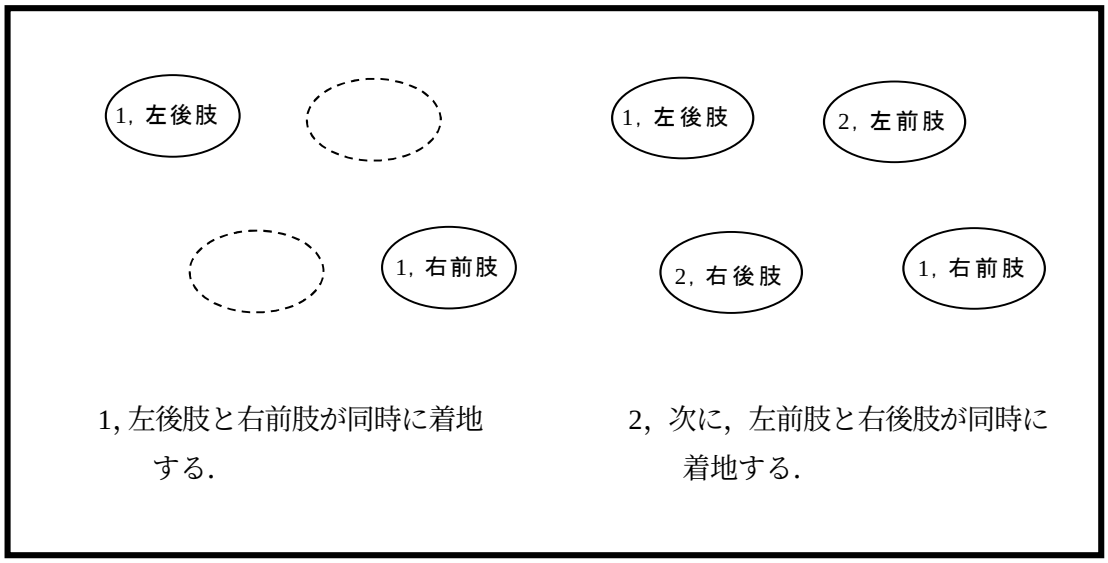
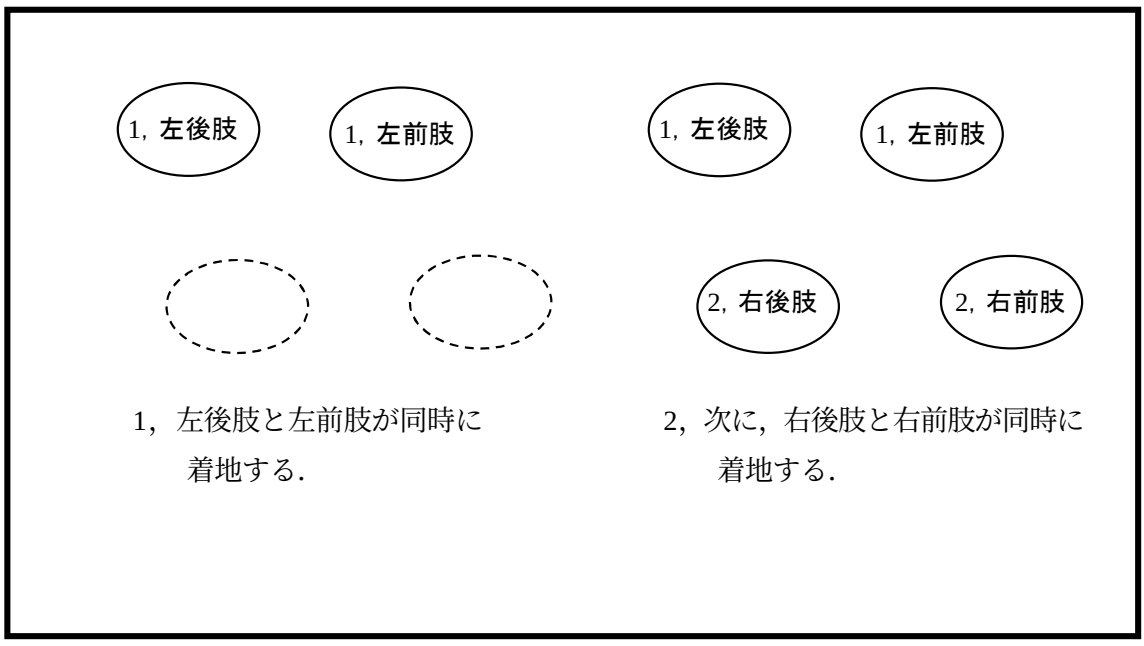


図 3-4　斜対歩の足あと

3.1.2.2　側対歩

側対歩（PACE）は同じ側の脚をペアにして行うもので，右前と右後（左前と左後）を同時に離地，着地させる．象やらくだ，キリンにとってはあたりまえの歩様だが，馬には珍しく，ドサンコや木曽馬，テネシーウォーカーなど一部の品種に見られる歩様.

側対歩には馬の背中の上下動が少ないという特長がある．この歩様は，馬の背中に荷物を積んだり，馬車を引いたりするのに適している．荷崩れしたり，馬車の揺れが少なくなる．そのため，馬に側対歩を教え込んだりすることもある．



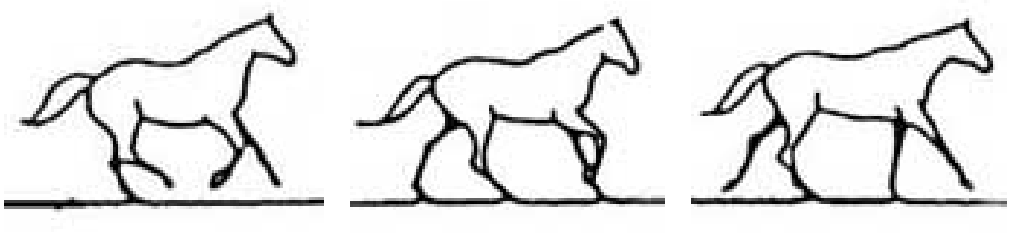
3.1.3 駈歩

図 3-5　側対歩の足あと

英語で「CANTER（キャンター）」と言う．馬が躍動するように走って，４肢が宙に浮く瞬間があるくらいのスピードが出る歩様．馬の疲労は最も大きく，それほど長い時間駈歩を続けることはできない．速さは１分間に３３０mほど．

3.1.3.1 右駈歩・左駈歩

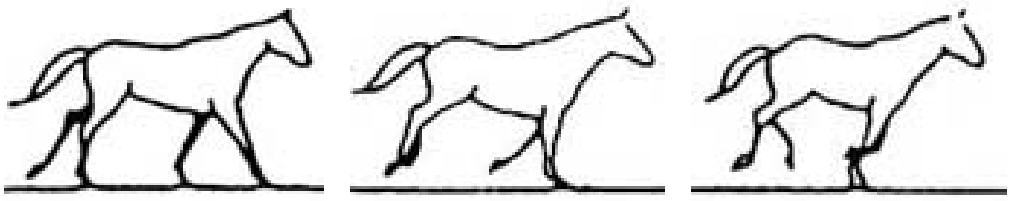
右駈歩と左駈歩があり，右駈歩の場合，まず左後肢が着地し，次に右後肢と左前肢が同時に，最後に右前肢となる３節運歩で右前肢がリードする走り方．左駈歩はその逆である．



①左後肢が着地する．

②左前肢と右後肢が着地する．右前肢が離地する．

③右前肢が前に出る．



④右前肢が着地する．
左後肢が離地する．

⑤左前肢と右後肢が離地する．

⑥左後肢が前に出る．

図 3-6 駈歩（右駈歩）

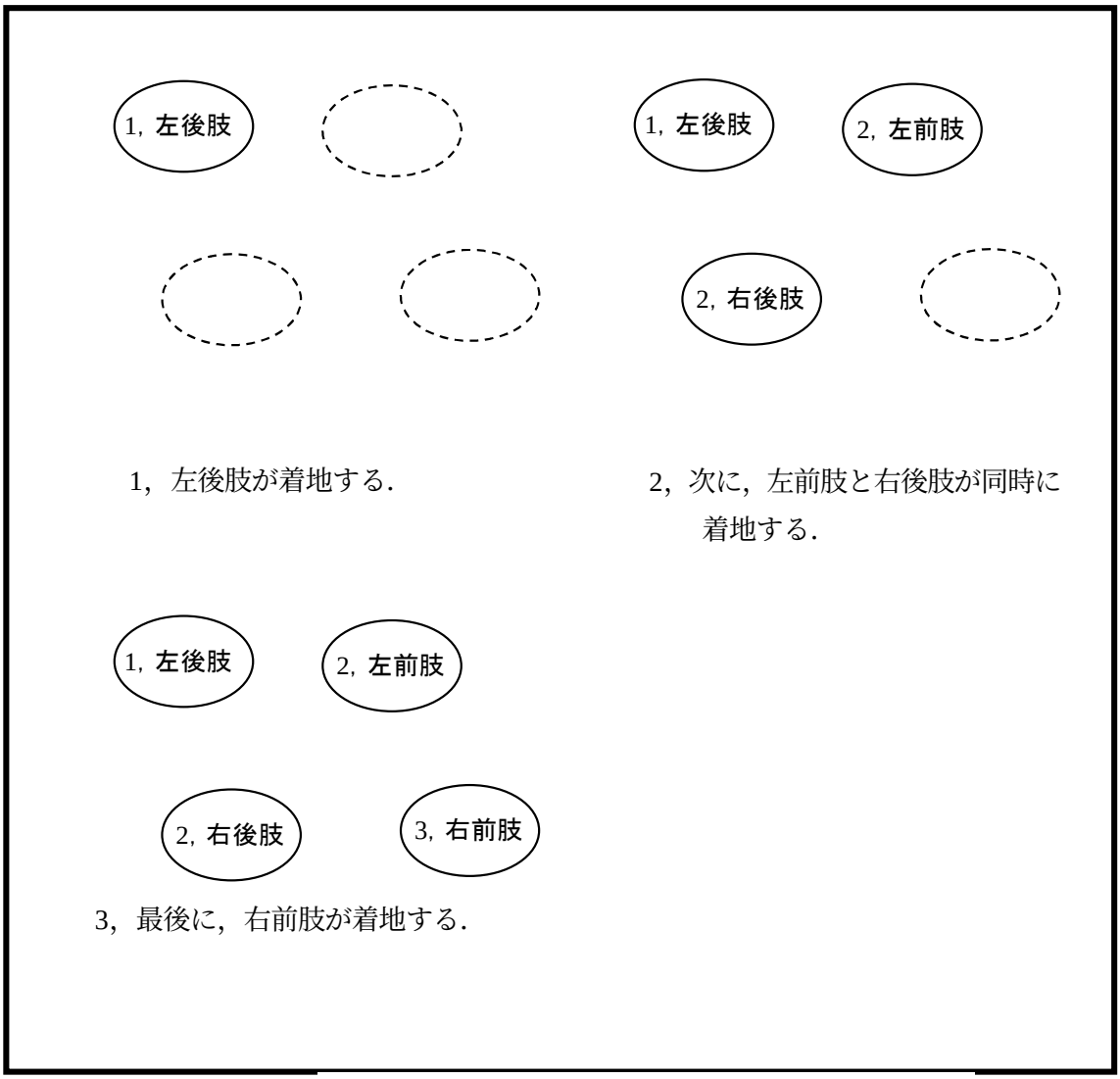


図 3-7　駈歩の足あと（右駈歩）

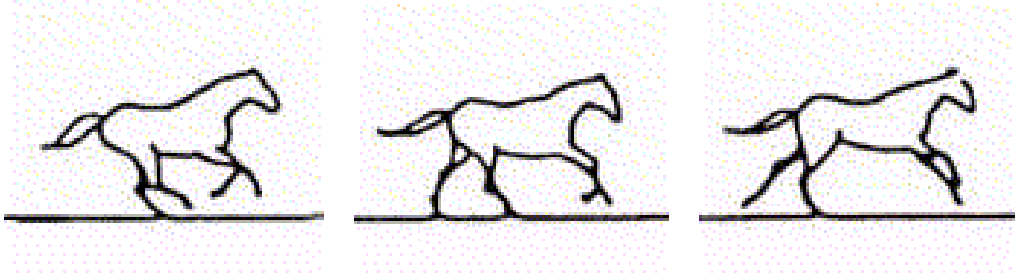
3.1.4　襲歩

英語で「GALLOP（ギャロップ）」という．４拍で，馬がもっとも速く走れる歩様．上下動がほとんど無いため，乗っていても非常に楽．しかし馬は疲れるのであまり長いこと襲歩を続けることは出来ない．競馬の時の馬は襲歩．速さは１分間に約１０００m 前後になる場合がある．

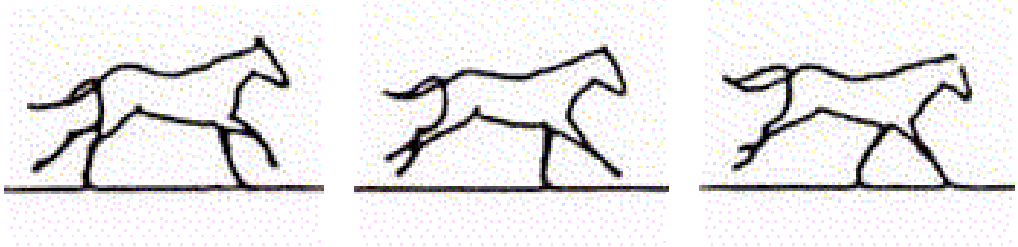
この歩様は交叉襲歩と回転襲歩に分けることが出来る．

3.1.4.1　交叉襲歩

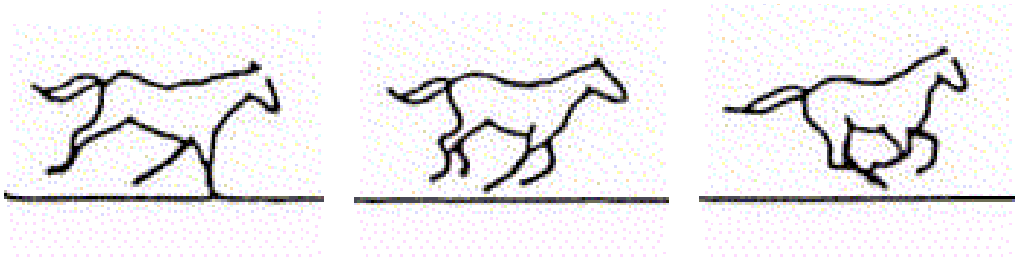
交叉襲歩というのは脊椎の柔軟性が無い馬や牛が行うもので，バランスがとりやすい歩様でもある．例えば，左後肢，右後肢，左前肢，右前肢の順に着地する．馬の４肢をこの順に結ぶと線分が交叉するところからこの名前がついた．



①左後肢が着地する．　②右後肢が着地する．　③左後肢が離地する．

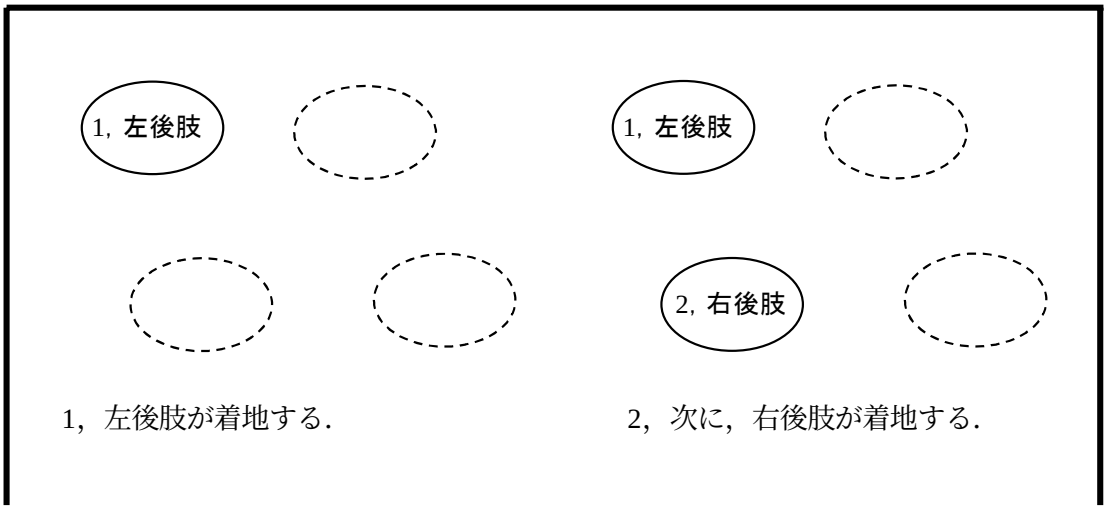


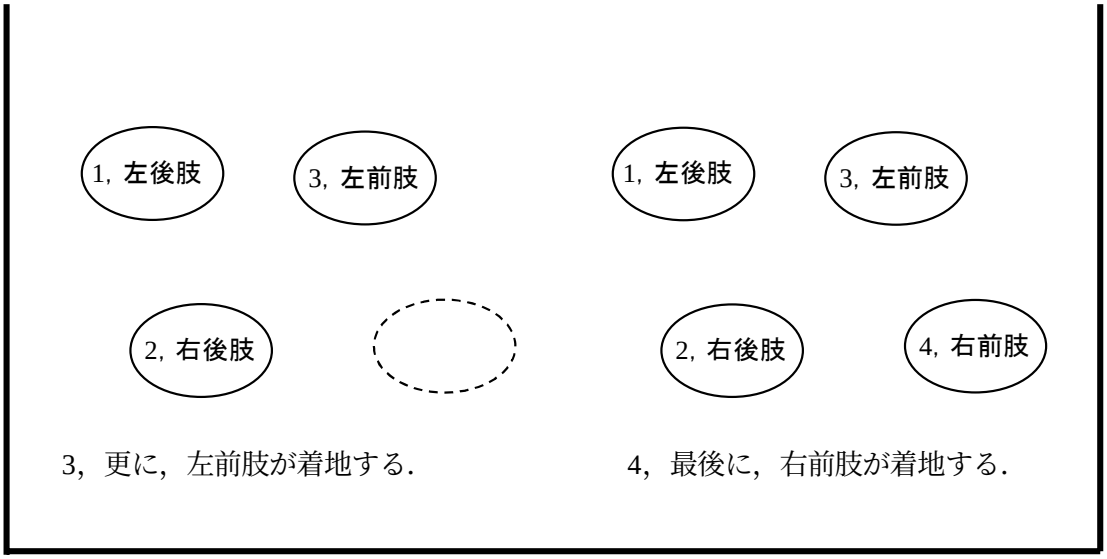
④左前肢が着地する．　⑤右後肢が離地する．　⑥右前肢が着地する．



⑦左前肢が離地する．　⑧一瞬，すべての肢が離地する．　⑨左後肢と右後肢が前に出る．

図 3-8　交叉襲歩



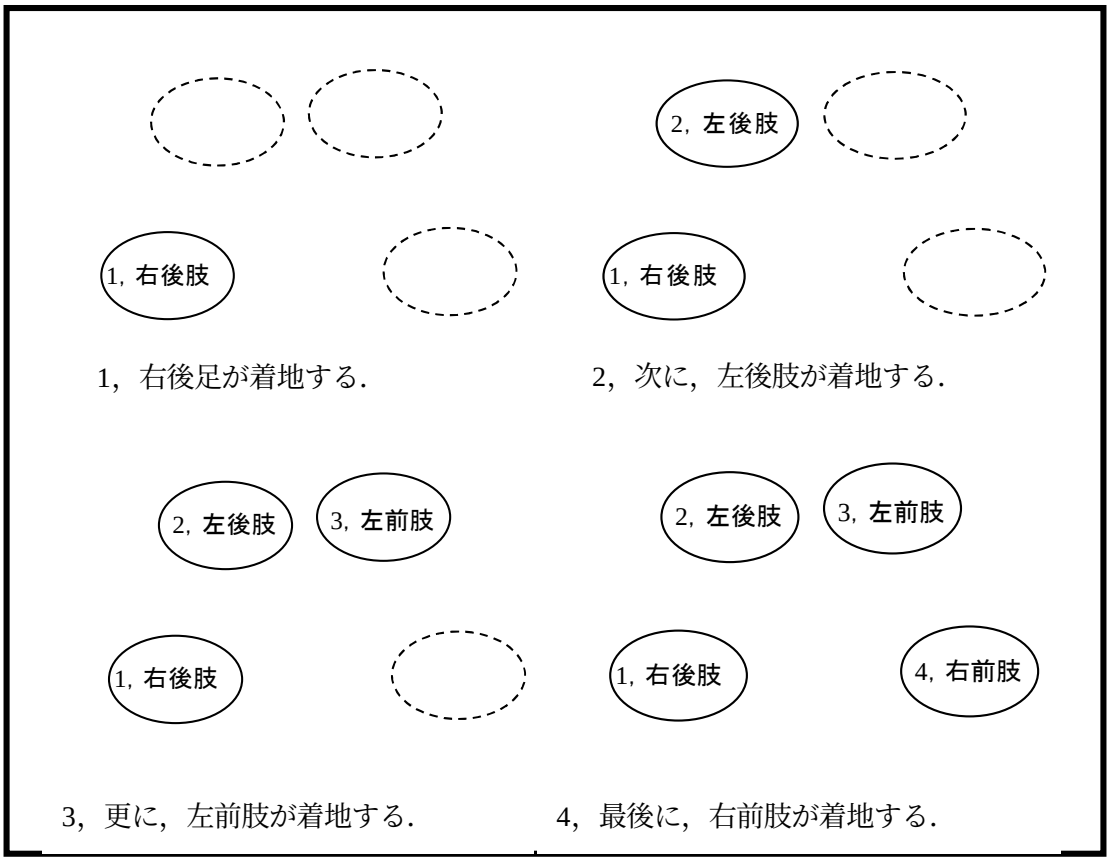


3，更に，左前肢が着地する．　　4，最後に，右前肢が着地する．

図 3-9　交叉襲歩の足あと

3.1.4.2　回転襲歩

回転襲歩は犬や鹿などが行うもので，右後肢，左後肢，左前肢，右前肢の順に着地する．馬も競馬の発送直後などは一時的に回転襲歩をとる．他にも猫のように後肢を同時に蹴り出すハーフバウンドという歩様もあるが，基本的に馬は交叉襲歩で走るものだと思ってよい．



1，右後足が着地する．　　2，次に，左後肢が着地する．

3，更に，左前肢が着地する．　　4，最後に，右前肢が着地する．

図 3-10　回転襲歩の足あと

3.1.5　手前

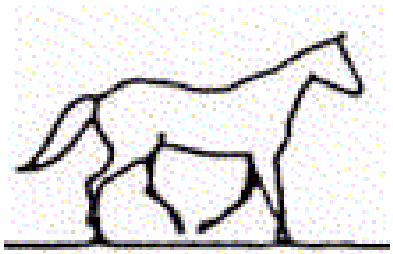
速歩と駈足，襲歩には手前というものがある．駈足と襲歩の場合，右前肢が一番前の方に着地するような走り方を右手前，その逆を左手前と言う．

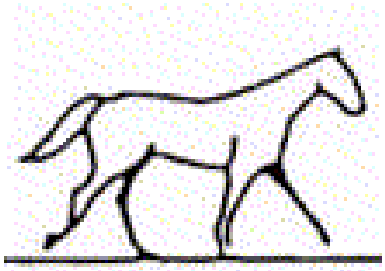
速歩の場合は左右対称の動きだが，人が乗って軽速歩をとった場合に手前が生じる．右前肢が着地した時に人が鞍に座る乗り方を右軽速歩といったりする．

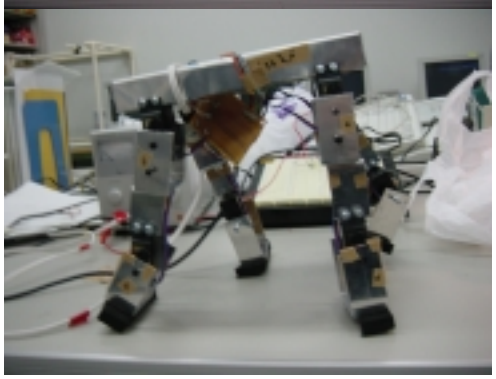
いつも同じ手前で走っていては馬が疲れてしまうので，時々馬も手前を変えて走る．馬が自分で変えないときは騎手がかえてあげたりもしている．

3.2　馬ロボットと生馬の歩様の比較（常歩）

生馬　　馬ロボット







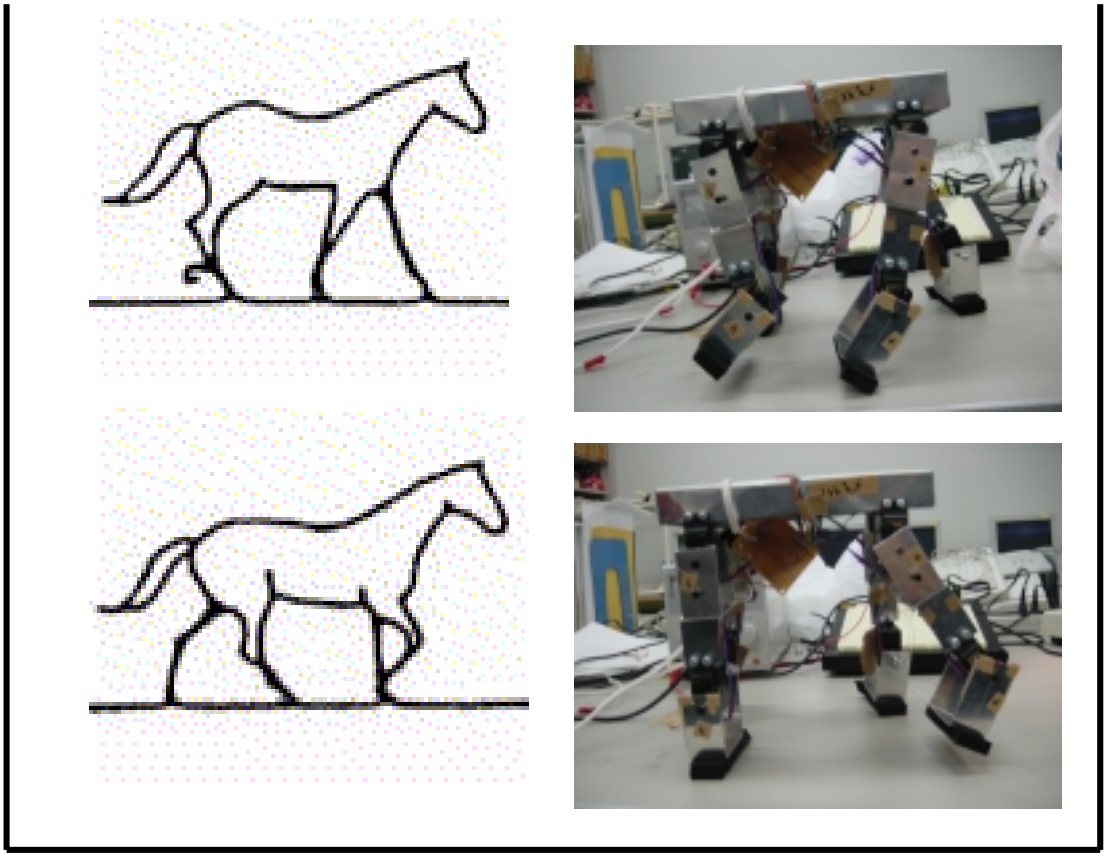


図 3-12

図 3-12 でわかるように馬ロボットは生馬の常歩に近い動きで歩行できていることがわかる．

3.3 考察

常歩，速歩は，揺れと速さ的に健康増進に適していると思われる．今後，馬ロボットで速歩でも歩くことができるようにすべきである．しかし，駈歩，襲歩は，揺れ，速さ的に健康増進には適していないと思われる．これは，揺れが激しすぎる点と，速さが安全性をかけるという点である．駈歩，襲歩は，生馬の乗馬でさえ，乗りこなすのに大変な経験と訓練が必要となっており，健康増進を目的とした乗馬ロボットと，目的が異なると思われる．

本実験ではこれら歩様のうち常歩をすることができたが，
他の歩様はまだできておらず，駈足，襲歩を馬ロボットで行うことは非常に困難であると思われる．

第４章　　結章

4.1　　本研究の結果

本研究ではＲＣサーボを馬の関節として使用し，ＰＩＣでそれを制御することで馬ロボットを歩かせることができた．これにより，これからの４足歩行馬ロボットでの乗馬療法の研究のため，役立たせることができるであろうと思われる．

4.2　　今後の課題

4.2.1　　サーボの角速度

本実験で製作した馬ロボットはただ歩いただけで，本来の目的である乗馬療法のための揺れまで再現されていない．技術的な問題点として関節であるサーボの角速度が制御できていないことがある．

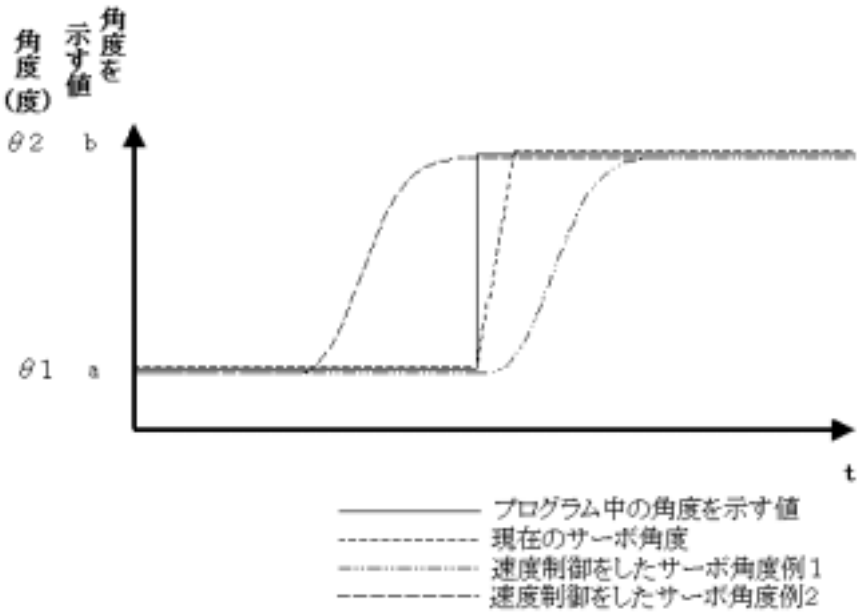


図 4-1

図 4-1 はプログラムの角度データの変化に対して現在のプログラムでのサーボの角度変化と速度制御によって期待できる角度変化の例である．現在のプログラムは「角度データの値」＝「パルス幅」であるから，角度データに変化が生じるとそれにあわせて瞬間的にパルス幅が変わるので，サーボは最高速度でそのパルス幅に合わせようとする．そして目的角度に達すれば角速度は瞬間的に０になる．このような動きでは振動が激しく，各部にかかる負担が大きいし，生馬のようになめらかに歩くことは不可能である．

そこで，図 4-1 の例のように滑らかに角度を変化させたいのだが，その方法としてまず挙げられるのが，細かくｔをわけ角度データを少しずつ変化させていく方法である．この方法を使えば例のように動かすことはできるだろう．しかしデータ量がかなり多くなると予想される．他の方法として考えられるのがファジィを用いてパルス幅を変化させる方法である．私はファジィについて，まだ勉強不足なのだがファジィに詳しい吉本氏によると目的角度のデータを一つ与えるだけで例のような変化が可能だという．できれば私もファジィを学び，馬ロボットを歩行させる技術として取り入れていきたい．

4.2.2 脚関節数

今回製作し，歩かせることのできた馬ロボットの脚関節の数は脚１本につき２個である．しかし，実際の馬ロボットは図 4-2 のように大きな関節は各脚で

３つである．

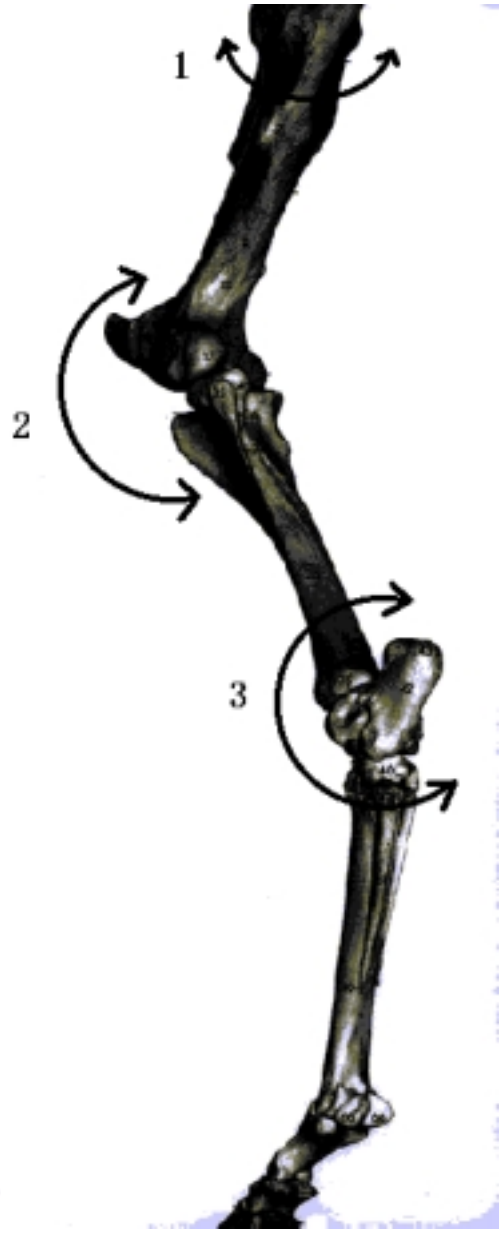


図 4-2

よって，馬ロボットも関節を各脚に３個ずつにすることでより生馬に近い動きが可能になるものと思われる．

すでに第２章で説明したが各脚に３個の関節をもつ馬ロボットは製作段階に入っており，より生馬に近い動きが期待される．

4.2.3 馬の操作方法

馬ロボットの操作方法は生馬に似た方法になると思われる．そこで，生馬の操作方法を紹介する．

発進＝ＧＯ

から④の手順で，推進（Ｐｕｓｈ）を行う．

①基本フォームを保ちながら，脚（ふくらはぎ全体の特にくるぶしあたりまで）を馬腹に密着させ，その位置で馬の動きに同調しながら，力が前方に向くようにタイミングよく瞬間的に圧迫．

②もし，馬が反応を示さなければ，これを 2～3 度繰り返す．

③それでも反応しなければ，踵（拍車も利用して）で，今度は力が後方に向くように軽打．それでも反応が無い場合は強打する．ただし，膝から上は動かさないこと．

④踵で強打しても反応がないようなら，長鞭（１００cm 程度）を腰角のあたりに軽く数回あてる．なお，長鞭のあて方は，ドアノブを回す要領．

脚を馬腹に密着させ，力が前方に向くように圧迫

⇩

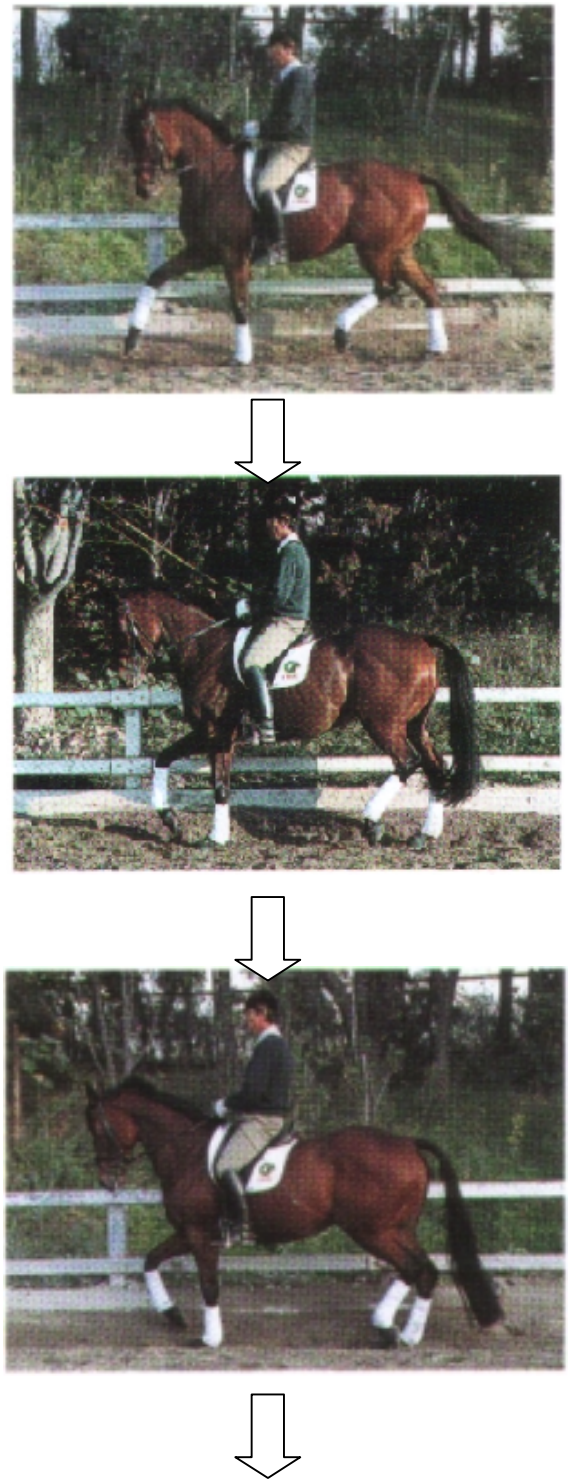
膝から上を動かさないように心がけつつ,
踵で力が後方に向くように軽打

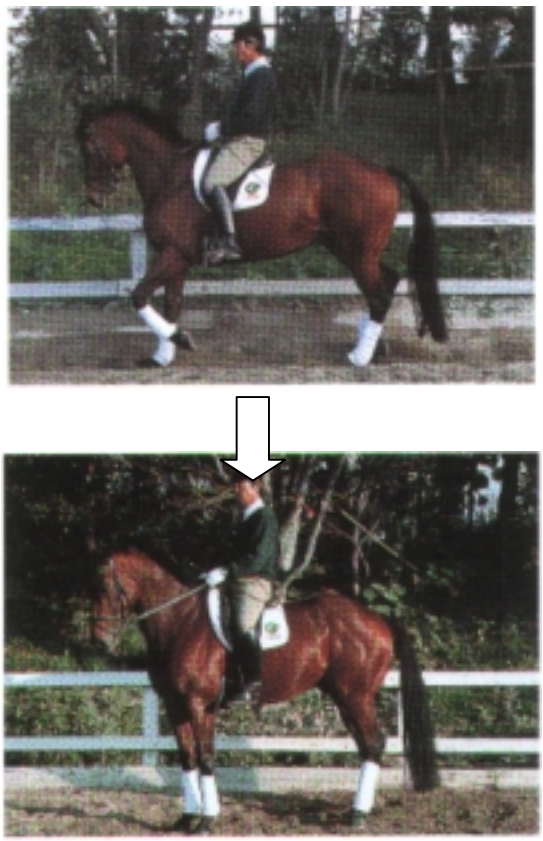
脚で推進する時には必ず手綱を使うことで
（＝手脚の一致），馬を屈撓させることができる
図 4-3

ブレーキ・停止＝Ｈａｌｔ
半停止＝Ｈａｌｆ　Ｈａｌｔ

基本フォームを保ちつつ，脚で馬腹を圧迫（前方へ）しながら徐々に手綱を控える．このとき，両脚と両拳を均等に使うことがポイント．それによって唐突的ではなく, ゆるやかに, しかも確実にブレーキをかけることができる. また, ホーラと声をかけながら行うとさらに効果的. Ｈａｌｆ　Ｈａｌｔ以上に強い指示を与えることで，よく調教された馬なら正しく停止をする．なお，４肢をそろえて停止をできれば理想だが，そのためには日々の調教で馬に習慣づける

必要がある.



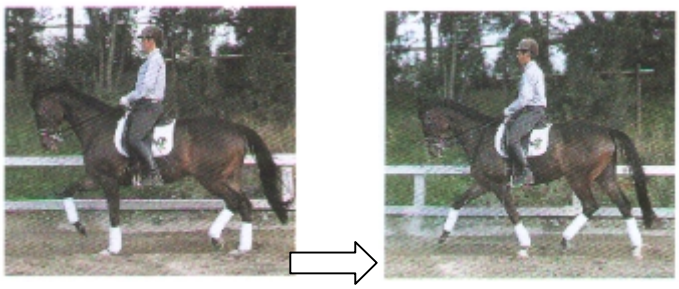


徐々に後肢が踏み込んできて，収縮の度合いが
大きくなっているのがわかる

図 4-4

伸長速歩

基本フォームを保ちながら，脚で馬腹を圧迫し，手綱を控えてアゴをゆずらせ，馬のクビがラウンドした体制を保つ．馬の首が前に出ないようにしながら，少し手綱をゆずりクビから背中を伸展させる．



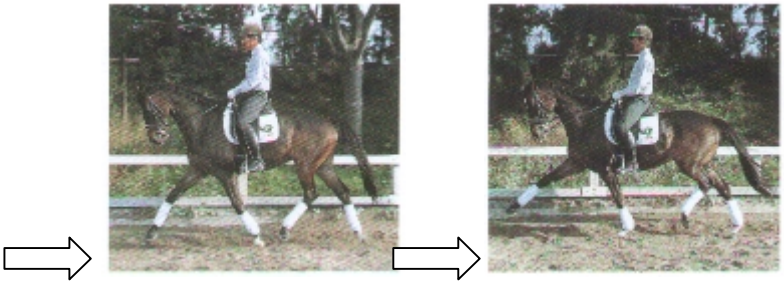


図 4-5

収縮速歩

基本フォームを保ちながら手綱を控え，馬のアゴをより深く求める．そして，力が前方を向くように脚を使って馬腹を圧迫し，後脚を 1 cm でも 2 cm でも踏み込ませるように意識する．

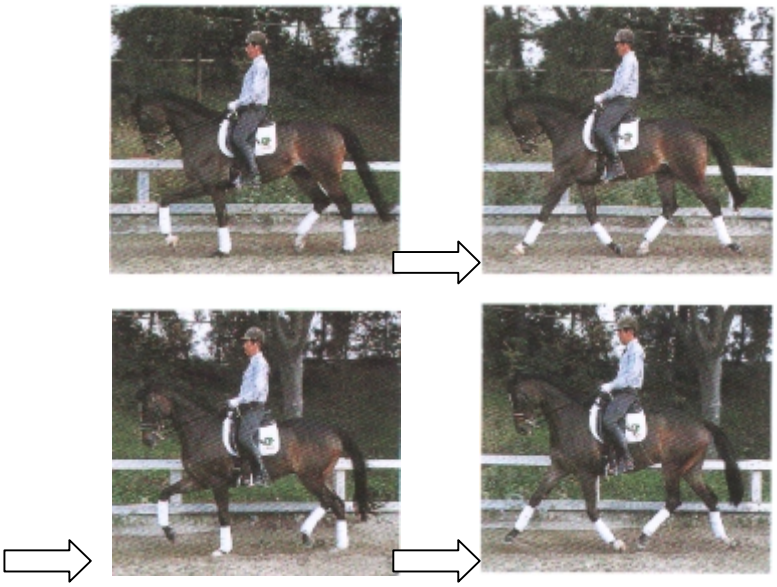
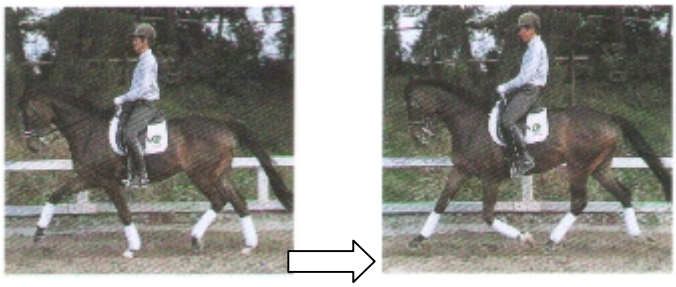


図 4-6

尋常速歩

伸長速歩と収縮速歩のちょうど中間にあたるのが尋常速歩です．



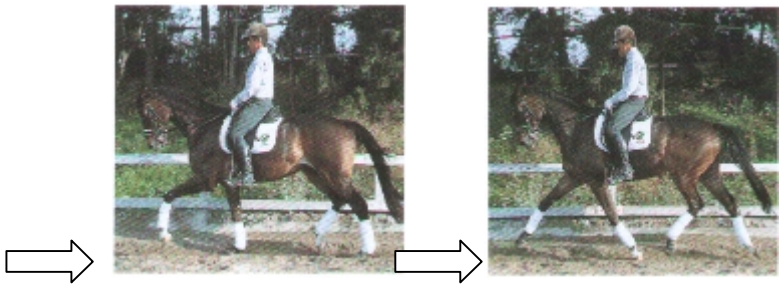


図 4-7

Running

ライダーがエネルギーやリズムを意識せず，脚で推進することばかりに気をとられているため，ただスピードが速くなるばかりで，馬のステップが大きくなっていない．

さらにエネルギー量も減少してしまっている．

このような状態になったら，いったん歩度をつめて再度，屈撓を求め直す．そして，改めて「発進＝ＧＯ」を行う．

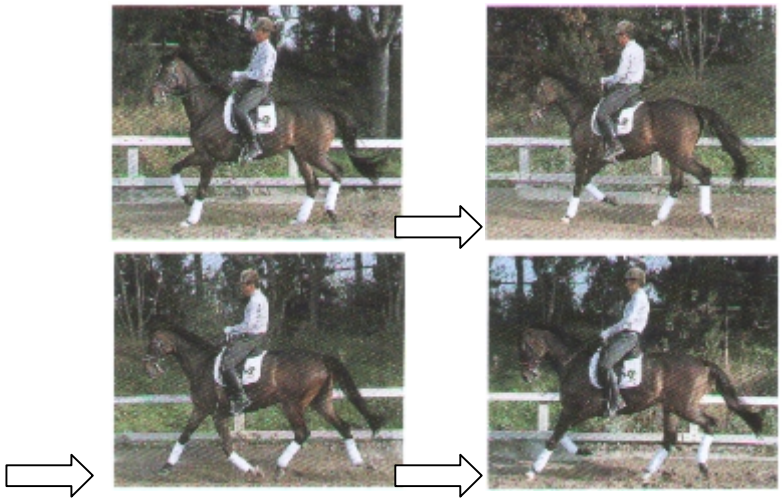


図 4-8

Swimming

ライダーが手綱だけで馬を抑えているため，後脚が踏み込めていない．馬のエネルギーが失われ，まるで泳いでいるかのように不安定な走りになってしまっている．歩幅は小さくなっているが脚は上がらず，弾発力に欠ける．

もちろん手綱を使うことも大切ですが，同時に座骨を安定させ，馬の動きに同調しながら両脚（特にふくらはぎ）で馬腹を前方に圧迫しなければならない．そのうえで，馬の後脚の踏み込みを意識する．

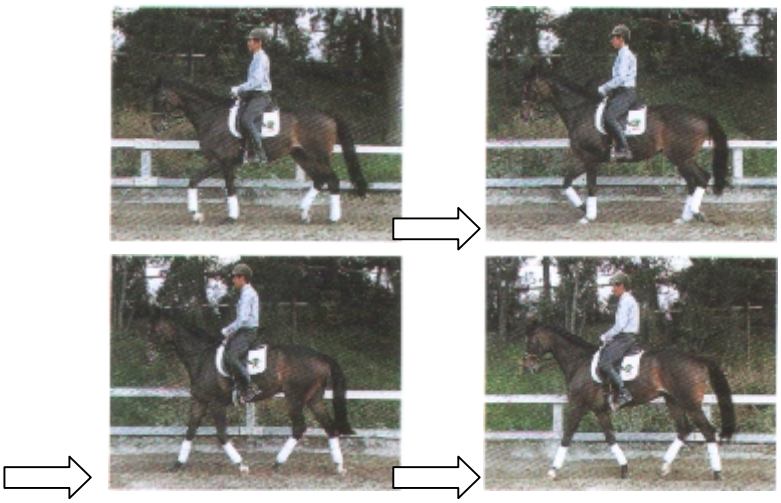


図 4-9

歩度の伸縮とエネルギー量

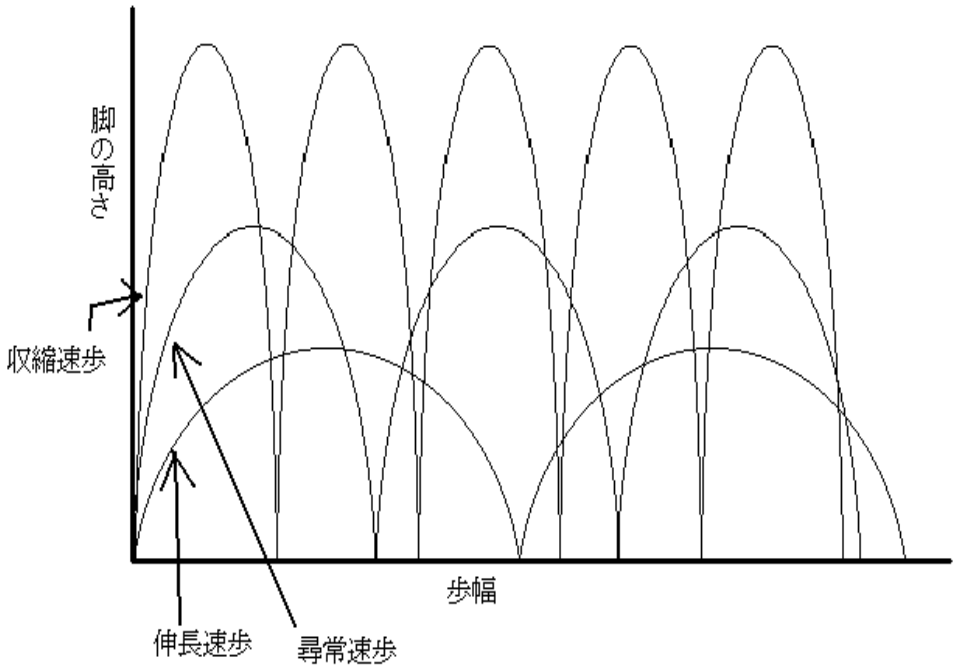


図 4-10

図 4-10 は歩度の伸縮度別に「馬の歩幅」と「肢の高さ」を示したものだ．どの歩度においても，それらの占める面積が同じだということが読み取れる．つまり，歩度の伸縮においては，一定のエネルギー量を保持した状態で歩幅の伸縮を求めることになります．そして，そのためにはよいリズムで走る馬の感覚をつかみ，それを常にイメージして乗ることが大切だ．

4.3 全体的な考察

本研究で馬ロボットは歩くことはできたが完成度は高いとはいいがたく，上記したように改良の余地が多々見られる．

本研究の最終目標である「馬ロボットに人が乗り，そして歩くことで健康が増進される．」という段階はまだ遠いかもしれない．しかし，本研究でそこに少しでも近づくことができたと思う．

参考文献

高知工科大学知能機械システム工学科知能ロボティクス研究室卒業論文集“健康増進用馬ロボットの制御を目的とする生馬の歩行パターンについて”
/菅野正人（２００１）

電子工作のためのＰＩＣ活用ガイドブック
/後閑哲也・株式会社技術評論社（２０００）

学生のためのＣ
/中村隆一　他・東京電機大学出版局（１９９５）

乗馬ライフ（２００２年2月号）
/オーシャンライフ株式会社

特集・ＰＩＣマイコンをつかおう
/CQ 出版株式会社（２０００）

はじめてのＰＩＣマイコン
/中尾真治・株式会社オーム社（２００１）

趣味の電子回路工作
/ http://www4.justnet.ne.jp/~sei.inoue/index.html

電子工作の実験室
/ http://www.picfun.com/

三和電子機器株式会社
/http://www.sanwa-denshi.co.jp/WebPages/index.html

PIC な日曜日
/http://www.kimurass.co.jp/picindex.htm

マイクロチップ・テクノロジー・ジャパン

/http://www.microchip.co.jp/

秋月電子通商

/http://www.akizuki.ne.jp/

CCS

http://ccsinfo.com/

Hori's Library

/http://adam.miyagi-ct.ac.jp/~s93936/index.html

謝辞

本研究は筆者が高知工科大学工学部知能機械システム工学科において行った研究をまとめたものである.

本研究を行うにあたってご指導ご鞭撻を下さった高知工科大学知能機械システム工学科王碩玉教授に対して深く感謝致します.

また，日ごろから研究において励まし，そしてアドバイスを下さった同大学知能ロボティクス研究室の皆様に深く感謝致します.

最後に，筆者の研究に対して理解を示し，支援して下さった両親に感謝致します.

2002 年 2 月 5 日