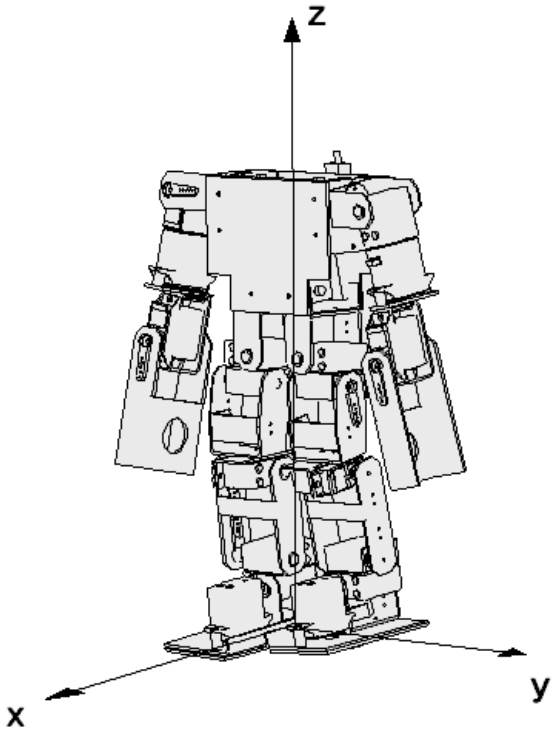
二足歩行ロボット入門



職業能力開発総合大学校東京校
生産情報システム技術科
小野泰二

目次

1　モータ

1.1　歩行ロボットに使われるモータ

最近ではベンチャー企業や個人で歩行ロボットを作る人たちも増えてきました。その多くが、比較的安価で制御しやすい RC サーボモータもしくはその改良版ともいえるロボット用モータを使っています。

図 1: 双葉電子工業 (株) の RC サーボモータ FP-S3003 の概観

いわゆる DC モータというのは、電流を流すと回転軸がぐるぐる回るのですが、歩行ロボットに使うモータは負荷がかかっても指定した角度を保持している必要があります。たとえば人のひざ関節を考えてみると、歩行の際に自分の体重を支えながら曲げ伸ばしを繰り返さなくてはなりません。このため、十分な負荷に耐えつつ指定角度を保持できるものでなくてはならないのです。このようなモータは、DC モータ以外に減速機 (ギア)、角度を読み取るセンサ (ポテンショメータ)、制御基板などを組み合わせて実現することができます。これらを組み込んでいるのが、RC サーボモータ (図 1) もしくはロボット用モータと言われるものなのです。

図 2 は図 1 の RC サーボモータ FP-S3003 の裏蓋をはずして制御基板を出したところです。この制御基板には DC モータが直接半田付けされています。

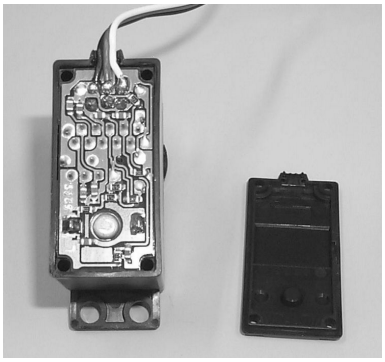


図 2: 制御基板

図 3 は制御基板をはずして DC モータ, ポテンショメータを見えるようにしたものです。モータの回転軸はポテンショメータから出ています。制御基板にはモータドライバなどの電子部品が付けられていることが分かります。FP-S3003 には使われていませんが、歩行ロボットに使われるモータの多くは、制御基板にマイクロ

コンピュータが使われています。マイクロコンピュータを使用したモータのほうが保持トルクが高いといわれています。

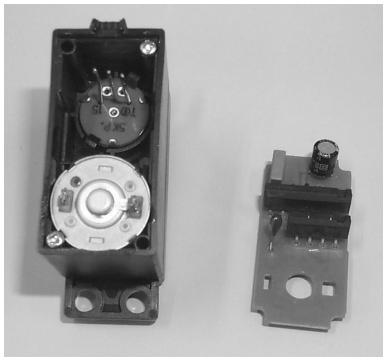


図 3: ポテンショメータ, モータ, 制御基板

図 4 にギアの配置を示します。FP-S3003 では 4 枚のギアが使われていることが分かります。それぞれのギアは大きいギアと小さいギアからなり、大きいギアで力を受けて小さいギアで次に伝えることでトルクを上げています。最終ギアは出力軸になっており、ポテンショメータにもつながっています。

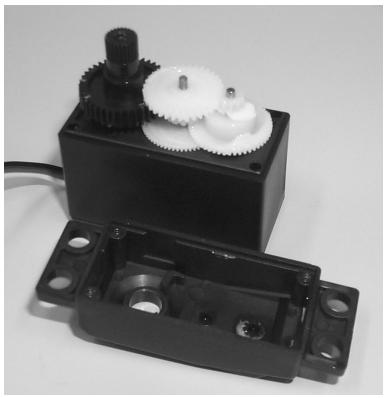


図 4: ギアの配置

この FP-S3003 はトルクが小さいために、2 足歩行ロボットに利用するためにはモータの数を減らすなどの工夫が必要です。ロボット用として発売されているモータも同様にギアとモータと制御基板を持ち、2 足歩行ロボットを作るのに十分なトルクを備えています。現在ロボット用モータは各社から様々なタイプのものが発売されていますが、その一例として ROBOTIS CO.,LTD. の Dynamixel AX-12 の外観を図 5 に、基本スペックを表 1 に載せておきます。

1.2 歩行ロボット用モータの制御方法

現在発売されているロボット用モータの制御方法は大きく二通りに大別されます。ひとつはパルスによる位置指定であり、もうひとつはシリアルデータによる位置指定です。いずれも統一企画のようなものはなく、詳細は各社まちまちのというのが現状です。

図 5: ROBOTIS CO.,LTD. ロボット用モータ Dynamixel AX-12 の外観

表 1: AX-12 の基本スペック

参考保持トルク	12(7V)kg·f/cm~16.5(9.6V)kg·f/cm
減速比	1/254
速度	0.269(7V)sec/60°~0.196(9.6V)sec/60°
最大動作角度	300° (Endless turn 可)
電源電圧範囲	DC7V~DC12V (Recommended voltage: DC9.6V)
最大電流	900mA
重量	55g
コマンドシグナル	デジタルパケット
プロトコル	半二重非同期通信
リンク方式	1 ワイヤ双方向 (5V TTL レベル)
通信速度	7343bps~1Mbps
フィードバック	位置, 速度, 温度, 負荷, 電源電圧
材質	エンジニアリングプラスチック
モータ	Cored Motor

1.2.1 パルスによる位置指定

まだロボット用モータが市販されてなかったころ、ベンチャー企業や個人で歩行ロボットを作る人たちは、ラジコン模型のステアリングなどに使う RC サーボモータを利用していました。このモータの制御方法がパルスによるものだったために、現在でも多くのモータがパルスによって位置指定を行っています。

各社のモータでパルスと回転角度が多少異なるばかりでなく、同じ会社のものでも型番が違うと違いがあったり、安価なものですと同じ型番でも個体差があったりします。しかし、おおむね 20mS の周期で ON と OFF を繰り返し、ON の幅が 1mS から 2mS 程度の範囲で位置の指定ができます (図 6)。

多くのモータがパルスを受け取るだけですが、最近ではモータから現在の角度に応じたパルスを返すものも

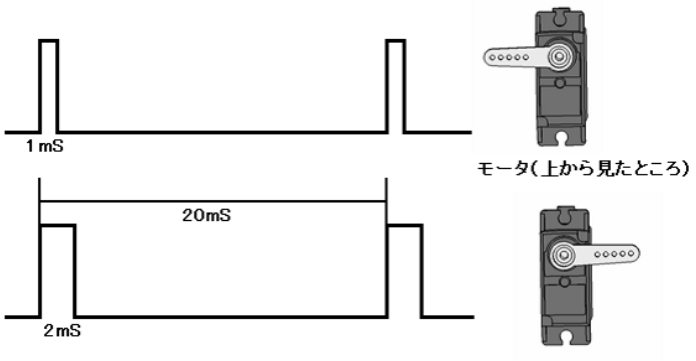


図 6: パルスによる位置指定

出ています。また、専用の機器やソフトを使って設定を変えることが可能なものもあります。

以下に GWS 社の RC サーボ MICRO 2BBMG のパルス幅とモータ角度の関係を示しておきます。

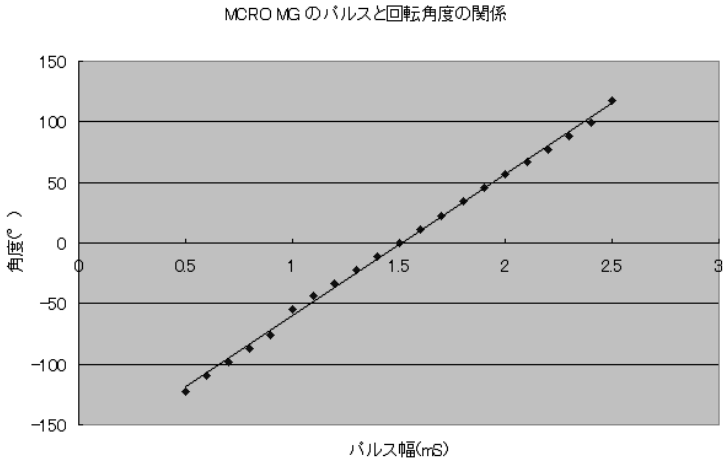


図 7: MICRO 2BBMG のパルス幅とモータ角度の関係

このグラフでは、パルスの ON 時間が 1.5mS のときのモータの角度を 0° としています。また、時計回りに正方向を取りました。

このモータは可動範囲が比較的大きく、安価でトルクもあるのですが、パルスとモータ角度の関係に多少のばらつきがあり、個体差も比較的大きいようです。

1.2.2 シリアルデータによる位置指定

前述のモータは、パルスの ON 時間といういわばアナログ的な位置指定を行うものでしたが、シリアル通信を用いて位置データを送信するものも発売されています。このタイプのモータの特徴としては、位置だけではなく電流や温度などといった多種類の情報を送受信でき、可動範囲やトルクなどモータの設定をデータ送信に

よって変更することもできるということです。また、一本のシリアル線を用いて、分岐させたり直列に繋いだりすることで複数のモータに指示を与えることも可能です。

シリアルデータによって制御するモータも複数の会社から発売されていますが、送受信できるデータの種類やデータの仕様は各社まちまちです。

表 2 に図 5 で紹介した AX-12 のモータ角度指定のための命令を示しておきます。ここでは 150° (中心位置:図 8 参照) を指定する場合を例にとりました。

表 2: AX-12 のモータ角度指定のための命令 (中心位置の場合)

例	0xFF	0xFF	0x01	0x05	0x03	0x1E	0xFF	0x01	0xD8
意味	スタート bit		ID	データ長	書き込み	位置指定	下位 bit	上位 bit	チェックサム

このモータの場合、最初にスタートビットとして 0xFF,0xFF と 2 バイト使います。その次の ID はモータの ID であらかじめモータに一意に設定した数値になります。データ長は、このビット以降のデータの長さです。ここでは「書き込み」から「チェックサム」までの 5 バイトになります。5 バイト目はオペレーションになります。ここでは「書き込み」の場合を示しましたが、他にも「読み込み」「PING」などが利用可能です。次の 1 バイトはメモリマップのどこに書き込むかの指定です。位置指定は 0x1E に 2 バイトのデータを書き込みます。位置とデータの関係は図 8 のようになっています。最後はチェックサムでデータのチェックに利用します。ID 以降のデータを足し上げ下位 8 ビットの NOT を取ります。上の例ですと、`~(0x01+0x05+0x03+0x1E+0xFF+0x01) & 0xFF=0xD8` となります。

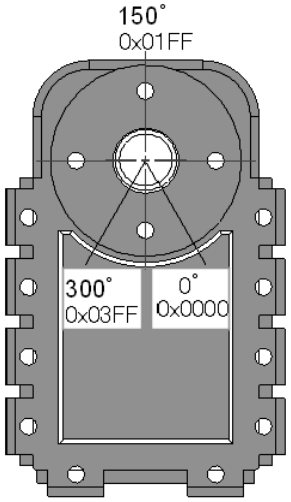


図 8: AX-12 の位置指定 (「いろいろ製作日記」 [4] の CAD データを加工)

モータの動かし方が分かったところで、モータの配置の仕方を考えてみましょう。

2 モータの配置と座標系

ロボットを作る場合に重要なのがモータをどのようにいくつ配置するかということです。市販の RC サーボモータやロボット用モータを使った場合で考えてみましょう。また、ロボットを動かそうとする場合まず大事なのが座標系の定義です。座標系に関する必要事項をまとめておきます。

2.1 モータの配置

多くの人型ロボットは 20 個前後のモータを利用していますが、これらの指示には通常マイクロコンピュータが使われています。それぞれのモータに位置を指定してやることによってロボットの姿勢を決めることができるわけです。

では、ロボット用モータを利用して人型二足歩行ロボットをつくる場合モータをどのように配置すればよいかを考えてみましょう。

人間の関節はひざのように 1 方向にしか回転しないところもあれば、股関節のように 3 方向に回転するところもあります。モータによってこのような動きを再現しようとすれば、回転する方向の数だけ用意しなくてはなりません。股関節など 3 方向に回転が可能な箇所は、モータは 3 個必要になります。人間の足の動きを再現しようとすれば、片脚で、股関節に 3 個、ひざに 1 個、足首に 3 個の計 7 個が必要になるでしょう。通常市販品の人型ロボットでは、モータの数を減らして片脚 5 個もしくは 6 個で構成しているようです。

また、上半身は片腕 4 個程度のものが多いようです。

図 9 にモータの配置の例を載せておきます。

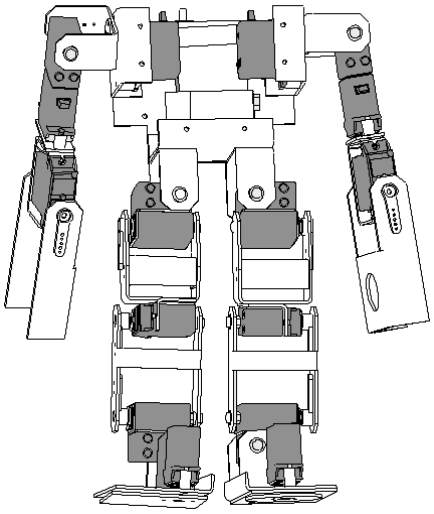


図 9: モータの配置例

このロボットでは、片脚で股関節に 2 個、ひざに 1 個、足首に 3 個モータを配置しています。腕は片方で 3 個もしくは 4 個のモータを使っています。

2.2 ワールド座標系

ロボットの各部の位置を指定するにあたり、座標系を定義する必要があります。通常、ロボットを直立させた状態で、胴体の中心から床に下ろした垂線の足を原点に取ります。

ロボットの前方を x 軸、左方向に y 軸、上方向に z 軸を取ります。

このような座標系をワールド座標系といいます (図 10)。

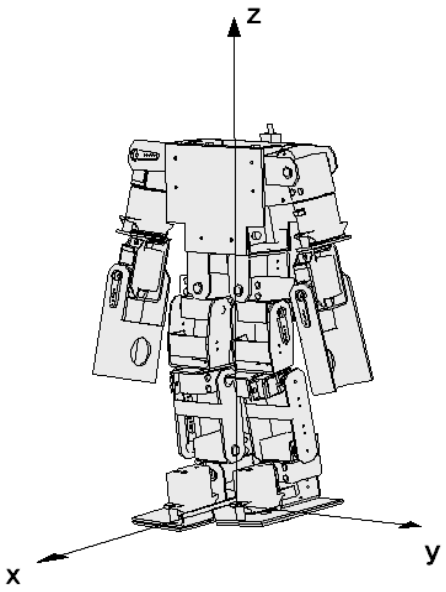


図 10: ワールド座標系

2.3 ロール、ピッチ、ヨー

x 軸周りの回転をロール (roll)、y 軸周りの回転をピッチ (pitch)、z 軸周りの回転をヨー (yaw) といいます (図 11)。

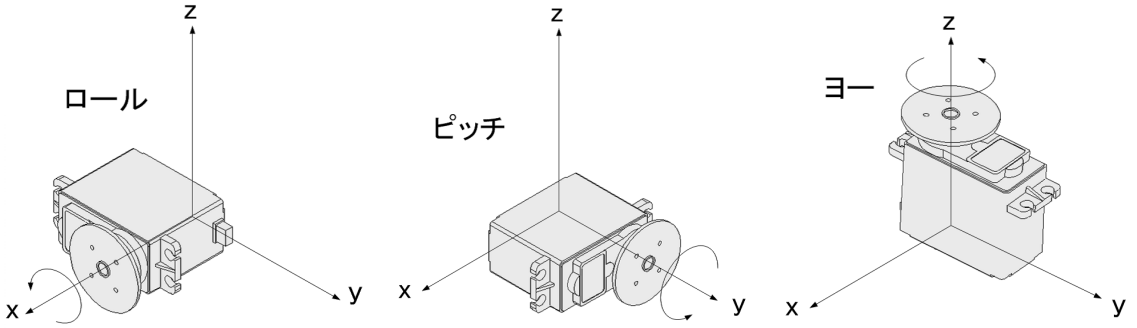


図 11: ロール、ピッチ、ヨー

図 10 のワールド座標系を考え、図 9 のモータの配置で言うと、左脚のモータはそれぞれ図 12 のようになります。

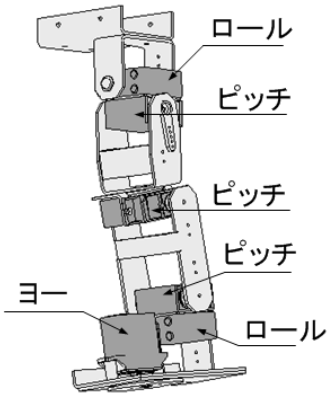


図 12: 片脚のロール、ピッチ、ヨー

この例では、ヨー軸が足首のところにありますが、股関節のところにヨー軸を持ってくることが多いようです。

モータの配置と座標系が分かったところで、足先や手先を目標位置に移動する方法についてみていきましょう。

3　運動学

ここでは、手先や足先を目標位置に移動するために必要な運動学について考えてみましょう。

3.1　順運動学・逆運動学

ロボットの関節位置や姿勢と、関節の角度の関係をつける理論を運動学 (kinematics) といいます。ロボットの各関節の角度から、関節などの位置を計算するのを順運動学 (forward kinematics) といいます。逆に、目標の関節位置から各関節の角度を求めることを逆運動学 (inverse kinematics) といいます (図 13 参照)。ロボットの手足を思った位置に移動させるには、逆運動学が必要になってくるわけです。逆運動学を解く場合には、目標位置によっては解が存在しなかったりしますので注意が必要です。

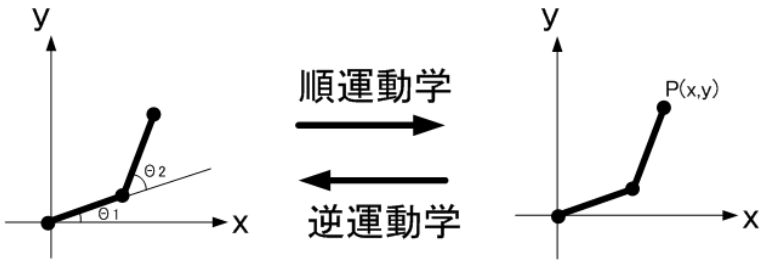


図 13: 順運動学・逆運動学

図 14 のに示す 2 自由度のロボットを用いて、順運動学と逆運動学を考えてみましょう。

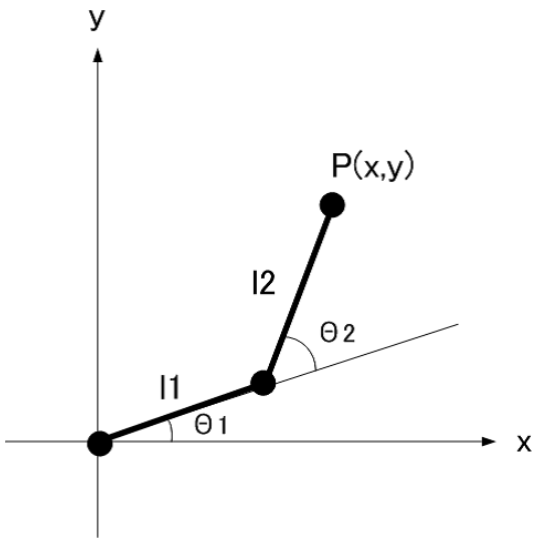


図 14: 2 自由度のロボット

順運動学は、二つの関節角度 (θ_1, θ_2) から手先の位置 $P(x, y)$ を求めるものです。これは以下のようになります。

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \tag{1}$$
$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \tag{2}$$

逆運動学は、手先の位置 $P(x, y)$ から二つの関節角度 (θ_1, θ_2) を求めるものです。これは余弦定理などを使って解くことができ、以下のようになります。

$$\theta_1 = \arctan(y/x) \pm \beta \tag{3}$$
$$\theta_2 = \pm(\pi - \alpha) \tag{4}$$
$$\alpha = \arccos(\frac{l_1^2 + l_2^2 - x^2 - y^2}{2l_1 l_2}) \tag{5}$$
$$\beta = \arccos(\frac{l_1^2 - l_2^2 + x^2 + y^2}{2l_1\sqrt{x^2 + y^2}}) \tag{6}$$

正と負の解が存在するのは、図 15 のに示すように、手先の位置 $P(x, y)$ を指定した場合、関節角度 (θ_1, θ_2) は 2 通りの解があるためです。

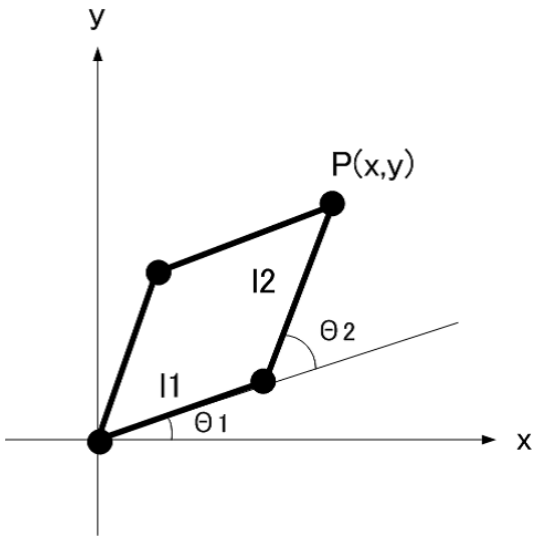


図 15: 逆運動学の 2 通りの解

このように順運動学の場合と違い、逆運動学の解は一つとは限りません。また、解が存在しない場合もあります。上の例では、原点からの距離が $x^2 + y^2 > l_1^2 + l_2^2$ であるような点 $P(x, y)$ を指定した場合には解が存在しないことが分かります。

解析的に解く方法は、自由度が多くなると複雑になり解を求めることが困難です。

このような場合には、Newton-Raphson 法を用いて関節の移動量を計算し、順運動学を用いて誤差を評価しながら収束させるという数値的解法を用いることで解を求めることが可能です (付録 A 参照)。

このように、逆運動学を用いれば足裏を思った位置に移動させることはできるわけです。

では、安定した歩行を実現するためには足裏をどの位置に持っていけば良いのでしょうか？それを考えるために、必要ないくつかの概念を説明し、歩行の場合の理論を見ていきましょう。

4　歩行

人型ロボットでは、歩行の制御は多くの研究成果が報告されています。ここでは、それらの一部を取り上げ概要を見ていくことにしましょう。

4.1　重心・支持多角形

重心とは、物体をそこで支えた場合に回転しない点のことです。

重心は物体の運動を考える上で非常に重要です。二足歩行ロボットの場合でも静歩行と呼ばれる歩行では、重心の位置によって歩行の安定性が評価できます。

支持多角形 (support polygon) とは、接地しているすべての点を囲った領域を言います。たとえば、二足歩行ロボットが片足で立っている場合には、足裏の接地面全体が支持多角形となるわけです。両足で立っている場合には図 16 に示す領域が支持多角形となります。図 16 左図で灰色の部分は足裏の接地面をあらわし、支持多角形はその足裏を直線で結んでできた多角形全体になります (図 16 右図)。

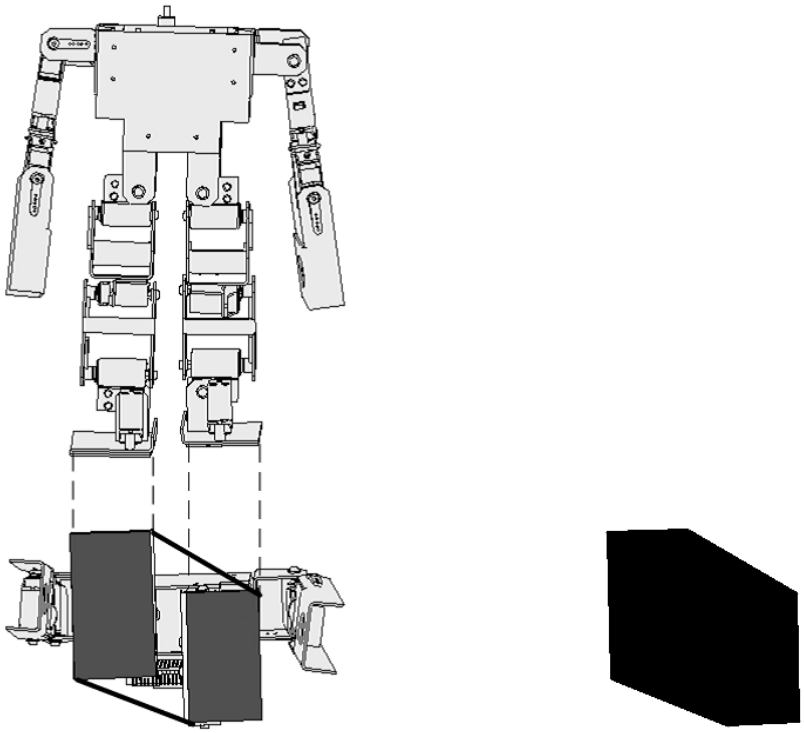


図 16: 二本足で立っている場合の支持多角形

4.2 静歩行と動歩行

重心の投影点が常に支持多角形の中にあるような歩行を、静歩行といいます。一方、重心の投影点が支持多角形の外に出ることがある歩行を、動歩行といいます。

静歩行は加速度の影響が小さい歩行であり、暗いところなどでそろりそろりと歩くイメージです。このような歩き方は、重心の投影点が支持多角形の内部にあればよいのですから、足裏の目標位置を比較的簡単に計算できます。反面、動歩行に比べてモータのトルクが必要であったり、ゆっくりとした歩きしか実現できないという難点があります。

われわれが通常行っている歩行は動歩行であり、加速度の影響を考慮した歩行になっています。この歩行では重心の投影点が支持多角形の外に出ることがあるので、重心の投影点では転倒の条件を考えることができないということになります。動歩行を安定させるための指標として、重心の投影点の代わりに ZMP(Zero Moment Point) という概念が導入されています。

4.3 静歩行の逆運動学

ここでは逆運動学を用いて、静歩行の動作を考えてみましょう。モータは図 17 の配置とします。

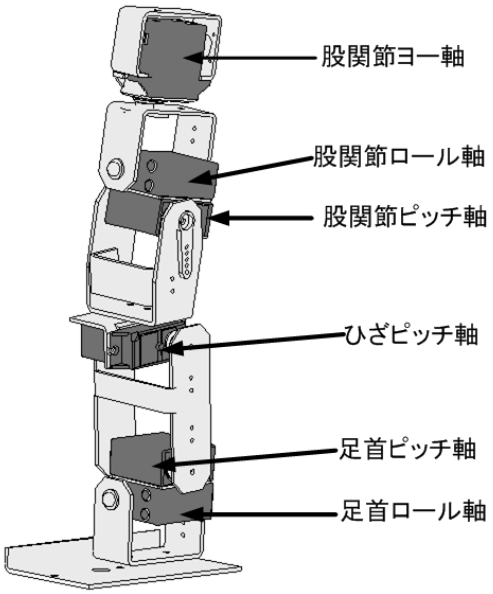


図 17: モータの配置

解析的に解けるように、各モータの役割を考えてみます。

股関節のヨー軸は主に旋回などの方向転換に利用します ($x-y$ 平面)。重心を横に移動するには、股関節のロール軸と足首のロール軸が使われます。このとき z 軸方向の座標も変化することに注意しましょう ($y-z$ 平面)。足を上げたり前後に出すには、股関節、ひざ、足首のピッチ軸を使います ($x-z$ 平面)。

このように分けて、座標 $P(x,y,z)$ と股関節のヨー軸の角度 ψ_{leg} が与えられたときの股関節のロール軸、

ピッチ軸の角度 ϕ_{leg}, θ_{leg}、ひざ関節のピッチ軸の角度 θ_{nee}、足首のロール軸、ピッチ軸の角度 $\phi_{foot}, \theta_{foot}$ を求めてみましょう。

ここでは計算がしやすいように片脚のみ考え、座標系として股関節のヨー軸の中心を原点とし、ロボットの前方に x 軸、左方向に y 軸、下方向に z 軸を取ることにします。また、片脚のみで考えてみましょう。関節の角度は、直立状態を 0 とし、回転は反時計回りを正とします。

股関節のヨー軸の角度を ψ_{leg} とすると、足の前方を新たな x' 軸、右方向を y' 軸とした座標 $P(x', y', z)$ は座標 $P(x, y, z)$ と以下の関係にあります。

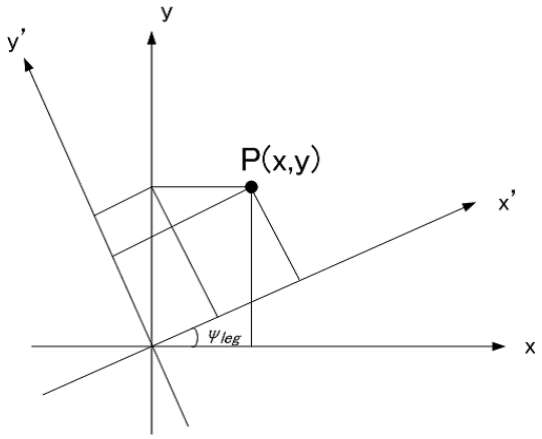


図 18: 座標 $P(x', y', z)$ と座標 $P(x, y, z)$ の関係

$$x' = x \cos \psi_{leg} + y \sin \psi_{leg} \tag{7}$$

$$y' = -x \sin \psi_{leg} + y \cos \psi_{leg} \tag{8}$$

$y' - z$ 平面で考えると、股関節のロール軸の角度 ϕ_{leg} は以下のように求められます (図 19 参照)。

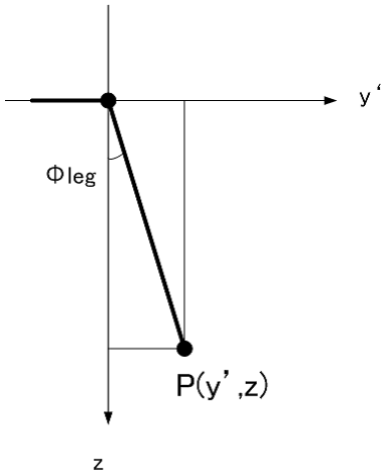


図 19: ϕ_{leg}

$$\phi_{leg} = \arctan(\frac{y'}{z}) \tag{9}$$

足首のロール軸は股関節の二つのロール軸をむすんだ線と足裏が平行になるようにするのなら、

$$\phi_{foot} = -\phi_{leg} \tag{10}$$

となります。

$x' - z$ 平面を考え、股関節のピッチ軸の角度 θ_{leg} と足首のピッチ軸の角度 θ_{nee} が求められます (図 20 参照)。式 (3)、(4)、(5)、(6) を求めたときと同様にして、

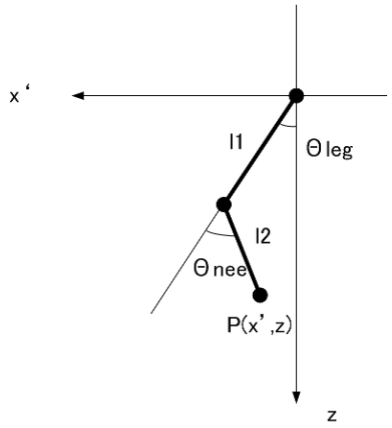


図 20: θ_{leg} と θ_{nee}

$$\theta_{leg} = \arctan(x'/z) \pm \beta \tag{11}$$

$$\theta_{nee} = \pm(\pi - \alpha) \tag{12}$$

$$\alpha = \arccos(\frac{l_1^2 + l_2^2 - x'^2 - z^2}{2l_1 l_2}) \tag{13}$$

$$\beta = \arccos(\frac{l_1^2 - l_2^2 + x'^2 + z^2}{2l_1\sqrt{x'^2 + z^2}}) \tag{14}$$

のように求まります。足裏を胴体と垂直にするためには、足首のピッチ軸の角度 θ_{foot} は、

$$\theta_{foot} = -\theta_{leg} - \theta_{nee} \tag{15}$$

とすればよいことが分かります。

モータを思った位置に動かせれば、静歩行は次のようにして作ることができます。

- 重心を片脚に乗せ
- 重心の乗っていないほうの足を上げて前に出し
- 足を下ろし
- 重心を両足の間に戻します

この動作を左右順番に行うことで歩行を行うわけです。

このとき注意すべきなのは、「足を下ろし」「重心を両足の間に戻す」動作です。片足で立っている間は支持多角形が片足の接地面ですが、両足がついた瞬間から支持多角形は両足の接地面を囲んだ多角形になります。この足を着く前の片足状態で、重心の投影点が支持多角形を出ないようにしなくてはなりません。また、重心を逆足に乗せかえる場合にも、その途中で重心の投影点が支持多角形を出ないようにしなくてはなりません。歩幅が大きくなればなるほどこれは難しくなります。

4.4 ZMP(Zero Moment Point)

静歩行では重心の投影点が支持多角形の中にあることが転倒しないための条件でした。しかし、動歩行では重心の投影点は支持多角形から出ても安定した歩行が可能になります。そのため重心の投影点を拡張し、加速度がある場合に歩行が安定するための条件を導く必要があります。このために導入されたのが ZMP というものです。

ZMP(Zero Moment Point) とは、接触面全体の加重を一点にまとめたとき、この力ベクトルが通過する作用点として定義されています (図 21 参照) 。

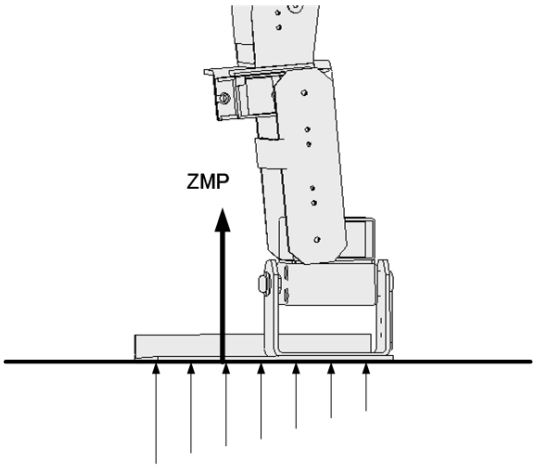


図 21: ZMP

これは、支持多角形の境界の内側の一点で支えたとき、前後方向や左右方向に傾かない点になっています。ZMP は常に支持多角形の中にあります。

ZMP が支持多角形の境界の内側にあれば、転倒しないことが知られています。ただし、この条件は十分条件であり、ZMP が支持多角形の境界上にあるからといって倒れるというわけではありません。

また、ZMP は加速度の無い直立の状態などでは、重心の投影点と一致します。

4.5 ZMP による重心移動の補正

ここでは、簡単なモデルを使って ZMP を計算し、加速度の効果を考慮した動きを作ってみましょう。

まず、全質量は重心位置に集中していて、他の部分は質量が無いものと仮定します。また、今回は横方向の加速だけを考えることにしましょう。

重心の横方向の座標を $x(t)$ とし、重心の高さを H とします。ゼロモーメントポイントの位置 x_{ZMP} は加速度の効果によって、

$$x_{ZMP} = x(t) - \frac{H}{g}\ddot{x}(t) \tag{16}$$

となります (図 22 参照)。

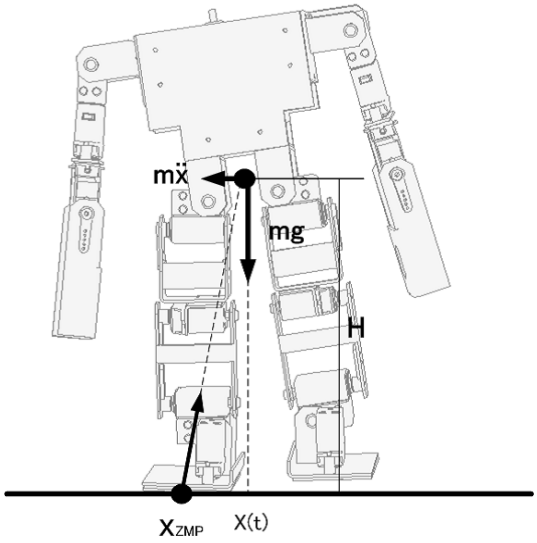


図 22: 加速度の効果

ここで $\ddot{x}(t)$ は $x(t)$ を時間で二回微分したものです。いま x_{ZMP} を支持脚の位置とし、重心 (の投影点) から足裏の中心までの距離を r とすると、重心の方程式は、

$$\ddot{x}(t) - \frac{g}{H}x(t) = -\frac{g}{H}r \tag{17}$$

となります。運動の周期が T の連続的なものになるように係数を決めれば、

$$x(t) = -\frac{r(\exp(\sqrt{\frac{g}{H}}t) + \exp(-\sqrt{\frac{g}{H}}t))}{\exp(\sqrt{\frac{g}{H}}\frac{T}{4}) + \exp(-\sqrt{\frac{g}{H}}\frac{T}{4})} + r \tag{18}$$

です。つまり、横方向については重心位置がこの方程式に従うようにすれば良いことになります。

重心が両足の中心にある状態 ($t = -T/4$) で、$T/4$ 後 ($t = 0$) に重心位置を

$$x_0 = \frac{-2.0r}{\exp(\sqrt{\frac{g}{H}}\frac{T}{4}) + \exp(-\sqrt{\frac{g}{H}}\frac{T}{4})} + r \tag{19}$$

に移動し、$T/4$ 後 ($t = T/4$) に足を切り替えることによって、任意の足幅 r、周期 T の体重移動になるわけです。

5　モーション作成

ここまでは、歩行動作を安定させるための理論についてみてきました。この章では、最近のロボットキットで利用されているモーション作成のためのソフトについてみておきましょう。

5.1　モーション作成ソフト

ここまでみてきたように、歩行というのは微妙なバランスを保つことで成り立っています。このバランス制御を行うために様々な理論が提唱されているわけです。これらの理論の多くは数値的な計算を必要をしており、実行するためにはかなりの CPU パワーを必要とします。また、歩行以外の一般的な動作についてこれらの理論を拡張することはけして簡単ではありません。

市販されている比較的安価な二足歩行ロボットでは、このような数値計算を行わず、モーション作成ソフトを使って動作を作ることが多いようです。

モーション作成ソフトの一例を図 23 に示します。

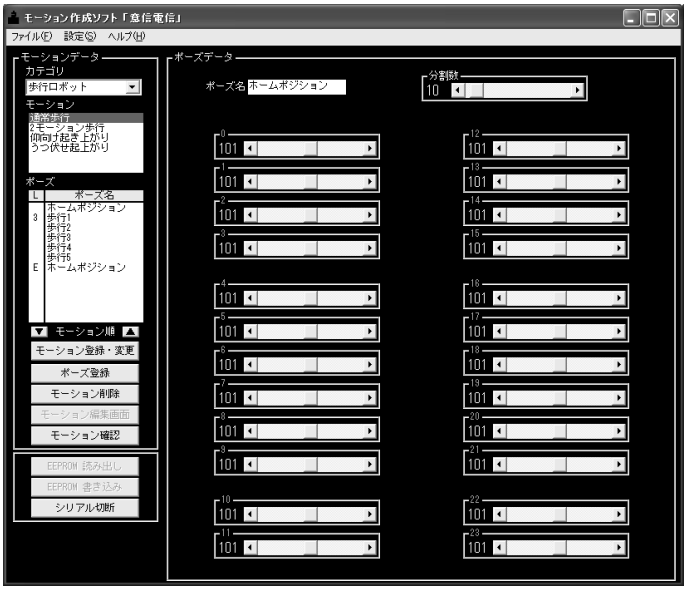


図 23: モーション作成ソフトの例

このソフトでは、スライダーを動かすことでモータを動かすことが可能です。全てのモータを思った位置に動かすことで「ポーズ」が出来上がります。この「ポーズ」と次の「ポーズ」との間をどのくらいの時間で補間するかを設定します。このようにして一連の「ポーズ」を次々と実行することで、動き「モーション」が実現するわけです。

これらのソフトでモーションを作る場合には、実際のロボットにポーズを取らせながら目で確認していきます。たとえば、静歩行をさせようとした場合には、途中のポーズを作っておき、間を計算で埋めていくことになります。ZMP が支持多角形の中にあるかは、目で見て確認することになります。動歩行の場合ですと、大体の「ポーズ」を作っておき、実行してみてうまくいかないところを修正していくことになるでしょう。

このような作り方は、直感的で扱いやすく、歩行以外の様々なモーションも難しい計算無しに作ることがで

きます。歩行などの動作を安定させようとしたら、ジャイロセンサなどのセンサを用ることができるでしょう。また、補間程度の計算しかしていませんので、CPU は比較的安価なものでも間に合います。

反面、生成したデータに汎用性が無く、歩幅を少し変更するだけでも全てのデータを作り直す必要が出てきたりします。一つの動作を作る場合でも試行錯誤を繰り返し、実際に動かしながら修正していくことになるでしょう。つまり、ロボットごとに、動作ごとにデータを作るという手間が必要です。こうして、出来上がったデータが最適なのかという評価の方法も主観的なものでしかありません。

運動学を用いる方法とモーション作成ソフトを用いる方法を場合によって使い分けることが現実的なのではないでしょうか。

付録A　逆運動学の数値的解法

ここでは、逆運動学の数値的解法について考えてみましょう。

A.1　Newton-Raphson 法

Newton-Raphson 法は、方程式 $f(x) = 0$ を、

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{20}$$

によって収束させていきます。これは、$f(x)$ が、ごく狭い範囲では 1 次式で近似できるという仮定の下に行っています。

図 24 で Newton-Raphson 法の意味をみてみましょう。

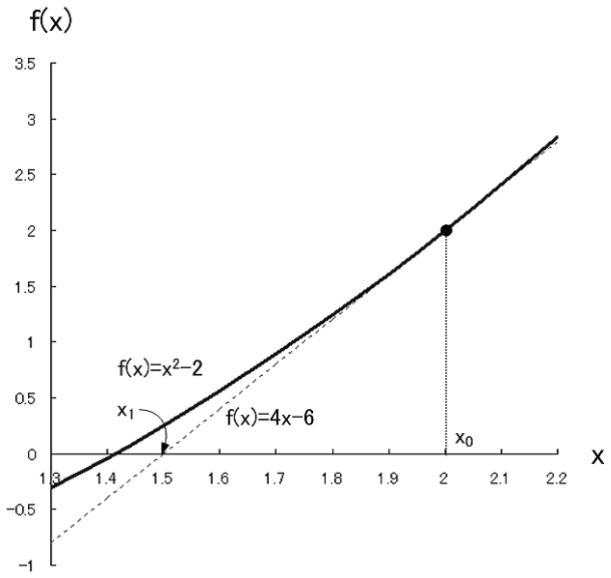


図 24: Newton-Raphson 法

- x_k のところで $y = f(x)$ に対する接線を引きます
- この接線が x 軸と交わる点を x_{k+1} とします
- $f(x_{k+1})$ があらかじめ設定しておいた値より小さくなったら終了です

たとえば $x^2 - 2 = 0$ を Newton-Raphson 法で解いてみましょう。$x_0 = 2.0$ からはじめて、$|f(x_k)| < 10^{-5}$ になるまで解くと、手計算でも解くことが可能で、$x_3 = 1.414215\cdots$ となります。

関数によっては $f'(x)$ が簡単には求まらないことがあります。このような場合には、x の値を微小量 dx 変化させて得られた $df(x_k)$ をもとに、$df(x_k) = f'(x_k)dx_k$ から求めることができます。

上の例で $dx = 0.01$ としてみると、$x_3 = 1.414226\cdots$ となることが分かります。

ここで注意すべきことは、極大点や極小点の近くでは $f'(x)$ が 0 に近い値になってしまうために値が収束せず、解が求まらなくなってしまうということです。二分法などの収束性は悪いけれど確実に解に向かう方法である程度まで収束させておいてから Newton-Raphson 法を用いるのが確実でしょう。

A.2 多関節の場合における逆運動学の数値的解法

上記の方法は、式 (1)、(2) を解く場合にも利用できます。現在の値を $\theta_1 = \theta_2 = \pi/6$、$l_1 = l_2 = 10$ とし、$x = 6, y = 8$ になるような θ_1, θ_2 を $|(x_n - x)^2 + (y_n - y)^2| < 10^{-5}$ という条件のもとで解いてみましょう。ここでは $d\theta_1 = d\theta_2 = 0.001$ としてみます。

$\theta_{10} = \theta_{20} = \pi/6 = 0.523599\cdots$ より、$x_0 = y_0 = 13.66025\cdots$ となります。この点における微少量 $d\theta_{10} = d\theta_{20}$ に対する x_0, y_0 の変化 dx_0, dy_0 は、

$$\begin{bmatrix} dx_0 \\ dy_0 \end{bmatrix} = \begin{bmatrix} -0.00866 & -8.66275 \\ 13.65342 & 4.995669 \end{bmatrix} \begin{bmatrix} d\theta_{10} \\ d\theta_{20} \end{bmatrix} \tag{21}$$

となります。逆行列をかけてやることによって、

$$\begin{bmatrix} d\theta_{10} \\ d\theta_{20} \end{bmatrix} = \begin{bmatrix} 0.042253 & 0.073269 \\ -0.11548 & -0.000073 \end{bmatrix} \begin{bmatrix} dx_0 \\ dy_0 \end{bmatrix} \tag{22}$$

が求まります。この変化が微小区間以外にも適用されるとして計算すれば、$\theta_{11} = 0.467104, \theta_{21} = 0.946183$ のようになり、$|(x_1 - x)^2 + (y_1 - y)^2| = 0.632702$ であることが分かります。

このようにして計算していくと、最終的には、$\theta_{13} = 0.534953, \theta_{23} = 0.895678$ のようになり、$|(x_3 - x)^2 + (y_3 - y)^2| = 3.78 \times 10^{-9}$ になることが分かります。

このように、自由度があがっても逆行列さえ求められれば計算は可能であることが分かります。逆行列を数値的に解く方法としては、Gauss-Jordan 法や LU 分解などが知られています。

参考文献

[1] 梶田秀司. 「ヒューマノイドロボット」. オーム社, 2005.

[2] 米田完, 坪内孝司, 大隅久. 「はじめてのロボット創造設計」. 講談社, 2001.

[3] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery. 「NUMERICAL RECIPES in C［日本語版］」. 技術評論社, 1993.

[4] いろいろ製作日記. http://isseisfactory.jugem.jp/.