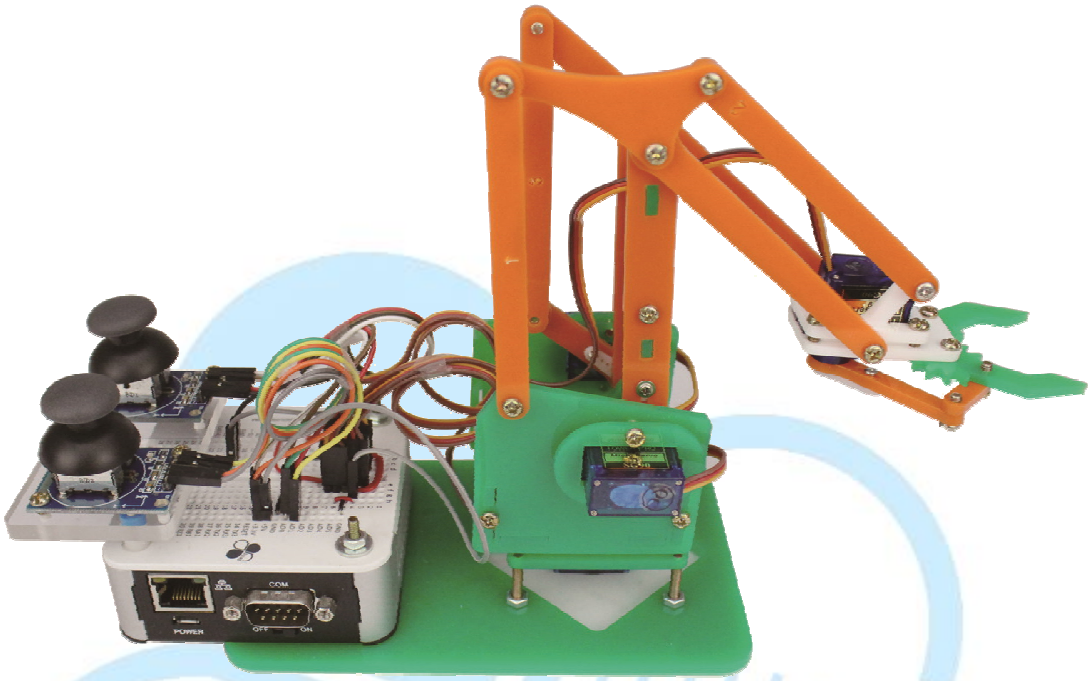
EduCake によるロボットアーム制御



一、 ロボットアームとは

図Ⅰの様なロボットアーム(RobotArm)は、各種工業生産の現場において広く持ちいられている。その用途は主に人手に代わり、煩雑な仕事(EX:自動車組み立て、製品箱詰め)や、重複性の高い仕事(EX:果物選別)、精密性の求められる仕事(EX:電気回路溶接)などである。これらの仕事を、決して我々人類が出来ないわけではないが、ロボットアームは 24 時間休むこともなく、それに見合う給与を求むこともなく、また、腹を立てることもなく高い水準を維持したまま、仕事に励むことが可能である。

その為、工業においてますます多くの企業が既にロボットアームを人手に代わって採用しているが、これにより大々的にコスト削減、生産効率及び良品質を向上させることが可能となった。これは既に産業における趨勢の向かうところであると言える。我々はこの章でロボットアームの構成及びそのコントロールについて述べたいと思う。

図 1. http://www.kuka-robotics.com/高名な Kuka ロボットアーム

二、 ロボットアームの基本構造

工業用ロボットアームの値段は数十～数百万台湾元以上である。構造は極めて複雑であり、大まかに大別すると、機械本体、フロント機能（エアブラシ、溶接、爪、吸盤、電磁石等異なる機能、ひいては人手を必要とするあるいは自動交換における機能が異なる等）、電源供給システム、PLC コントロールシステム、感測システム、コンピュータ監視/警報システム、UI コントロール等の部品により構成されている。これらの部品を理解するとなると、膨大な時間がかかってしまうので、我々はここではまず先にロボットアームについて基本的な事のみ、述べたいと思う。

作業者はロボットアームを使用する前にまず、基本的事項を学ぶ必要がある。訓練所で学ぶロボットアームはかなり以前より、ロボットアームの特徴を模倣し作成した、図2の様な様々な簡易版アームを用いて行われている。それは筆者が常用しているロボットアームでもあり、基本的な構造と容易に入手できるモーターを用い、ボードをコントロールすることにより構成されていおり、そのコストはおおよそ数千～一万台湾元である。

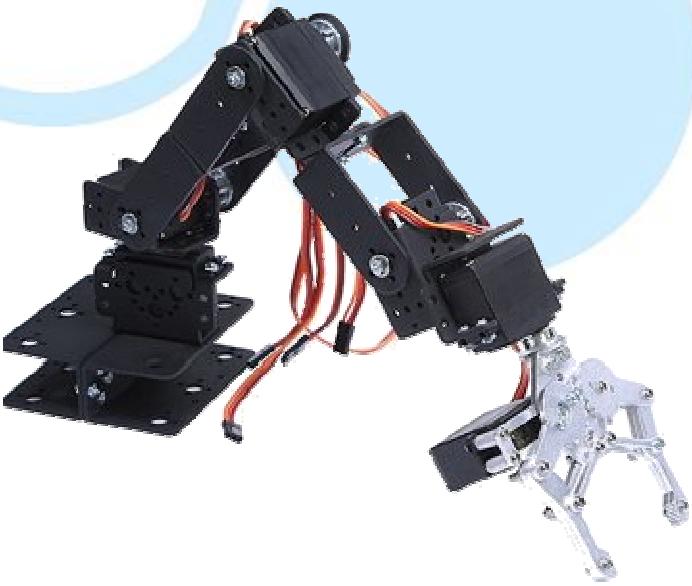


図 2. 一般的な教学用ロボットアームの一例

この様なロボットアームは約10数年前に考案され、この時は基本的な機能しか持ち合わせてはいなかったが、その多くが生産現場で用いられる中、絶え間なく改良が続けられ、様々なタイプのロボットアームが生まれていった。筆者が最も多く遊んだタイプは大体2～10軸であり、自身でも10軸あまりの様々なロボットアームを設計したことが有る。長年来基本的な教学方法は一貫して10年一日の如くであったが、2013年にようやく大いなる飛躍を遂げることとなる。まずはネットコンペの流行により、おもちゃ屋を改行したい人たちに機会が与えられ、それは大体2013年半ばのことであったが、uArmと呼ばれるロボットアームが出現し、同作品は2014年初頭にkickstarterにおいて空前の成功を収め、七百万台湾元を超える賞金を獲得した。
https://www.kickstarter.com/projects/ufactory/uarm-put-a-miniature-industrial-robot-arm-on-your。

この様なロボットアームの成功と、オープンソース設定（設計図、回路図、プログラムは全てネット上でシェアされている）により、コンペ終了後ネット上に様々な改良バージョンが出現した。例えば、playArm、liteArm、meArm、xxArm…などである。そのうちのmeArmこそが、我々がこの章で選んだ対象である。作者は関連するデータを全てhttp://www.thingiverse.com/thing:360108に公開することにしたが、ここでのロボットアームの外観はおおそよそ図3の如くであり、様々な異なる材質やカラーが使用されている為、それぞれが独特の風格を呈している。

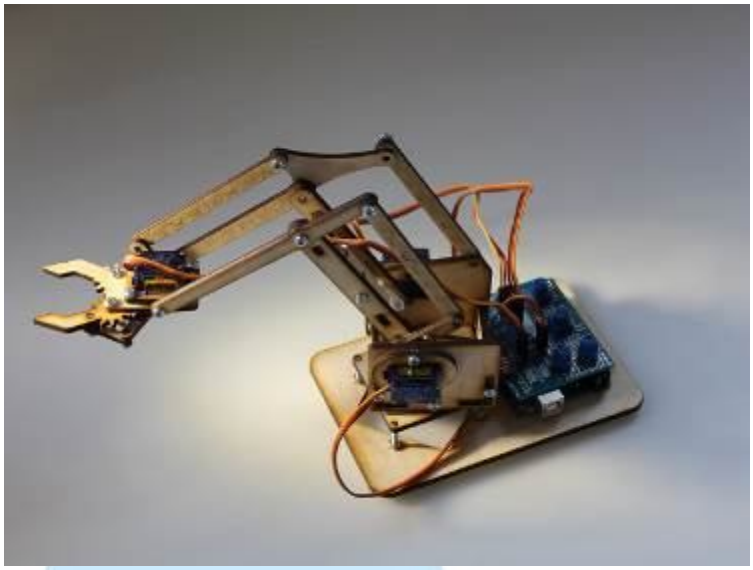


図 3. 2014 年最も人気の高かった教学用ロボットアーム。

これは材料に木材を使用したタイプ。

なぜ、このロボットアームが好評を博したのだろう？なぜならその構造は非常に単純（明らかに uArm に比べ簡略化が進んでおり、組み立てにはわずかに 30 分ほどしかかからない）で、工業ロボットアームの機能原理が容易に表現されており、非常に安価で、関連する全てのデータが作者によりネット上に公開されたからである。一昔前は一台当たり二万台湾元したこのに対し、このたった二千台湾元であったため、この安価なロボットアームは瞬く間に普及してゆき、教育の現場の注目と支持を獲得していったのである。我々はこのアームを用いて、いくつかのテーマについてお話していきたいと思う。

三、 ロボットアームの構造

meArm タイプのロボットアームの主要構造は、メカニズム、モーター、電子回路ボード、電気供給システム、コントロールプログラム等から構成される。ネット上のシステムは既に関連する各種バージョンが人々にシェアされており、それに基づき厚さ 3 ㎜程の合金や木材等からなる各種材料を使用し組み立てることが可能となっている。また、図 4 の様な厚さ 2~5mm のタイプも存在する。

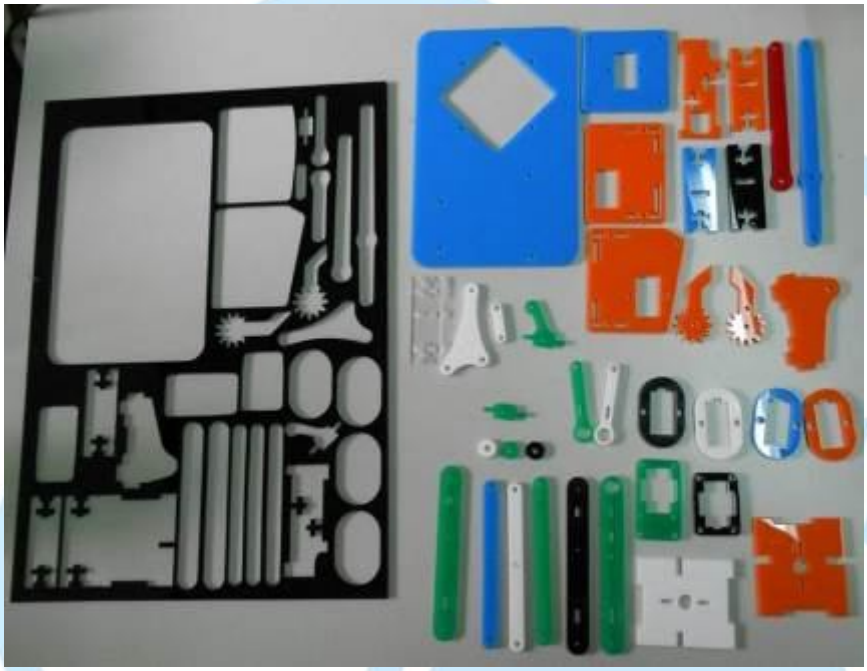


図 4. 必要とされる全てのパーツ。各パーツごとに切り取り済み。

ここで使用する材料の厚さは全て 3 ㎜に統一しなくてはならず、さもなければ設計を変更する必要が出てくる。これはつまり、もし、厚さが先に挙げた 3 ㎜より薄ければ、組み立て後にぐらつきが生じ、厚ければ、そもそも組み立てることが出来ないからである。カッターなどの工具で、ネジ穴を広げる方法もあるが、この種のやり方はロボットアームに設計に重大な瑕疵を生じさせ、使用する材料自体にも重大な問題を生じさせてしまう。

モーターは主に SG90 を使用し、このモーターは先の PWM の章ですでに紹介した、制御方式を使用しており、安価で入手しやすい。

しかしながら筆者はこのモーターに対して多くの問題点が有ると考えている。機能部分についてはとりあえずは置いておくとしても、モーターの外殻の許容誤差範囲が大きすぎ、千個のモーターを買ってきたとすると、外殻のサイズの誤差は大体 1 ㎜に達し、これは構造設計上、非常に大きな影響を及ぼす。穴も小さすぎたりし、万が一、これに当ってしまったとしたら、そもそもモーターの外殻を取り付けることすらできない。反対に、穴が大きすぎ、規格よりモーターの外殻が小さかったとしたら、取り付け後に締まりのない感じを見るものに与えるだろう。そしてモーターの軸部分に誤差が有ったとしたら、アームの運用動作が安定しない。

国外の設計者はこの問題を発見し、解決方法を見出し、無理にでもこの問題を処理しようとしたが、しかしながら、このやり方では、組み立て時に細心の注意を払い、調整しなくてはならない。この章では主にロボットアームの入門について紹介しようと考えているので、とりあえずはこの問題は置いておきこのモーターを使用することとして、後面で精密制御についてお話しする時に、改めて類似のサイズのモーターを使用して、再度設計を行おう。これはロボットアームの運用上の安定と精密性に大きく貢献し、もし、このモーターを使用して実作を行いたいのであれば、http://www.roboard.com/servo_0263.html を参照されたい。速度はより早く、馬力も大きく、精密度も明らかに向上し、さらには金属により作成されている。

また、更には EduCake の 13bit の PWM は運用時にその真価を発揮し、制御もより精密なものとなる。ここでも SG90 と言う初級モーターを使用したいと思う。

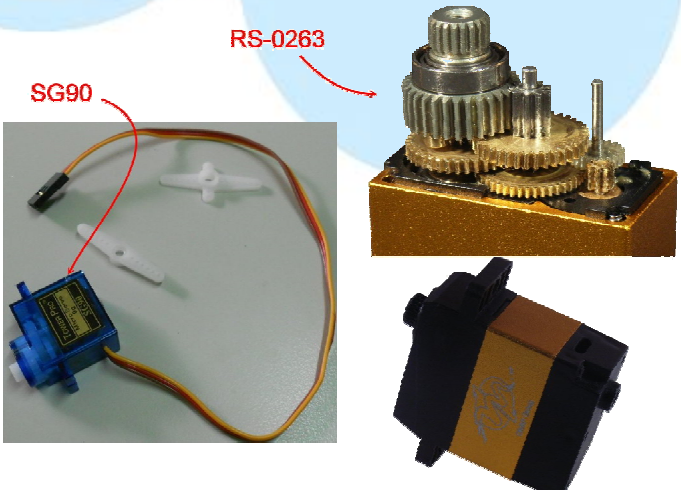


図 5. SG90 とより精密で安定した RS-0263

電気回路ボードを制御するには、やはり弊社の 86duino EduCake を使用しよう。しかしながら使用者がより簡単に操作できるように、ジョイスティック等を使用するための周辺モジュールを、図 6 の通りとしよう。

図 6.前後左右とボタンが制御可能なジョイスティックモジュール

この種のモジュールは使用が大変容易で、主に 5 個のピンからなり、Vcc(5V)、GND、X、Y、button に区分される。その中の X、Y は analog ピンに接続され、ジョイスティックが前後左右に動く数値を読み取る。その他の内部構造は 10K オームの VR に過ぎず、一般的にその数値は 0~1023 である。或いは EduCake 特有の analogReadResolution()を使用し、11bit 解析度か、0~2047 の数値を獲得する指令を発する。

しかしながらこの種のモジュールは、通常生産されている部品に誤差が有った場合、ひずみが生じてしまう。数値も予測した 512 から 470、或いは 600 というかけ離れたものとなってしまう。その為、使用する前にまずテストを行い、その数値が決定範囲内にはたして置かれているのかを確かめる必要がでてくる。テスト用に仮設定された数値は 500 であり、EX:470～530 は動作範囲外とする。この様にすることでアームの誤作動を防ぐことが可能であり、非常に大切である。

当然、その後私たちは、より複雑な演算方法を書いてこの問題を処理することが可能であり、button は小型のボタンであり、ジョイスティックモジュール内部に組み込まれているが、押しづらく、やや硬く、その上ジョイスティックの真下に直接組み込まれているため、押したとき XY 数値が影響を受けることは免れず、アームの揺れに連動してしまうため、通常、私たちはこのボタン組み込みを採用するこは稀である。

最後に、これら一切を組み込んだものは図 7 の様になり、組み込み工程は先に述べたネット上にシェアされた文章をご参考願いたい。ここには詳細なステップが掲載されており、その中の arduino はそのまま EduCake に使用しても、何ら問題はないが（ただし、ピンの位置は確認する必要がある）、当然、ここで私たちは自らより効果の高いものを書きたいと思う。

図 7 組込み完了後のロボットアームの状態

組み込み完了後のアームは、後部褐色部分のモーターが回転すると、アーム前半部分の爪が連動し、図 8 の様に動作し、その動作において、この種の平衡機構の特徴は、アーム後部が如何なる運動をしたにせよ、前半部の爪、つまり図中の緑色の部分だが、それの形状に変更を加えられることはない。ただプログラミングの際、注意が必要なだけである。

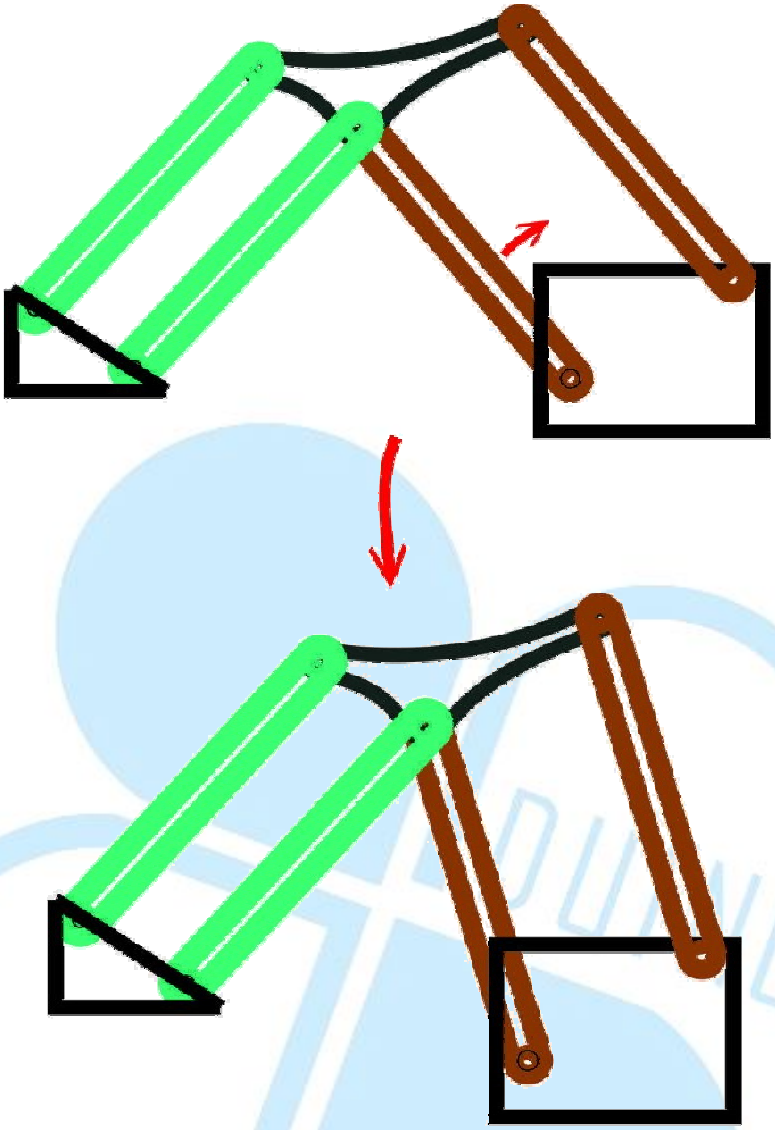


図 8 アーム後部の作動状態図

図 9 はアーム全部の動作状態図であり、図中の赤色の部分のモーターと連動し、爪の上下が制御される。

これはさほど難しいものではなく、直接、モーターを図中の赤色の部分に取り付けるというものである。しかしながら、この種の設計は主に工業用であり、一般的に出力を大きくするため、モーターは大きく重いもので、もしジョイント部分に取り付けると、アームを水平にした際に、アーム後部のモーターが非常に大きな負担となり、使用上の問題が生じてしまう。

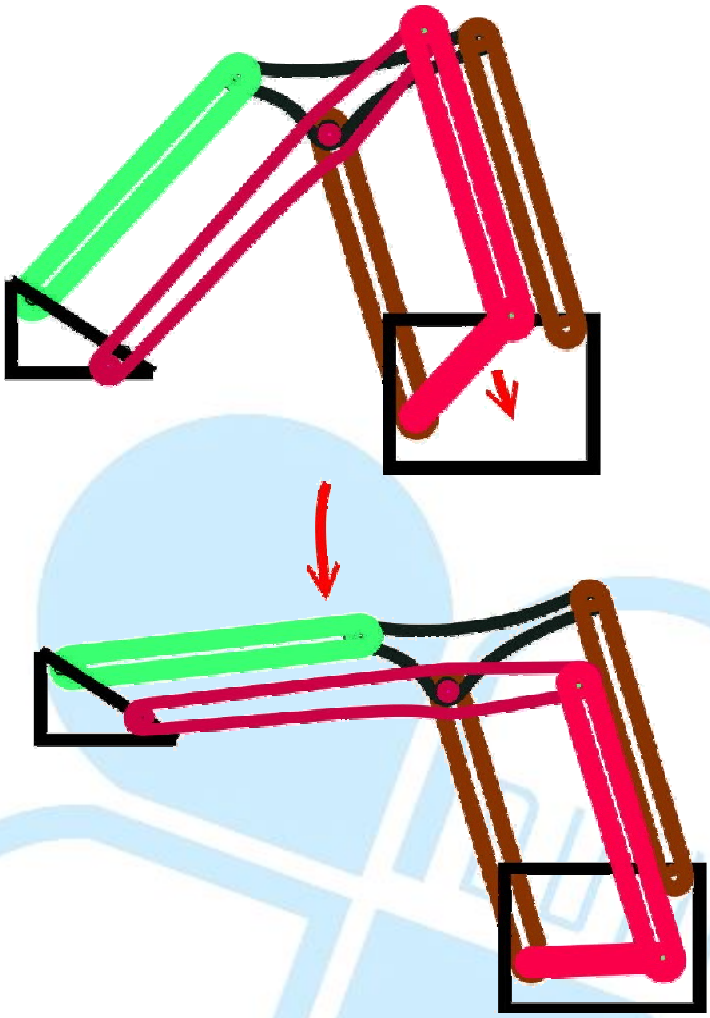


図 9 アーム前部の動作状況図

図 10 は爪の部分のモーターが、オリジナルの設計は連接棒に連動して爪が動し、物をつまむ。この設計は先に挙げた図 2 のロボットアームに使用されている平行タイプの爪に比べると、その長所は体積が比較的小さく、そしてつまめる範囲がやや大きいという点である。

しかし、これは連桿システムを使用し連動により爪を動かしているが、いくつか、設計上の重大な瑕疵が有る。第一はボルトの穴に問題が有り、固定後の爪が依然として明らかにぐらつくため、爪の部分の動作を精密に比べることが出来ない。実際、テストでは大体モーターは 20° に角度を変更しているが、爪の部分は開いた時最大の状態になるようにしてあったものを、完全にぴたりと閉じた状態になるように変更し、平行タイプの爪の場合と同様の状況下において、モーターは角度 100° 余りの時完全に開ききることが可能となるが、精密な度数と比べてみると、幾倍もの

差異が生じる。その為、meArm の設計は主に物品をつまむことが出来る、もしくは開く動作が出来るという事に焦点があてられているので、使用する際は細心の注意でつまむとき絞りすぎないよう、あまり長くつまんでいないようにすることが必要である。さもないと、モーターに長時間高い電流が流れ過ぎ、焼けてしまう(モーター本体には保護システム無し)

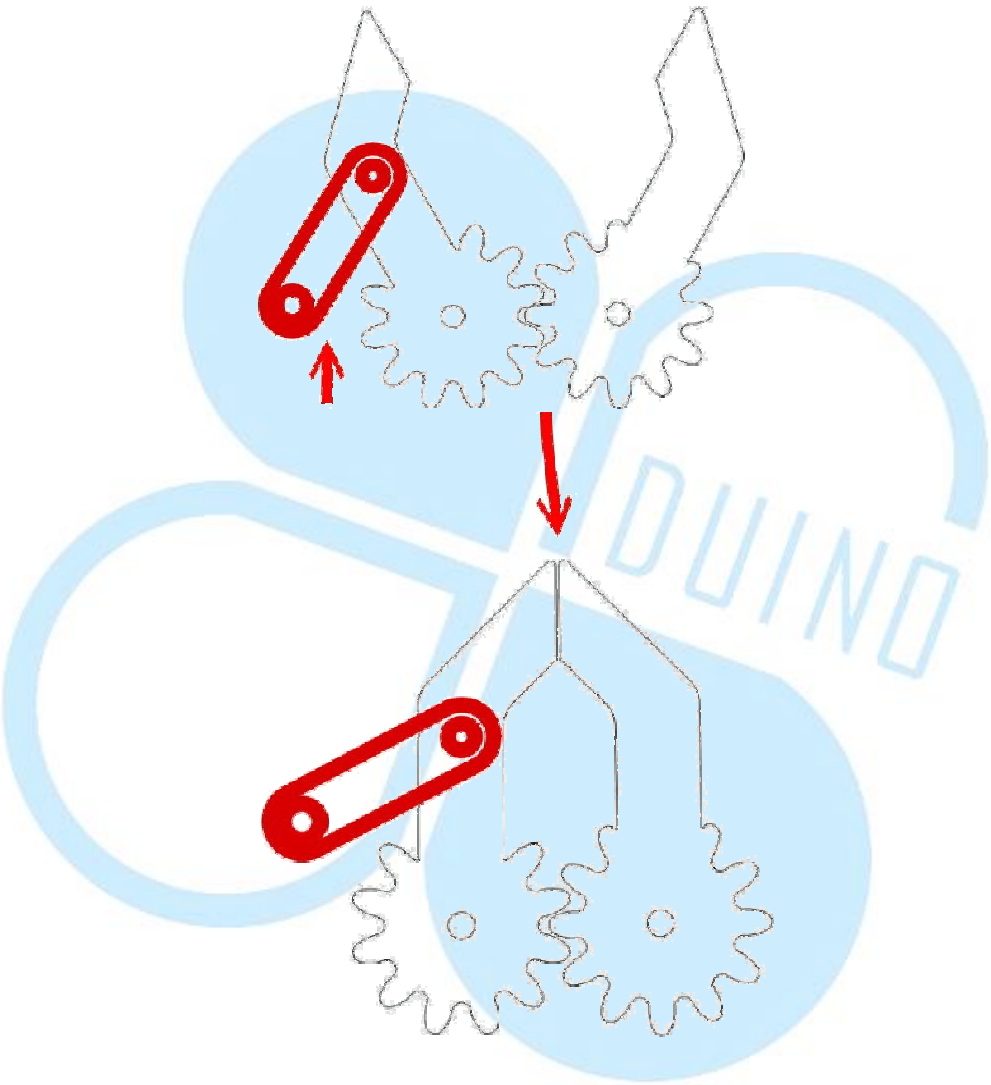


図 10 爪の動作状況

以上に挙げたいくつかの図により、ロボットアームのジョイント部分の基本動作状況が、簡単に理解できたことと思う。この部分の共同執筆者は、喜びながら、直接アイスキャンディー程の長さで細い形をしたものを用いて、毎回の運動状況及び構造上のシステムが、アームに及ぼす影響をテストした。多少の修正を加えたのち、その効果はなかなかのものであった。

四、ロボットアームの基本的コントロール

先に紹介した大体の構造に加えて、基本的な四軸構造を備えたロボットアームにおいては、その中の最も重要な部分は、モーターの角度の定位置である。

なぜなら、SERVO は中間で左右に運動する範囲を表すのみであり、SG90 は大よそ正負角度 80° の範囲内において比較的安全であり、システムの間の干渉問題についても、考慮する必要がある。その為、軸を取り付けるときには、紙若しくは図を書いて運動範囲をまずイメージしてみて、その後、モーターを図 11 の様に設置したほうがよい。図中のステップ 1 は、この運動範囲と中点はどこの位置なのか確認することであり、ステップ 2 は、まず簡単なモーターで回路図（PWM の章をご参考願いたい）をテストし、このモーター1500us の位置を中間におくことであり、ステップ 3 はこのシステムをモーター上に組み込み、その過程でモーターを取り外しできないようにしっかりと固定することである。

この動作を書学者に説明するのは比較的困難である。なぜなら、中点の位置は水平方向あるいは垂直方向に異動する可能性が有り、正確な位置をつかむことは困難である。その為、ロボットアームの設計は必ずこのことを考慮し、システムをモーターの中心に置くようにし、容易に直接垂直もしくは水平に取り付けるようにしなくてはならない。

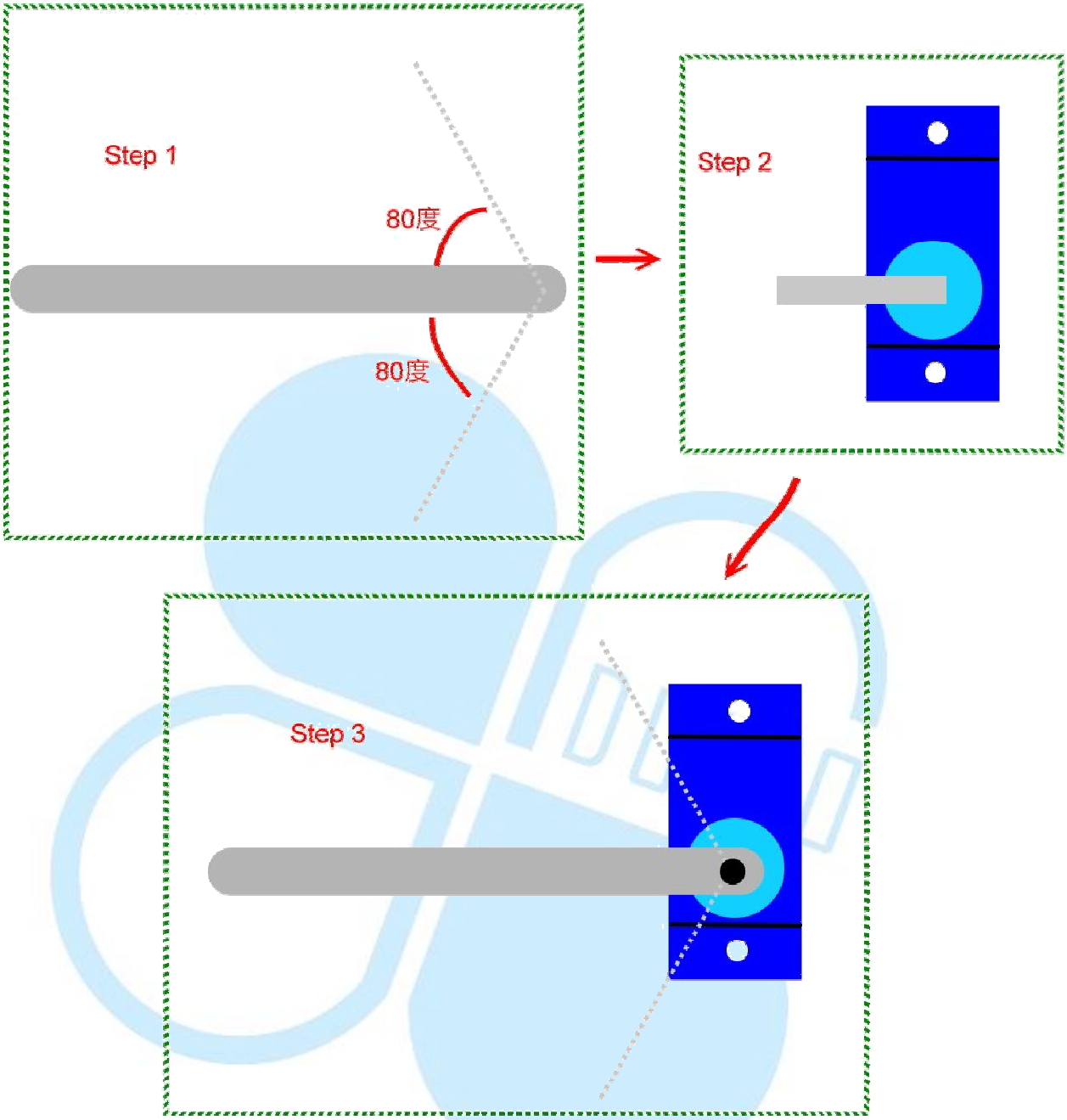


図 11. モーターの取り付け方法

モーターを中点に位置させるプログラムは簡単であり、以下のようなコードを入力すればよい。もしくは、出来合いのモーターを調節し、回路に取り付ければ、完了である。通常、この動作を除いて、モーターと機器をつなげた後、モーターを左右にゆっくりと極限まで揺らし、異なる角度でモーターが別の機器に影響いしたり、或いは、ボルトが飛び出ないかチェックすること（これは solidwork 等のソフトェアにおいて、シュミレーション可能）

もしくは、VR を EduCake に取り付けて、VR を読み取ることで、モーターの位置を

知ることもできる。モニターに出力し、モーター運動の左右の極限値を記録することで、続いてプログラミングを入力する時、用いることが可能だ。

簡単なテストプログラミングは以下の通り:

```
#include <Servo.h>
Servo myservo;
void setup()
{
   myservo.attach(3); // digital ピン 3 に接続
}
void loop()
{
    myservo.write(90); // 90 度の位置に移動
   // 或いは writeMicroseconds(1500)を使用; 1500us への移動は更に良い選択である
}
#include <Servo.h>
Servo myservo;
void setup()
{
   myservo.attach(3); // digital ピン 3 に接続
Serial.begin(9600);
}
void loop()
{
int a;
   a=analogRead(0); // VR のピンを analog ピン 0 につなぐ
a=map(a,0,1023,0,180); // 読み取った VR 値を 0~180°角度とする
    // ここで注意するのは、起動前に必ず VR の位置を大よそ中心の位置にすること
   // もし VR の位置が両端なら、起動後モーターが食い違う可能性有り
   // モーター損傷の可能性有り
```

```
    // モーター損傷の可能性有り
myservo.write(a); // 90 度の位置に移動
    // 或いは writeMicroseconds(1500)を使用し; 1500us 移動するのがより良い選択である
Serial.println(a); // モニターに表示・目下の位置を記録 }
```

正しい位置に設置後、回路を接続し、アームを図 12 の様に、その中の一つのモーターをベース回転とし、図中の 1 のモーターをアームの全運動用とする:図中の 2 のモーターは前後運動用:図中の 3 のモーターは物品を挟む為のものとする。

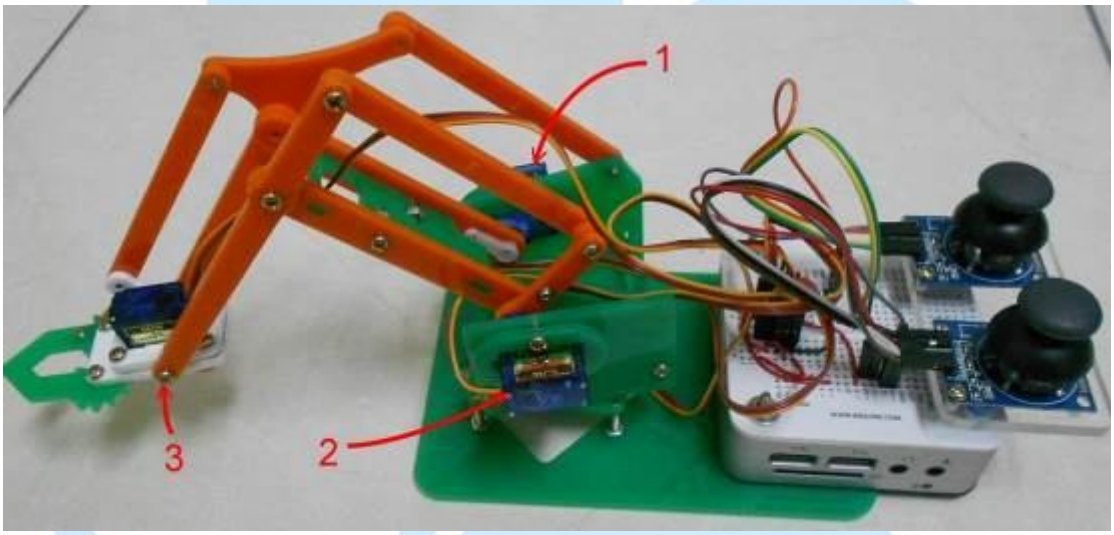


図 12．アームの電気回路図接続後の様子

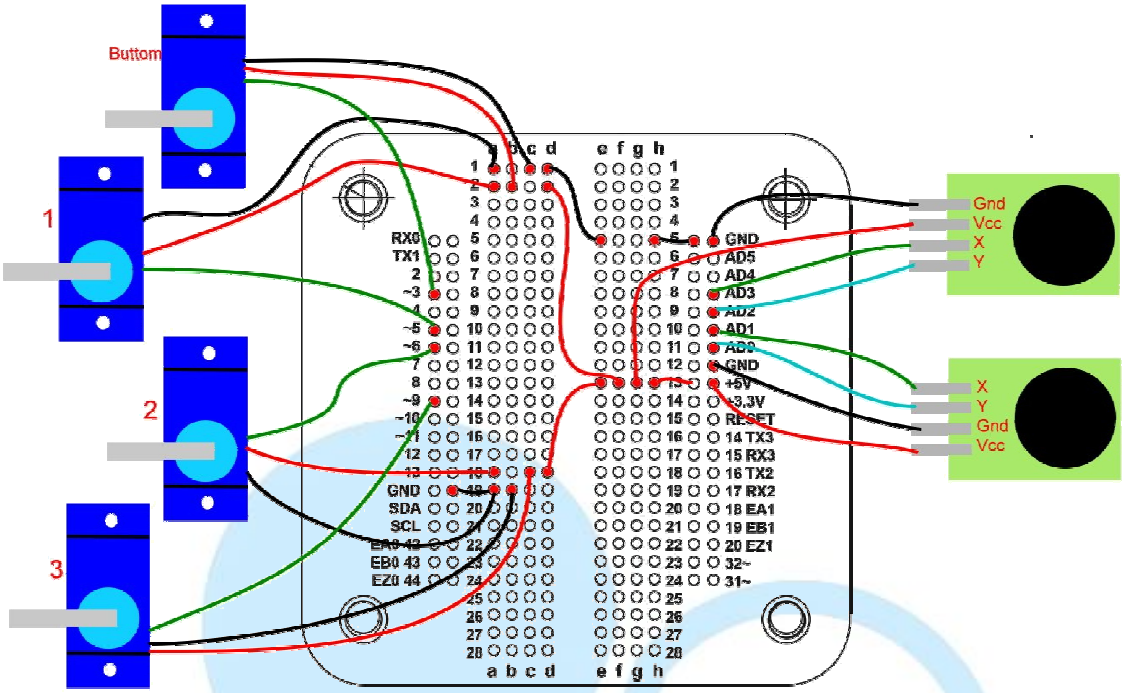


图 13.配線図

図 13 は配線図であり，モーターは digital ピンの 3、5、6、9 四個の PWM 出力ピンを使用し，二つのジョイスティックの XY を analog ピン 0~3，に分け、アームの左右、前後の動きと、挟むなど四個の機能を別々に制御させる。

五、 ロボットアーム全体の制御について

二つのロボットアームの組み立てと、モーターの設置が完了したら、続いて全てのプログラミングを完了させよう。

```
#include <Servo.h>

Servo myservo1; //  シャシ回転モーター
Servo myservo2; //  アーム後部昇降モーター
Servo myservo3; //  アーム前部モーター
Servo myservo4; //  アーム爪部分モーター

void setup()
{
Serial.begin(9600);
  myservo1.attach(3);
  myservo2.attach(5);
  myservo3.attach(6);
  myservo4.attach(9);
}

void loop()
{
inta,b,c,d,e;
  // abcd 四個の変数からまずジョイスティックの数読み取り
  //  並びに前項のテスト方式によりモーターの制御値読み取り
  // ジョイスティックごとのモジュール状況がことなる為・以下の数値を参考にすること
  a=analogRead(0);// ジョイスティック 1、左から右を 1023~0 とし，中点を 497 とする
  b=analogRead(1);// ジョイスティック 1、前後を 0~1023 とし,中点を 508 とする
  c=analogRead(2);// ジョイスティック 2、前後を 0~1023 とし,中点を 490 とする
```

```
 d=analogRead(3);// ジョイスティック 2 、左右を 1023~0 とし・中点を 500 とする
Serial.print(a); // ウィンドウに出力してみる
Serial.print(",");
Serial.print(b);
Serial.print(",");
Serial.print(c);
Serial.print(",");
Serial.println(d);

  // シャシ制御・設置後・その運動角度を 20~160 度とする
  a=map(1023-a,0,1023,20,160);
  // 直接転換角度を servo1 と入力することで・シャシ転換の制御可能
myservo1.write(a);
Serial.print(a);
Serial.print(",");

 // アーム後部制御・運動角度を 90~150 度とする
b=map(1023-b,0,1023,95,150);
Serial.print(b);
Serial.print(",");
  myservo2.write(b);

 // アーム前部制御・運動角度を 40~140 度とする
c=map(1023-c,0,1023,40,140);
Serial.print(c);
Serial.print(",");
  myservo3.write(c);

  // 爪の制御・開閉時の運動角度を 70~90 度とする
d=map(1023-d,0,1023,70,90);
Serial.print(d);
```

シャシの転換をその両辺の中央値を 70 度前後とする。前後アームを設置し，機器状況が制限されているため，転換時の角度がやや特殊である；ハサミ部分が更に小さくなり，わずかに 70~90 度角の間である。この数字はモーターやジョイスティックモジュールを交換したためにより，結果が異なってくるため、テストしなおす必要がある。

別に myservo.write();という指令を使用し、その内部のパラメーターに角度値を用いること。ただし、この角度値は実際のところ、ただの参考に過ぎず、異なる種類のモーターはその仕様が異なる為、”度”という単位ではなく、EX:が SG90 を用いて 120 度とすると、その中央から左へ回転したときの角度が 30 度となる(90 度- - > 120 度)が、同様の指令を MG995 というモーターに出した場合、回転の幅は 26 度に過ぎず、モーターが 60 度に回転した場合は値が大きすぎる為、テストをし直す必要がある。

もしこの myservo.write();の指令が不正確であったら、myservo. writeMicroseconds ();を使用すること。パルスを変更して制御するが、ここで必要となるモーターの精度は、これで十分であり、SG90 はこの指令を必要としない。理由は、これ自身があまり正確とは言えないからである。

この指令は角度制御、パルス制御の為のものであり、実際、内部の実作はパルスを転換してモーターを制御し、両者の最大範囲は 0~180 度、大よそ 500~2500us(或いは 700 ~2300us)であるが、これは参考に過ぎず、モーターの中には、20~160 度が 800~2200us に相当するものもあり、もしくは 35~145 度が 1000~2000us、或いは 25~155 が 900~2100us など、変化の幅は広い。

通常モーターは使用される際、まず説明と制限をチェックする必要がある。モーターの中には、指定の範囲を超えると食い違いが生じたり、焼けたりすることが有る為、注意が必要である。

六、 終わり

ロボットアームは素晴らしく楽しいアイテムであり、遊んだり、教学或いは工業／生活において様々に応用されている。ここでは、ロボットアーム教学文章を用いているが、他にも様々な応用、例えば、オートマチックの箱詰め、オートマチックの配列、オートマチックの運搬などである。これら実作について、後程また改めてお話ししたいと思う。