ロボット博士養成講座

ロボティクスプロフェッサーコース

六脚（ろっきゃく）ロボット①

第2回

六脚（ろっきゃく）ロボットの組み立て（後編）

講師用

目　　次

○　授業開始にあたって

授業のはじめは、着席させ、大きな声であいさつしてから始めます。

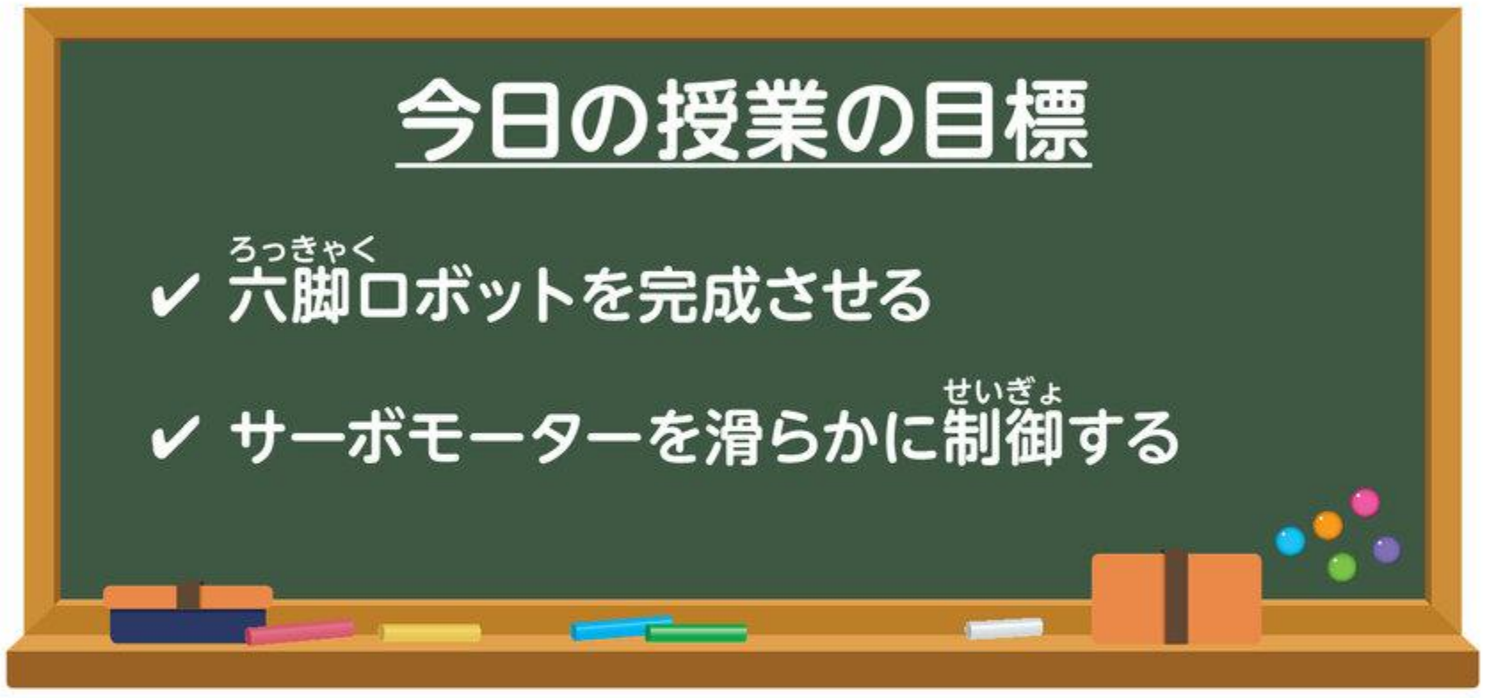
○　今回の目標をパネルで用意するか、黒板に予め書いておきます。

（授業の目標を明確化することは大変重要なことですので、生徒によく理解させます）

目安時間は授業時間 120 分のうち、休憩 10 分程度取ることを想定しています。
生徒の進捗状況により、休憩時間などを調整して授業を行ってください。

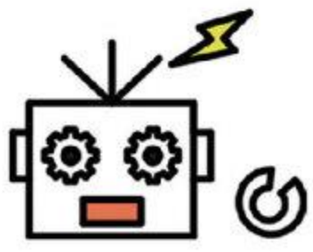
0. 六脚ロボットの組み立て（後編）（目安５分）

0.0. 「六脚ロボットの組み立て（後編）」でやること



今回は、六脚ロボットの組み立ての後編です。前回、サーボモーターの原点調整をして、個々の部分にいろいろな部品を取り付けましたが、今回は、これらを連結して六脚ロボットを完成させます。複数のサーボモーターを同時に動かすため、思わぬ動きで指を挟まれたりしてケガをすることがあります。六脚ロボットを持つときは背中側を持ち、脚やツノの近くを持たないように注意しましょう。
六脚ロボットをリアルに動かすためには、サーボモーターの調整は欠かせないポイントですので、こちらのチェックもしていきましょう。

図 0-0　六脚ロボットのイメージ



6本の脚で歩行するには、どうしたらイイのかな～?

0.1. 必要なもの

第1回めの「必要なもの」をチェックしましょう。

なお、以下は六脚ロボットの完成体です。前回組み立てをした前脚、後脚などと中脚、ツノをボディに組み付けていきましょう。

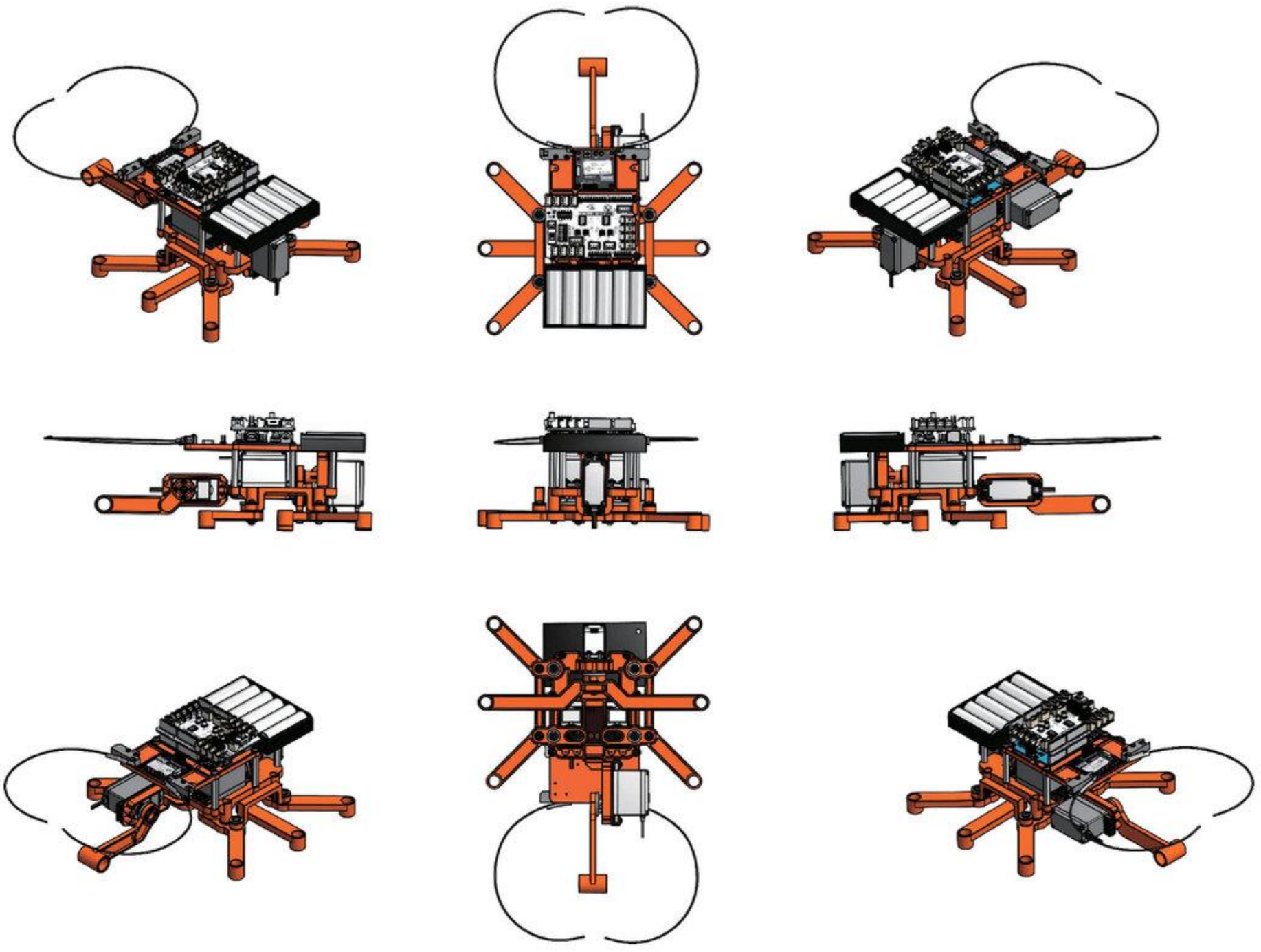


図 0-1　六脚ロボットの完成図

1. 六脚（ろっきゃく）ロボットを完成させる（目安 60 分）

1.0. 組み立てと配線

前回は、サーボモーターとパーツを組み立てて、ロボットのボディ、前脚（あし）、中脚（あし）、後脚（あし）、ツノになる部分をつくりました。第1回の組み立てが途中の人は、続きから作業を進めてください。

＜組み立て手順①＞

A-1、A-2パーツに組み付けたサーボモーターの出力軸（じく）に、B-1パーツ（×3）を取り付けます。
まず、サーボモーターの出力軸（じく）が原点になっていることを再確認してください。
組み立てる前に、プログラム「ServoTest0」を実行して調整しておきましょう。
原点調整が確認できたら、B-1パーツのツマミが**図1-0**の図の向きになるよう取り付けます。

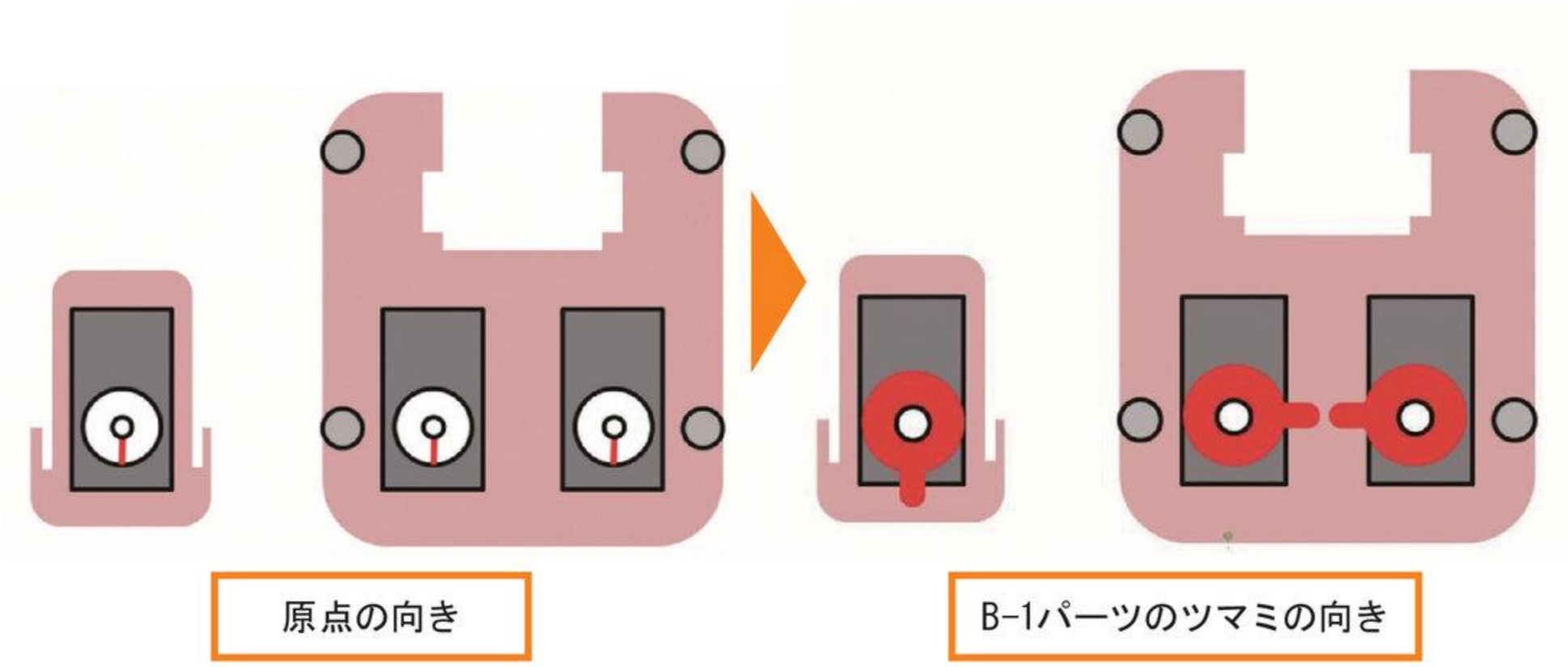


図 1-0　サーボモーターの原点の方向とB-1パーツの取り付け方向

講　A-1パーツにはB-1パーツのツマミ部分が向き合いになるように取り付けますが、サーボホーンとモーターの出力軸のはめ合いで多少のずれが出ます。僅かなずれは、プログラムで調整します。

＜組み立て手順②＞

＜組み立て手順①＞でB-1パーツをA-1パーツのサーボモーターに取り付けたら、M3L6ネジ（×2）をサーボモーターの出力軸に組み付けて固定します。同じように、A-2パーツに取り付けたB-1パーツもM3L6ネジでサーボモーターの出力軸に組み付けて固定します。

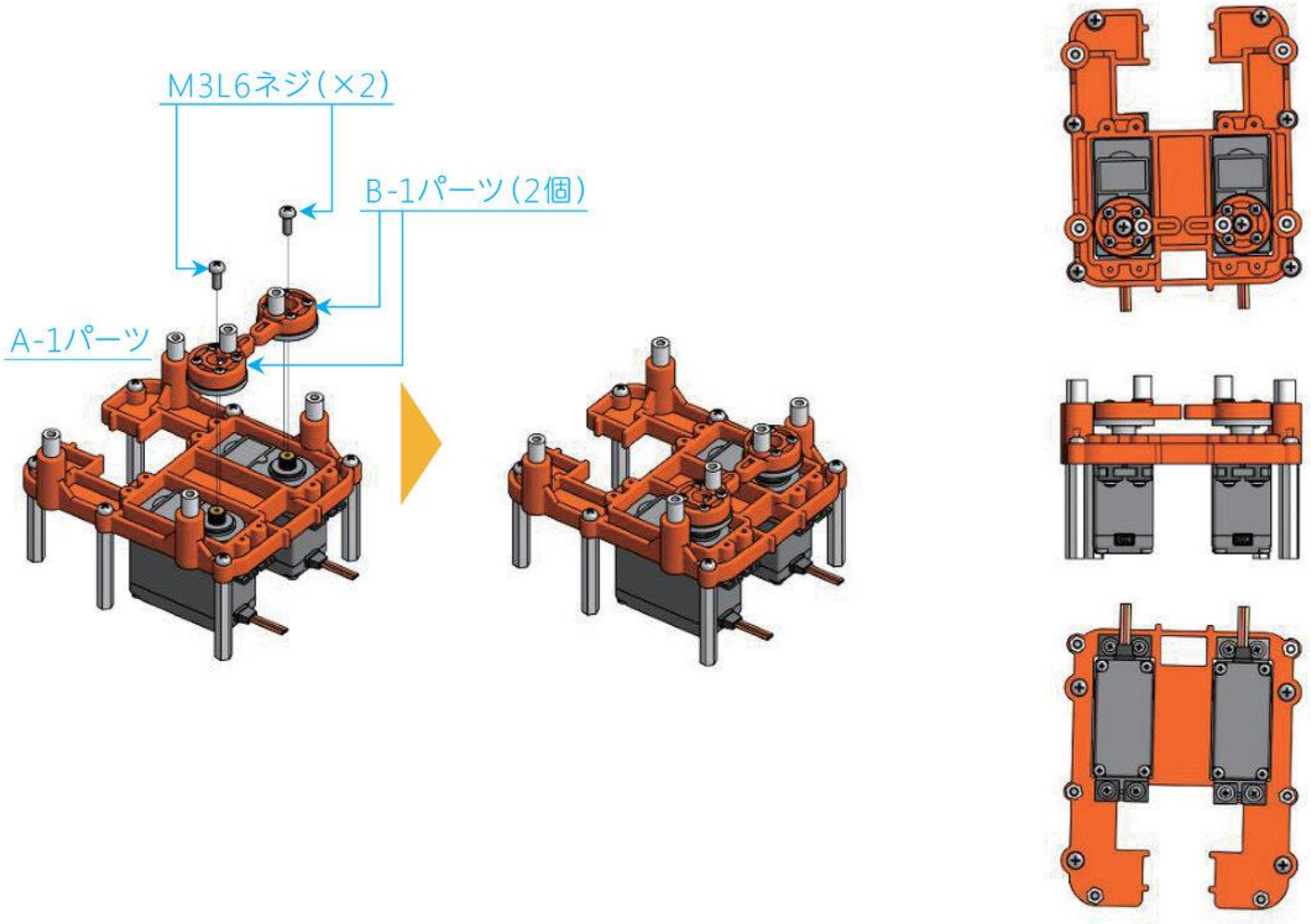


図 1-1　B-1パーツの取り付け①

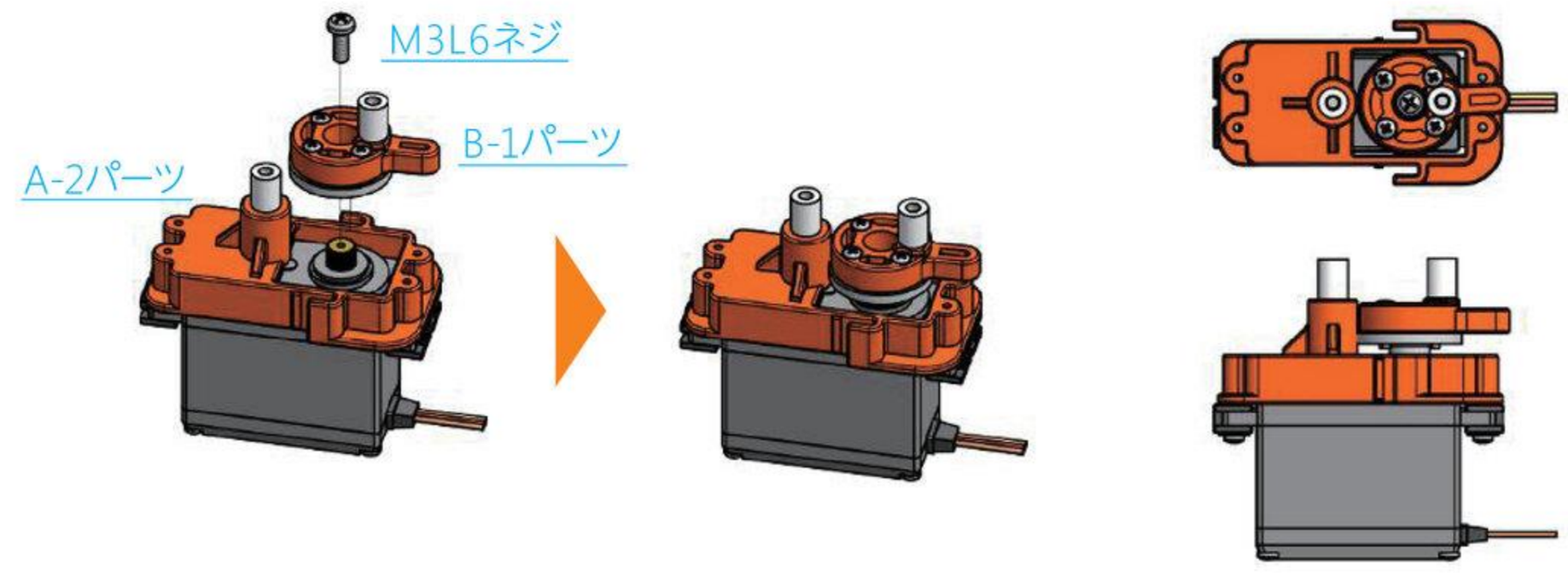


図 1-2　B-1パーツの取り付け②

＜組み立て手順③＞

B-1パーツを取り付けたA-2パーツに、前回**図1-3**のように組み立てた中脚(あし)部分を組み付けます。B-1パーツの8mm丸スペーサーにポリスライダーワッシャーを取り付け、その上からオイレブッシュ（×2）を取り付けたC-6パーツを取り付け、C-6パーツの楕円形(だえんけい)の穴の上からポリスライダーワッシャーを取り付け、M3L4ワッシャー付ネジ（×2）を組み付け固定します。

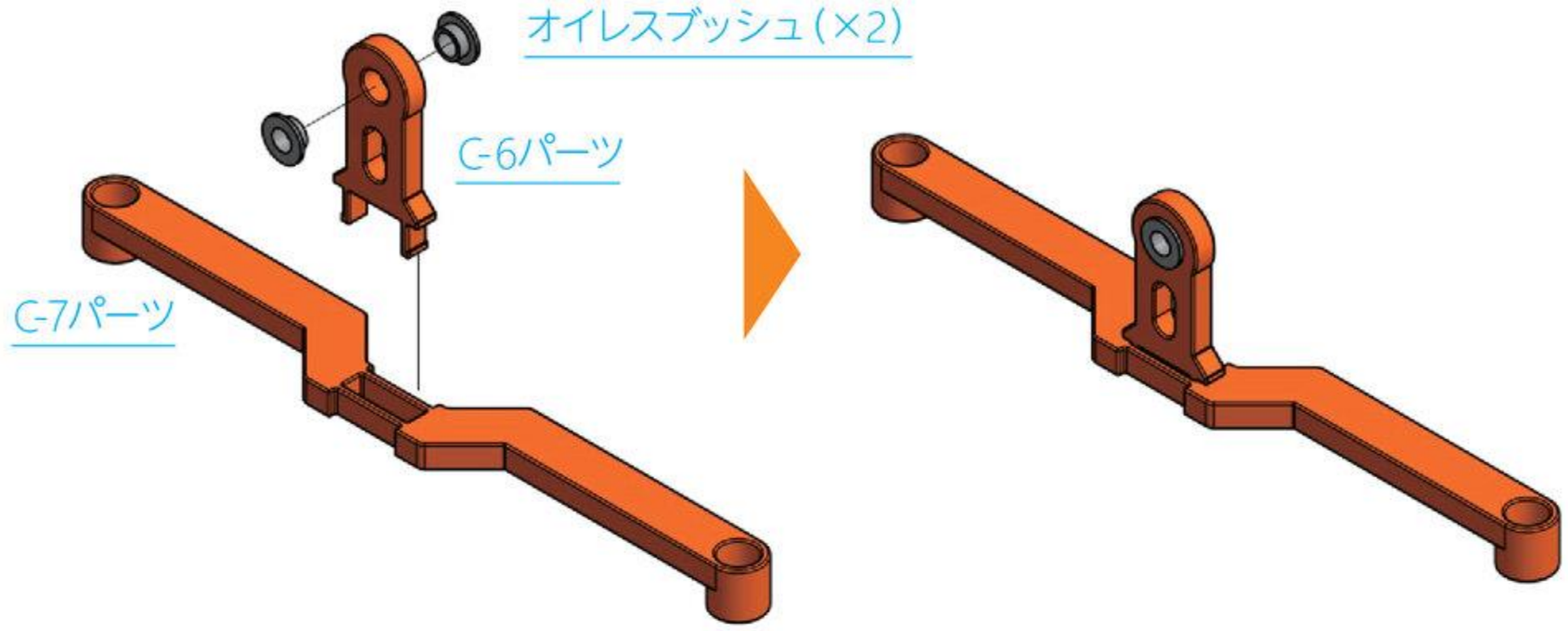


図 1-3　C-6パーツとC-7パーツを連結した図（中脚(あし)）

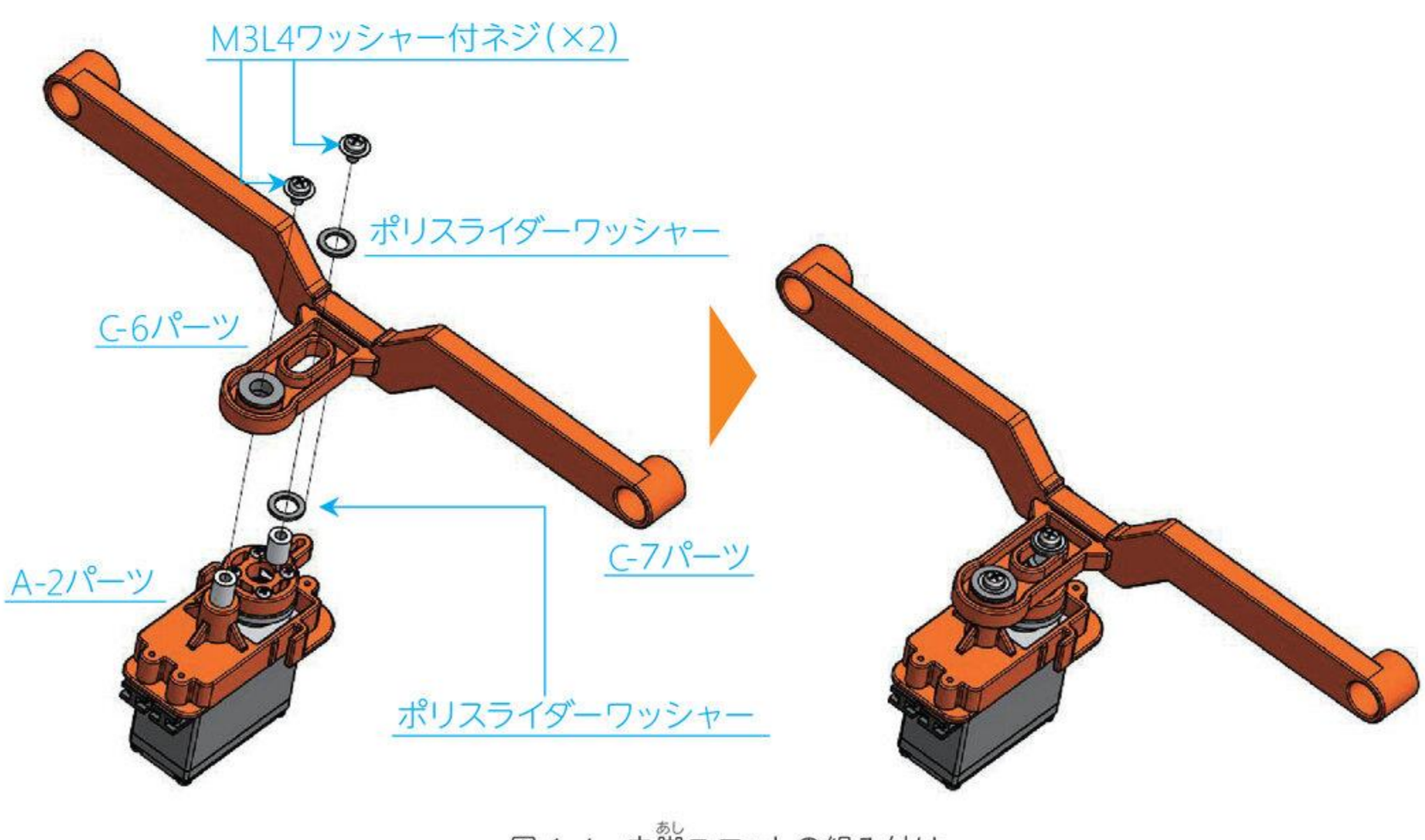


図 1-4　中脚(あし)ユニットの組み付け

＜組み立て手順④＞

A-1パーツのユニットとC-6、C-7パーツのユニットを連結します。**図1-5**のように、サーボモーターを組み付けたA-2パーツ部分をA-1パーツにはめ込みます。
さらにC-7パーツを上向きにして、B-1パーツに組み付けた8mm丸スペーサー（×2）にポリスライダーワッシャー（×2）を取り付けます。

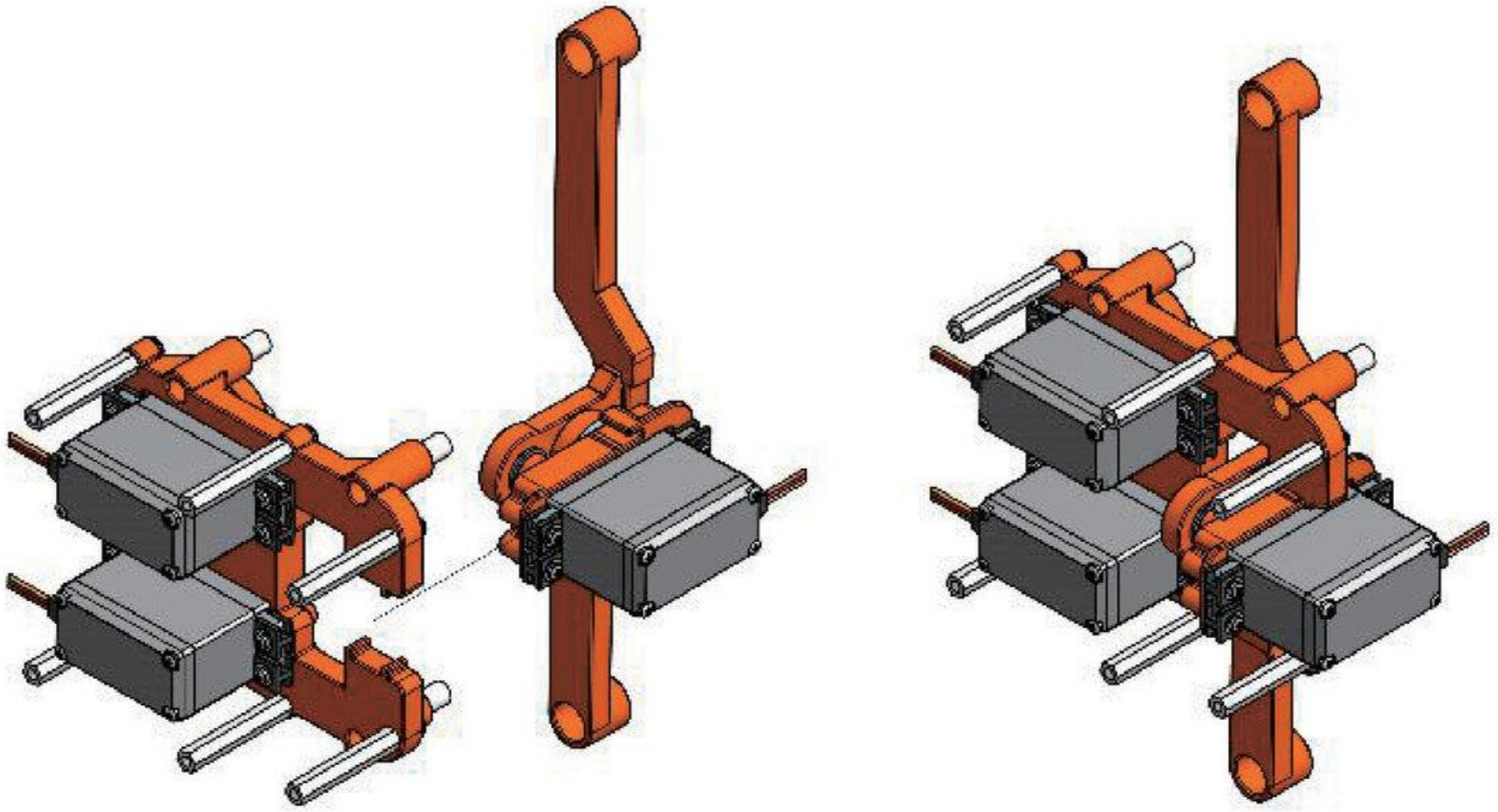


図 1-5　A-1ユニット（ボディ）とC-6、C-7ユニット（中脚）の取り付け

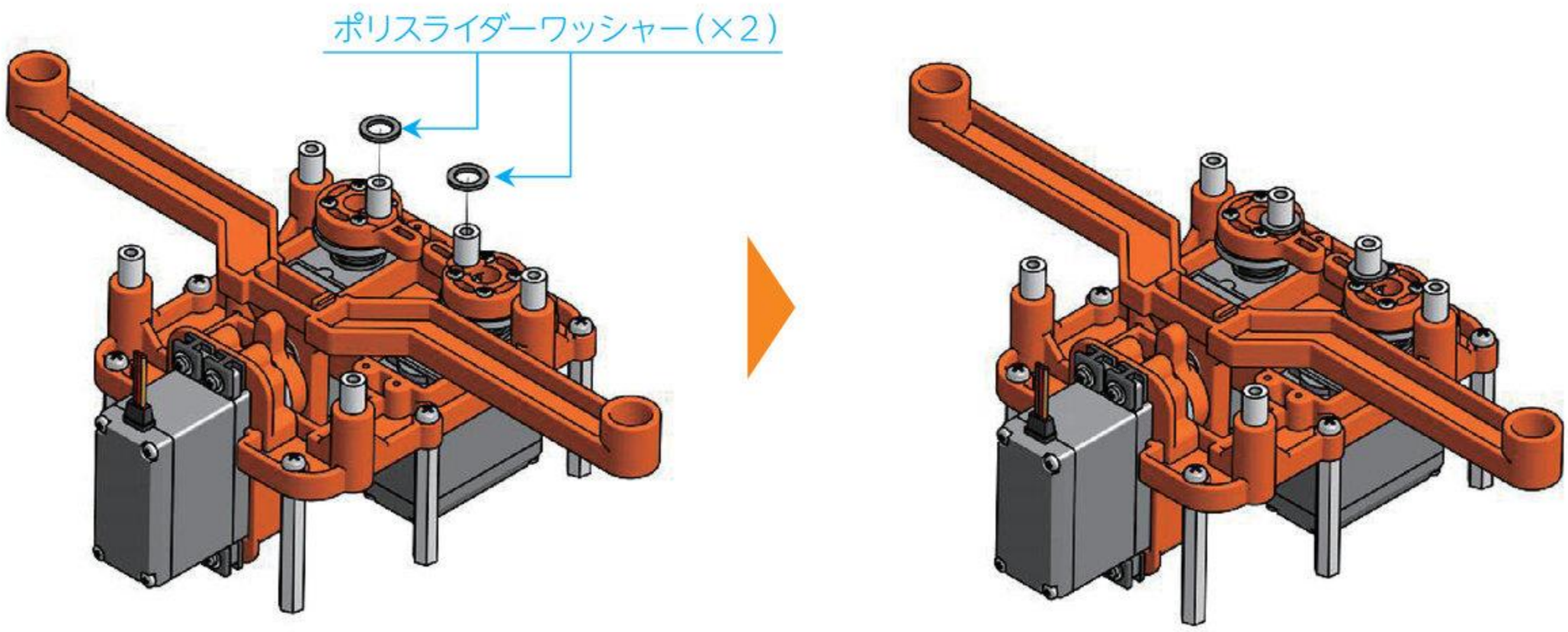


図 1-6　ポリスライダーワッシャーの取り付け

＜組み立て手順⑤＞

C-1、C-2、C-3、C-4パーツをM3L4ワッシャー付ネジ（×6）で図のように8mm丸スペーサーに組み付けます。C-1、C-2パーツの楕円形の穴にはポリスライダーワッシャー（×2）を取り付けてからM3L4ワッシャー付ネジを組み付けます。

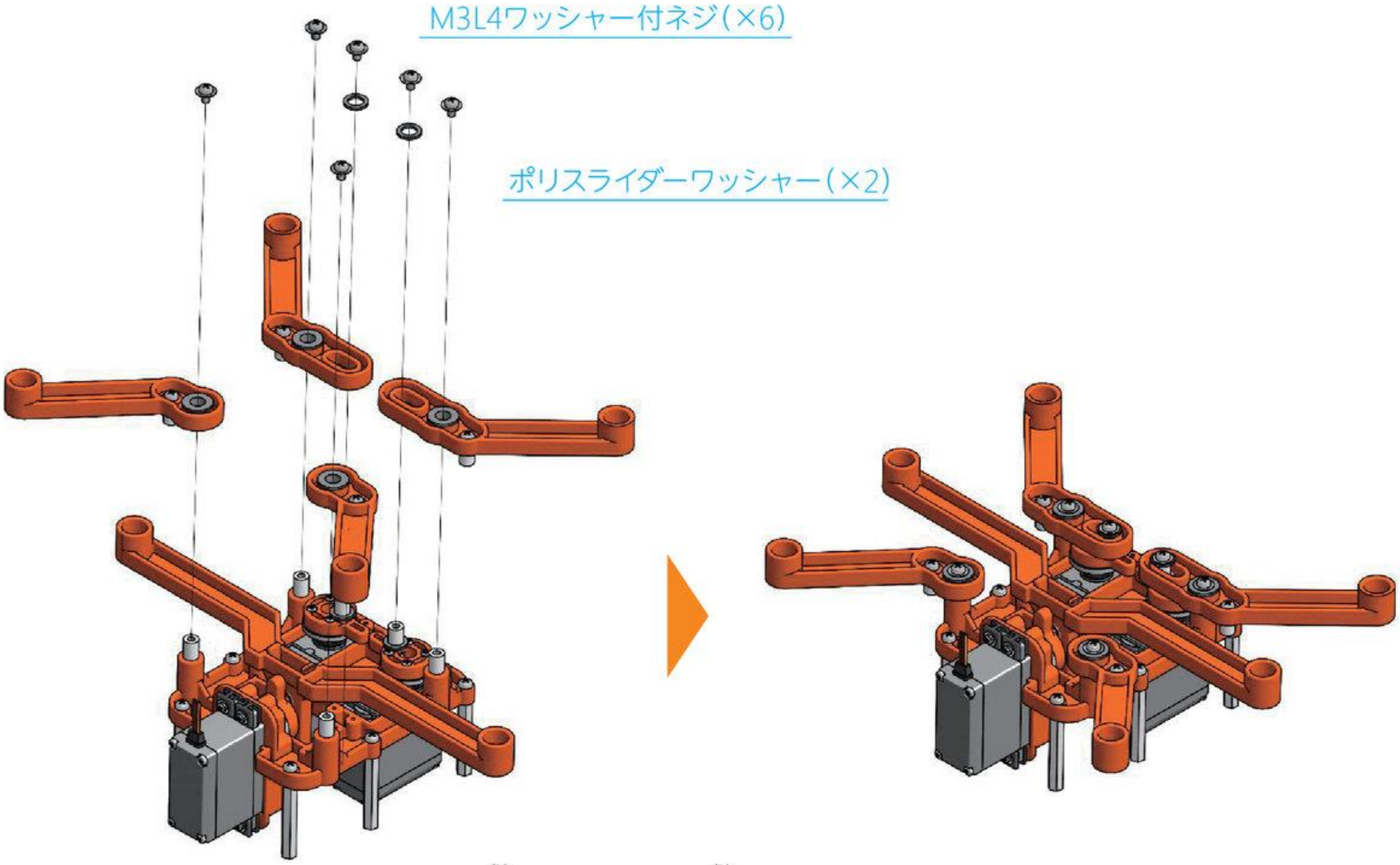


図 1-7　前脚（C-1、C-2）、後脚（C-3、C-4）の組み付け①

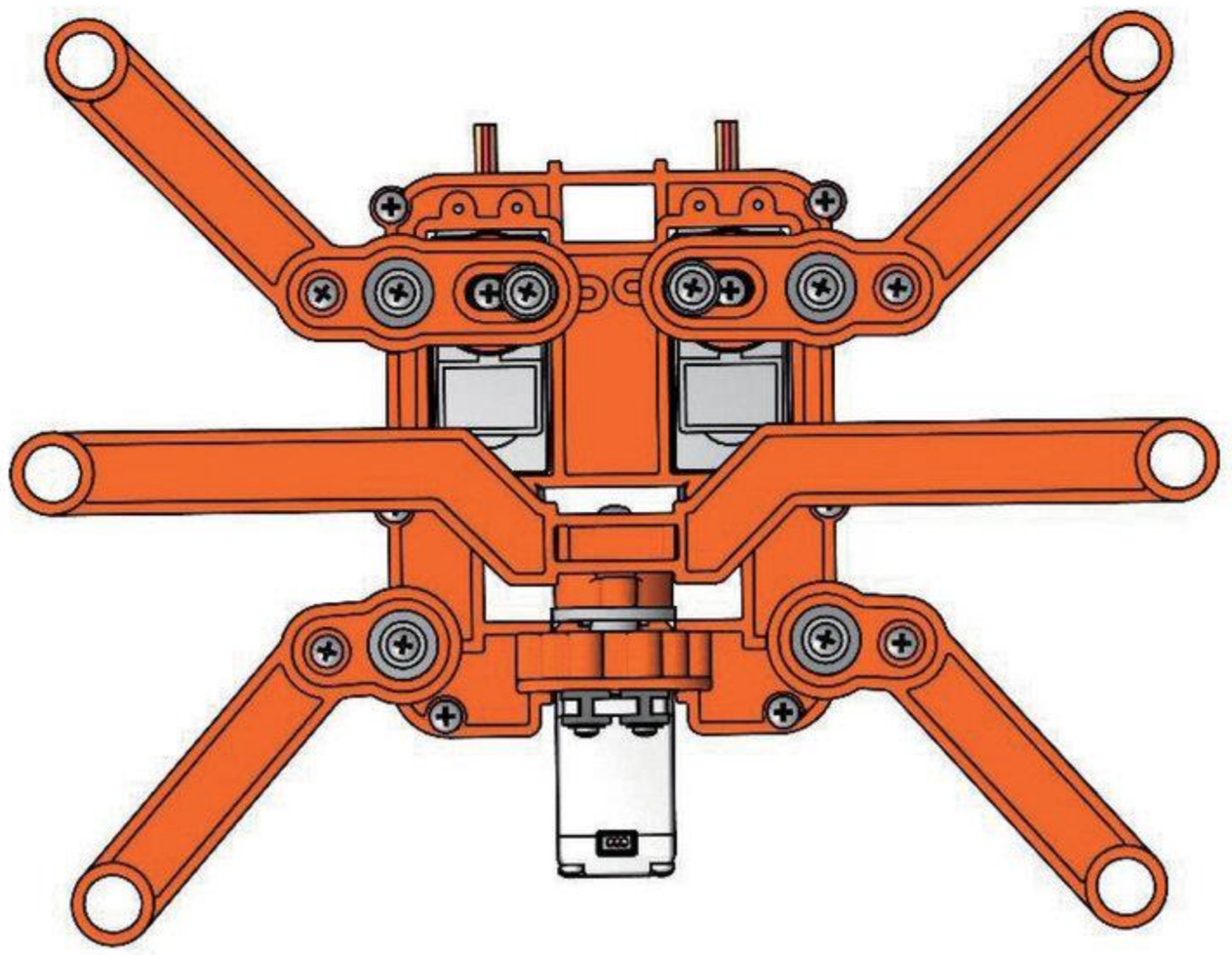


図 1-8　前脚、後脚の組み付け②

＜組み立て手順⑥＞

オイレスブッシュ（×4）がC-5パーツの2つの穴に両面から取り付けられていることを確認し、C-5パーツ（×2）をA-1パーツの左右の前脚（C-1、C-2）と後脚（C-3、C-4）に組み付けた8mm丸スペーサーに取り付けリンクさせます。その上から8mm丸スペーサーにM3L4ワッシャー付ネジ（×4）で組み付けてください。C-5パーツは前脚と後脚の動きを連動させるリンク機構の役割になります。次に、サーボモーターを組み付けたD-1パーツをA-1パーツのスリットに引っかけるように取り付けます。

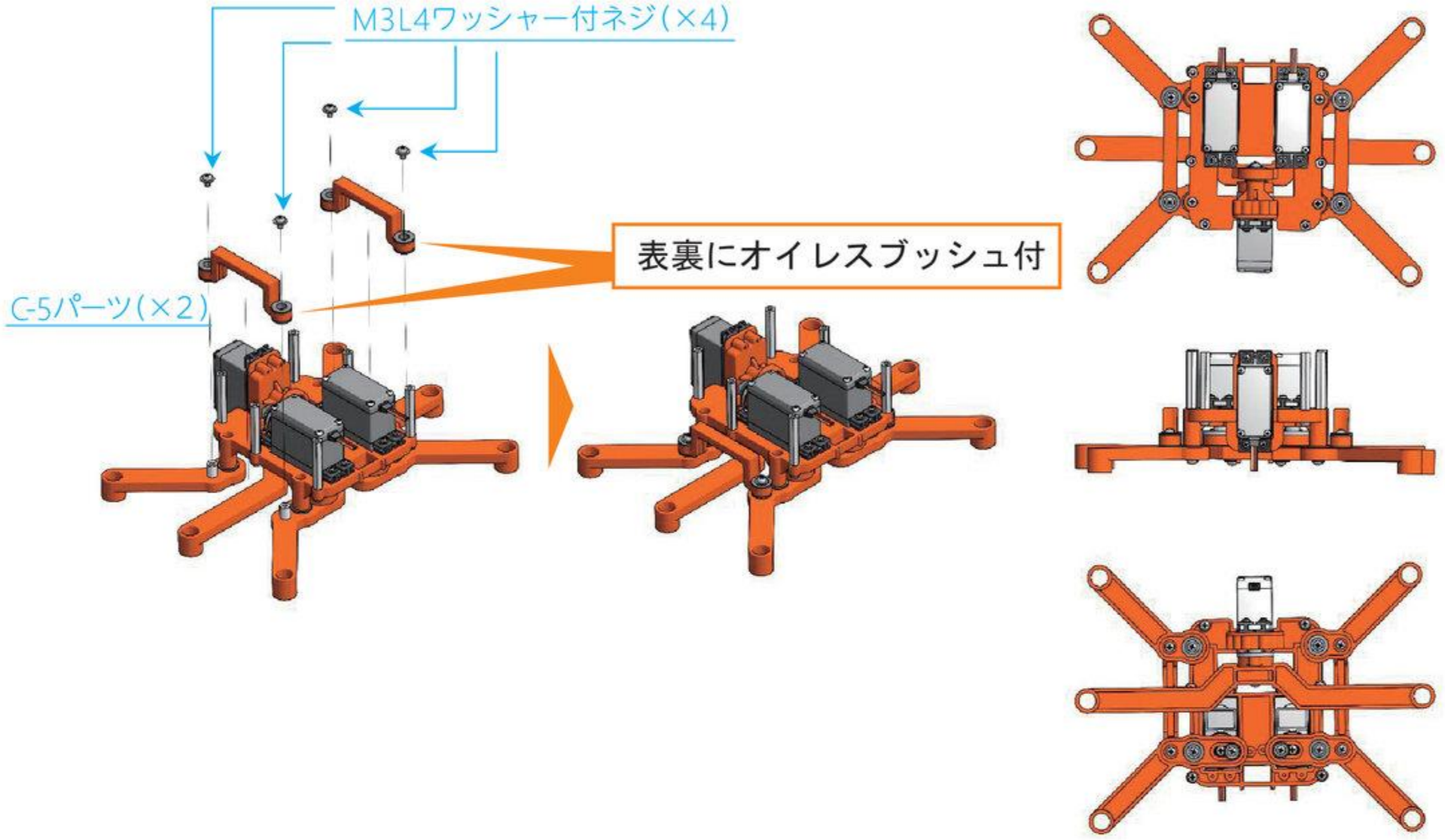


図 1-9　前後脚連結リンクの組み付け

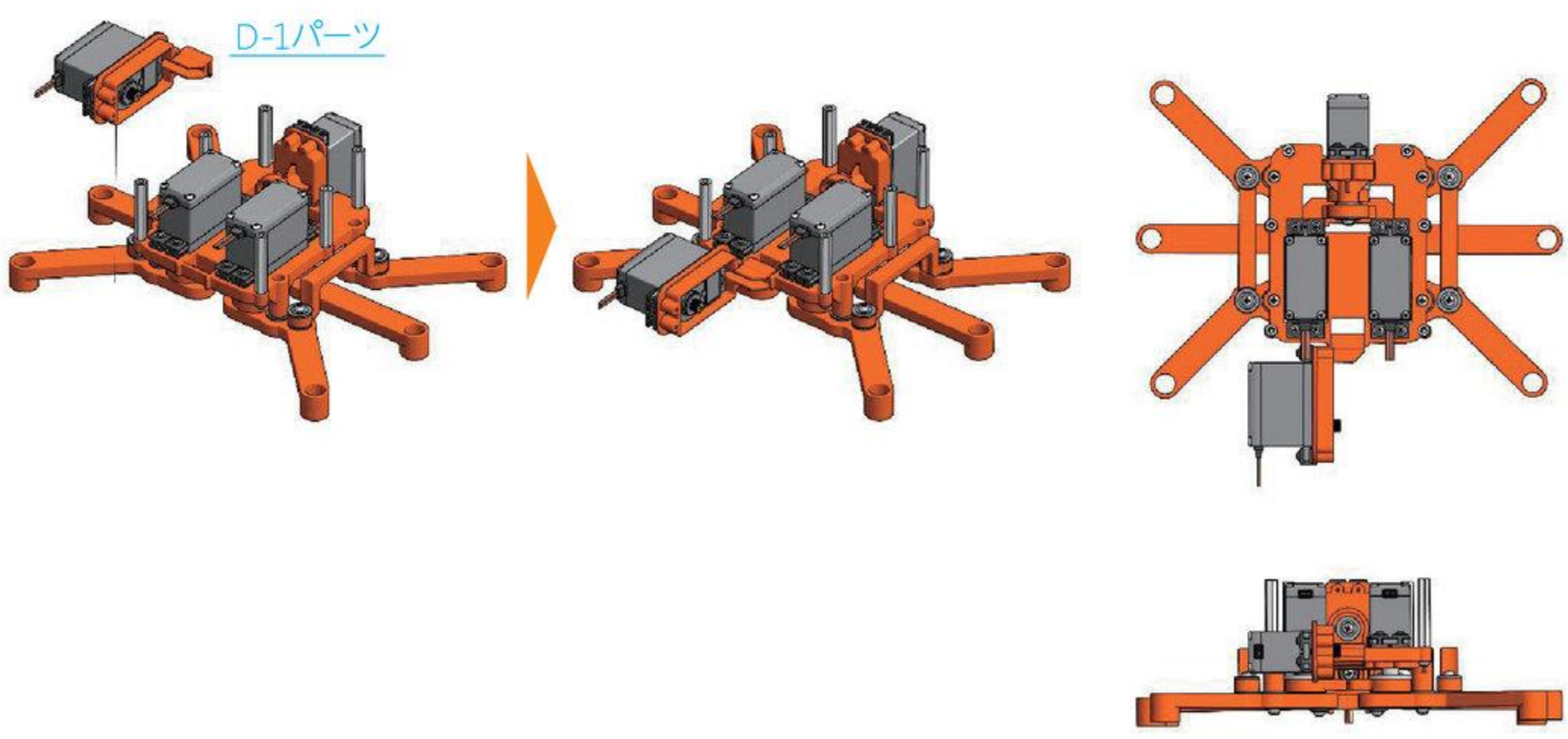


図 1-10　ツノ部サーボモーターの取り付け

＜組み立て手順⑦＞

D-2パーツのサーボホーン部分をD-1パーツのサーボモーター出力軸にはめ込み、その上からM3L6ネジで組み付けてください。組み付ける前にサーボモーターの出力軸が原点になっていることを確認してください（**図1-12**を参照）。

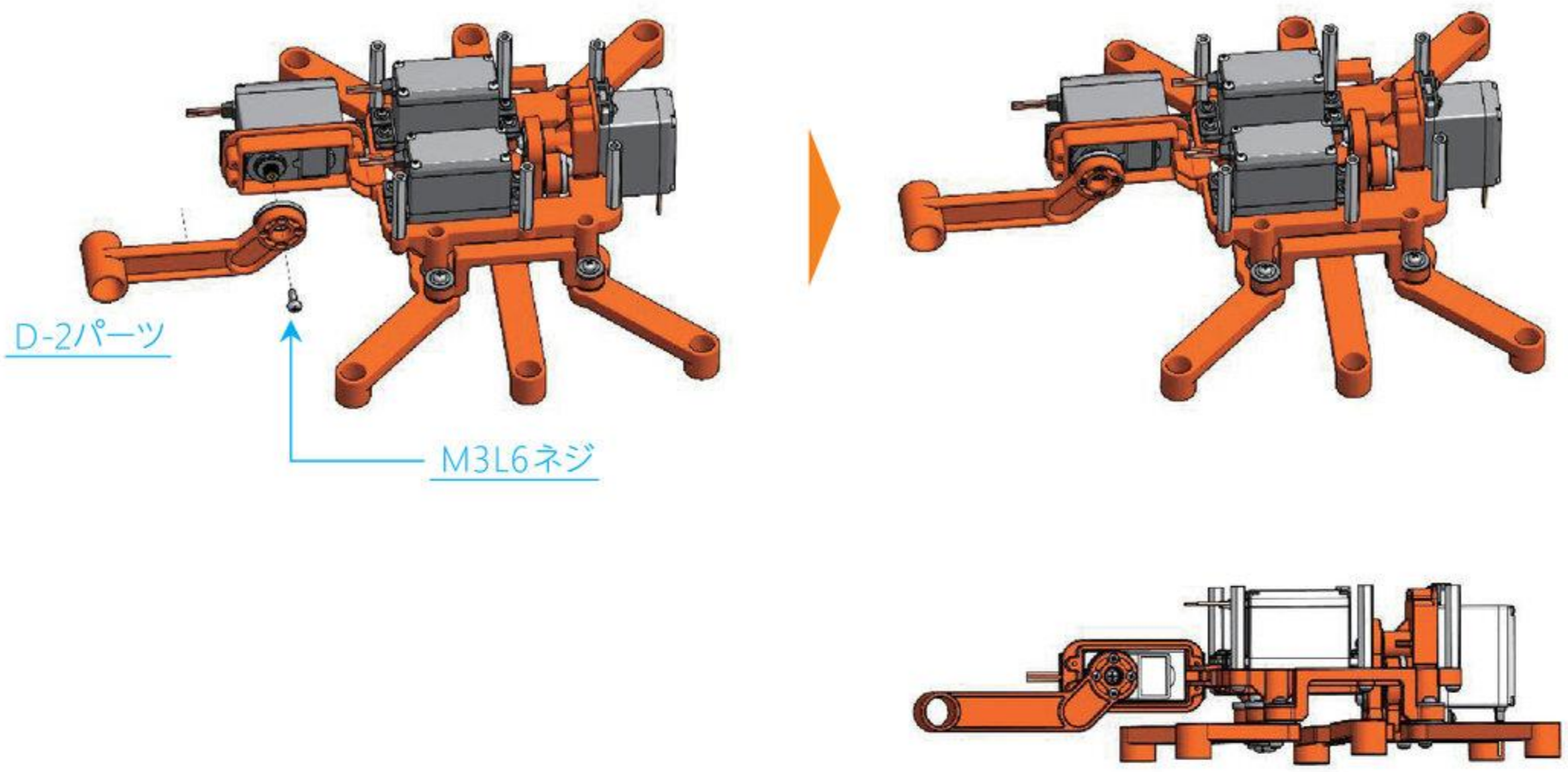


図 1-11　ツノ（D-2）の組み付け

図 1-12　ツノ（D-2）の取り付け方向

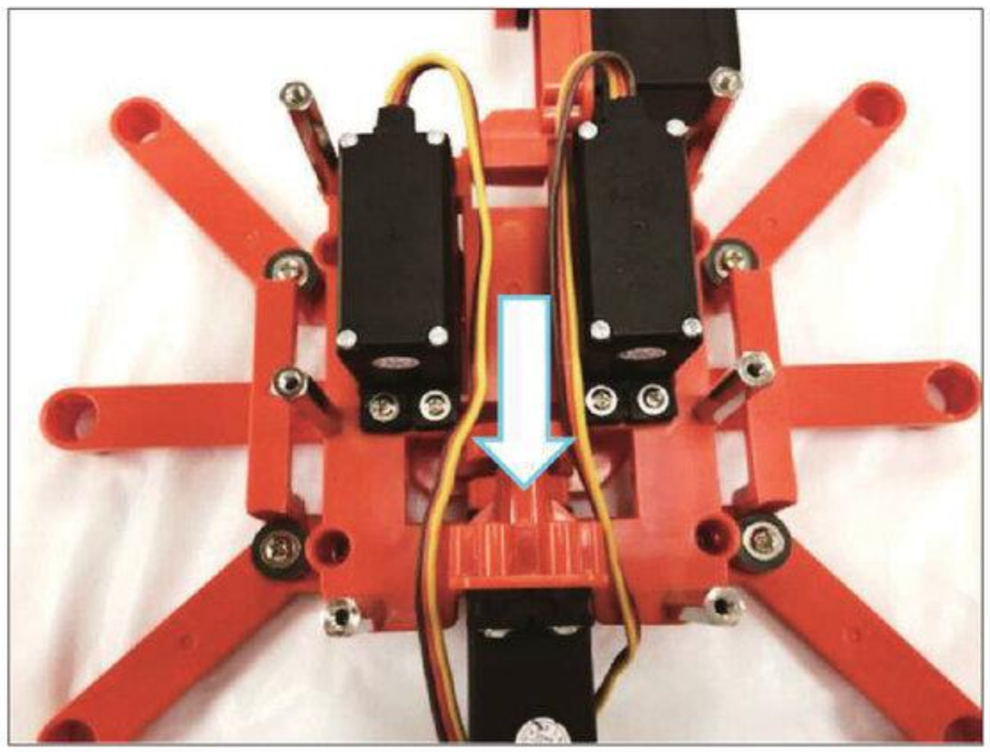


図 1-13　サーボモーターケーブルの配線

＜組み立て手順⑧＞

A-3パーツをM3L12ネジ（×4）で、A-1パーツの33mm角スペーサー（×4）に組み付けます。
A-3パーツの前後方向に注意して取り付けてください。
また、マイコンボードと無線受信モジュールをM2L5タッピングネジB（×6）でA-3パーツに組み付けます。

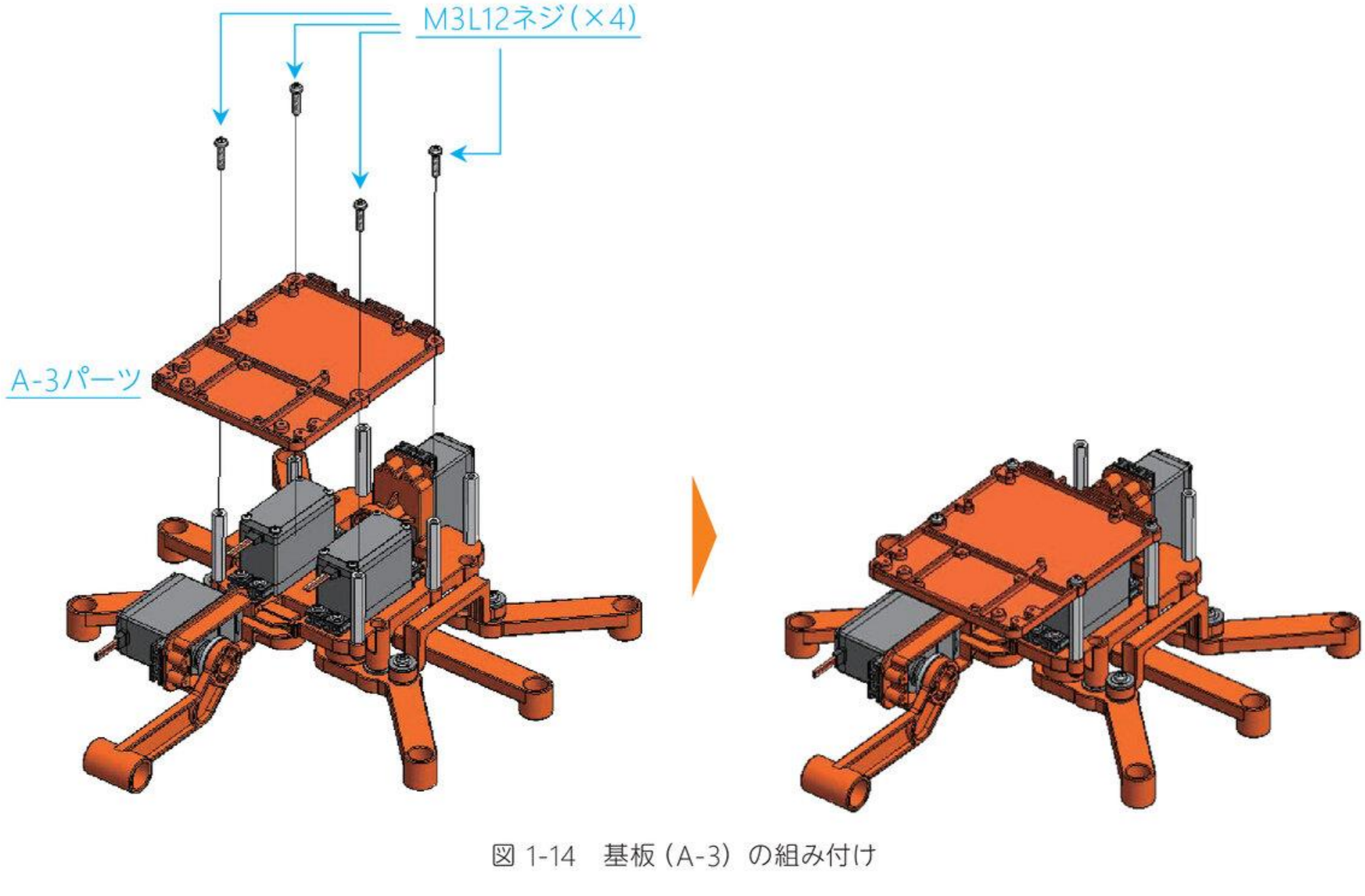


図 1-14　基板（A-3）の組み付け

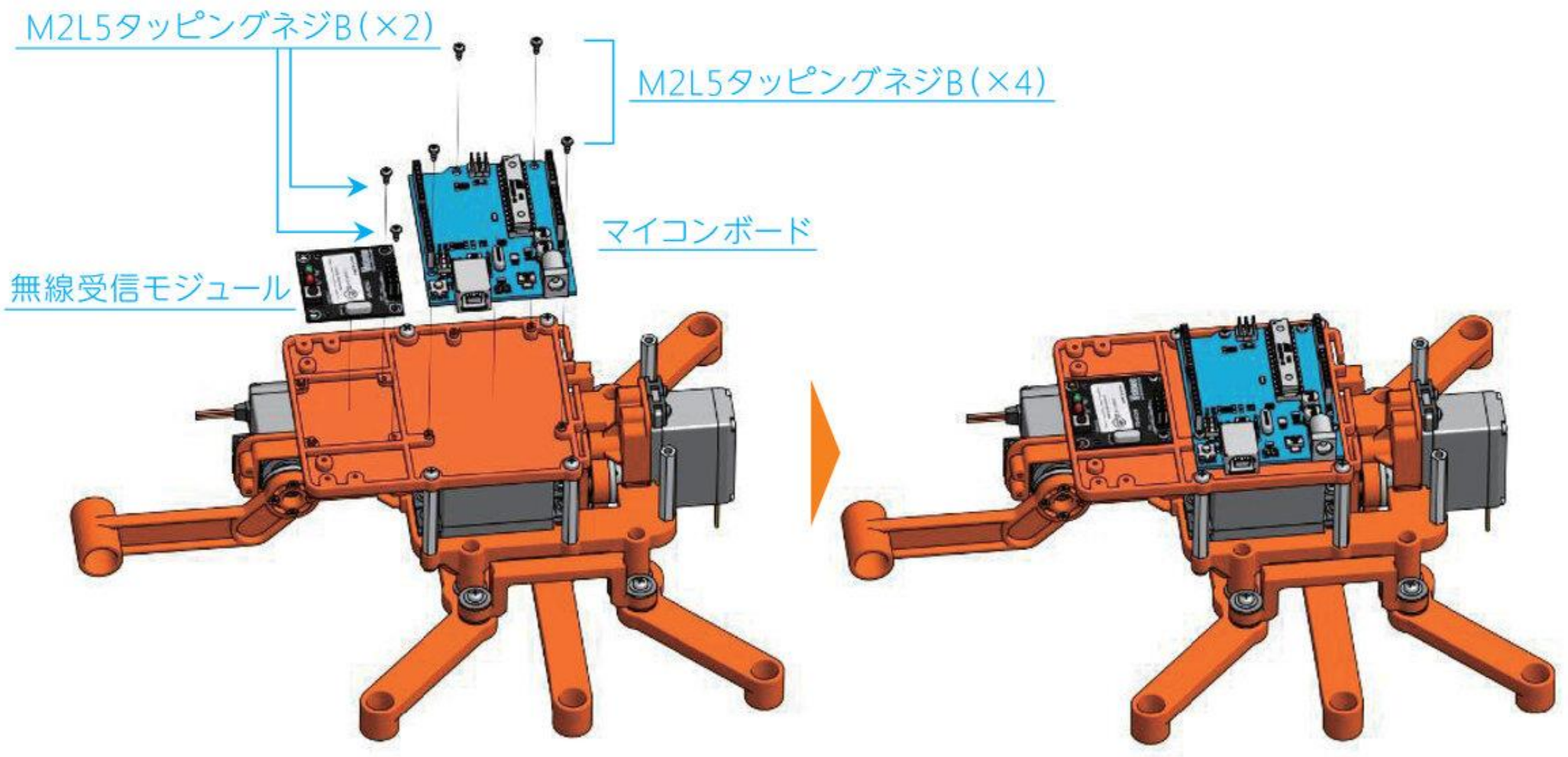


図 1-15　マイコンボード、無線受信モジュールの組み付け

＜組み立て手順⑨＞

マイコンボードにロボプロシールドを接続し、左脚側のサーボモーターのコネクターをロボプロシールドの S0 、右脚サーボモーターのコネクターをロボプロシールドの S1 、中脚サーボモーターのコネクターをロボプロシールドの S2 、ツノのサーボモーターのコネクターをロボプロボードの S3 に接続します。配線は、**茶→G、赤→V、橙→S**になるように接続してください。次に、リボンケーブルをロボプロシールドのコネクターと無線受信モジュールのコネクターに接続します。

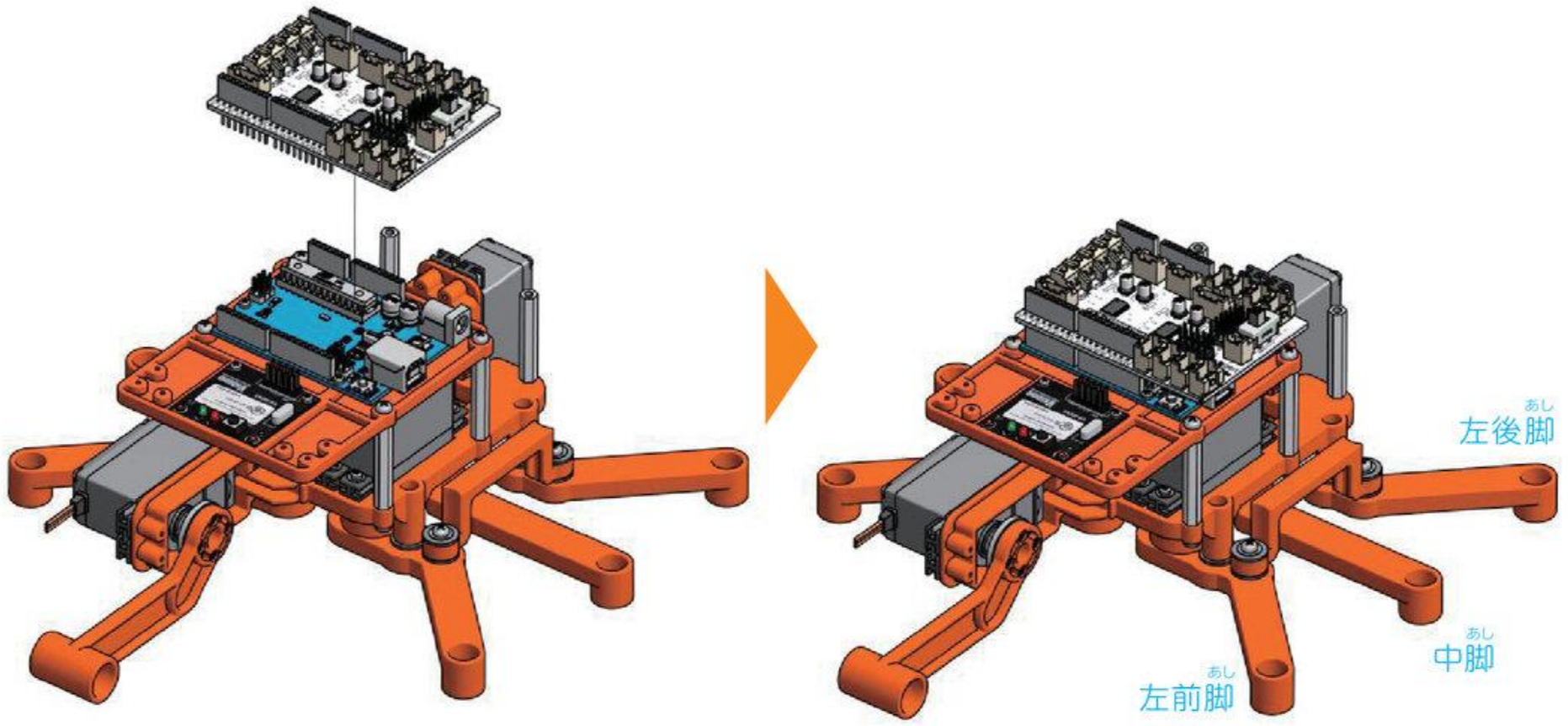
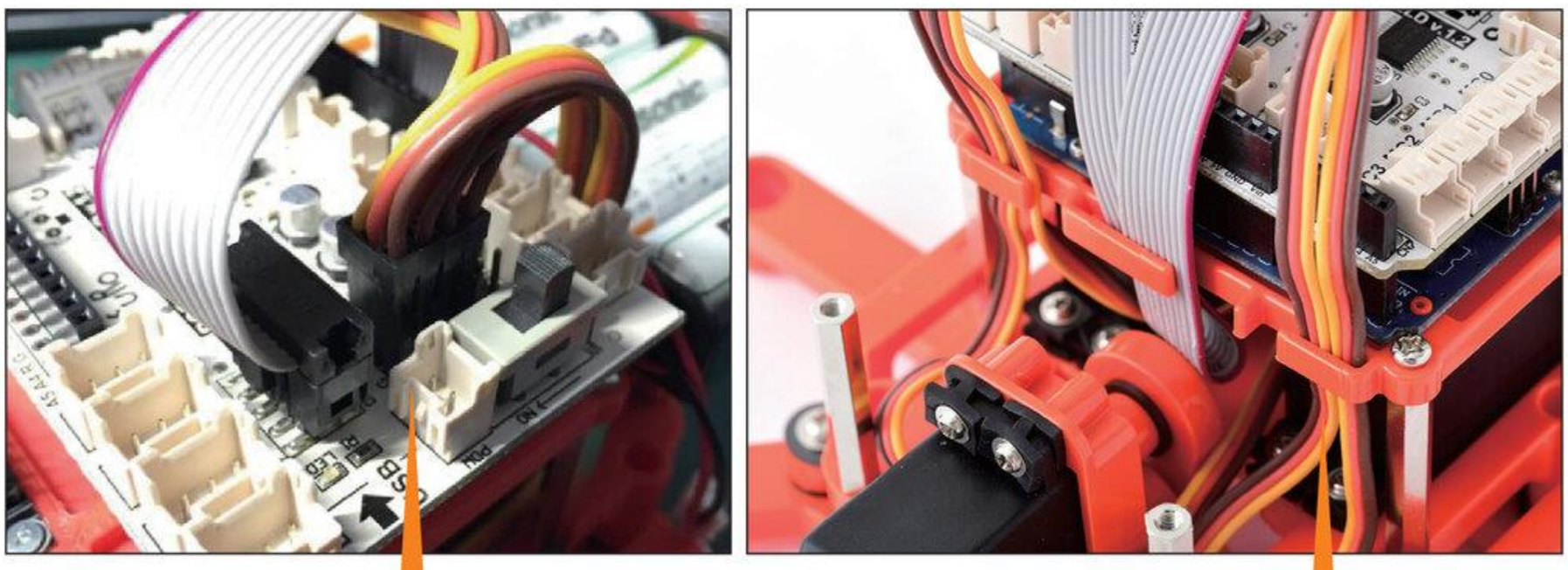


図 1-16　ロボプロシールドの接続



サーボモーター	ロボプロシールド端子
左脚	S0
右脚	S1
中脚	S2
ツノ	S3

サーボモーターのケーブルとリボンケーブルは、写真のようにA-3パーツのフックに通しておく

図 1-17　ロボプロシールドとの接続と配線

＜組み立て手順⑩＞

A-1パーツの後方に組み付けた33mm角スペーサーに、M3L6ネジ（×2）を使用して電池ボックスを組み付けます。

さらに、タッチセンサー（×2）をM2L12タッピングネジB（×4）を使用して、A-3パーツに組み付け、200mm針金（×2）を15mmビニールチューブ（×2）でタッチセンサーに固定します。

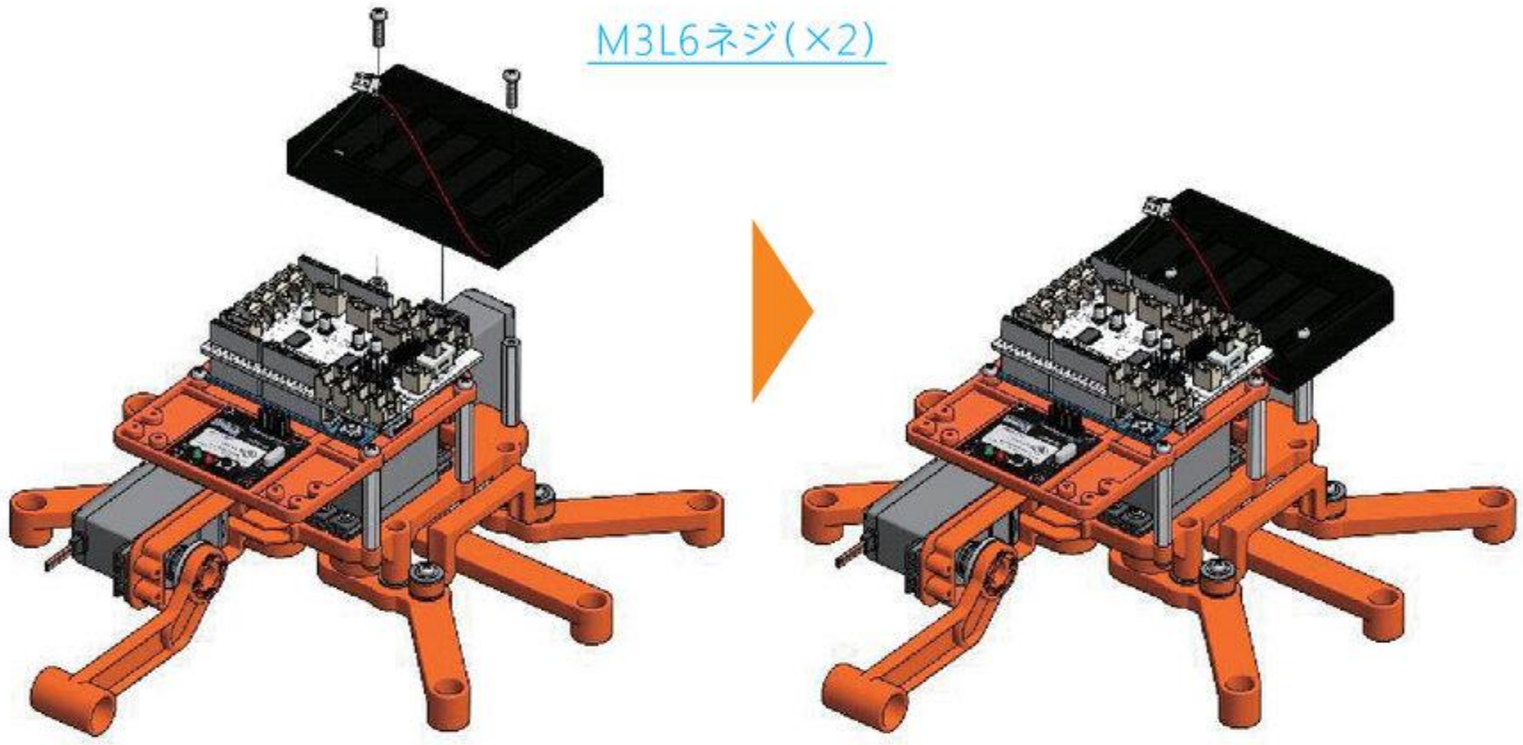


図 1-18　電池ボックスの取り付け

図 1-19　タッチセンサーと針金の取り付け

左側のタッチセンサーはD2に、右側のタッチセンサーはD3に接続

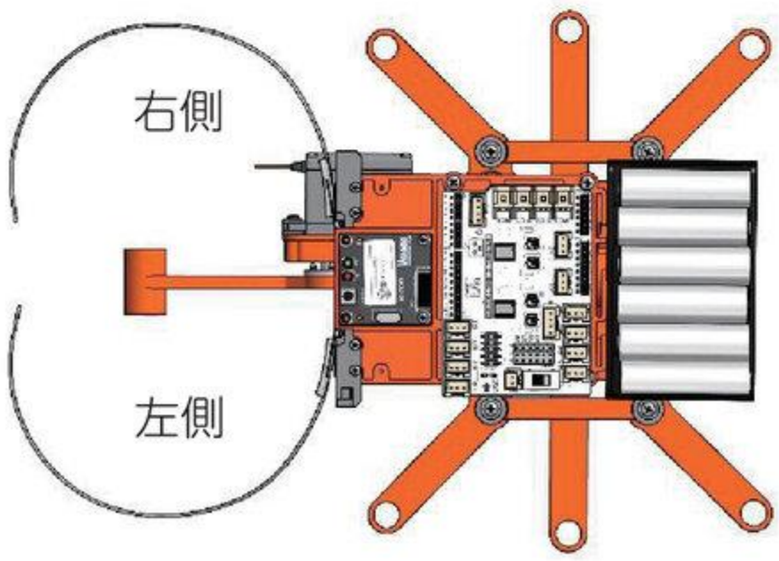


図 1-20　タッチセンサーの配線

2.　六脚(ろっきゃく)ロボットを動かす（目安 40 分）

2.0.　サーボモーターの動作確認

六脚(ろっきゃく)ロボットが完成したので、早速歩かせたいところですが、歩行制御(せいぎょ)のプログラミングを突然はじめるのは難易度がとても高いので、まずは、サーボモーターを上手に制御(せいぎょ)するところから学んでいきます。以下のプログラムを実行してください。

プログラムの書き込み

RoboticsProfessorCourse3　>　HexRobot2　>　ServoTest2

実行結果：左右のタッチセンサーがオンになると、ツノが上下に動きます。
このプログラムの流れをフローチャートにすると、次のようになります。

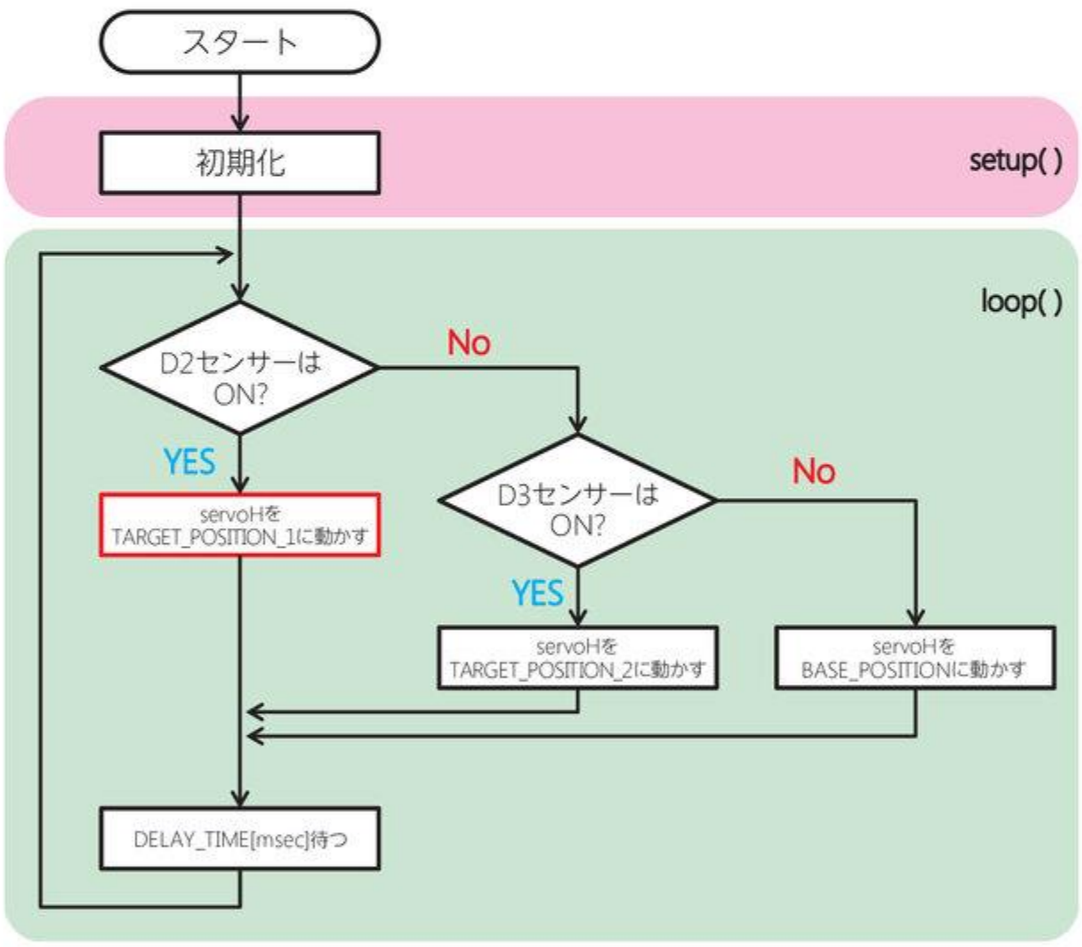


図 2-0　プログラム「ServoTest2」のフローチャート

やってみよう！

このプログラムでは、ツノのサーボモーターの目標位置が3種類あるね！
それぞれ何度か、プログラムから読みとってみよう！

D2 センサーが ON のとき：10 度

D3 センサーが ON のとき：100 度

どちらも OFF のとき：55 度

2.1. サーボモーターを滑らかに動かそう

プログラム「ServoTest2」でサーボモーターを動かすと、ロボプロシールド上の青色LEDが一瞬暗くなるのがわかりますか？　サーボモーターに瞬間的に大きな電流が流れ、電池の電圧が下がったためです。この現象がひどくなると、電池の電圧が下がってマイコンがリセットされてしまったり、動作が不安定になったりしてしまいます。
この問題にはどのように対処すればいいでしょうか。以下のプログラムを実行してみましょう。

プログラムの書き込み

RoboticsProfessorCourse3 ＞ HexRobot2 ＞ ServoTest3

実行結果：サーボモーターの動きがゆっくりになります。
青色LEDの明るさも変化しなくなりましたね。プログラム「ServoTest3」の loop() 内は、タッチセンサーの状態をみて条件分岐をしたあと、サーボモーターを1度だけ動かすという処理になっています。10度の位置にあったサーボモーターを55度の位置にしたければ、loop() を45回くり返せばいいことになります。
また、前回説明したとおり、サーボモーターは10~20msecの周期で信号を受け取ります。逆に、これより短い周期で命令を出しても受け取れません。そのため、loop() 内に20[msec]の待ち時間を入れてあります。これにより、サーボモーターにはおよそ20msecに1回だけ信号が送られることになります。

プログラム「ServoTest3」より抜粋

```
#define    SERVO_PERIOD          20
・・・
delay(SERVO_PERIOD);
```

「一気に目標位置まで動かす」という命令を「何回かの動きに分けて少しずつ動かす」という命令にかえたことで、モーターの負荷が減ったわけですね。皆さんも、夏休みの宿題が大量に出たら、最終日にあわてて全部片づけるよりも毎日少しずつこなしていくほうが楽だと思うのではないでしょうか？
loop() 内の処理を詳しく見ていきましょう。

講 ServoTest2 と ServoTest3 を同時に開いて比較すると、どの部分が違うかわかりやすくなります。
サーボモーターの性質を理解し、制御方法を学習することで、第 3 回のコントローラーを使った制御を進めやすくなります。

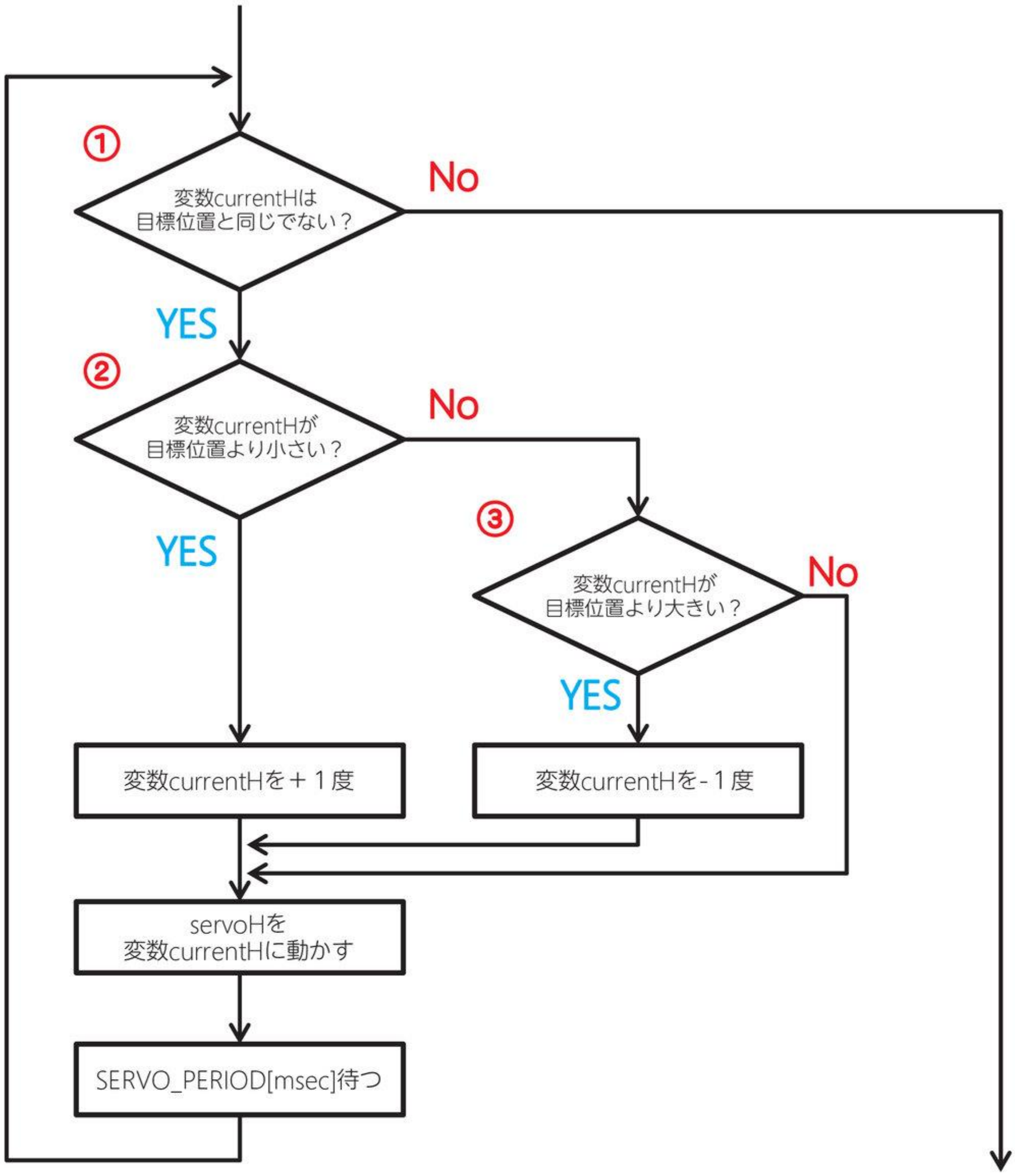


図 2-1　プログラム「ServoTest3」のフローチャートの一部

フローチャートは**図2-1**のようになります。

やってみよう！

図2-1のフローチャートの3つの分岐（ひし形のブロック）のうち、削除しても処理が変化しないものが1つあるよ！
①～③のどのブロックか考えてみよう！

③

講 ①と②どちらも NO だった時点で、変数 currentH は目標位置と同じでも小さい値でも無い、つまり目標位置より大きい値であることが確定しています。よって③の分岐（if 文）は不要です。

ところで、プログラム「ServoTest3」の loop() 内にある、タッチセンサーの状態で条件分岐を行ったあとの処理を見てみましょう。

プログラム「ServoTest3」より抜粋

```
void loop(){
    if (digitalRead(D2) == HIGH){
    //もしD2に接続されたスイッチが押されたら
        while (currentH != TARGET_POSITION_1){
        //servoHの現在位置とTARGET_POSITION_1が異なる間はくり返す
            if (currentH < TARGET_POSITION_1){
            //もし現在位置のほうが小さかったら
                currentH++;                    //現在位置に1を足す
            }
            else if (currentH > TARGET_POSITION_1){
            //もし現在位置のほうが大きかったら
                currentH--;                    //現在位置から1を引く
            }
            servoH.write(currentH);            //新しい現在位置をservoHに反映
            delay(SERVO_PERIOD);               //SERVO_PERIODミリ秒待つ
        }

    }
    else if (digitalRead(D3) == HIGH){
    //もしD3に接続されたスイッチが押されたら
        while (currentH != TARGET_POSITION_2){
        //servoHの目標位置を TARGET_POSITION_2 に設定
            if (currentH < TARGET_POSITION_2){
                currentH++;
            }
            else if (currentH > TARGET_POSITION_2){
                currentH--;
            }
            servoH.write(currentH);
            delay(SERVO_PERIOD);
        }

    }
    else {
    //そうでなかった(D2・D3に接続されたスイッチがどちらも押されていなかった)ら
```

```
        while (currentH != BASE_POSITION){
        //servoHの目標位置を BASE_POSITION に設定
            if (currentH < BASE_POSITION){
                currentH++;
            }
            else if (currentH > BASE_POSITION){
                currentH--;
            }
            servoH.write(currentH);
            delay(SERVO_PERIOD);
        }
```

```
    }
}
```

赤枠で囲まれている3つの部分は、目標位置の文字列以外はまったく同じであることがわかります。このように、同じような処理を複数かくときは「関数」を作成すると、手間が減らせるうえに文字の打ち間違いなども防ぐことができましたね。

やってみよう！

moveHead() という関数を作成して、プログラム「ServoTest3」をまとめてみよう。
void moveHead() 内の処理内容をどうすればいいか考えてみてね。

ヒント

delay には3つの赤枠すべて同じ値が入るけど、while の判定条件は枠ごとに異なるね。つまり、この部分だけは関数をつくるときに値を指定できるようにする必要があるよ。
「引数」の機能を覚えているかな？

以下のプログラムが解答例です。

プログラムの書き込み

RoboticsProfessorCourse3　＞　HexRobot2　＞　ServoTest3b

チャレンジ課題

プログラム「ServoTest3」では、一度タッチセンサーが押されると、その後離しても目標位置に到達するまでサーボモーターの動作は止まらなかったね。これでは、状態の変化に対して素早くロボットが動作をかえることができないので、今度は、この問題を改善する方法を考えよう。

以下のプログラムが解答例です。

プログラムの書き込み

RoboticsProfessorCourse3 ＞ HexRobot2 ＞ ServoTest4

実行結果：タッチセンサーを押すと、サーボモーターがゆっくりと上下に動き、離すと、もとの位置にもどりはじめます。

このプログラムのフローチャートは、**図2-2**のようになります。
「ServoTest4」では、2つの変数 targetH と currentH を使って、servoH で管理されるツノのサーボモーターを制御しています。
targetH の値は、タッチセンサーの状態によって TARGET_POSITION_1（10度）、TARGET_POSITION_2（100度）、BASE_POSITION（55度）のいずれかとなります。この targetH とツノのサーボモーターの現在位置 currentH とを比べるので、タッチセンサーの状態が変化することでサーボモーターの目標位置も変化するのです。
そして、この2つの変数の管理が loop() のくり返し時間、約20ミリ秒毎に毎回行われるようになっています。そのため、タッチセンサーの状態変化は最大でも約20ミリ秒で、ツノの動きに反映され、素早い反応が可能になりました。

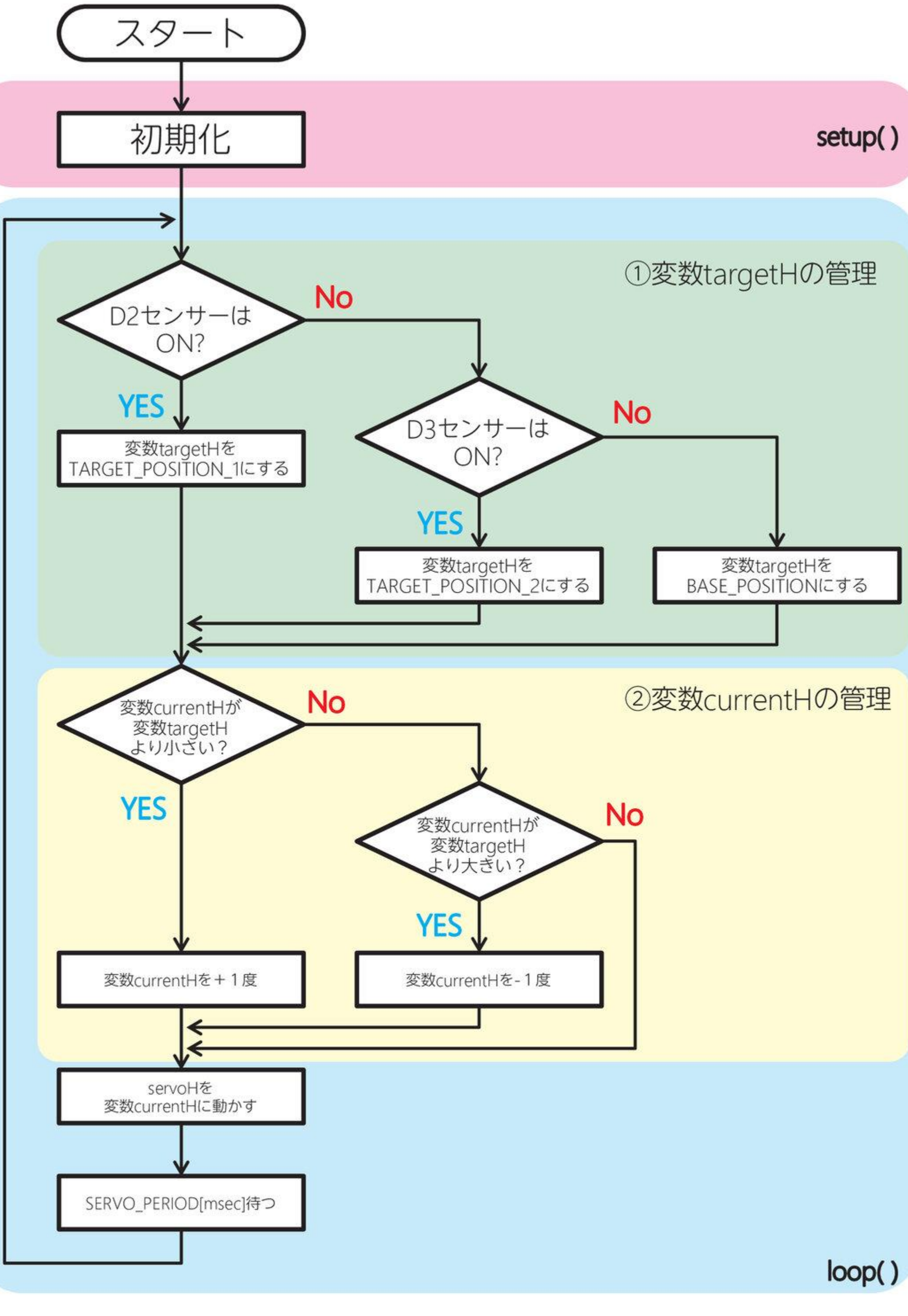
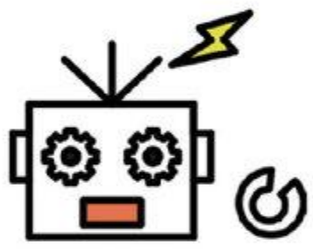


図 2-2　プログラム「ServoTest4」のフローチャート

講　制御プログラム文が長くなっていますので、フローチャートとプログラムを対比させながら、読み解けるように時間を与えてください。実際に、フローチャートの分岐に沿って、ロボットの動作を実行させてください。

3.　まとめ（目安５分）

今回は、六脚(ろっきゃく)ロボットを完成させました。そして、ロボットを動かす上で最も基本となるサーボモーターを上手に制御(せいぎょ)するための考え方を順番に説明してきました。このようにフローチャートを使ってロボットのプログラムを紙の上で設計すると、プログラミングがスラスラできるようになります。これは、プログラミングだけではなく、毎日の生活の中で、何をどの順番でやるのか段取りを考えるということにも役立ちます。
次回はいよいよ、ロボットを「歩かせる」ことに挑戦します。次回までに「普段無意識に自分がしている歩行動作はいったいどういう動きが組み合わさっているのだろうか?」と考えてみましょう。



次回はコントローラーでリズミカルに操作するよ。
お楽しみに！

≪次回必要なもの≫
次回は、今回つくったロボットと以下のパーツを持ってきてください。

USBケーブル 1	コントローラー 1	ACアダプター 1
		

図 3-0　次回必要なもの

講

○　以下の授業の目標を再確認します。
　・六脚ロボットを完成させる
　・サーボモーターを滑らかに制御する
○　今回の授業で学んだ感想や面白かったことなどを、生徒から聞いてみましょう。
○　次回のテーマは「六脚ロボットの歩行制御（前編）」であることを告知します。