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

二足歩行ロボット③

第6回

二足歩行ロボットのプログラミング③

講師用

Human ヒューマンアカデミー ジュニア STE"AMスクール

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

2024年3月授業分

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。

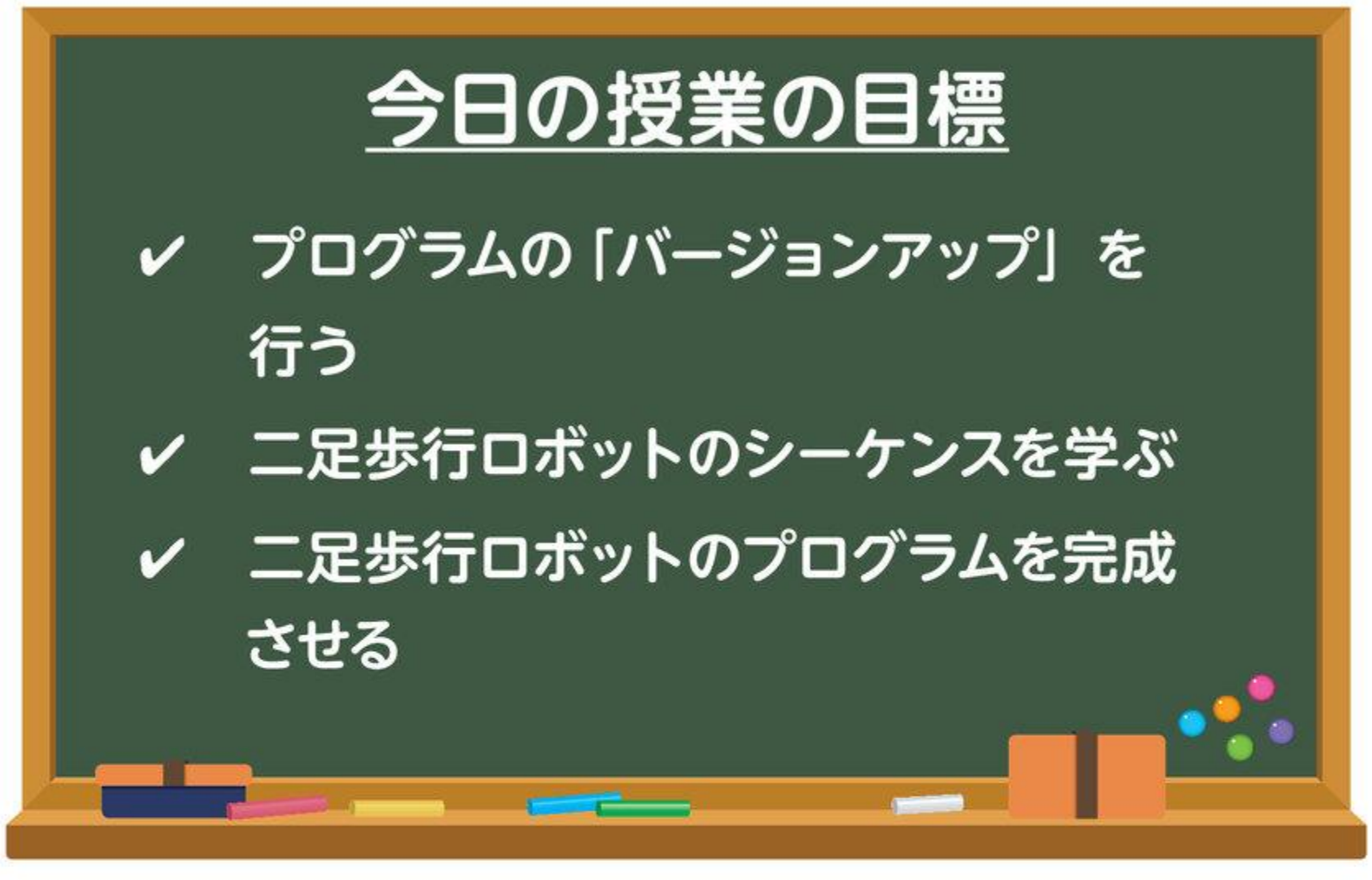
○　今回の目標をパネルで用意するか、黒板に予め書いておきます。

（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）

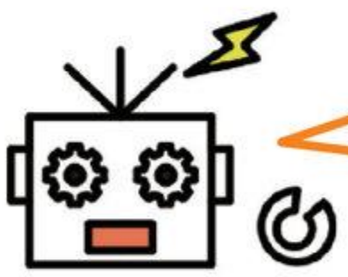
目安時間は授業時間 120 分のうち、休憩 10 分程度取ることを想定しています。
生徒の進捗状況により、休憩時間などを調整して授業を行ってください。

0.　二足歩行ロボットのプログラミング③（目安５分）

0.0.　「二足歩行ロボットのプログラミング③」でやること



第5回の授業では、二足歩行ロボットのすべての機能を問題なく発揮できるように、マイコンの機能などを学んでさまざまなトラブルを回避しました。
今回の授業では、二足歩行ロボットのシーケンス（あらかじめ定められた動作手順）を確認して、ロボットのプログラムを完成させるまでの過程を学びます。
第6回では二足歩行ロボットのプログラムを完成させて動かしましょう。
そして、ちょっとした便利な機能も付け加えて、二足歩行ロボットのコントロールの方法を覚えて、楽しみながら、この授業を締めくくりたいと思います。



今まで勉強したことの集大成ダヨ！

0.1. 必要なもの

前回使った「二足歩行ロボット」とコントローラーを使用します。
また、電源はACアダプターを使うので準備しましょう。

講

一気にサーボモーターを動かすと電力が不足する可能性があるため、動作を５回に分けて行ったり、１回のループで5度までしか動かないように制限をしたりするようにプログラムを工夫してください。
詳しくは解答例のプログラムをご確認ください。

1．　二足歩行ロボットを動かす（目安100分）

1.0.　プログラムのバージョンアップ

前回の課題で、二足歩行ロボットの機能のうち胴体サーボモーター、両足サーボモーター以外のすべての機能が使えるようになりました。

今回は胴体・両足サーボモーターの動作をプログラムすることはもちろん、次のような各種機能を追加することで、より高性能な二足歩行ロボットを目指していきましょう。

POINT

・胴体、両足サーボモーターをそれぞれ、コントローラーの各種ボタンやスティックで操作できるようにする。
・スピーカーから音が出なくなる「ミュート（消音）モード」を実装する。
・コントローラーの十字ボタンを押すだけで、対応した向きに歩いていく「歩行モード」を実装する。

皆さんは実は、これまでの授業でプログラムを書いていくのに必要な各種命令はすでに学び終えています。
「ここではどんな処理をつくればいいのかな」「どうすればこの動作を実現できるかな」というのを考えながら、自分で1つずつ機能を追加していくことで、二足歩行ロボットのプログラムを完成させましょう！

なお、ベースにするのは以下のプログラムです。前回の授業で課題に出た、各種機能のループ時間を適切に調整したプログラムですね。

プログラムの書き込み

RoboticsProfessorCourse3 > BiRobot5 > BiRobotTest2_lib

1.1.　胴体および両足サーボモーターの実装

まずは、両手サーボモーターに続いて胴体・両足のサーボモーターも、コントローラーを使って操作できるようにしましょう。
目標は、以下のような操縦(そうじゅう)ができるようになることです。

図1-0　サーボモーターの操縦(そうじゅう)方法

既にアナログスティック上下（両手サーボモーターの回転）はプログラムしてあるので、これを参考にしてみるのもよいでしょう。

やってみよう！

プログラム「BiRobotTest2_lib」を書きかえ、胴体・両足サーボモーターの操縦機能(そうじゅうきのう)も追加してみよう！

ステップアップ

モーターの追加に成功したら、さらに次のように直すためにはどうすればよいか考えてみよう！

・両手、両足サーボモーターへの回転指令値が-38 ～ +218の範囲(はんい)になってしまっているのを、0 ～ +180の範囲(はんい)内におさまるように修正する。

この機能を追加したのが以下のプログラムです。

プログラムの書き込み

RoboticsProfessorCourse3 > BiRobot6 > BiRobotTest3_lib

このプログラムでは、アナログスティックの読み取り値 lx 、 ly 、 rx 、 ry にそれぞれ5/8をかけています。
アナログスティックの読み取り値は本来± 128までですが、5/8をかけることで± 80までの範囲に変化します。
結果、回転指令値の範囲が +10 ～ +170 となります（限界まで回転させると負荷がかかるので、少し余裕をもたせてあります）。

プログラム「BiRobotTest3_lib」より抜粋

```
targetLF  = 90 + ( lx * 5 / 8 );
// アナログ左スティックの横方向真中の位置を 90 にした角度司令値に変換する
targetRF  = 90 + ( rx * 5 / 8 );
// アナログ右スティックの横方向真中の位置を 90 にした角度司令値に変換する
targetLA  = 90 - ( ly * 5 / 8 );
// アナログ左スティックのたて方向真中の位置を 90 にした角度司令値に変換する
targetRA  = 90 + ( ry * 5 / 8 );
// アナログ右スティックのたて方向真中の位置を 90 にした角度司令値に変換する
```

また、このプログラムでは各サーボモーターの初期位置など、はじめに設定する調整値をまとめて「AdjustParam.h」というタブにまとめてあります。
もしこのプログラムを今後使っていくようなら、必要に応じて値を調整し、次のプログラムに移植していきましょう。

1.2.　ミュート機能の追加

続いて、アニメーションのたびにスピーカーから効果音が出ていますが、これを消音するミュート機能を追加しましょう。
先ほどまでのプログラムに、さらに以下のような操作を追加します。

図1-1　ミュート機能の制御方法

やってみよう！

先ほど自分で書きかえたプログラム、またはプログラム「BiRobotTest3_lib」を書きかえ、START・SELECTボタンでオン・オフできるミュート機能を実装してみよう！

ミュート機能を実装したのが以下のプログラムです。

プログラムの書き込み

RoboticsProfessorCourse3 > BiRobot6 > BiRobotTest4_lib

SELECTボタンを押したらtrue、STARTボタンを押したらfalseになるような変数を「フラグ」として使い、音を鳴らす前にifを使って分岐するようなプログラムになっています。

今回の解答例では、フラグに使う関数の型がいつものint型と異なります。

プログラム「BiRobotTest4_lib」より抜粋

```
bool mute;

(中略)

if (mute == false){                          // ミュート機能無効のとき
    updateTone(spriteIndex, distance);       // スピーカーからの単音出し
}
```

変数muteは、boolという型で宣言されています。

これまでにもたまに登場した「ブーリアン型」ですが、覚えているでしょうか？

範囲内の整数であればどんな数になろうが自由なint型と違い、ブーリアン型変数はtrueかfalseのどちらかの値にしかなれません。

今回のような「オンかオフか」といった、2つに1つの分岐をさせたいケースでは無駄なメモリを消費しないため役立ちます。

1.3.　歩行動作の状態遷移の確認

最後に、歩行モードを追加します。
これは、コントローラーの十字キーを押している間、対応する方向にロボットが自動で歩き続けてくれる機能です。
たとえば十字キーの上を押している間は前に向かって歩き続け、押すボタンを右に切りかえたら右へ向き直るように足踏みをする、といった動作を作りたいわけです。

しかし、一言に「歩く」といっても、具体的にどんな動作がどのように組み合わさってできているのかを指示しないと、ロボットには実行できません。
まずは、ロボットに「歩く」という動作の順序（シーケンス）を教えるため、動作内容を整理するところからはじめましょう。

やってみよう！

まずは現在のプログラムを使って、コントローラーを操作してロボットが前進するように歩かせてみよう！
それができたら、「歩行」がどんな動作の組み合わせでできているか、しっかり観察しておこう！

ロボットを前に歩かせることはできましたか？
成功した人はよくわかると思いますが、前進歩行をしているロボットは、大きく分けて6つの状態を順番にくり返していますね。

POINT

STEP1：直立している
STEP2：左足を上げている
STEP3：両足をねじっている
STEP4：直立している
STEP5：右足を上げている
STEP6：両足をねじっている
STEP1：直立している
…

これを1つの図にまとめると、**図1-2**のようになります。片方の足を上げている状態から、もう片方の足を上げる状態に移るとき、かならず直立の状態をはさんでいることがわかりますね。
このような、ロボットなどの状態の移り変わり（遷移）をわかりやすく表した図を「状態遷移図」といいます。

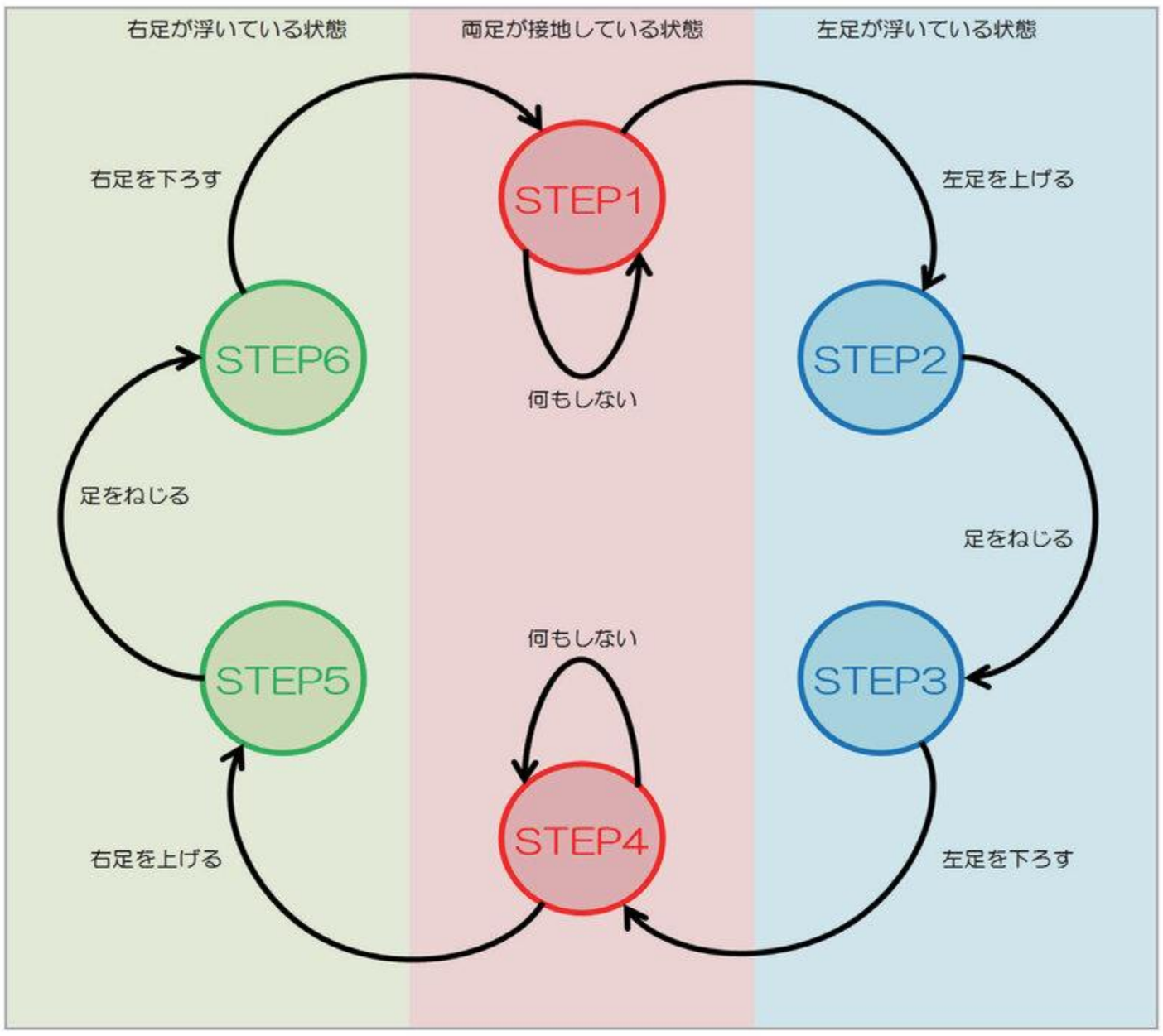


図1-2　二足歩行ロボットの歩行動作の状態遷移図

ステップアップ

前進歩行と同じように、後退、右旋回、左旋回もそれぞれ6つのステップに分けることができるよ！
「前進」の行をヒントにして、次の表を完成させてみよう！

	STEP1	STEP2	STEP3	STEP4	STEP5	STEP6
前進	直立	左足上げ	両足ねじり	直立	右足上げ	両足ねじり
後退	直立	左足上げ	両足ねじり	直立	右足上げ	両足ねじり
右旋回	直立	左足上げ	右足ねじり	直立	右足上げ	左足ねじり
左旋回	直立	左足上げ	左足ねじり	直立	右足上げ	右足ねじり

一覧表を完成させてみると、STEP1、STEP2、STEP4、STEP5はどの向きに歩くときでも同じ動作であることがわかりますね。
また、STEP1とSTEP4では必ず胴体のモーターが原点位置に戻り、両足が地面に接地します。歩行動作を途中で止めたり、歩く向きを切りかえたいときは、STEP1かSTEP4のときに行えばよさそうです。

1.4.　歩行動作の自動化

さて、いよいよ二足歩行ロボ、最後のプログラムを完成させます！
ここまでロボプロの授業で培(つちか)ってきた知識・スキルをフル活用して、思い通りに歩き回るプログラムを作りましょう！

ステップアップ

先ほど自分で書きかえたプログラム、またはプログラム「BiRobotTest4_lib」を書きかえ、前後左右の歩行モードを実装してみよう！

次のページから、少しずつヒントを紹介していきます。
はじめは何も見ないで挑戦してみて、途中でつまってしまったら1ページずつ読み「自分で進められそう！」と思ったらまたプログラミングを再開してみましょう！

講

なお、完成版の例は以下のプログラムです。
RoboticsProfessorCourse3 > BiRobot6 > BiRobotTest5_lib

1）歩行モードのフラグを立てる

まずは、「前進」「後退」「左旋回」「右旋回」の各モードになっていることを判定するための変数（フラグ）があると良いですね。

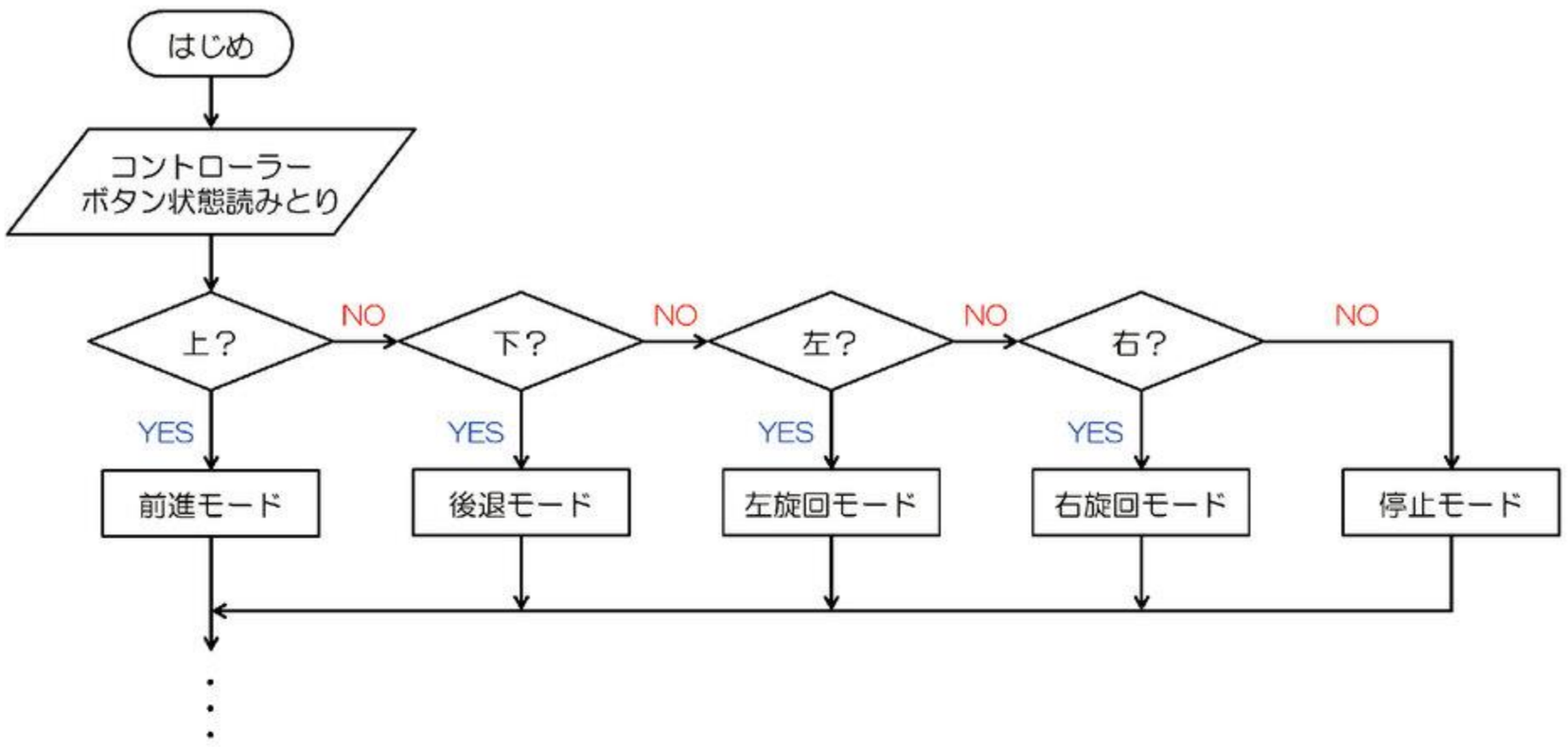


図1-3　歩行モードのフラグを管理

もとのプログラムでは、コントローラーのボタンやスティックの状況から、モーターの回転指令値などの変数を決定する処理を関数 updateAction 内で行っていました。
ここに、**図1-3**のようなフラグ用変数の処理も付け加えてしまいましょう。

なお、今回完成させる二足歩行ロボットは、次の**図1-4**のようなボタン操作を想定します。

図1-4　完成版二足歩行ロボットのコントローラー操作

講

解答例ではフラグ用の変数 mode_flag を作り、上・下・左・右ボタンが押されていたらそれぞれ1・2・3・4を代入することで現在のモードを管理しています（ボタンが押されていないときは0を代入し、停止モードフラグとします）。

プログラムがうまく作れないと嘆いているようなら、この解説を参考にして再チャレンジさせてください。
次のページからは、実際にサーボモーターの回転角度を決めるプログラムのヒントを紹介します。

2）歩行モードにあわせた動作ステップをつくる

「今の歩行モード」が判別できるようになったので、あとは実際の動作をプログラムしていくだけです。
すでに「ステップアップ」内で完成させた表が役立ちますね。
動作モードが4つあり、それぞれ6つのステップに分かれているので、最大24の動作命令をつくる必要があります。

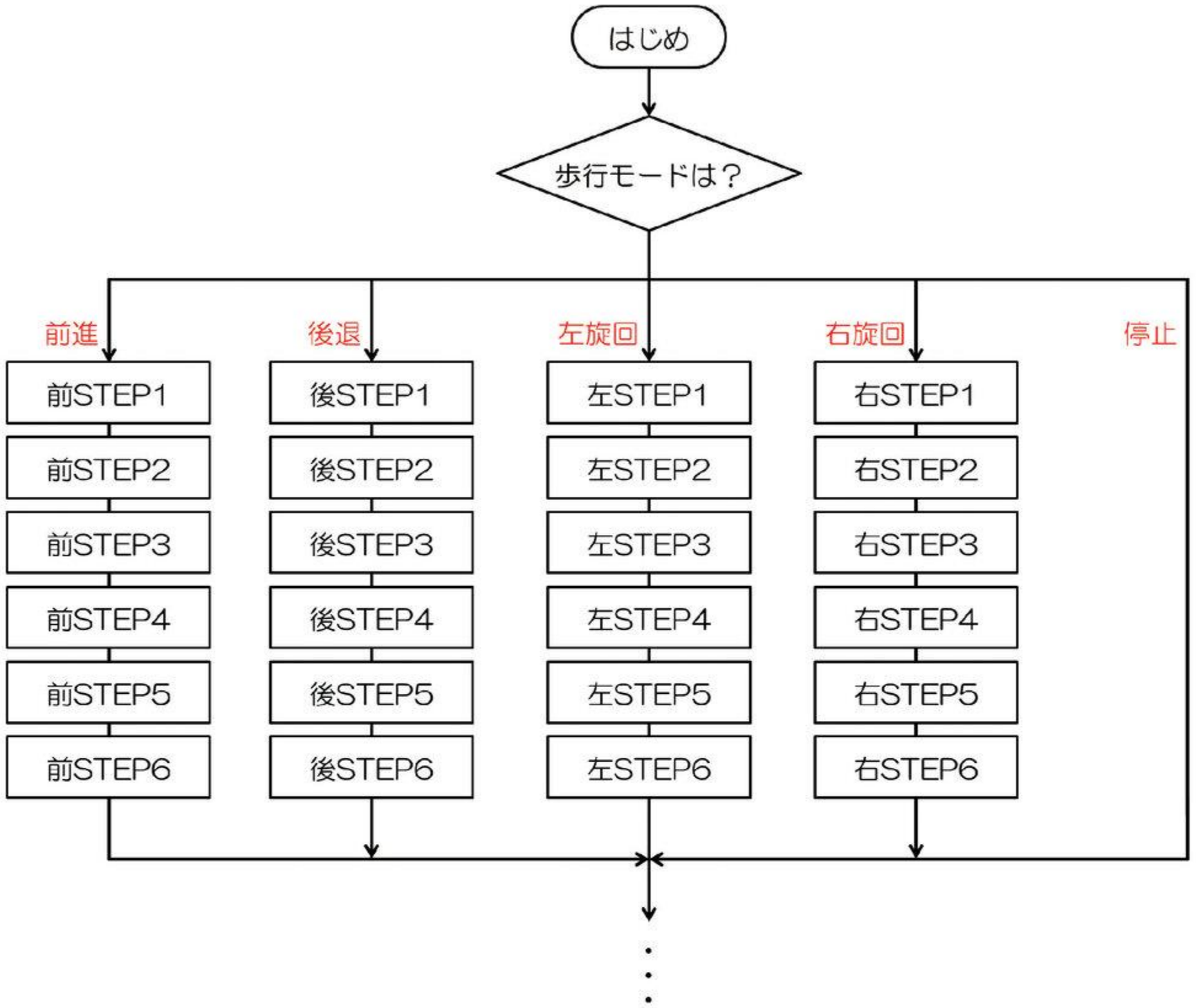


図1-5　全ての動作命令を分けた場合のフローチャート

図1-5のようにプログラムをしても、十字キーで操作できることは確かです。
ただ、このフローチャートだと`void loop()`1周でSTEP1～6を全て行うことになります。1つのループにあまり長い時間をかけるのは避(さ)けたいですし、歩行モードを途中で切り替えることもできなくなるので、少し改良が必要ですね。また、先ほどSTEP1、2、4、5はどの歩行モードでも同じということがせっかくわかったのですから、これらのステップでは歩行モードで分岐(ぶんき)させない方がプログラムがまとまるはずです。

チャレンジ課題

図1-5での問題が解消されるように、フローチャート図を作りなおしてみよう！

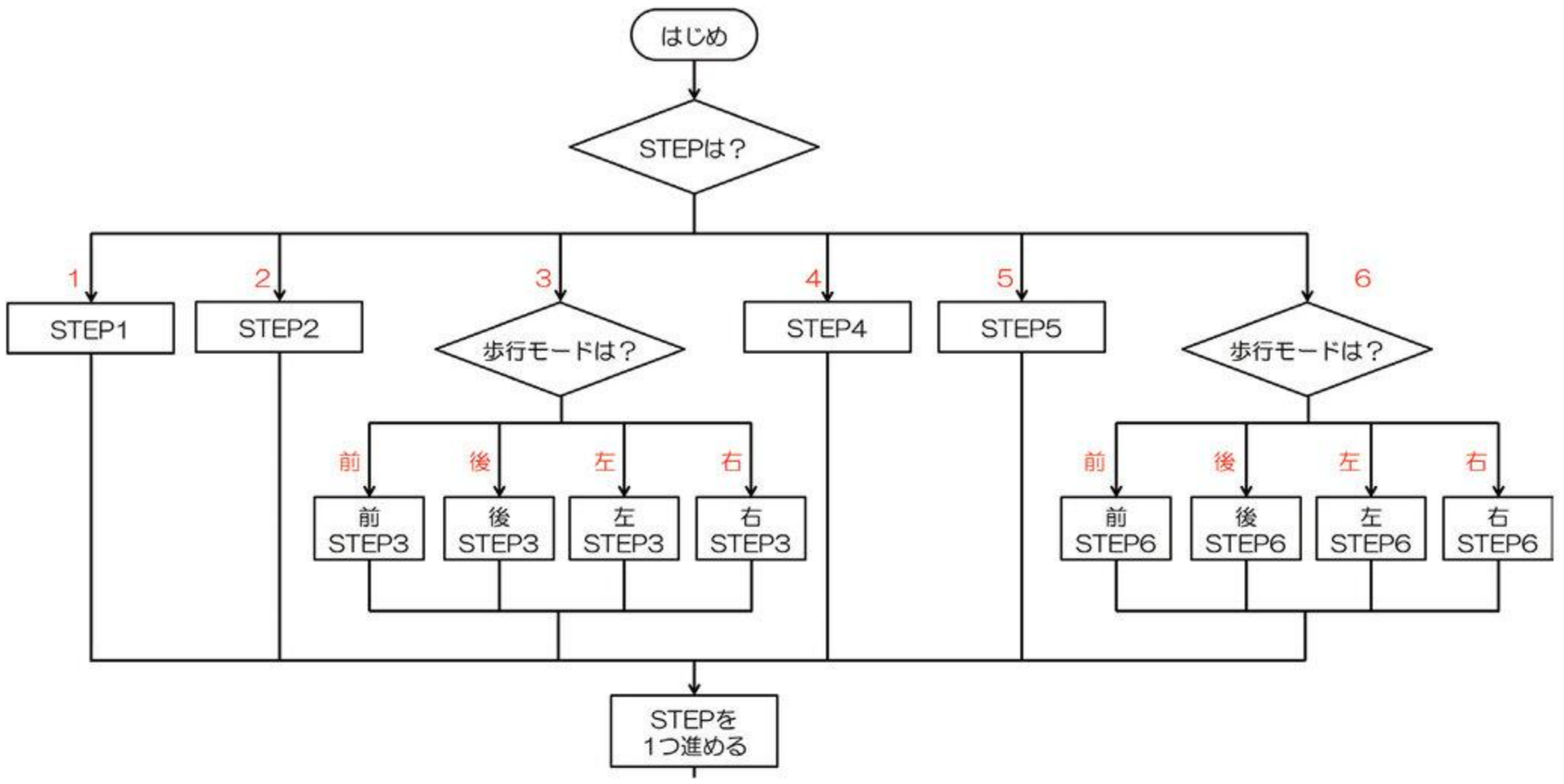


図1-6　改良版フローチャート例

1回のループで1つのステップだけ実行されるように、「歩行モード」より先にまず「今のステップ」で分岐するようにしました。
こうすれば、STEP3かSTEP6のときだけ歩行モード分岐をさせればよいことになりますね。

サーボモーターが目標位置に到達しておらず、まだ動作途中である場合は次のステップに進むことはできません。
よって、目標位置（targetLFなど）と実際の位置（currentLFなど）が等しくなるまでは、次のステップに進む処理を停止させる必要があります。

また、今のステップが1または4であれば、ロボットは接地状態にあります。
もし十字キーが押されていなければ、今の状態をキープできますね。

ここの追加が、プログラムの完成を目指す中で一番かく量が多く大変です。
すべてとは言わないまでも、できそうな部分は少しでも自力でかき進められるようにできるといいですね！

3）プログラム中の数値をわかりやすい形にする

今回追加していく部分には、「歩行モード」が4つ、「歩行ステップ」が6つあり、しかもそれぞれ「回転指令値」を決めなければならないため、さまざまな数値が必要になってきます。
これをすべて数字で書いてしまうと、どの数字がなんの値なのか混乱してしまいますね。

たとえば、歩行モードを変数 mode_flag を使って管理するとしましょう。この変数は、十字キーの上が押されていたら1、下が押されていたら2、左なら3、右なら4、何も押されていなければ0と値を変えていくようにして、あとで分岐（ぶんき）に使うことにします。

しかし、 mode_flag == 2 などといった条件式を見て「2 って前後左右どのモードだっけ……」と悩んでしまうのはミスの元ですし、何より面倒です。もし mode_flag == BACK などと書ければ、後退（バック）であることがすぐにわかるでしょう。

この書きかえは、これまでにも何度も登場した #define を使えば良さそうですね。必要に応じて、さまざまな値を #define で置きかえてしまいましょう。

今回のプログラムの数値を置きかえるのに、役立ちそうな英単語をまとめておきます。

POINT

FORWARD（前へ）	BACKWARD（後ろへ）	LEFT（左）	RIGHT（右）
BODY（胴体）	FOOT（足）	ARM（腕）	
STOP（停止）	ANGLE（角度）	FLAG（フラグ）	
MODE（モード）			

2.　まとめ（目安５分）

二足歩行ロボットを構成するいろいろな要素、音、映像、センサー、サーボモーターをプログラム上で統合していく中で起こったことを振り返りましょう。

1）組み立て工程

動かしたときに、関節の駆動範囲内でメカがぶつからないように設計することで、開発中に故障するような事故を防ぐことができます。

2）超音波距離センサーの誤動作対策

超音波距離センサーは電源ノイズに敏感で誤動作をしていましたが、電源ノイズの原因になるマトリクスLEDの点灯を、超音波距離センサーの計測中だけ止めることで、誤動作を回避することができます。電気回路が原因の問題ですが、電気回路を回収することなく、プログラムでも問題解決が可能です。

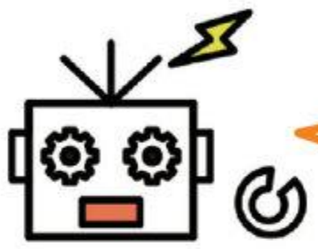
3）マイコンのタイマー機能

マイコンボードに使われているAVRマイコンのタイマー機能は３つしか搭載されていないうえ、そのうちの１つはシステムに専有されているため、タイマー機能を使うライブラリは２つまでしか使えません。タイマーを使わなくても処理できる機能についてはタイマーを使わないようにするなど工夫をすることで、この問題も回避できます。どうしても回避できない場合は、より多くのタイマーを搭載したマイコンにシステムを移行することになります。

4）歩行動作の状態遷移

複数のサーボモーターなどを使用してロボットの歩行システムをつくるときには、あらかじめ状態遷移図やフローチャートで動作の仮説（推測）をたてて、動作順序を決めていきます。
一方で、動作順序をプログラムに落とし込むときには後々にプログラムを変更・編集しやすいように `#define` やデータ型変数を用意して管理することが大事です。

今回で、二足歩行ロボットを使った、ロボットを構成するいろいろな要素を統合していく手順を実体験する講習はおしまいです。本当に、いろいろな問題が起こりましたが、プログラムを駆使して、全て回避することができました。
オリジナルのロボットを作ろうとすると、教科書はありませんので、起こった問題は自分で解決するしかありません。そんなときは、いろいろな問題を解決した今回の経験を思い出して、諦めないで問題を解決しようとする探究心を奮い立たせてください。
問題が起こるということは、問題を起こしている原因があります。その原因さえわかれば、問題を解決する方法は必ずあります。



二足歩行ロボットはこれにて終了だよ。
今回学んだことをいかして、将来オリジナルロボット作りに役立ててね。

講

○　以下の授業の目標を再確認します。
　・プログラムの「バージョンアップ」を行う
　・二足歩行ロボットのシーケンスを学ぶ
　・二足歩行ロボットのプログラムを完成させる
○　今回のターム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。
○　次回は「不思議アイテムⅢ-1」になります。

《次回必要なもの》

次回は、以下のパーツを持ってきてください。

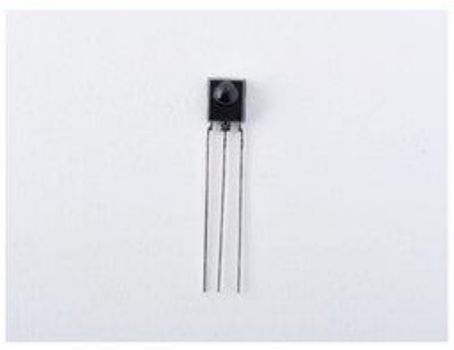
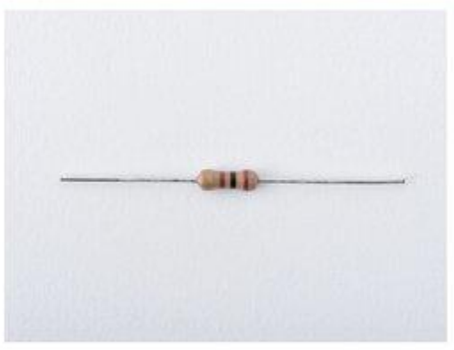
ラジオペンチ 1	USB ケーブル 1	マイコンボード 1	ロボプロシールド 1
			
マトリクス LED シールド 1	マトリクス LED 1	スピーカー 1	301 ブレッドボード 1
			
ジャンパー線 65	10 kΩ抵抗 10	タクトスイッチ 10	赤外線リモコン 1
			
赤外線 LED 1	赤外線受光素子 1	100 Ω抵抗 1	
			

図2-0　次回必要なもの