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②

第4回

六脚(ろっきゃく)ロボットの歩行制御(ほこうせいぎょ)（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。

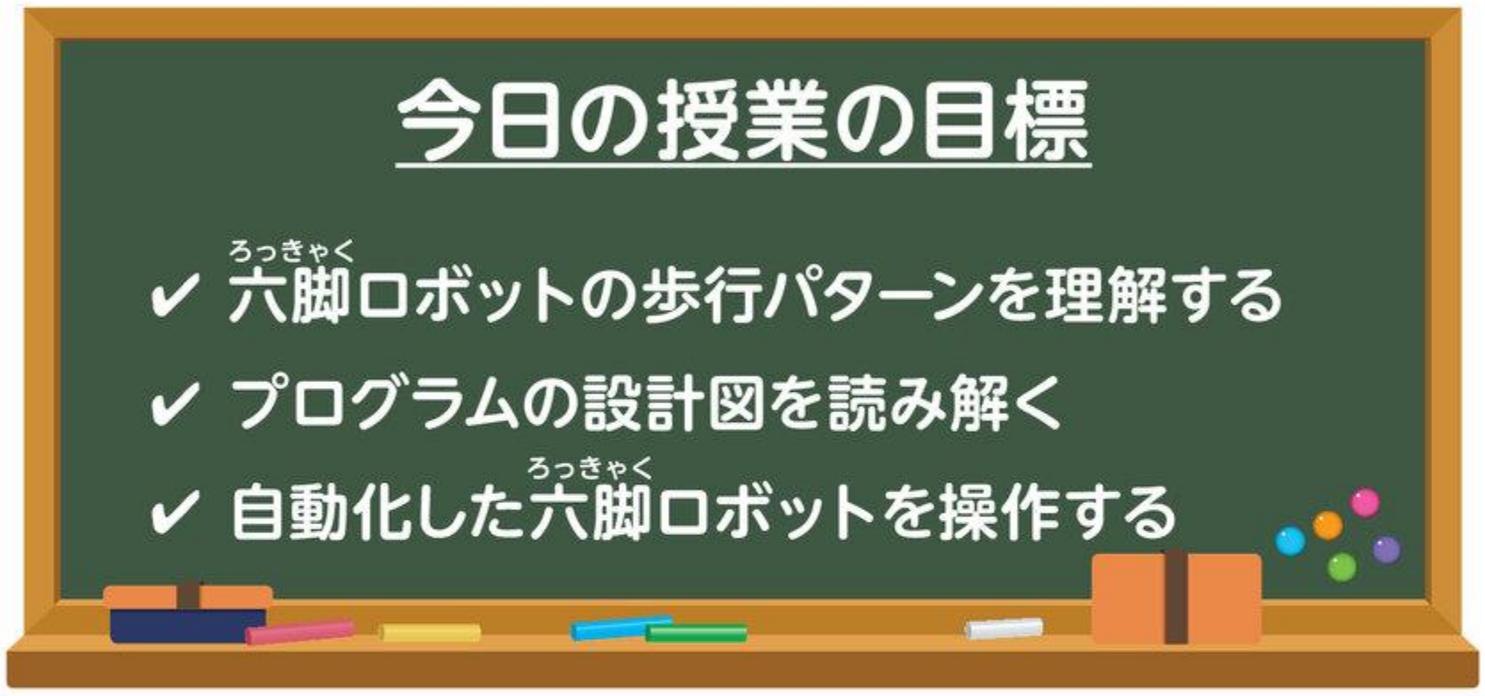
○　今回の目標をパネルで用意するか、黒板に予め書いておきます。

（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. 六脚ロボットの歩行制御（後編）（目安５分）

0.0. 「六脚ロボットの歩行制御（後編）」でやること



今回の授業は、「六脚ロボットの歩行制御（後編）」です。
前回は、六脚ロボットをコントローラーのボタン操作で歩行させました。
しかし、ボタン1個につき、ひとつのサーボモーターの制御をするような操作は、慣れるまではとても大変で面倒だったかと思います。
今回は、この歩行運動をマイコンボードにプログラミングすることによって、ボタンひとつで前進・後退・左右旋回などを自動的に再現する方法を勉強していきます。

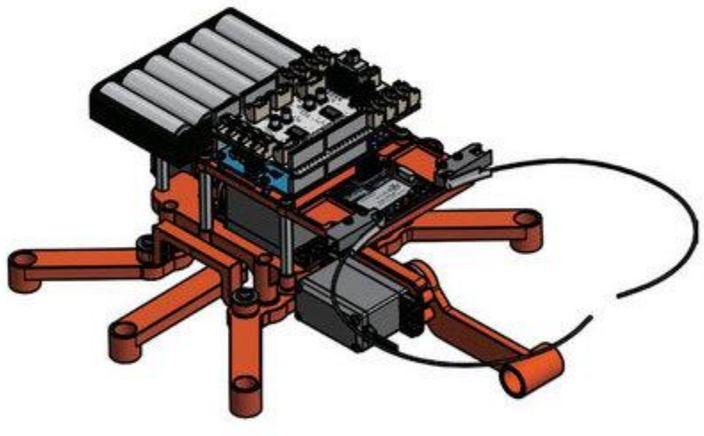
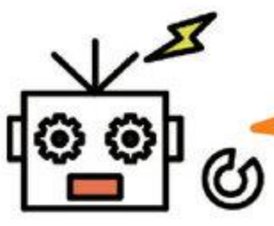


図 0-0　六脚ロボットの完成図



モーションの設計には、ジミチな観察が必要ダヨ。

0.1.　必要なもの

前回までに製作した六脚ロボットを使用します。コントローラーを使いますので準備してください。昆虫の触角のかわりに、タッチセンサーと針金、固定用ビニールチューブで障害物センサーをつくっています。タッチセンサーの固定には、M2L12タッピングネジを使います。

USBケーブル 1	コントローラー 1	ACアダプター 1
		

図 0-1　必要なもの

0.2.　六脚ロボットの注意点

前回は、プログラムで原点位置を補正するテクニックも勉強しましたね。
ロボットの点検方法は前回のテキストを参考にしましょう。

⚠注意！

複数のサーボモーターを同時に動かすと、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。ロボットを持つときは、背中の方を持ち、脚や角の近くを持たないように注意しましょう。

1.　六脚ロボットの歩行動作を設計する（目安 70 分）

1.0.　前回のおさらい

「RemoteTest2」では、**図1-0**のボタン操作で六脚ロボットを歩行させました。
ボタン操作の組み合わせによって、脚の運び方をかえて前後左右への移動を実現しました。
冒頭でもふれたように、この操作にとても不便さを感じたり、操作途中で混乱して挫折した人もいるのではないでしょうか？
しかし、この操作のパターンを分析することは、もっと簡単な操作を考えるうえで欠かせません。もう一度、しっかりと操作手順を整理して、歩行動作のプログラムを作りましょう。
こうしたひとつひとつの工程を分析して、プログラムの処理の流れを詳細にまとめることを、「プログラムの設計」といいます。

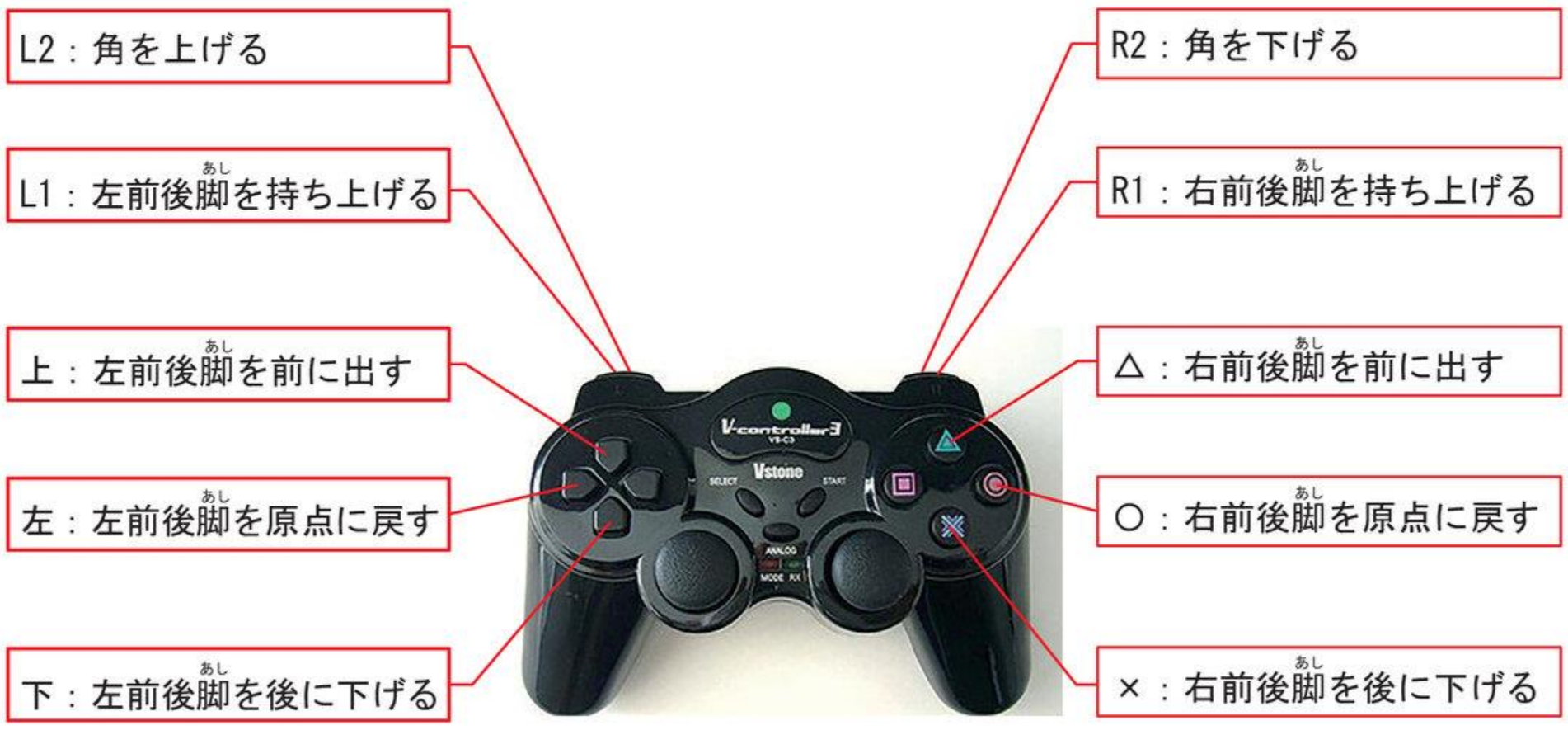
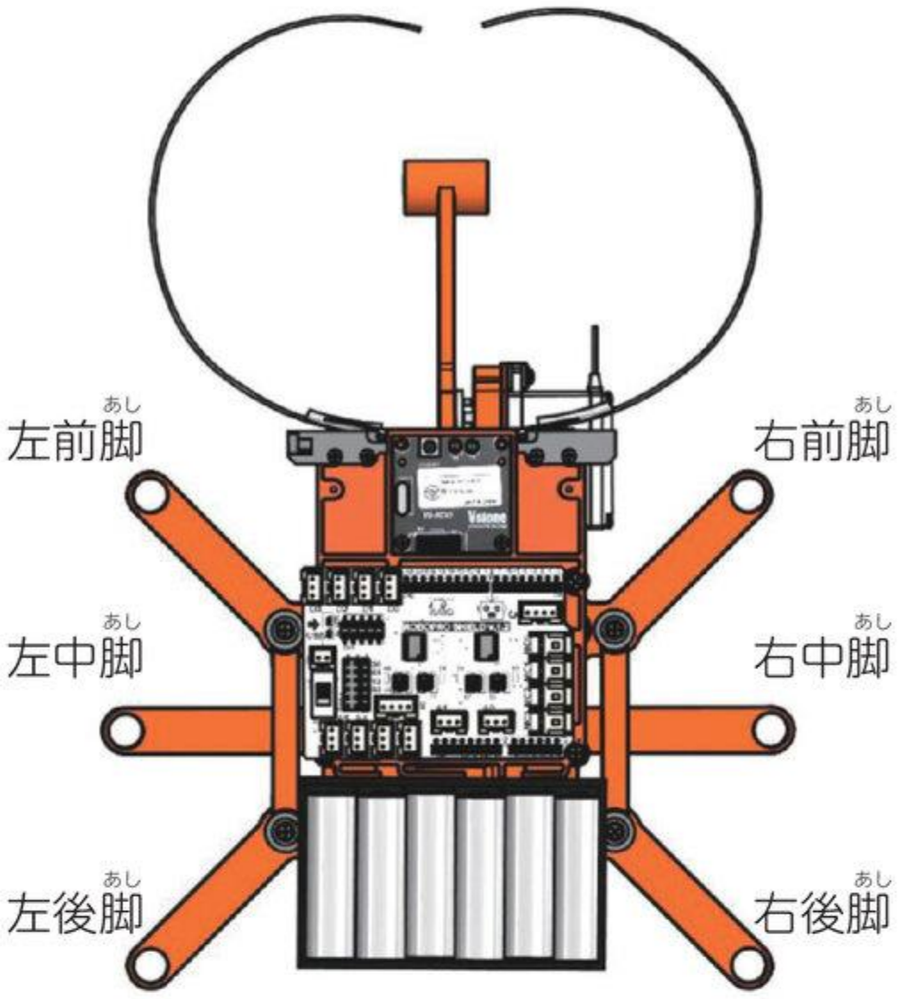


図 1-0 「RemoteTest2」でのボタン操作

では、歩行パターンの復習です。六脚ロボットの中脚をコントローラーの L1 、R1 ボタンで操作します。
図1-2の前回使った歩行パターンのメモの図を使いますので、見方を思い出しておきましょう。
三角形のそれぞれが、左右の前後脚の運び方を表現したものです。

【上から見たところ】



【下から見たところ】

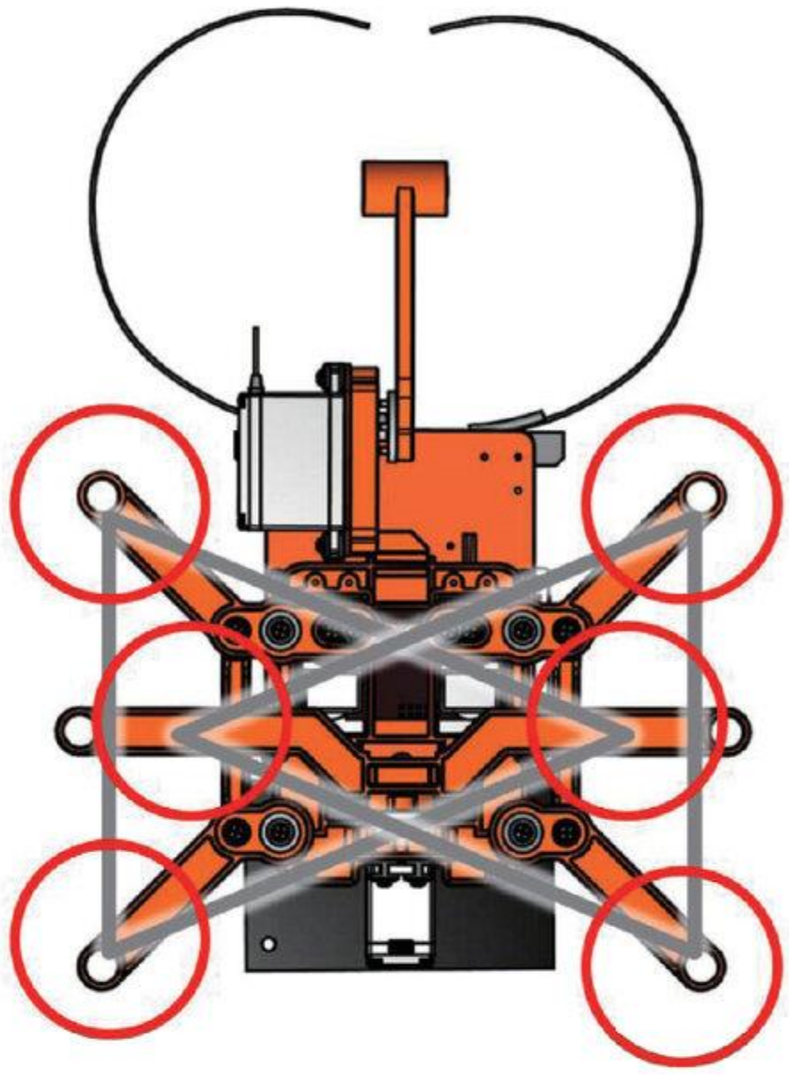
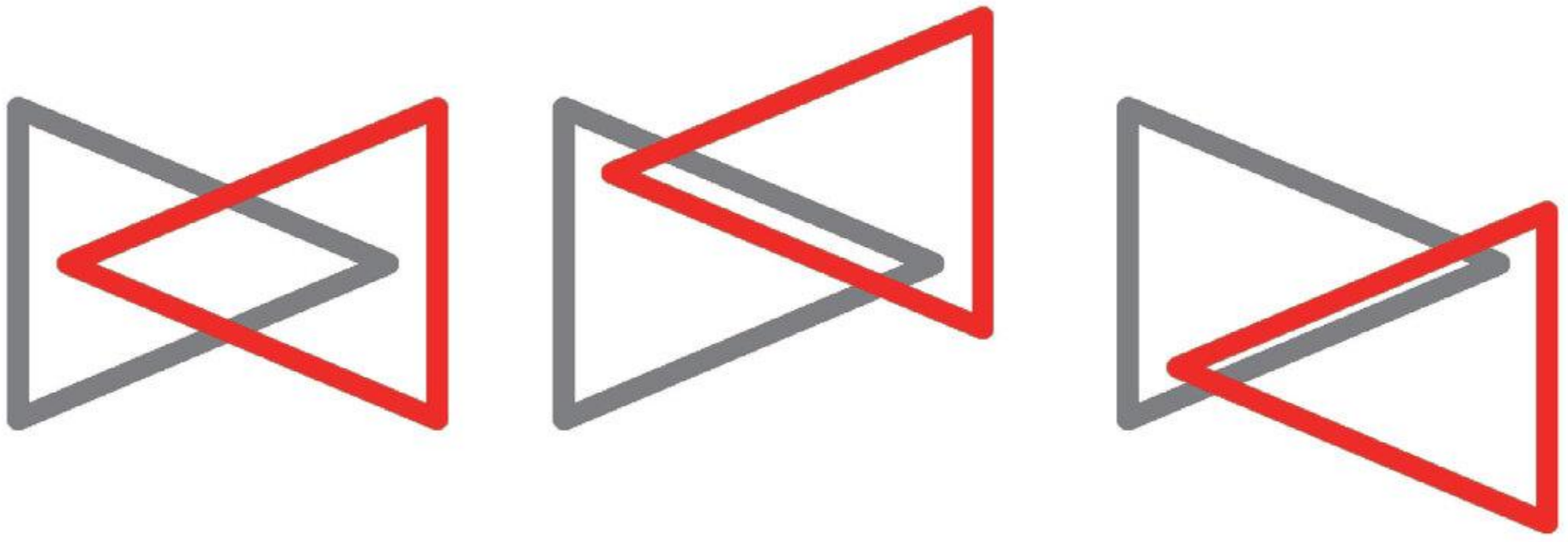


図 1-1　左右脚のリンク



R1ボタンを押して右前後脚を遊脚させた状態	R1ボタンを押して右前後脚を遊脚させた状態で、さらに△ボタンを押して右前後脚を前に出した状態	R1ボタンを押して右前後脚を遊脚させた状態で、さらに×ボタンを押して右前後脚を後に出した状態

図 1-2　六脚ロボットの歩行パターンのメモ例

プログラムの書き込み

RoboticsProfessorCourse3 ＞ HexRobot 3 ＞ RemoteTest 2

使用するコントローラーのボタンは、十字ボタンの上、下、左と、○、×、△ボタン、そして、L1、R1ボタンです。
図1-0のコントローラーのボタン操作と、**図1-2**の歩行パターンを確認しながら六脚(ろっきゃく)ロボットの歩行動作をマスターしましょう。

図 1-3　使用するコントローラーのボタン

1.1.　歩行動作の設計①　歩行動作を図解してみる

ここでは、前回の最後にあった「チャレンジ課題」の答え合わせをしていきましょう。前回取り組んでいない人は、ここで確認しましょう。

1）　前進歩行の動作手順

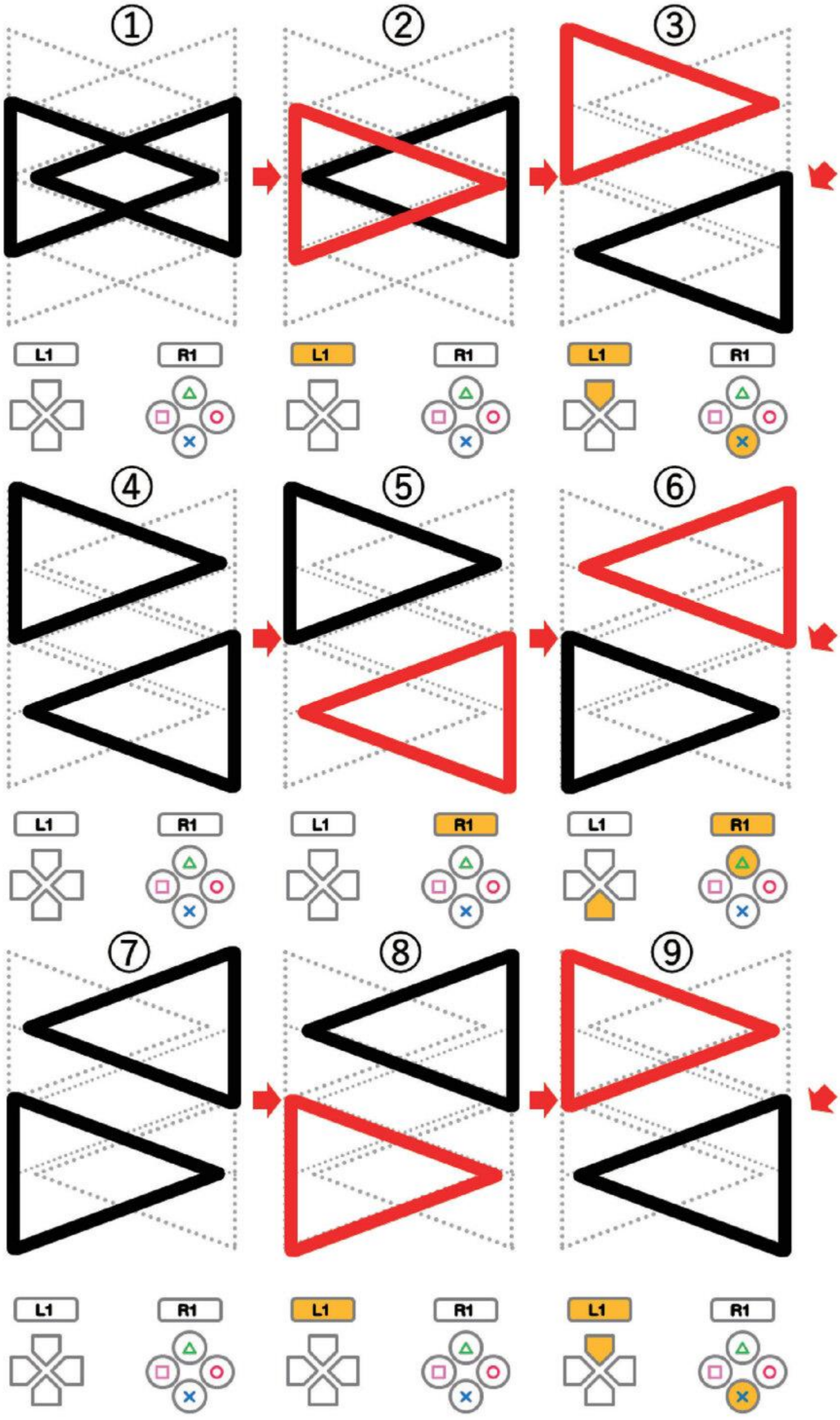


図 1-4 前進歩行の動作手順

2）　後退歩行の動作手順

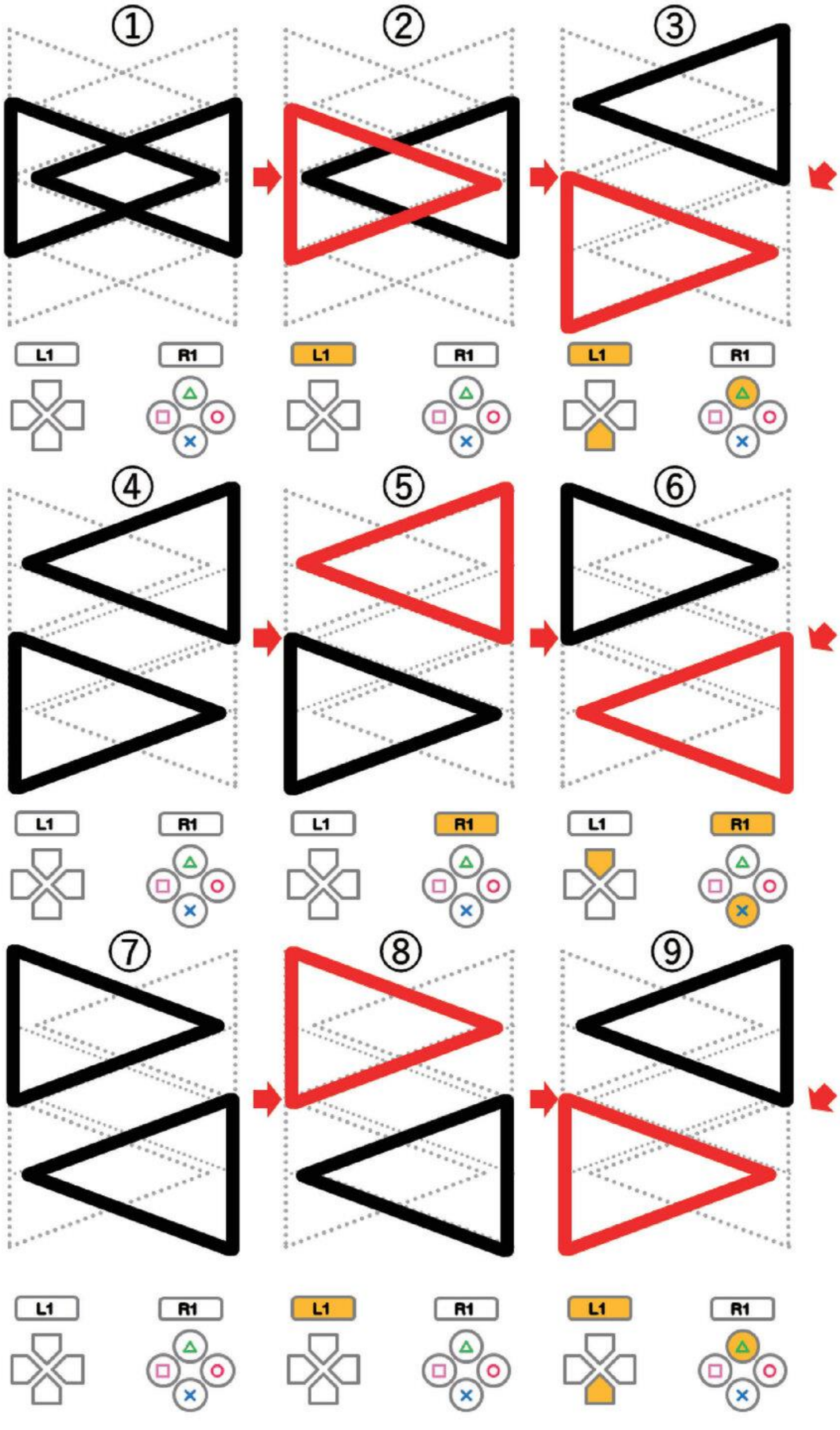


図 1-5　後退歩行の動作手順

３）　左旋回の動作手順

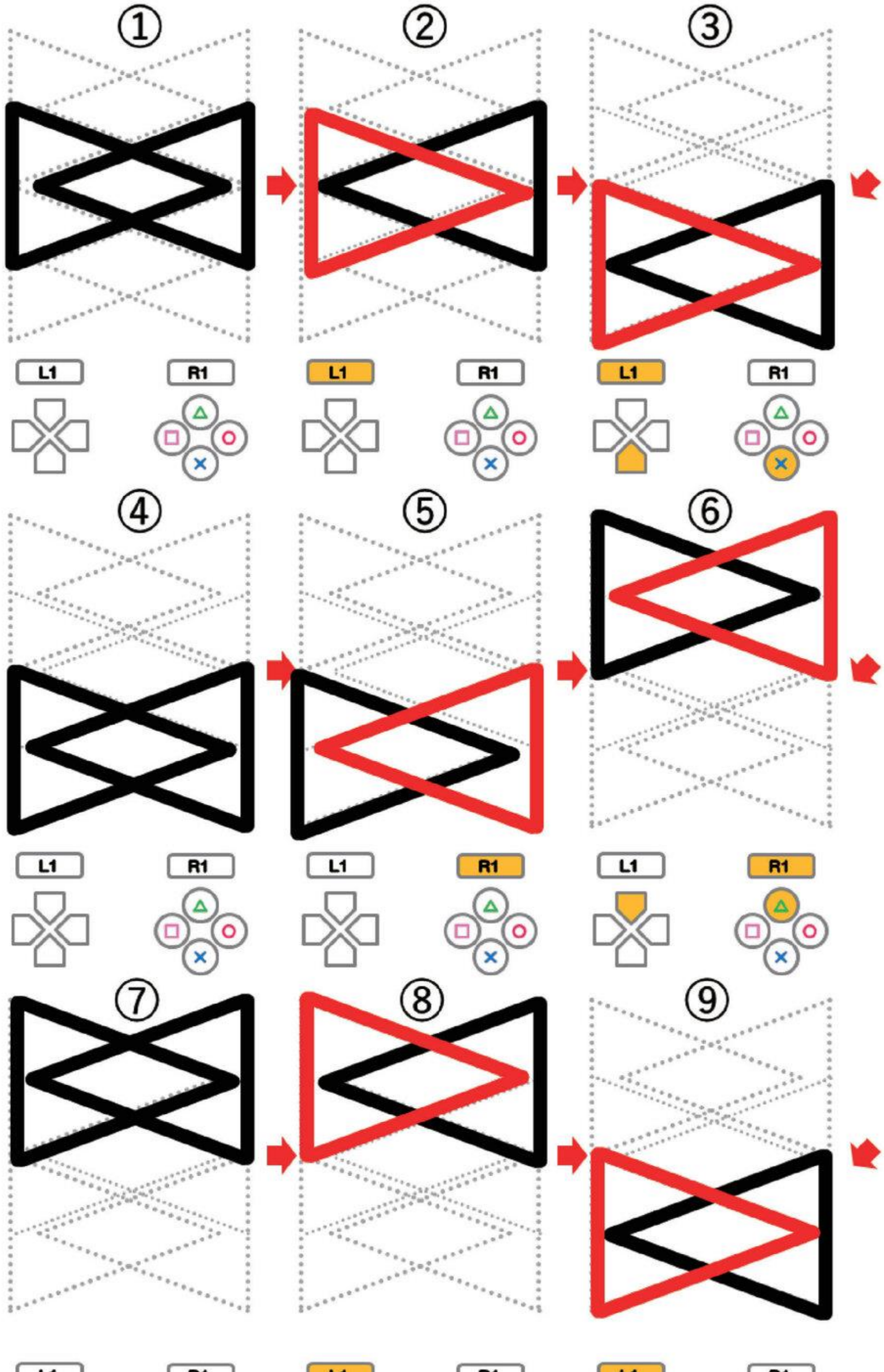


図 1-6 左旋回の動作手順

4)　右旋回（せんかい）の動作手順

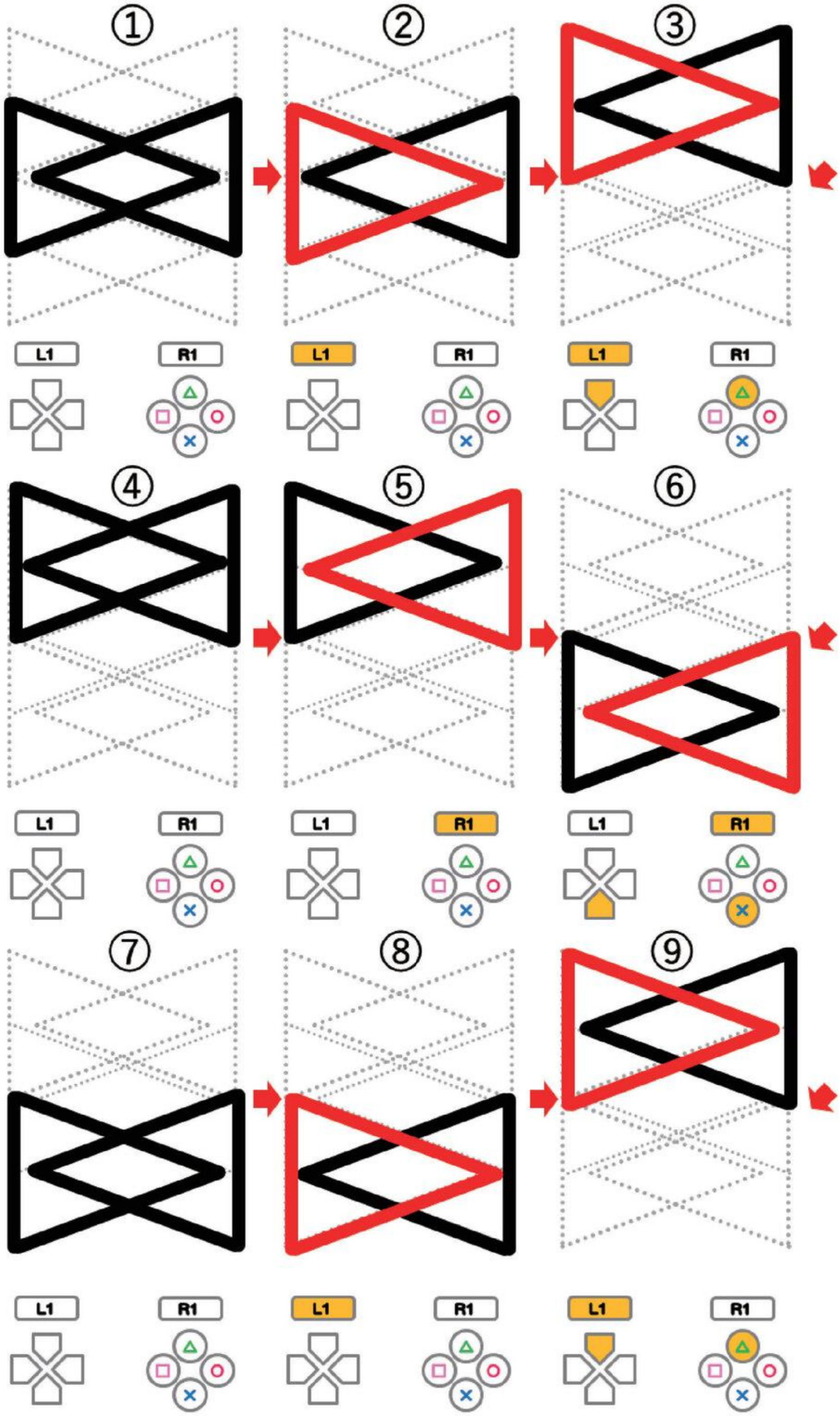


図 1-7　右旋回（せんかい）の動作手順

1.2.　歩行動作の設計②　歩行動作を表にまとめる

コントローラーで操作し、手順を確認することはできましたか？ 次はこの動作をプログラムによって自動化させるために、先ほどの操作の手順を表にまとめて整理していきます。
図1-4～**図1-7**で9つに分割（①～⑨）した手順を表に整理したものが、**表1-0**です。

表 1-0　歩行動作の一覧

	①	②	③	④	⑤	⑥	⑦	⑧	⑨
前進	左右接地	左遊脚	左脚前 右脚後	左右接地	右遊脚	左脚後 右脚前	左右接地	左遊脚	左脚前 右脚後
後退	左右接地	左遊脚	左脚後 右脚前	左右接地	右遊脚	左脚前 右脚後	左右接地	左遊脚	左脚後 右脚前
左旋回	左右接地	左遊脚	左脚後 右脚後	左右接地	右遊脚	左脚前 右脚前	左右接地	左遊脚	左脚後 右脚後
右旋回	左右接地	左遊脚	左脚前 右脚前	左右接地	右遊脚	左脚後 右脚後	左右接地	左遊脚	左脚前 右脚前

表1-0をよく見ると、①～③と⑦～⑨の手順は同じであることがわかります。
また、①、②、④、⑤はそれぞれ、全ての歩行パターンで同じ動作をしていますね。

やってみよう！

上の文章を読みとって、歩行動作の一覧表をもっと単純にまとめてみよう。
①～③の欄を参考に、手順④から先の欄を記入してみてね。不要な欄があれば×をかいてなくしてしまおう！

	①	②	③	④	⑤	⑥	⑦	⑧	⑨
前進	左右接地	左遊脚	左脚前	左右接地	右遊脚	左脚後	×	×	×
後退			右脚後			右脚前			
左旋回			左脚後			左脚前			
右旋回			右脚前			右脚後			
			左脚後			左脚前			
			右脚後			右脚後			
			左脚前			左脚後			
			右脚前			右脚前			

講　重複する動作は、プログラムもできるだけまとめて単純化したほうが見やすく、データ量も手間も省けます。こうした点は、表に整理することで発見しやすくなります。

次に、「やってみよう!」で単純化できた内容をプログラムに適した文章にすると、次のような内容になります。

POINT

ステップ①：中脚の目標位置を原点に設定する。

ステップ②：中脚の目標位置を、左前後脚を遊脚させるように設定する。

ステップ③：歩行モードに応じて、左前後脚と右前後脚に所定の目標位置を設定する。
- ・歩行モード前　進：目標位置は、左前後脚は前で右前後脚は後
- ・歩行モード後　退：目標位置は、左前後脚は後で右前後脚は前
- ・歩行モード左旋回：目標位置は、左前後脚は後で右前後脚も後
- ・歩行モード右旋回：目標位置は、左前後脚は前で右前後脚も前

ステップ④：中脚の目標位置を原点に設定する。

ステップ⑤：中脚の目標位置を、右前後脚を遊脚させるように設定する。

ステップ⑥：歩行モードに応じて、左前後脚と右前後脚に所定の目標位置を設定する。
- ・歩行モード前　進：目標位置は、左前後脚は後で右前後脚は前
- ・歩行モード後　退：目標位置は、左前後脚は前で右前後脚は後
- ・歩行モード左旋回：目標位置は、左前後脚は前で右前後脚も前
- ・歩行モード右旋回：目標位置は、左前後脚は後で右前後脚も後

※全ての脚が目標位置に到達すると、次のステップに移行する。ただし、⑥の次は①に戻る。

しかし、まだステップ①～ステップ⑥の流れでは、肝心の歩行モードがどのタイミングで設定されるのかがかかれていません。前回のプログラムと同様に、コントローラーの状態取得は「タイマー割り込み」で行い、その中で歩行モードを設定する必要があります。ですから、この①～⑥の処理に**図1-8**のようなタイマー割り込み処理を入れていきます（フローチャートは、**図1-10**です）。

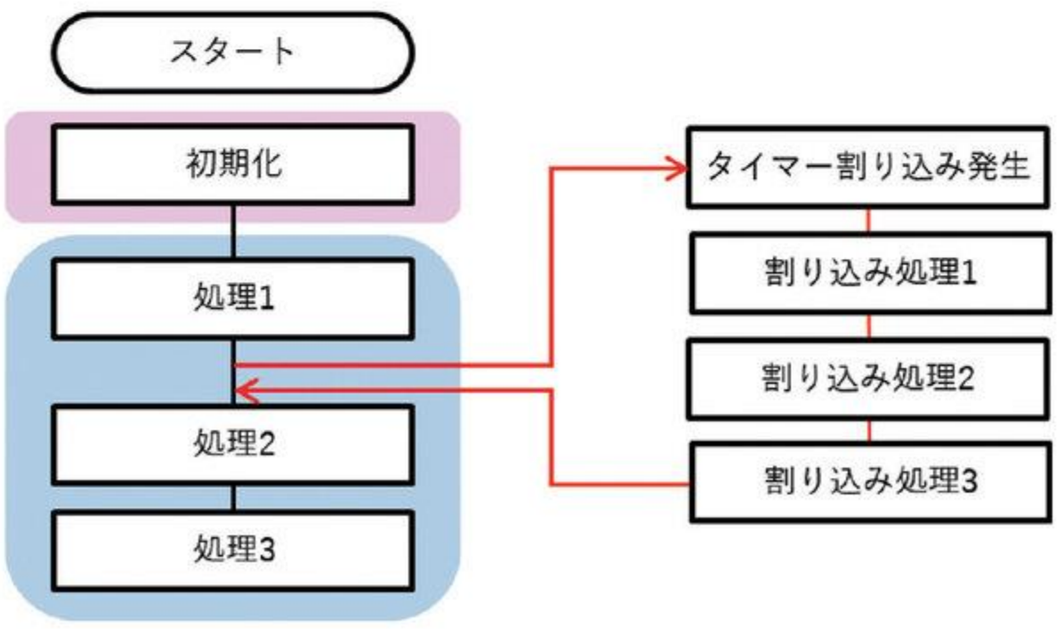


図 1-8　メインループ処理と割り込み処理イメージ

1.3.　歩行動作の設計③　フローチャートにまとめる

では、フローチャートにして検証します。**図1-9**はタイマー割り込み処理の部分です。コントローラーの情報の取得（歩行モードの管理）をしています。見てみると、前進でも後退でも左旋回でも右旋回でもない場合、つまりコントローラーの十字ボタンがどの方向にも入力されていない場合の処理がぬけていることに気づきませんか？　このままだとプログラムに問題を残すことになります。追加しておきましょう。

やってみよう！

図1-9の A にはどんな処理が入るかな？
必要だと思う処理の内容を、以下に記入しよう。

歩行モードを停止にする

答えはこの後確認するよ。

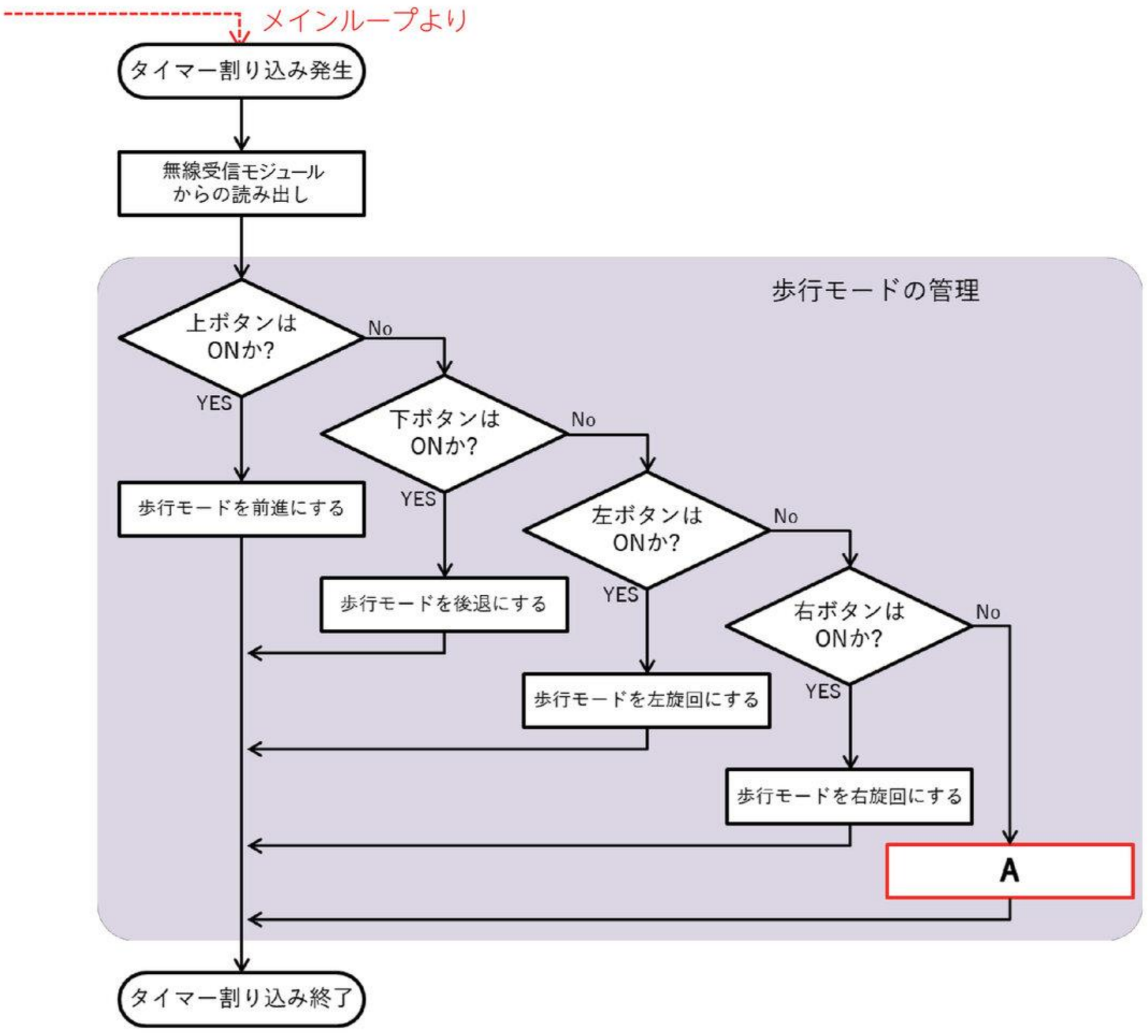


図 1-9　タイマー割り込みのフローチャート

続いて、メインループの歩行動作のフローチャートです。見えにくい人は、巻末に拡大図があるので、そちらで確認しましょう。前の部分でまとめた①～⑥の手順（ステップ）ごとに分岐させていますね。全体の流れをとらえましょう。なお、B と C にも A と同じく停止に関する命令が入ります。具体的にどういう命令が入るのか考えてみましょう。答えは、この後確認します。

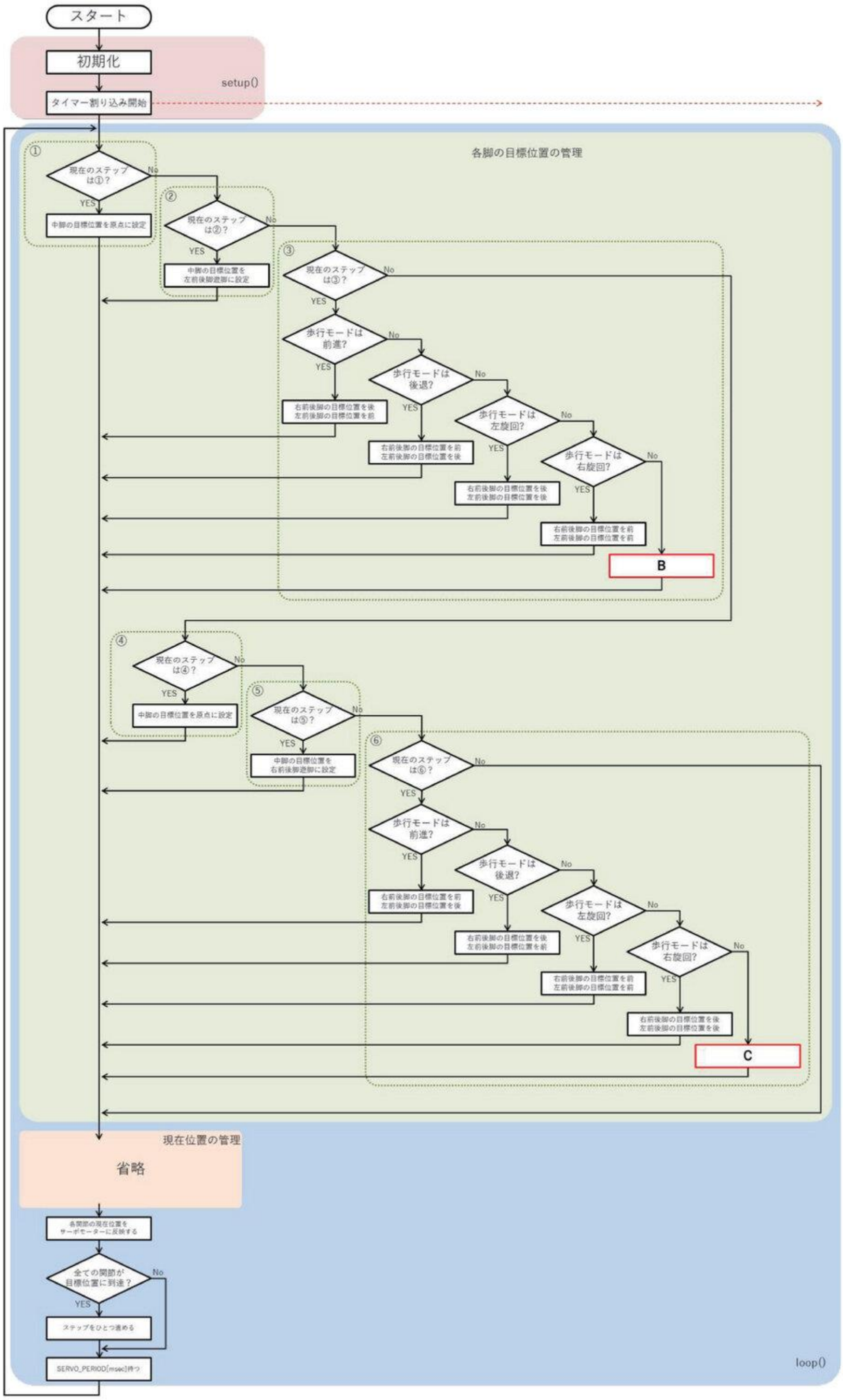


図 1-10　歩行動作のフローチャート

1.4. 歩行動作の再設計

六脚ロボットの動きを自動化するために、歩行動作のパターンを様々な形で整理しました。ロボットのプログラムを完成させるには、このような欠陥を見落とさないような地道なチェックを重ねることがとても大切です。なお、さきほどのフローチャートの A ～ C の空欄部分には具体的な停止命令として、以下のようなものが入ります。

A：歩行モードを停止にする。
B：右前後脚の目標位置を原点、左前後脚の目標位置を原点にする（停止）。
C：右前後脚の目標位置を原点、左前後脚の目標位置を原点にする（停止）。

チャレンジ課題

プログラム「RemoteTest2」をかきかえ、**図1-9**、**図1-10**の動作をするプログラムにしてみよう！

ヒント

「RemoteTest2」から変更しなければならないのは「歩行モードの実装」「動作ステップの実装」「コントローラーのボタンを押されたとき、目標位置の更新をするのではなく、歩行モードの変更を行うようにする」「現在の動作ステップと歩行モードから、次の目標位置を決めるようにする」などの点だね！

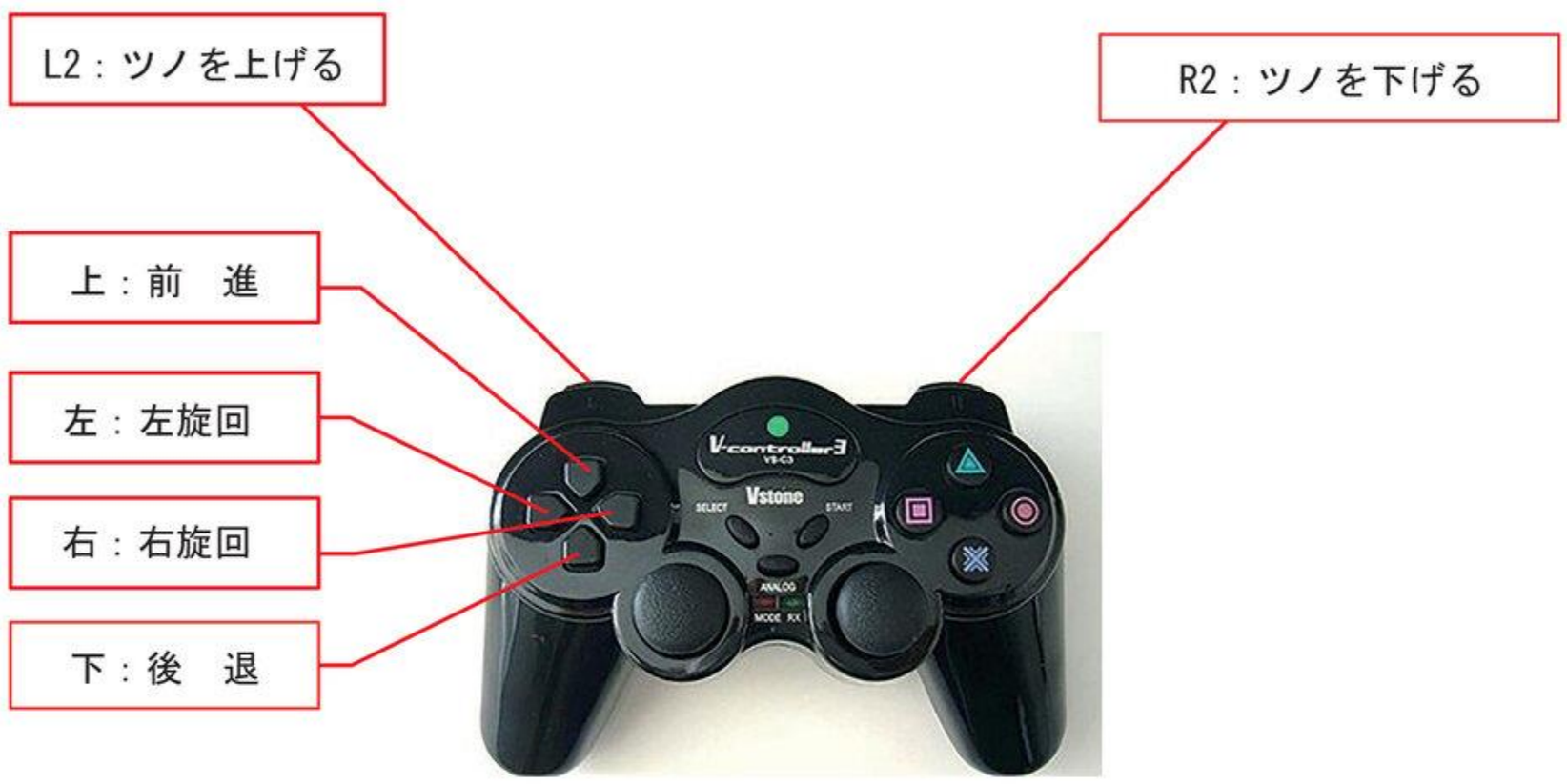
図1-9、**図1-10**のフローチャート通りにプログラミングしたものが、「RemoteWalk0」です。
ではプログラムを実行してみましょう。

プログラムの書き込み

RoboticsProfessorCourse3　＞　HexRobot 4　＞　RemoteWalk 0

やってみよう！

先ほどのフローチャートの流れに従って実際にロボットを動かしてみよう。急遽（きゅうきょ）付け足した、「停止」の命令で、思い通りの動作になっているかな！？ ちなみにフローチャートにかかれていない、L2 、R2 ボタン操作によるツノの動作もおまけでプログラムされているよ。



さあ、どうでしょうか。思い通りの動きができましたか？

講

「RemoteWalk0」を実行してわかることですが、図 1-9 と図 1-10 のフローチャート通りにプログラミングをすると、コントローラーの十字キーを触っていない「停止」の状態でもロボットは足踏みを続けています（中脚が動き続けます）。
そのまま止まらずに動きつづけても特に問題はありませんが、電源を切ったりするときに扱い難いため、「完全停止」するための対策を生徒に考えさせてください。

ステップアップ

プログラム「RemoteWalk0」を実行すると、歩行モードが停止のときも中脚が動き続けてしまうことがわかるね。この問題を解決する方法を考えよう。
方法はたくさんあるけど、たとえば**図1-10**のフローチャートの中に「歩行モードが停止かそれ以外か」の条件分岐を追加することでも解決できるよ！
フローチャートのどの部分に条件分岐をつけたせばいいか、また、分岐したあとどのような動作をさせればいいか考えて、説明をかいてみよう！
また、その解決策通りになるよう、プログラムをかきかえてみよう！

ステップ②および⑤のときに歩行モードが停止かどうかをみて、もし歩行モードが停止であれば中脚の目標位置を変更せずに次のステップに進むようにする。歩行モードが停止ではない場合は、中脚の目標位置を左前後脚遊脚に設定する。

次のページに、解答例のフローチャートを示します。うまく説明がかけなかったという人は、フローチャートを参考にして改めて説明を考えてみるとよいでしょう。

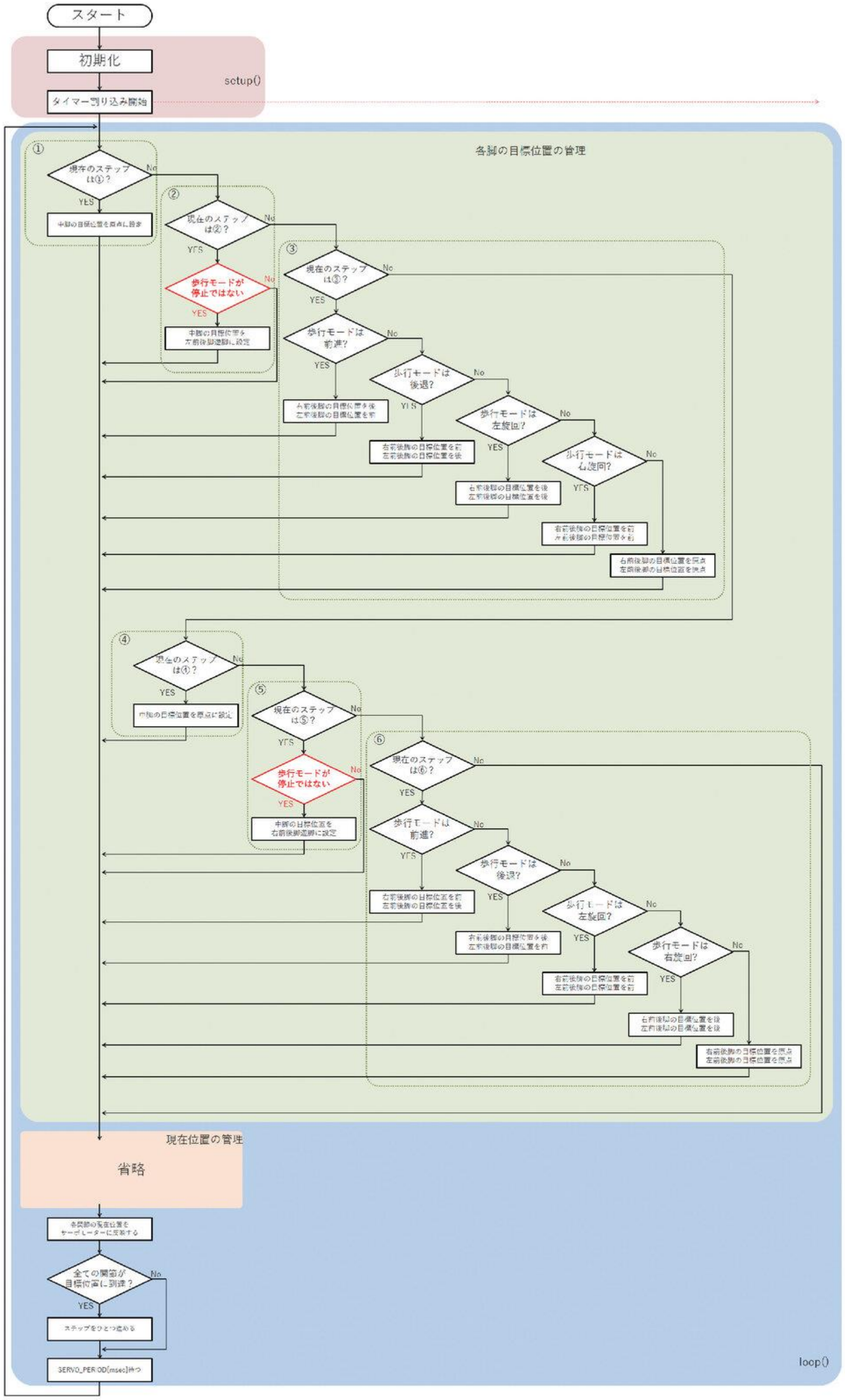


図1-11 「RemoteWalk1」のフローチャート

図1-11のフローチャートではステップ②とステップ⑤で、遊脚のため中脚の目標位置を設定する際に、動作モードが「停止」となっている場合には、中脚の目標位置を原点位置に戻して、脚を上げないよう（足踏みをさせないよう）にしました。

この改良を行ったのが、以下のプログラムです。実行して動きを確認しましょう。

プログラムの書き込み

RoboticsProfessorCourse3　＞　HexRobot 4　＞　RemoteWalk 1

解決方法は、この他にもいくつか考えられます。

たとえば「そもそも、動作モードが停止であれば、歩行ステップを進める必要が無い」という考え方でもプログラムを改良できますね。

チャレンジ課題

プログラム「RemoteWalk0」をかきかえ、上の文章通りの動きになるようにしてみよう！
条件分岐、つまりif文をどこに追加すればいいか、またどんなif文を作成すればいいかわかるかな？

ヒント

「AかつB」というときの条件式は「A && B」、「CまたはD」というときの条件式は「C || D」だったね！

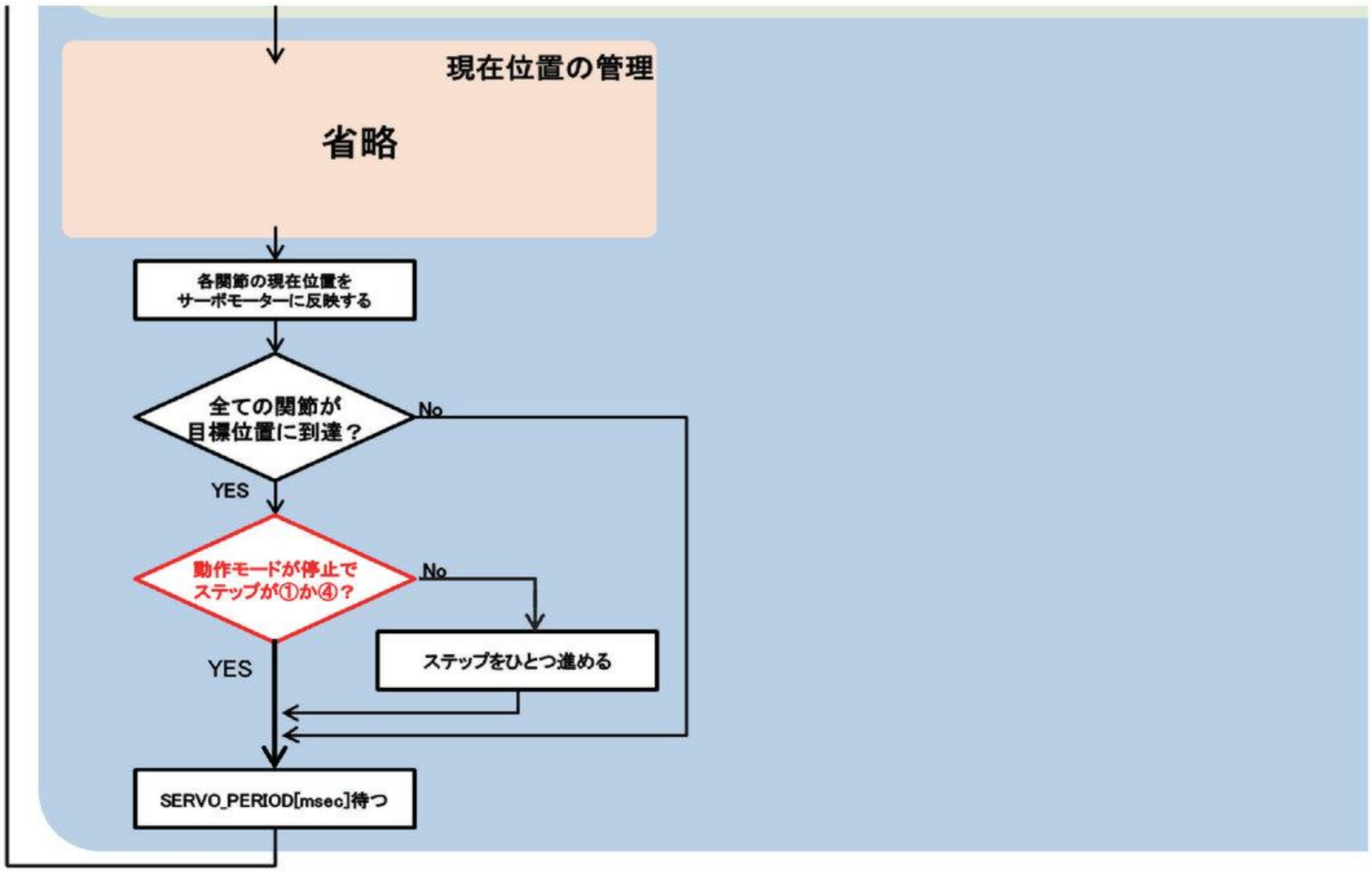


図 1-12　２つめの解決方法の例

講

解答例は以下のプログラムです。
RoboticsProfessorCourse3　＞　HexRobot 4　＞　RemoteWalk 2
紹介する解決方法は、あくまでも例になります。YES ／ NO の条件分岐のパターンは、他にも考えられます。プログラム設計は、開発者の考えによって詳細にも簡素にもなります。また、フローチャートは巻末にも拡大図を用意しております。見難い場合は、参照するようにご指導ください。

2.　プログラムのまとめ（目安 30 分）

2.0.　「RemoteWalk2」のポイント

プログラム「RemoteWalk2」より抜粋

```
#define    MODE_STOP          0    // 歩行モード停止
#define    MODE_FORWARD       1    // 歩行モード前進
#define    MODE_BACKWARD      2    // 歩行モード後退
#define    MODE_TURN_LEFT     3    // 歩行モード左旋回
#define    MODE_TURN_RIGHT    4    // 歩行モード右旋回

#define    STEP_1             0    // 動作ステップ１
#define    STEP_2             1    // 動作ステップ２
#define    STEP_3             2    // 動作ステップ３
#define    STEP_4             3    // 動作ステップ４
#define    STEP_5             4    // 動作ステップ５
#define    STEP_6             5    // 動作ステップ６

int    mode_flag;          // 歩行モードの管理用
int    step_flag;          // 動作ステップの管理用
```

図1-9の「タイマー割り込み」で更新される5つの歩行モード（停止、前進、後退、左旋回、右旋回）を他の処理に反映させるため、mode_flagという変数を用意します。
ここで、5つの歩行モードを数字で表してやり取りさせるわけですが、プログラム中に直に数字をかいていくと、あとから見ると、どの歩行モードを意味していたのかわからなくなることがあります。そこで、MODE_STOPのように、文字列で置きかえるように定義しておくと、プログラム中で歩行モードがわかりやすくなります。
同じように、「やってみよう!」でまとめた歩行動作の6段階の動作ステップを管理するための変数step_flagも、動作ステップを文字列で定義します。

図1-11のフローチャートの緑色の部分、各脚の目標位置を歩行モード(mode_flag)と動作ステップ(step_flag)に応じて管理しています。

2.1. タイムアタックレース

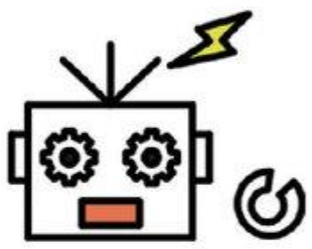
障害物（しょうがいぶつ）や三角コーンなどを用意してコースをつくり、実際につくったロボットを操作して、タイムアタックレースを開催してみましょう。 プログラムの以下の部分を調整してもOKです！

プログラム「RemoteWalk2」より抜粋（ばっすい）

```
#define    STRIDE_ANGLE    60  // 脚（あし）を前後に動かすときのサーボモーターを回す移動量
#define    LIFT_ANGLE      60  // 脚（あし）を上げるときのサーボモーターを回す移動量
#define    MOVE_RATE        5  // 現在値の更新をするときの１回あたりの進み量
```

3.　まとめ（目安５分）

今回は、コントローラーの十字キーで六脚ロボットを自由に歩行させられるよう、プログラムの改良を行いました。
フローチャートは、プログラムの設計図です。この設計図をつくっていく手順、考え方を、歩行動作を例にして紹介しました。さて、今日のところで六脚ロボットはだいぶロボットらしくなってきましたが、今のままでは、ただのラジコンとも言えます。次回は、プログラムをさらに改良して、六脚ロボットにちょっとした人工知能を搭載させてみましょう。



身近な歩行動作だけでもこんなに複雑だなんて。
人の心をプログラムしたらどうなるんだロウ？

≪次回必要なもの≫
今回使用したロボットと乾電池の他に以下を準備しておきましょう。

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

○　以下の理解度を確認します。
・六脚ロボットの歩行パターンを理解する
・プログラムの設計図（フローチャート）を読み解く
・自動化した六脚ロボットを操作する
○次回のテーマは「六脚ロボットの自律化」です。
引続き六脚ロボット本体とコントローラーを使います。