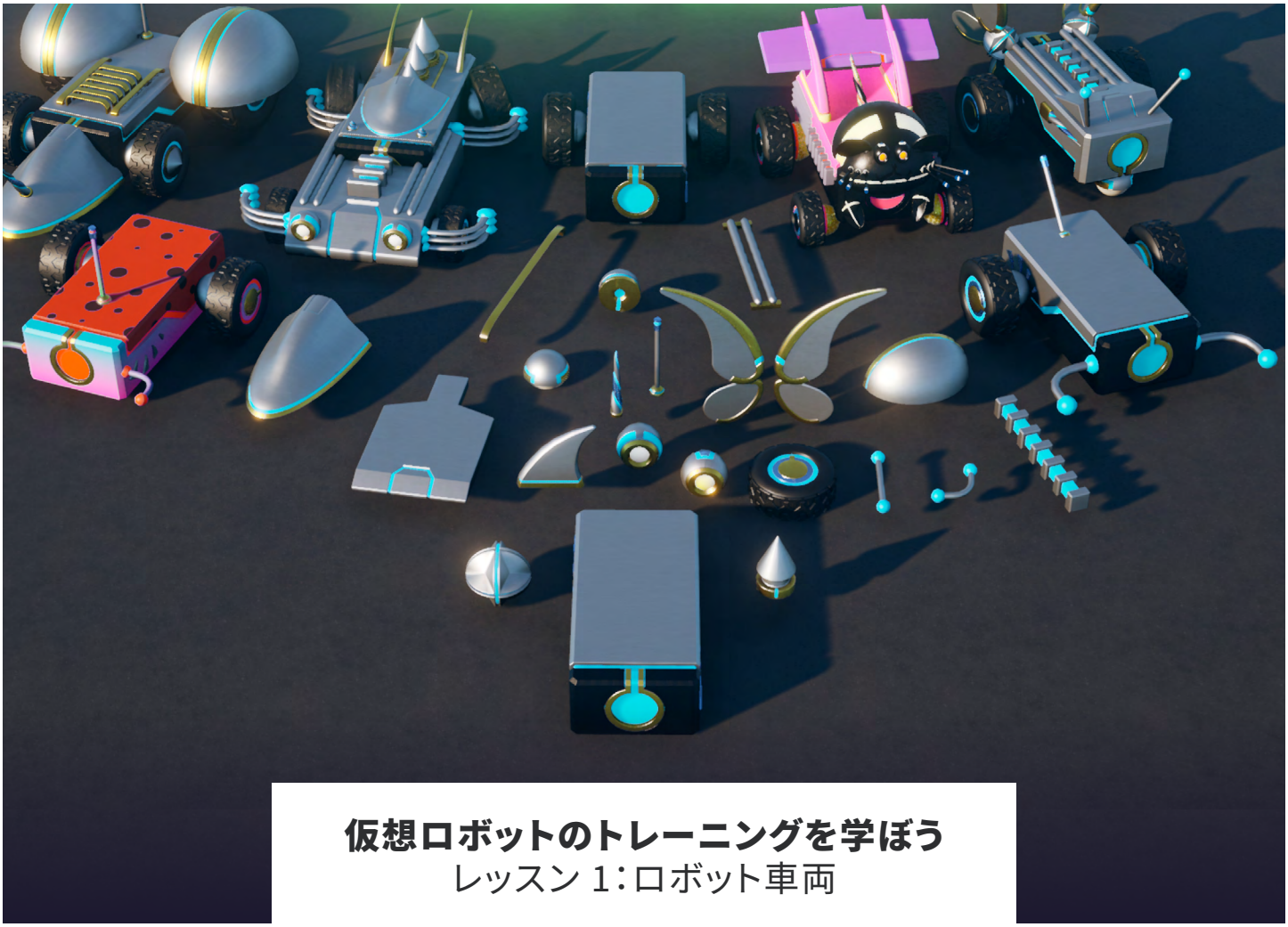
仮想ロボットのトレーニングを学ぼう

レッスン 1：ロボット車両

UNREAL ENGINE

生徒向けガイド

目次



仮想ロボットのトレーニングを学ぼう
レッスン 1：ロボット車両

目的

この Unreal Learning Kit レッスンは、生徒が仮想オンライン ロボットをコーディングして戦車のような動きで動かし、最も効果的なコーディングを選択して特定のターゲットに到達するように設計されています。生徒はロボット工学の原理の初歩を学びますが、コンピュータ サイエンスの概念と結び付けてロボットの動きをコーディングすることに重点を置いています。

はじめに

入門編ガイド

Unreal Engine を初めてインストールおよび使用する場合は、このアクティビティに進む前に「入門編ガイド」を完了してください。このガイドには、このアクティビティをうまく行えるよう、Unreal Learning Kit のファイルをインストールする手順が記載されています。

「入門編ガイド」はこちらにあります。

はじめに (続き)

レッスンの目的

生徒はこれらすべてのレッスンを順序どおりに進めることをお勧めします。そうすることで、Unreal Learning Kit の全体像、および Unreal Engine 環境でロボット制作とモーター/センサーのコーディングの両方がどのようにサポートされているかを把握できます。

Unreal Learning Kit では、応用物理を使用してロボット工学の概念を探ります。また、生徒にコーディングに関するフィードバックを速やかに与え、今後扱う可能性がある他の物理ロボット システムに応用できる原則やコーディング言語の知識を提供します。

アクティビティ 1 – ロボット車両

はじめに

このアクティビティでは、最小限の数のコマンドを使用して戦車のような動きで特定のターゲットに向けて移動するように、二輪ロボットをコーディングする方法を学習します。自分の部屋を最後に掃除したのはいつですか。先週から残っている食べかけのスナックを片付けてくれるロボット掃除機があれば便利ですね。飼っている猫や落ちているイヤホンが吸い込まれないようにすることはもちろんですが。
最近のロボット掃除機では、毎回迅速かつ効果的に掃除するために、多くの複雑なコマンドが、自動化された動きやアクションに単純化されています。エンジニアやプログラマーは必要とされる動きやコーディングをどのようにして判断したのでしょうか。

アクティビティの概要

このアクティビティでは、最小限の数のコマンドを使用して戦車のような動きで特定のターゲットに向けて移動するように、二輪ロボットをコーディングする方法を学習します。このアクティビティで扱う仮想ロボットの動作は、多くの家庭で使用されているロボット掃除機に似ており、前進、後退、右旋回、左旋回、回転に対応し、転倒しないように設計されています。また、一貫した結果を生み出せる安定したロボットの重量、車輪の位置、滑走球について、適切に設計されたロボットの工学と物理学の原理も解説します。

タスクを達成するようにロボットをプログラムするには、出力アクションを理解して伝えなければなりません。このやりとりは、人間とロボット装置の両方が理解できる特定のソフトウェア言語で実現しなくてはなりません。ここでは、Unreal Engine Robotics Learning Kit でブループリントのノードベースのコーディングを使用して、ロボットの動きを制御します。

プログラム コマンドを作成して、ロボットをテストします。そして、指定した目標を達成しなかった命令設定をコーディングに戻って調整し、目標が達成されるまで、仮想ロボットでコードを実行します。

コーディングのつながり

この最初のアクティビティで、生徒は Unreal Engine 内で移動し、コンテンツを構築することに慣れます。Unreal Engine は、Fortnite や Rocket League など人気のある多数のビデオ ゲームの制作に使用されているゲーム開発プラットフォームです。

レッスンを始める

ロボットの動き

目標まで移動するようにロボットにうまく指示するには、以下のスキルを順序どおりに習得する必要があります。

1. **Unreal Engine Robotics** 環境での移動と用語を学ぶ
2. **ロボットの設計**、および特定の強みのためにどのように構築されているかを探る
3. **戦車のようなロボットの動き**の概念、モーターの機能、コマンドのコーディングについて学ぶ
4. 作業とドキュメントの試行、成功、課題の結果を**観察する**
5. 学習したことを**復習**して、このレッスンの概念について完全に理解する

アクティビティ 1 では、車輪を回転させるようにモーターを制御することで、仮想ロボットを動かし始めます。現行のロボット モデルにはモーターが 2 つあり、それぞれを「モーター A」と「モーター B」と呼びます。それぞれのモーターが、右側と左側のある車輪を動

図 1 - 1

かします。前進するには、両方の車輪が同じ速度で同じ方向に同じ時間だけ回転する必要があります。モーターには方向、速度、持続時間のアクションを制御する設定があります。

実習

実習 1：バランスと設計

ロボットは通常、ある環境内で動作する物理オブジェクトであるため、その設計方法が動作に影響を及ぼします。適切に設計されたロボットに不可欠なコンポーネントを確認することから始めます。低摩擦の滑走球を 3 つ目の接地点とする二輪ロボット車両でのすべてのアクティビティに注目します。このモデルでは、ユーザーがわかりやすいように、ロボットの上面にある矢印がロボットの前部を示しています。

各モデルの動作を確認しましょうこれらのアニメーションを試してみるときに、ロボットのバランスに注目してください。

- **コンテンツ ブラウザ**で、**「Content」->「LearningKit_Robots」->「Maps_Robots」**フォルダに移動し、**「L1」**を選択します。
- **コンテンツ ブラウザ**で**「Map_1-1_BalanceAndDesignA」**レベルを**ダブルクリック**して開きます。
- **ナビゲーション ショートカット キー**を使用して、ビューポート内で動き回ります。

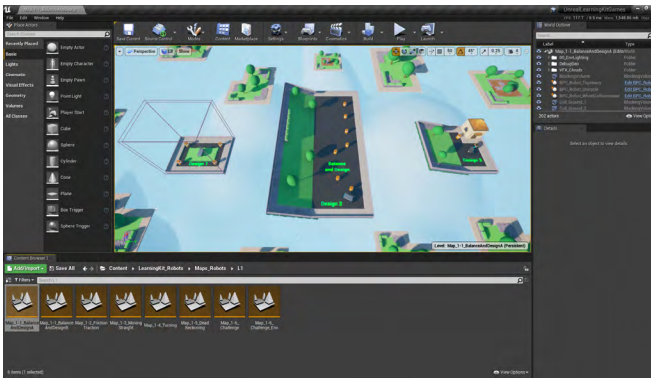


図 1 - 2

設計 1：車輪の位置がロボットのボディに対して近すぎる

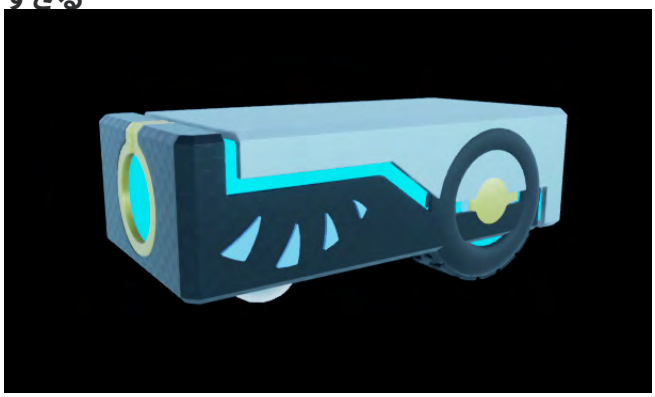


図 1 - 3

車輪がロボットのフレームとくっつきすぎていると、自由に回転できないためロボットは動くことができません。ロボットの設計時に、車輪が車両のボディに接触しないようにします。

動作を確認する

- **ビューポート**内で**クリックします。**
- **1** キーを押して、カメラを最初のデモの位置に動かします。
- ツールバーにある **[Play (プレイ)]** ボタンを押します。

図 1 - 4

- デモを**観察し**、原因と結果について検討します。
- **[Stop (停止)]** ボタン (または **ESC キー**) を押します。

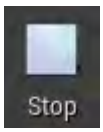


図 1 - 5

設計 2：車輪の間隔が狭すぎる

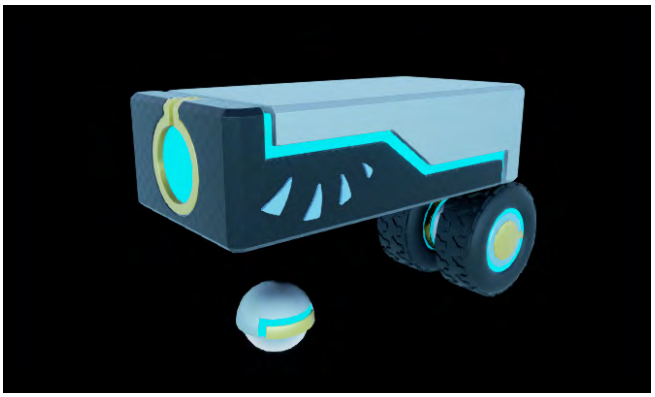


図 1 - 6

車輪の間隔が狭すぎると、ロボットは横向きに転倒しやすくなります。

動作を確認する

- **ビューポート**内で**クリックします。**
- **2** キーを押して、2 番目のデモに移動します。
- **[Play (プレイ)]** を押します。
- デモを**観察し**、原因と結果について検討します。
- **注目点：**車両がどの程度横向きに転倒しやすいか。
- **注目点：**車輪の間隔がどの程度狭すぎるか。
- **解決策：**車輪の間隔を十分に広くして、横に倒れにくいようにします。
- **[Stop] ボタン (または ESC キー)** を押します。

設計 3：車輪の位置が滑走球に対して近すぎる

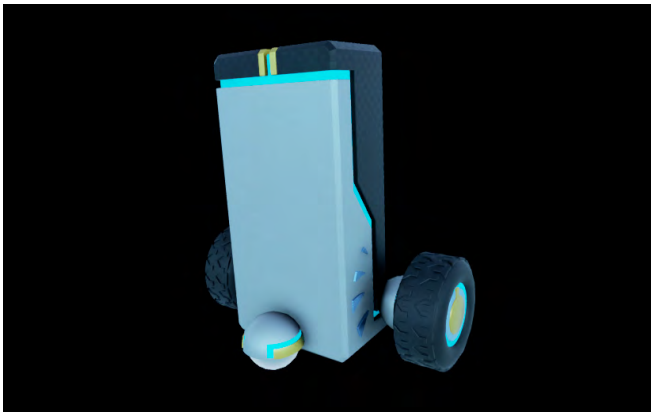


図 1 - 7

ロボットのボディが車輪より高く、車輪の位置がロボットの前面に対して近すぎると、前から後ろにひっくり返ることがあります。

動作を確認する

- **ビューポート**内で**クリックします。**
- **3** キーを押して、3 番目のデモに移動します。
- **[Play]** を押します。
- デモを**観察し**、原因と結果について**検討します。**
- **注目点：**車両の高さと薄さ。
- **注目点：**車輪と滑走球との間隔。
- **注目点：**車両が前方または後方にどの程度ひっくり返りやすいか。

ロボットの設計時に、車輪の間隔 (左側と右側) を広くして、滑走球と車輪の間 (前部と後部) を十分取ることで、車両の安定性が高まり、ひっくり返りにくくなります。

- [Stop] を押します。

この実習で学んだことのまとめ

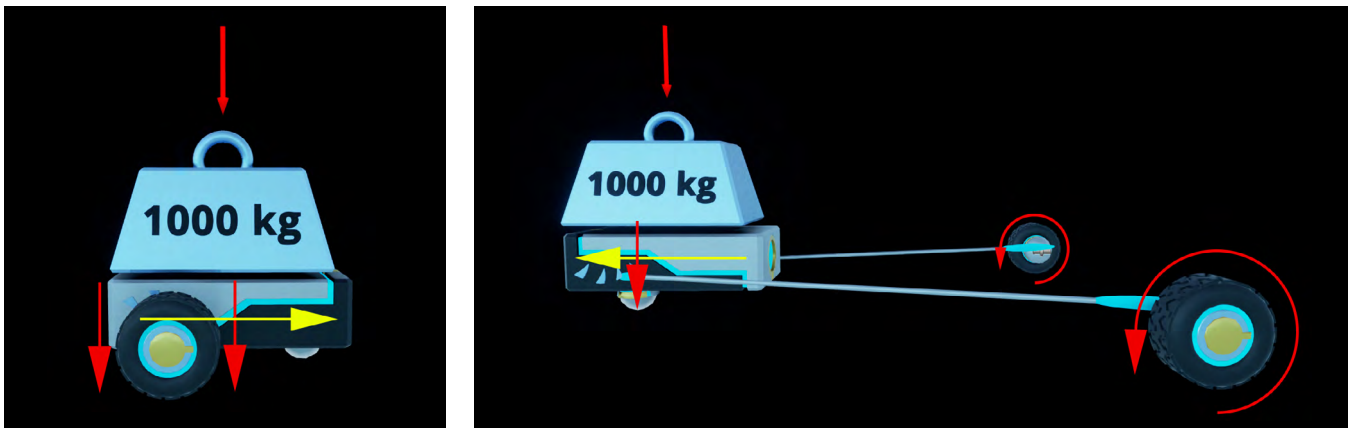
バランスの取れた設計では、車輪はフレームに近いが、フレームに接触していない。

ロボットが横向きに転倒しやすくならないように、2 つの車輪の間隔 (左側と右側) を広くする。

ロボットが前方または後方にひっくり返らないように、滑走球と車輪の間 (前部と後部) を十分に取る。

ロボットのボディは、(高くするのではなく) 横型にし、2 つの車輪と滑走球が三角形を形作るようにします。子供が乗るのに安全な三輪車の 3 点接地をイメージする。

実習 2：移動時の摩擦とトラクション



ボディの重量によって車輪が押し下げられて、地面に接触するようになる

車両の重量によって滑走球が地面に押しつけられるので、車輪が車両を動かしにくくなっている

車輪が地面の上に載っているので、車両を動かすことなく、空転して横滑りしやすい

図 1 - 8

ボットの構築時に、トラクションが重要であれば、車輪がより強く地面と接触するように、動輪をボディ/質量の近くに配置します。

- **コンテンツ ブラウザ**で「**Map_1-2_FrictionTraction**」を開きます。

傾斜路の上り坂を進もうとしている 2 台の車両を観察します。前述の情報に基づくと、どちらの車両が坂道を登るためのトラクションが大きいでしょうか。

例 1

- **ビューポート**内で**クリックします。**
- **1 キーを押して**、最初のデモに移動します。
- **[Play]** を押します。
- デモを**観察し**、原因と結果について検討します。

図 1 - 9

- **注目点:**左側の車両の車輪がどの程度ボディに近く配置されていて、右側の車両の車輪がどの程度ボディから離れて配置されているか。
- **注目点:**トラクションが小さいために車輪がスリップするので、右側の車両がどの程度坂道を登ることができないか。*(これは、鉛筆を垂直に持って線を引くことと、鉛筆を水平に持って線を引くことを比べるのに類似しています。*
  濃い線を上手に引くには、どちらの方法で鉛筆を持つ方が簡単ですか)。
- **注目点:**どの程度まで左側の車両が一番上までずっと坂道を登れるか。
- 車両の重量を利用して、車輪が地面に接触し続けるようにします。車両のボディと重量が車輪の近くにあるようにします。車輪の近くに重量を加えることで、車輪への下方圧力が増えるので、車輪のトラクションが大きくなります。
- **[Stop]** ボタンを押します。

例 2

- **ビューポート**内で**クリックします。**
- **2 キーを押して**、次のデモに移動します。

ここには 2 台の車両があり、正面どうしの押し出し競技が始まるところです。どちらの車両も同じパーツで構成されていて、モーター速度も同じです。主な違いは車輪の位置だけです。

[Play] を押す前に、「*どちらの車両がこの押し出し競技に勝つか*」**を自問してください。**

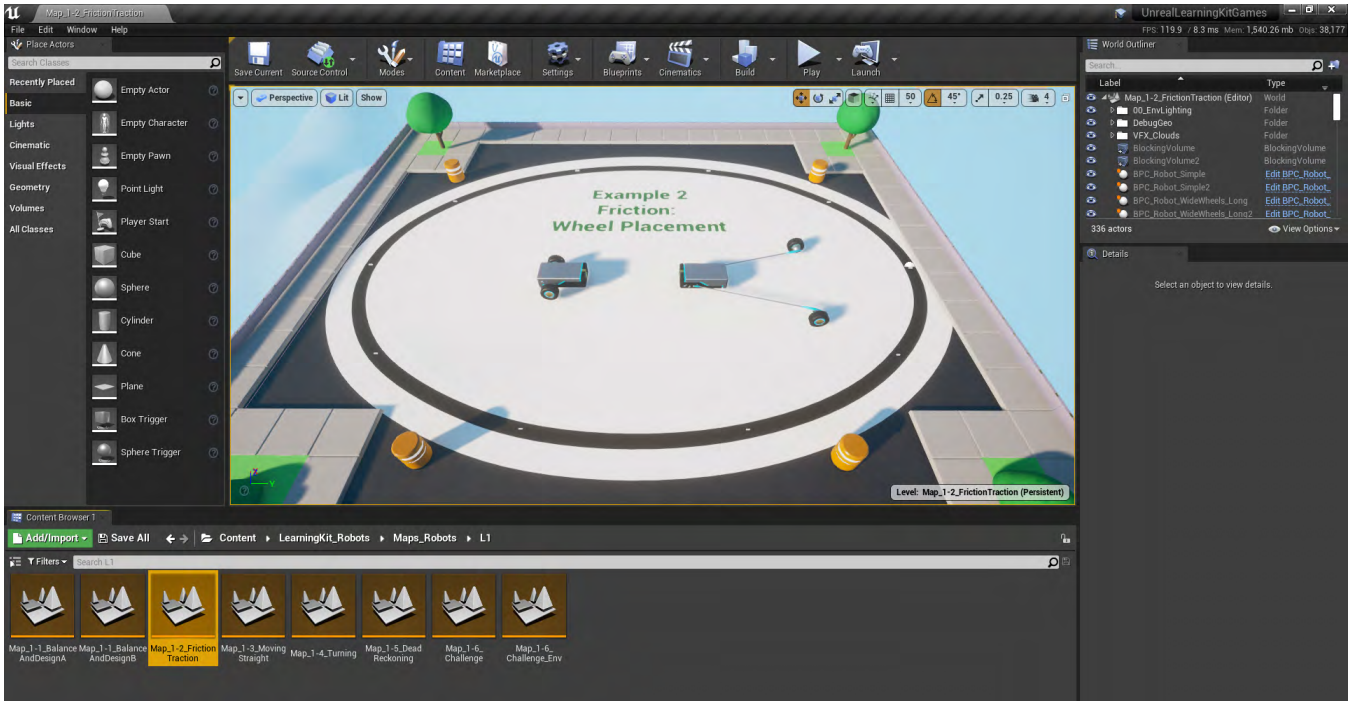


図 1 - 10

- **[Play]** (または、ショートカット キーである **Alt** + **P**) を押します。
- デモを**観察し**、原因と結果について**検討します。**
- **注目点:**左側の車両が毎回どのように勝っているか。
- **注目点:**右側の車両の車輪は回転しているが、トラクションが小さいためにスリップしているので、車両を押し出していない。
- 車両の重量を利用して、車輪が地面に接触し続けるようにします。車両のボディと重量が車輪の近くにあるようにします。車輪の近くに重量を加えることで、車輪への下方圧力が増えるので、車輪のトラクションが大きくなります。
- **[Stop]** ボタン (または **Esc** キー) を押します。

この実習で学んだことのまとめ

- ロボットの構築時に、トラクションが重要であれば、車輪がより強く地面と接触するように、動輪をボディ/質量の近くに配置します。
- 車両の重量を利用して、車輪が地面に接触し続けるようにします。車輪が車両のボディと重量の近くに配置されるようにします。
  車輪をボディと重量の近くに配置することで、車輪への下方圧力が増えるので、車輪のトラクションが大きくなります。
- 最初の設計で車輪への下方圧力が大きくなることで、車輪のトラクションが大きくなっています。

実習 3：ロボットを動かす

ロボットがどのように動くかを調べ、適切に設計されたロボットで使用できるさまざまなタイプの旋回について学びます。これは二輪ロボットであるため、車輪を駆動する各モーターの速度と方向を変えることで、動きを制御します。移動と旋回のデモを見てみましょう

モーターを制御する

モーターは、その速度値を -100 と 100 の範囲の数値に設定することで制御されています。この値を変えることで、モーターの回転の速度と方向の両方を確認できます。この値が 0 (ゼロ) であれば、モーターは回転しません。値が -1 ～ -100 の範囲であればモーターは反時計回りに回転し、-1 が最も遅く、-100 が最も速く回転します。モーターを時計回りに回転させるには正の値を指定する必要があり、1 が最も遅く、100 が最も速く回転します。**注記：**このアクティビティの最後で、ロボットがさまざまなターゲットに向かって走行するように、ロボットのモーターの値を編集します。

図 1 - 11

車両でのモーターの配置

このロボットには 2 つのモーターがあり、それぞれが 1 つの車輪を回転させます。2 つの車輪はそれぞれのモーターに取り付けられています。1 つのモーターは上下逆さまに取り付けられています (物理的な接続の便宜上)。各モーターの速度を調整すると、その車輪の動きを調整することになります。このアクティビティでは、一方のモーターを「モーター A」、もう一方を「モーター B」と呼びます。モーター A は正しい向きで取り付けられ、モーター B は上下逆さまに取り付けられています。速度は -100 ～ +100 の範囲で設定できます。

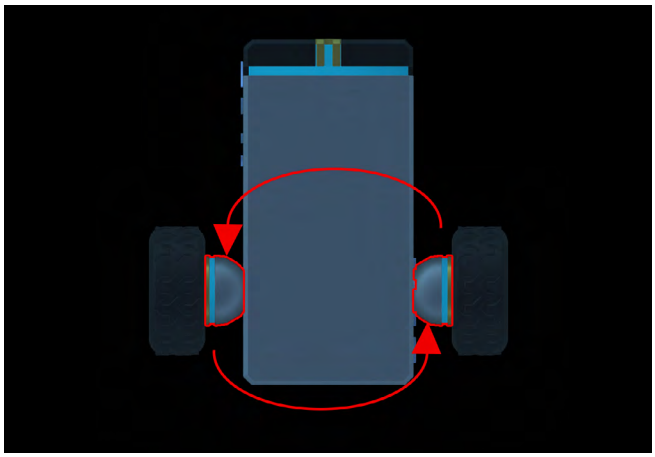


図 1 - 12

モーターは車両のどちら側でも、180° 回転して取り付けられています。そのため、モーターに同じ動きの指示を与えると、モーターは車両に対して逆向きに回転します。

車両を直進させるには、一方のモーターを逆向きに回転させて、両方の車輪を車両に対して同じ方向に回転させなければなりません。

例：直進

このデモは、ロボットを前進および後退させるためのモーターへの指示を理解できるように設計されています。前述のとおり、ロボットを前進または後退させるには、両方のモーターが逆向きに回転する必要があります。2 つのモーターが車両に対して互いに逆向きに取り付けられているからです。

両方の車輪が同じ速度で、ロボットのボディに対して同じ向きに旋回すると、ロボットは真っすぐに走行します。たとえば、モーター **A** が +100 に設定されていて、モーター **B** が -100 に設定されていると、ロボットは 100 の速度で前進します。モーター A が -100、モーター B が +100 に設定されていると、ロボットは 100 の速度で後退します。

モーター A と B の速度設定を観察します。一方のモーターが正の数値であり、もう一方が負の数値であることに注目してください。ロボットを真っすぐに前進または後退させるための設定を思い出すには、このデモを参照してください。

- **「Map_1-3_MovingStraight」**を開きます。
- **ビューポート**内でクリックします。
- **1 キーを押して**、このデモに移動します。
- **[Play] を押します。**
- ロボットが前進および後退する**を観察します。**モーター A とモーター B の速度設定を**メモします。**
- **Esc** キーを押して、デモを停止します。

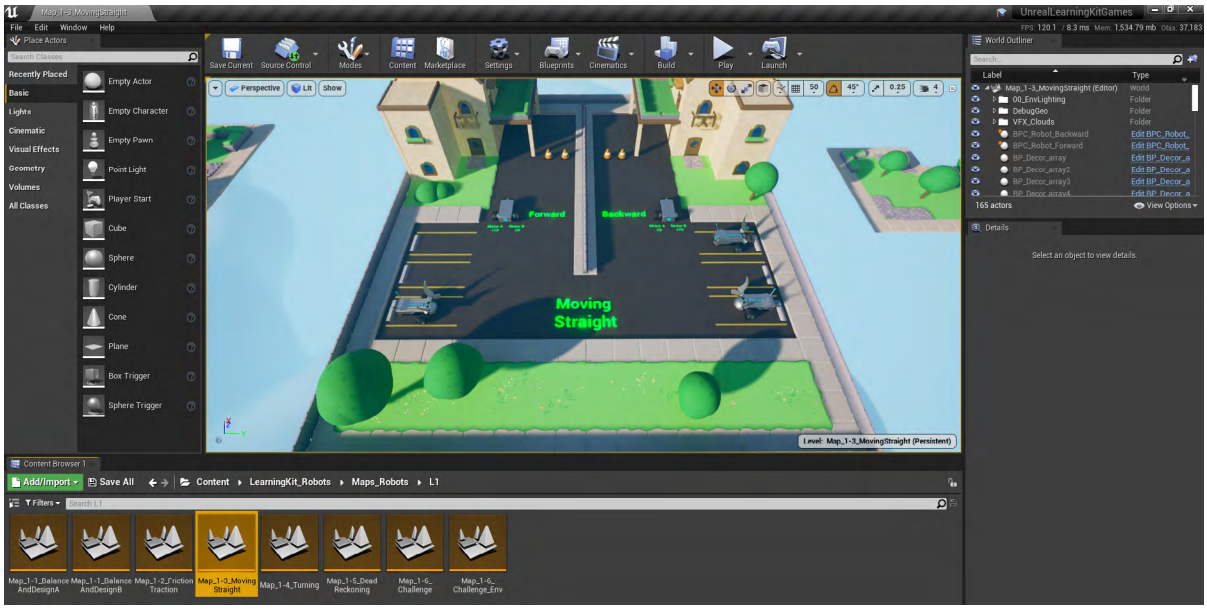


図 1 - 13

実習 4：旋回する

ここまでで、車両を直進させる方法がわかったので、次は二輪車両を旋回させる方法を見ていきましょう。一方の車輪が他方の車輪よりも速い速度で前方に旋回すると、ロボットは曲線状に走行します。速度の差によって、曲線の弧が決まります。

車輪が旋回している方向に**注目してください**。

- ロボット A – ピボット旋回 (一方の車輪は回転していて、もう一方の車輪は停止している)
- ロボット B – スピン旋回 (両方の車輪が同じ速度で互いに逆向きに回転している)
- ロボット C – アーク旋回 (両方の車輪が異なる速度で同じ向きに回転している)
    - 「**Map_1-4_Turning**」を開きます。

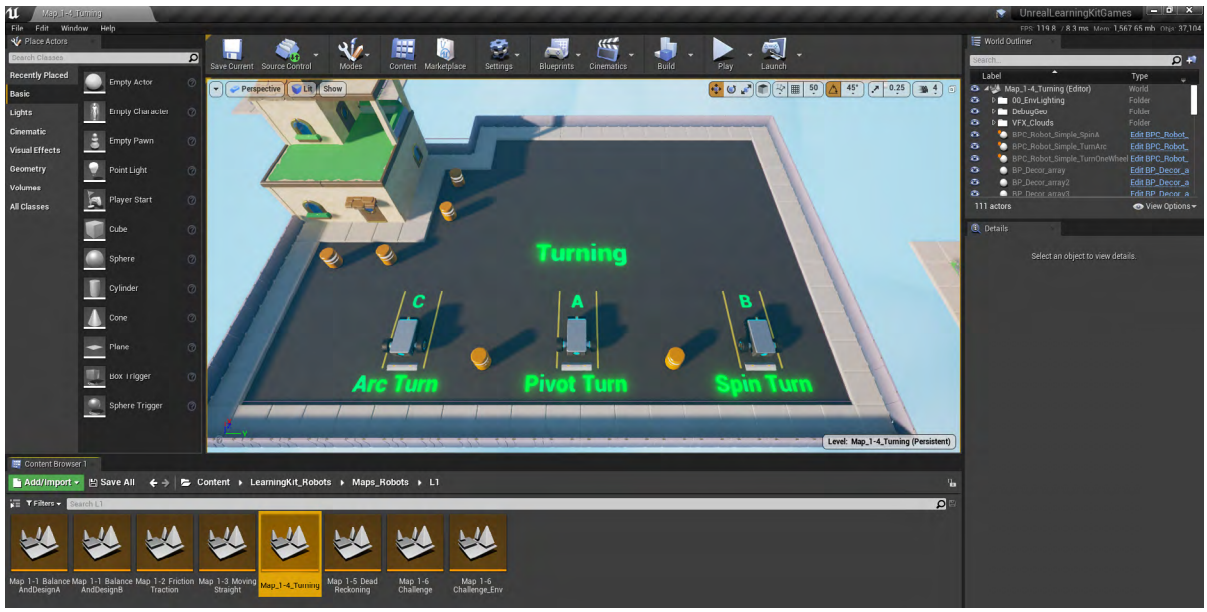


図 1 - 14

ロボット A：ピボット旋回 – 一方の車輪は回転していて、もう一方の車輪は停止している

- **ビューポート**内で**クリックします。**
- **1** キーを押して、最初のデモに移動します。
- [Play] を押す**前に**、「一方の車輪は回転していて、もう一方の車輪は停止している場合に、この旋回はどのように見えるか」を自問します。
- **[Play]** を押します。
- デモを**観察し**、原因と結果について検討します。
    - **注目点：**停止している車輪が回転軸の役割を果たして車両がどのように回転しているか。
- **[Stop]** を押します。

ピボット旋回は、一方の車輪が回転していて、もう一方の車輪は静止したままである場合にのみ起こるので、ロボットは静止している車輪の向きの回転軸で旋回します。

ロボット B：スピン旋回 – 両方の車輪が同じ速度で互いに逆向きに回転している

- **ビューポート**内で**クリックします。**
- **2** キーを押して、2 番目のデモに移動します。
- [Play] を押す**前に**、「両方の車輪が同じ速度で互いに逆向きに回転している場合に、この旋回はどのように見えるか」を自問します。
- **[Play]** を押します。
- デモを観察し、原因と結果について検討します。
    - **注目点：**車両が、車両の中心を回転軸とする狭い円内でどのように回転しているか。
- **[Stop]** を押します。

スピン旋回は、一方の車輪がある向きに回転 (前進) し、もう一方の車輪が同時に逆向きに回転 (後退) する場合に起こり、ロボットは実際にはスピンします。

ロボット C：アーク旋回 – 両方の車輪が異なる速度で同じ向きに回転している

- **ビューポート**内で**クリックします。**
- **3** キーを押して、3 番目のデモに移動します。
- **[Play]** を押す前に、「両方の車輪が異なる速度で同じ向きに回転している場合に、この旋回はどのように見えるか」を自問します。
- **[Play]** を押します。
- デモを**観察し**、原因と結果について**検討します。**
    - **注目点：**両方の車輪が異なる速度で同じ向きに回転している場合に、車両が特定の向きにどのように横滑りするか。
- **[Stop]** を押します。

アーク旋回は、両方の車輪が同時に同じ向きに回転 (前進) しているが、速度が異なる場合に起こり、ロボットは実際には円弧を描いて旋回しながら前進します。旋回の円弧の大きさは、2 つのモーターそれぞれの速度設定の差に応じて変化します。

さまざまな種類の旋回が有用である実世界でのシナリオを考えてみてください。

- ピボット旋回 – 円形の旋回やオブジェクトを軸とする旋回を行う
- スピン旋回 – 半径ゼロでの狭い旋回を行う
- アーク旋回 – 前進しながら緩やかな旋回を行う

この実習で学んだことのまとめ

- 次の 3 つのタイプの旋回について学びました。
  1. ピボット旋回 – 円形の旋回やオブジェクトを軸とする旋回を行う。一方の車輪が回転していて、もう一方の車輪は静止したままであるため、ロボットは静止している車輪の向きの回転軸で旋回する。
  2. スピン旋回 – 半径ゼロでの狭い旋回を行う。一方の車輪がある向きに回転 (前進) し、もう一方の車輪が同時に逆向きに回転 (後退) する。
  3. アーク旋回 – 前進しながら緩やかな旋回を行う。2 つの車輪が異なる速度で同じ向きに回転する。

この実習で学んだことのまとめ

- 次の 3 つのタイプの旋回について学びました。
  1. ピボット旋回 – 円形の旋回やオブジェクトを軸とする旋回を行う。一方の車輪が回転していて、もう一方の車輪は静止したままであるため、ロボットは静止している車輪の向きの回転軸で旋回する。
  2. スピン旋回 – 半径ゼロでの狭い旋回を行う。一方の車輪がある向きに回転 (前進) し、もう一方の車輪が同時に逆向きに回転 (後退) する。
  3. アーク旋回 – 前進しながら緩やかな旋回を行う。2 つの車輪が異なる速度で同じ向きに回転する。

この実習で学んだことのまとめ

- 次の 3 つのタイプの旋回について学びました。
  1. ピボット旋回 – 円形の旋回やオブジェクトを軸とする旋回を行う。一方の車輪が回転していて、もう一方の車輪は静止したままであるため、ロボットは静止している車輪の向きの回転軸で旋回する。
  2. スピン旋回 – 半径ゼロでの狭い旋回を行う。一方の車輪がある向きに回転 (前進) し、もう一方の車輪が同時に逆向きに回転 (後退) する。
  3. アーク旋回 – 前進しながら緩やかな旋回を行う。2 つの車輪が異なる速度で同じ向きに回転する。

実習 5：推測航法による一貫性のない結果

- **コンテンツ ブラウザ**で**「Map_1-5_DeadReckoning」**を**ダブルクリック**します。

このデモは、特定の時間だけアーク旋回を行うことで、ある位置に移動するための簡単な方法を示しています。方向、速度、および持続時間を指定してナビゲートすることは、一般に「推測航法」と呼ばれています。風、気流、地形などの外力によって現在位置を把握するのが難しくなるため、これは信頼性の高いナビゲーション方法ではありません。

このデモの目的は、まったく同じ条件で複数回実行した場合でも、ロボットが達成する結果が異なることを示すことです。このことを知っていると、長距離を走行する場合にロボットが一貫性のない結果を出す可能性があることを理解するのに役立ちます。ロボットが走行する距離が長いほど誤差の範囲が大きくなることが見込まれます (ヒント：ロボットをもっと正確に走行させるための解決策は、リアルタイムのフィードバックをコードに提供できるセンサーを付加することです。2 番目のレッスンでそれを行います)。

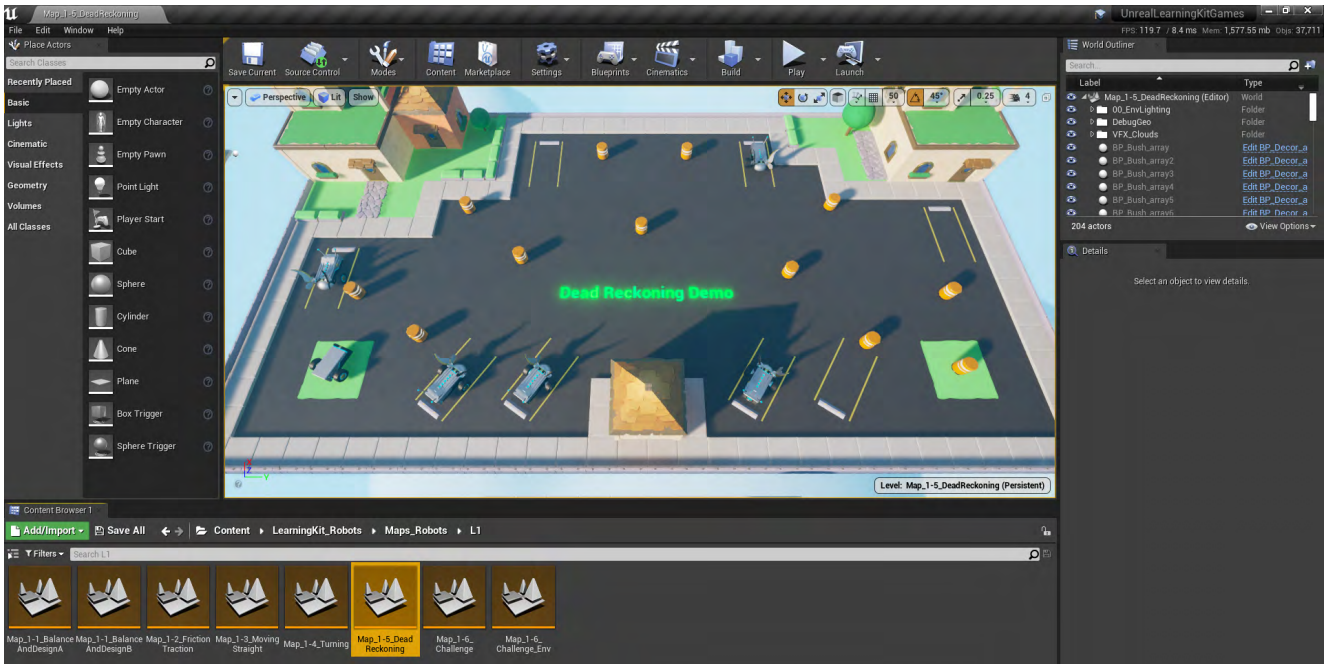


図 1 - 15

- **[Play]** を押します。
- 緑色の区画にある**ロボットを観察します**。
- このサンプル レベルを数回**実行**して、まったく同じ設定で異なる結果がどのように生み出されることがあるかを確認します。

この実習で学んだことのまとめ

- 方向と速度を指定するがセンサーからの入力を使用せずにロボットを一定時間走行させることは「推測航法」と呼ばれている。
- 風、気流、地形などの外力によって現在位置を把握するのが難しくなるため、推測航法は信頼性の高いナビゲーション方法ではない。
- センサーからの入力を使用せずにロボットをナビゲートしようとするのは困難であることがある。数回の試行でのわずかな違いによって結果が毎回異なるからである。

復習

- うまくいったこと
- どのような課題があったか。
- まだ練習する必要があること

課題

課題に進む準備ができたら、先に進みましょう。

- **コンテンツ ブラウザ**で**「Map_1-6_Challenge」をダブルクリック**して、課題マップを開きます。

目的

ロボットをさまざまなターゲットまでナビゲートする、モーターの速度と持続時間を見つけ出します。この課題のレベルでは複数のターゲットが提示されます。1 回のアーク旋回でターゲットに到達するようにコーディングしなければなりません。各テストを実行する前に、各モーターの速度を設定し、方向を決定します。モーターの持続時間を設定することで、ロボットが走行する距離を制御できます。

次のエンジニアリング プロセスを使用します。**予測**、**テスト**、**観察**、**反復**。

試行をログ記録し、そのデータを利用してその後の各試行に対する判断材料にします。

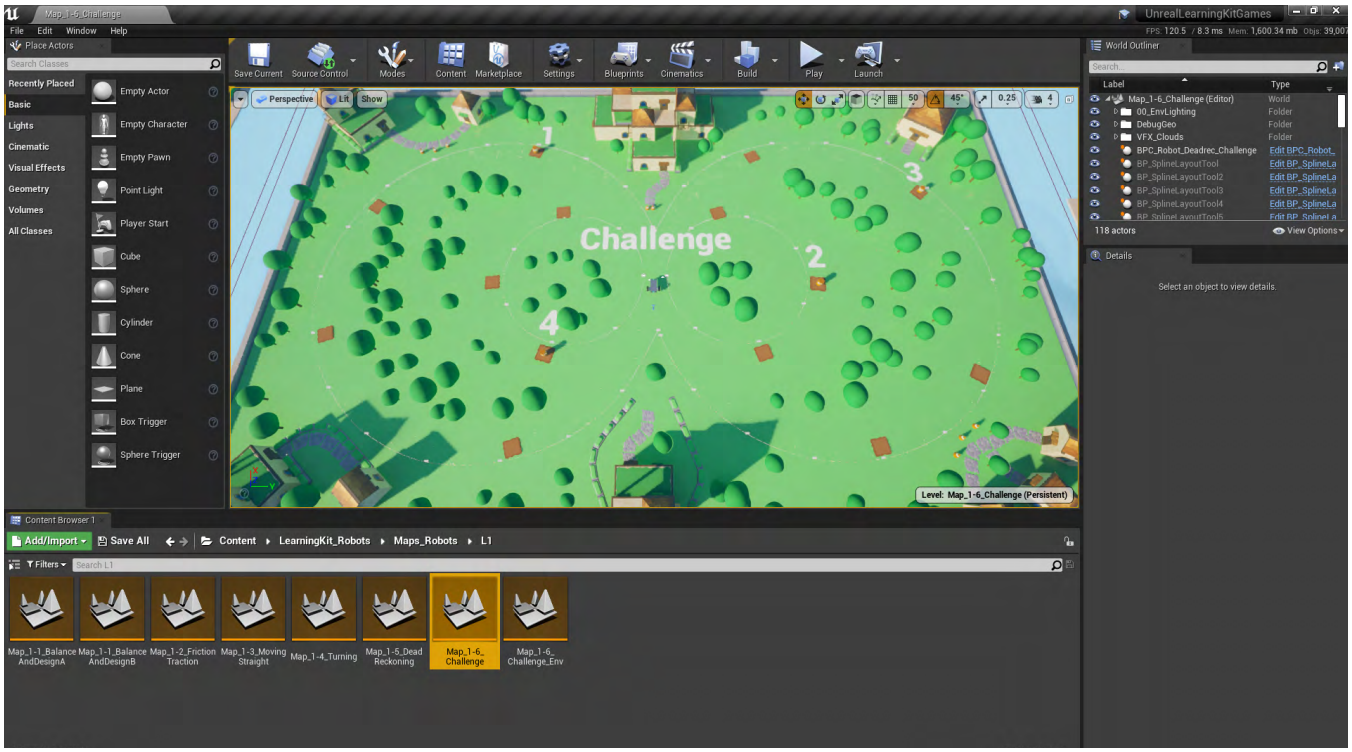


図 1 - 16

重要な注記:

- *現行のロボット設計では、一方のモーターが上下逆さまに取り付けられています (接続の便宜上)。そのため、ロボットのそちら側の車輪の移動方向は、モーターの時計回りの向きと逆になっています。そのため、両方の車輪の移動方向が同じになるように、設定では正の値と負の値が組み合わされます。*
- モーターの速度の値は 0 ～ 100 の範囲です。
- 0 ～ 100 または 0 ～ -100 の範囲の数値で速度を指定します。0 であれば回転せず、100 (または -100) であれば最大速度で回転します。下記のルールに従います。ただし、速度の数値は自分が選んだ値に置き換えてください。
- 一方の車輪が他方の車輪よりも速い速度で前方に旋回すると、ロボットは曲線状に走行します。速度の差によって、曲線の**弧**が決まります。
- 注記:速度が速いほど、走行結果がより不安定になることがあります。テストする際は、低い速度で始めて、望ましい速度を決めます。
- モーターの速度を変更すると、走行距離にも影響が及ぶことに注意してください。モーターの速度を速くすると、同じ距離に到達するまでの持続時間を短くする必要があります。

次の表を利用して、モーターの速度と移動方向を検討します。

移動方向	モーター A の設定	モーター B の設定	アクションの説明
前進	+50	-50	真っすぐに前進するには、モーター A は正の値で、モーター B は負の値で、同じ速度である必要があります。
後退	-50	+50	真っすぐに後退するには、モーター A は負の値で、モーター B は正の値で、同じ速度である必要があります。
左向きのピボット旋回	+0	-25	モーター A は停止していて、モーター B は前方に旋回する (負の値) 必要があります。
右向きのピボット旋回	+25	-0	モーター B は停止していて、モーター A は前方に旋回する (正の値) 必要があります。
左向きのスピン旋回	-25	-25	モーター A は後方に旋回し、モーター B は前方に旋回する必要があります。
右向きのスピン旋回	+25	+25	モーター A は前方に旋回し、モーター B は後方に旋回する必要があります。
左向きのアーク旋回	+0 ～ +20	+0 ～ +20	前進するときにわずかに左向きに旋回するには、モーター B がモーター A よりも高速に旋回する必要があります。
右向きのアーク旋回	+0 ～ +20	+0 ～ +20	前進するときにわずかに右向きに旋回するには、モーター A がモーター B よりも高速に旋回する必要があります。

- **ビューポート**にある**ロボットをクリックします。**

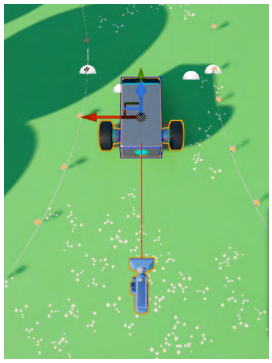


図 1 - 17

- **[Details (詳細)] パネル**内でクリックします (編集している項目が **BPC_Robot_Deadrec_Challenge** であることを確認するか、アウトライナを使用して「BPC_Robot_DeadRec_Challenge」を検索して選択する)。

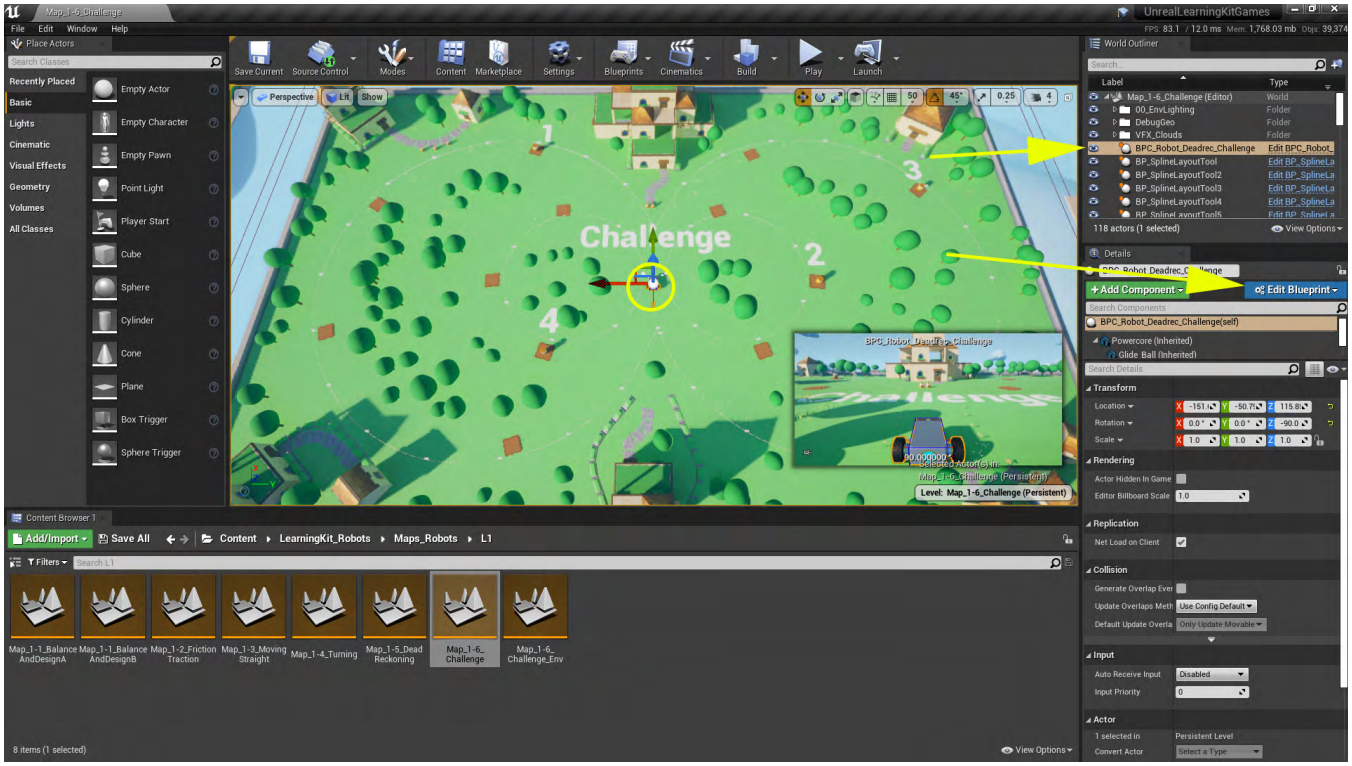


図 1 - 18

- **[Details]** パネルで **[Edit Blueprint (ブループリントの編集)] をクリックします。**
- ドロップダウン メニューで **[Open Blueprint Editor (ブループリントエディタを開く)] を選択します。**

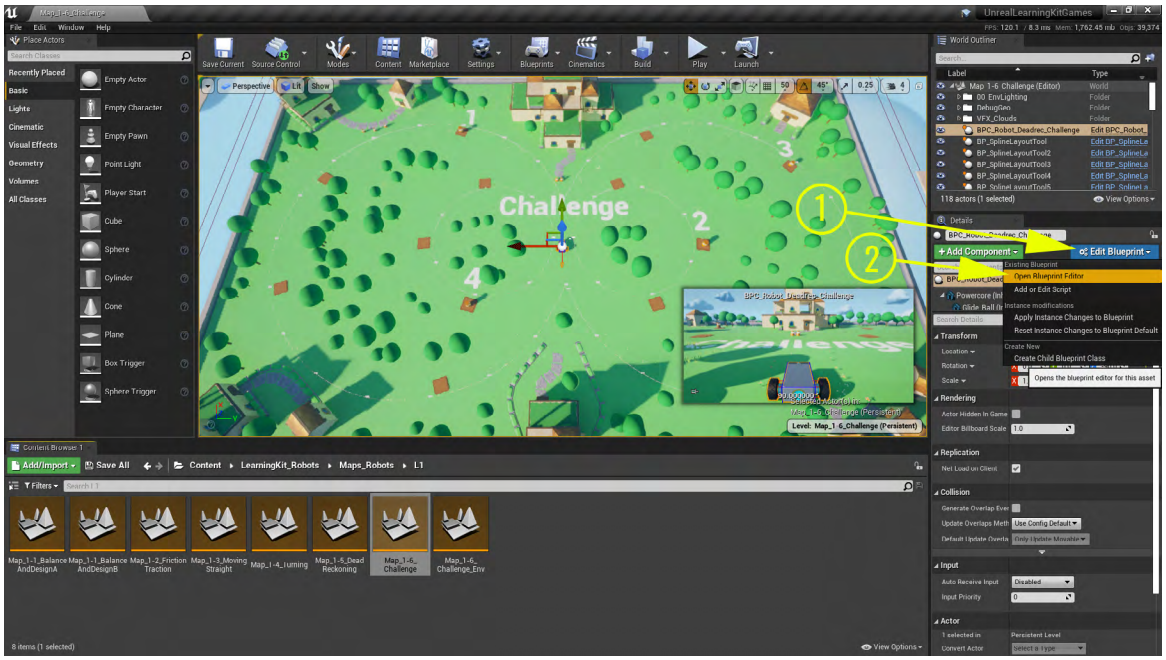


図 1 - 19

ヒント

ビューポート内でクリックしてから、**0** キーを押すと、**すべてのターゲット**が表示され、**1** キーを押すと**ターゲット 1** が拡大され、**2** キーを押すと**ターゲット 2** が拡大され、**3** キーを押すと**ターゲット 3** が拡大され、**4** キーを押すと**ターゲット 4** が拡大されます。

- ブループリント グラフ内でモーター A とモーター B をクリックしてから **新しい値を入力すると、Motor Speed** と **Time** の値が、ターゲット 1 に到達するための自身の経験に基づく推測に変更されます。

図 1 - 20

ブループリントのコードを変更したら、**[Compile (コンパイル)]** をクリックしてから **[Save (保存)]** をクリックします。[Blueprint] タブを閉じると、ビューポートに戻ります。

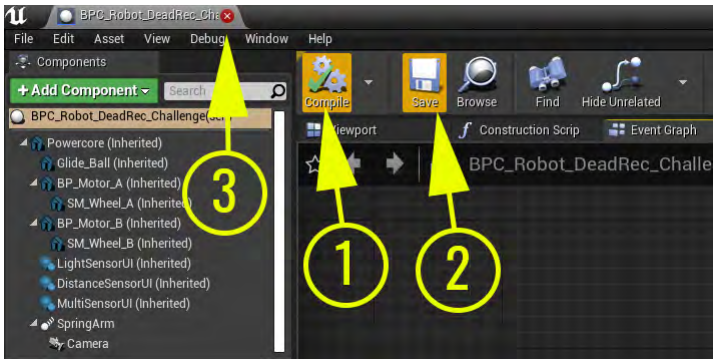


図 1 - 21

- **ビューポート**内でクリックし、数字キー **1、2、3、4 のいずれか**を押すと、それぞれの課題でのカメラにフォーカスが移ります。
- **[Play]** を押します。
- **試行**を**ログ記録**し、**結果**を**メモします**。
- トラブルシューティングを行い、設定を修正し、ターゲットに到達するまで必要なだけ繰り返します。
- ターゲット 1 ～ 4 のそれぞれに対して上記の手順を複数回繰り返し、各試行をログ記録します。

ヒント

モーターの速度の数値には小数を使用できます。これはコーディングでは一般的に Float (浮動小数点数) と呼ばれるものです。速度 31 では遅すぎて 32 では速すぎる場合は、31.5 を試してみます。

リソース

基本的なチュートリアル

Unreal Engine のプログラムを開いているときに、**[Help (ヘルプ)]** メニューをクリックし、**[Tutorials (チュートリアル)]** > **[Basics (基本)]** を選択します。

- ナビゲーション早見表 (ナビゲーション – この環境内で快適に作業できるようにカメラを操作できることの重要性)

拡張アクティビティ

- どれだけ速くターゲットに到達できるか
- 速度が向上するように試してみて、結果の正確さと一貫性はどうなるか
- 複数のコマンドをつなぎ合わせることで、複雑なパスを作成する
- ロボットを複数の目的地に移動させる
- 勝利のダンスを踊るようにロボットをコーディングする
- ロボットへの指示の正確さと信頼性を向上させることができる解決策を検討する
- ロボットの設計についてあれこれ試してみる (ヒント：ロボットのブループリントを編集するときに、[Viewport] タブを使用する)

評価

基準

概念	抜群	熟練	適格	未熟
ロボット設計の概念	ロボット設計のコンポーネントとそれがどのように機能するかを説明できる。プロジェクトでロボットのすべてのモーターと回転軸を使用のデモを行うことができる。ロボット設計のコンポーネントとそれがどのように機能するかを説明できる。	ロボットのモーターの使用のデモ行うことができる。ロボットのコンポーネントについて完全に理解していることが、言葉またはプロジェクトのプレゼンテーションで示されている。	ロボットを動かすことができ、モーターについて基本的なことを理解している。ハードウェア コンポーネントについて理解している基本的な根拠が、生徒のプレゼンテーションで示されている。	ロボットのパーツとその機能を理解している根拠がない。生徒のプレゼンテーションでハードウェア コンポーネントについて触れていない。
ソフトウェアの概念	コマンドの複雑な組み合わせとプロジェクトの目標が示されている。モーター、ロボットのバランス、回転軸を使用するために必要なコマンド グループについて説明できる。複数のコマンドまたはコマンド グループが生徒のプレゼンテーションまたはドキュメントに記載されている。	生徒はモーター、ロボットのバランス、回転軸を使用するために必要なコマンド グループを理解している。複数のコマンドまたはコマンド グループが生徒のプレゼンテーションに記載されている。	生徒はモーター、ロボットのバランス、回転軸を使用するために必要なコマンド グループについて基本的な知識を持っているが、生徒のプレゼンテーションでは触れていない。	コマンドおよびその機能について理解している根拠がない。生徒のプレゼンテーションでコマンドについて触れていない。
コーディングの概念	コードのエラーをデバッグできる。コードの複雑な組み合わせと独自の使用方法が示されている。ほとんどのセクションのコードを説明できる。生徒のプレゼンテーションまたはドキュメントにコードが記載されている。	デフォルト設定、ループ、条件付きコマンドについての知識の実例を示している。生徒は自身のプロジェクトで各コマンド タイプのコードをそれぞれ 1 つ以上参照している。	生徒はコードについて基本的な知識を持っているが、生徒のプロジェクトのプレゼンテーションでは触れていない。	コマンドのコードおよびその機能について理解している根拠がない。生徒のプレゼンテーションでコードについて触れていない。
実世界の概念	実世界でのロボット、コーディング、移動の利用を生み出す革新的なアイデアを持っている。理解していることを示しており、それをドキュメントに記載している。	実世界のアプリケーションでのコーディング、移動、ロボットの利用について理解している。生徒のプロジェクトのプレゼンテーションで 1 つ以上の例が記載されている。	実世界のアプリケーションでのコーディング、移動、ロボットの利用について基本的な知識を持っている。促されれば言葉で説明できるかもしれないが、プレゼンテーションでは触れていない。	コーディング、移動、ロボットの実世界のアプリケーションでの利用について理解している根拠がない。
課題アクティビティ	各試行をログ記録し、それを判断材料にして各反復で改良することで、課題で効率的にターゲットに到達できている。同じターゲットに到達するための異なる複数の構成を識別できる。	複数のターゲットに到達できている。各試行のログを利用して、各反復で改良を行っている。	1 つまたは複数のターゲットに到達するできている。各試行のログ記録を行っている証拠が限定的であるか、またはターゲットを素早く見つけることにログ記録を効果的に利用していない。	どのターゲットにも到達できていない。各試行のログ記録を行っている証拠がなく、モーター設定でどのようにロボットを動かすかについての知識が欠如している。

生徒向けガイド

仮想ロボットのトレーニングを学ぼう

レッスン 1：ロボット車両

UNREAL ENGINE