仮想ロボットのトレーニングを学ぼう

レッスン 5：遠隔操作できるサッカー ロボット

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 5：遠隔操作できるサッカー ロボット

目的

このレッスンは、モーターがキーボードのキー押下にマッピングされていて、キーボード入力から仮想遠隔操

作を行う仮想オンライン ロボットを、生徒がコーディングできるように設計されています。そうすることで、ロボットの動きをキーボードで操作および制御できるようになります。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。

レッスンの目的

すべての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。

レッスン 5：遠隔操作できるサッカー ロボット

はじめに

遠隔操作を使用する理由

ここまでのアクティビティのコーディングでは、ユーザーの介入なしで自律的に走行するロボットに重点を置いていました。ユーザーの介入なしでロボットをコードによって動かし続けるのを見るのは面白いものです。思い切りがよく好奇心の強い新しいコード作成者が与えた行動をロボットが忠実に行うことで、多くの笑いが起きることは
間違いありません。

新たな楽しみ方を作り出しましょう！ロボットの動きを前もってコーディングするのではなく、ユーザーからの入力に直接反応するようにロボットをプログラムします。遠隔操作レシーバーをロボットに追加することで、キーボードまたはコントローラーからの入力に反応するようにコーディングできます。

どのようなタイプのロボットが遠隔操作されるのでしょうか。爆弾処理ロボット、ドローン、救助ロボット、さまざまなタイプの娯楽用ロボットなど多数あります。

ここでは、遠隔操作できるサッカー ロボットを作成しましょう！**ゴーーーーーール！**

レッスンの概要

キー押下をマッピングする

このレッスンでは、モーターをキーボードのキー押下にマッピングして、キーボード入力による仮想遠隔操作を行う方法を学びます。そうすることで、ロボットの動きをキーボードで操作および制御できるようになります。

遠隔操作は、ビデオ ゲーム、テレビ、プロジェクター、ドローン、モデル カーやボート、他の多くのデバイスで日常的に使用されています。遠隔操作を使用することで、オブジェクトに迅速かつ簡単に指示することができます。設定の変更や望ましい仕様への移行をオブジェクトに迅速かつ簡単に指示することができます。デベロッパーは、遠隔操作で指示できるアクションをコーディングする必要があります。

ここでの目標は、キーボード入力に基づいて前進、後退、任意の向きの旋回を行うようにロボットを操作できることです。

ロボットの各モーターがキーボードの特定のキーに「マッピングされている」必要があります。コードのフローとしては、イベントを使用してキーボードのキー押下を検出し、指定された速度と向きでモーターを動かすコードを実行します。

コーディングのつながり

メイン ループではプログラムを実行し、キーボード入力をリッスンし続けます。イベントを使用して、ユーザー インタラクションを検出し、ゲーム内応答を生成します。演算関数を使用して、キー押下に基づいた動きの速度と方向を決定します。

ゲームを始めましょう!

レッスンを開始する

概念を学ぶ

このレッスンは、この「仮想ロボットのトレーニングを学ぼう」シリーズの中で、プレイヤーが操作する唯一のアクティビティであるため、独特です。他のすべての実習は、プログラマーが決めたコマンドを実行する自動運転車です。遠隔操作を追加することで、新たなタイプのプログラミング課題および楽しませるコンテンツを作り出す新たな方法が導入されています。

遠隔操作の概念を理解する

遠隔操作は有線でも無線でも行うことができます。無線の遠隔操作は、電波、赤外線、または他の方法で行うことができます。2 つの主要コンポーネントは、ユーザー側の「トランスミッター」とロボット側の「レシーバー」です。レシーバーがトランスミッターからの信号を受け取ったときに、さまざまな入力値ごとに特定のアクションを行うようにプログラミングできます。

実習

実習 1:計画を立てる

- **コンテンツ ブラウザ**で、**「Content」->「LearningKit_Robots」->「Maps_Robots」**フォルダに移動します。
- **「L5」**フォルダを**ダブルクリック**してを開きます。
- そして、**「Map_5-1_RemoteControl」**を**ダブルクリック**して開きます。

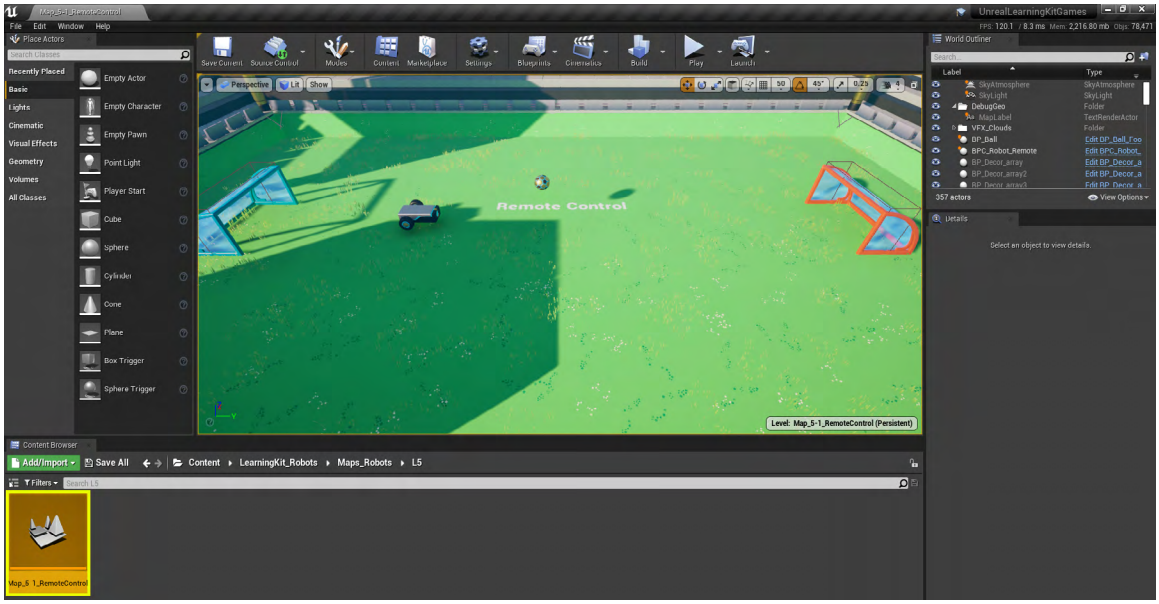


図 5-1

プレイ領域とロボットの開始位置をざっと確認します。

- 車両を使用してボールをゴールに導くために**必要とされる動き**を (口頭または書面で) **伝えて計画**します。
- よく使用される遠隔操作のキーボード レイアウトについて、**他の生徒と共同作業**します。
- キー名と動きのラベルを記載した**大まかなスケッチを作成**します。

遠隔操作車両の必要性を理解する

車両のどのような動きをプレイヤーが操作できるようにすべきか。

- 前進
- 後退
- 左旋回
- 右旋回
- 停止

どのようなキーボード入力を使用して車両を操作すべきか。

- それぞれの動きに対してキーボードのどのキーが最適な選択肢か。
- 片手で簡単にロボットを操作するためにどのようなキー一式を使用できるか。
- ゲーム内カメラの移動には W、A、S、D キーを使用できるので、車両の操作にどのような代替キー一式を使用できるか。

どのようなアクションを異なるキー押下にマッピングすべきか。

前進する

- ロボットはどの向きに動くべきか
- ロボットが動く時間はどれくらいにすべきか
- ロボットが動く速度はどれくらいにすべきか

後退する

- ロボットはどの向きに動くべきか
- ロボットが動く時間はどれくらいにすべきか
- ロボットが動く速度はどれくらいにすべきか

停止する

- どのようなタイプのブレーキをモーターに適用すべきか
- ロボットが停止している時間はどれくらいにすべきか
- ロボットがブレーキをかける速度はどれくらいにすべきか

左旋回する

- ロボットはどの向きに動くべきか
- どのモーターが動いているべきか
- モーターの回転速度はどれくらいにすべきか
- ロボットが動く時間はどれくらいにすべきか
- ロボットが動く速度はどれくらいにすべきか

左旋回する

- ロボットはどの向きに動くべきか
- どのモーターが動いているべきか
- モーターの回転速度はどれくらいにすべきか
- ロボットが動く時間はどれくらいにすべきか
- ロボットが動く速度はどれくらいにすべきか

実習 2：遠隔操作レシーバーをロボットに追加する

遠隔操作レシーバーをロボットに追加すると、車両のコーディングでユーザー入力に反応できるようになります。

アクティビティ

BP_Antenna (遠隔操作センサー) をロボットに追加し、キー押下を特定のアクションにマッピングできるブループリント コーディング領域にアクセスします。すぐに使い始められるように、車両の基本的な動きの関数として次の関数を **BP_Antenna** ブループリントに用意しています。

1. Move Forward
2. Move Backward
3. Turn Right
4. Turn Left
5. Stop Motors

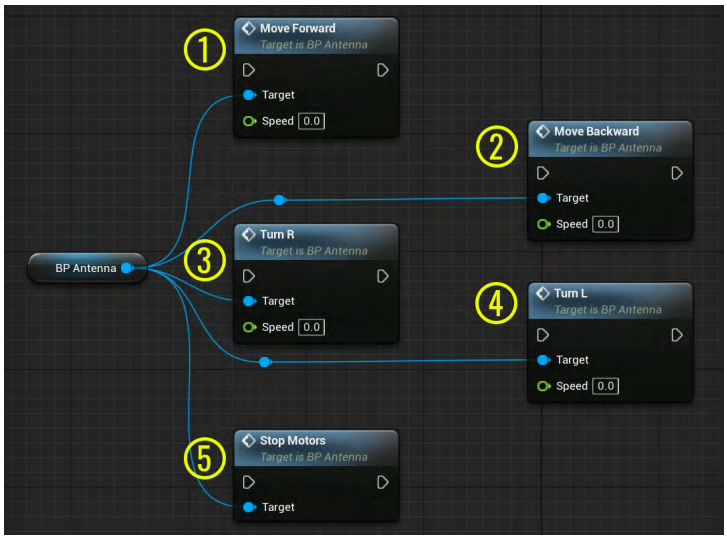


図 5-2

BP_Antenna を追加する方法を以下に示します。

- フィールドにある **BPC_Robot_Remote** を選択 (**クリック**) します。
- そのブループリントを開きます ([**Details**] パネルで、[**Edit**] > [**Open Blueprint**] を**クリック**する)。

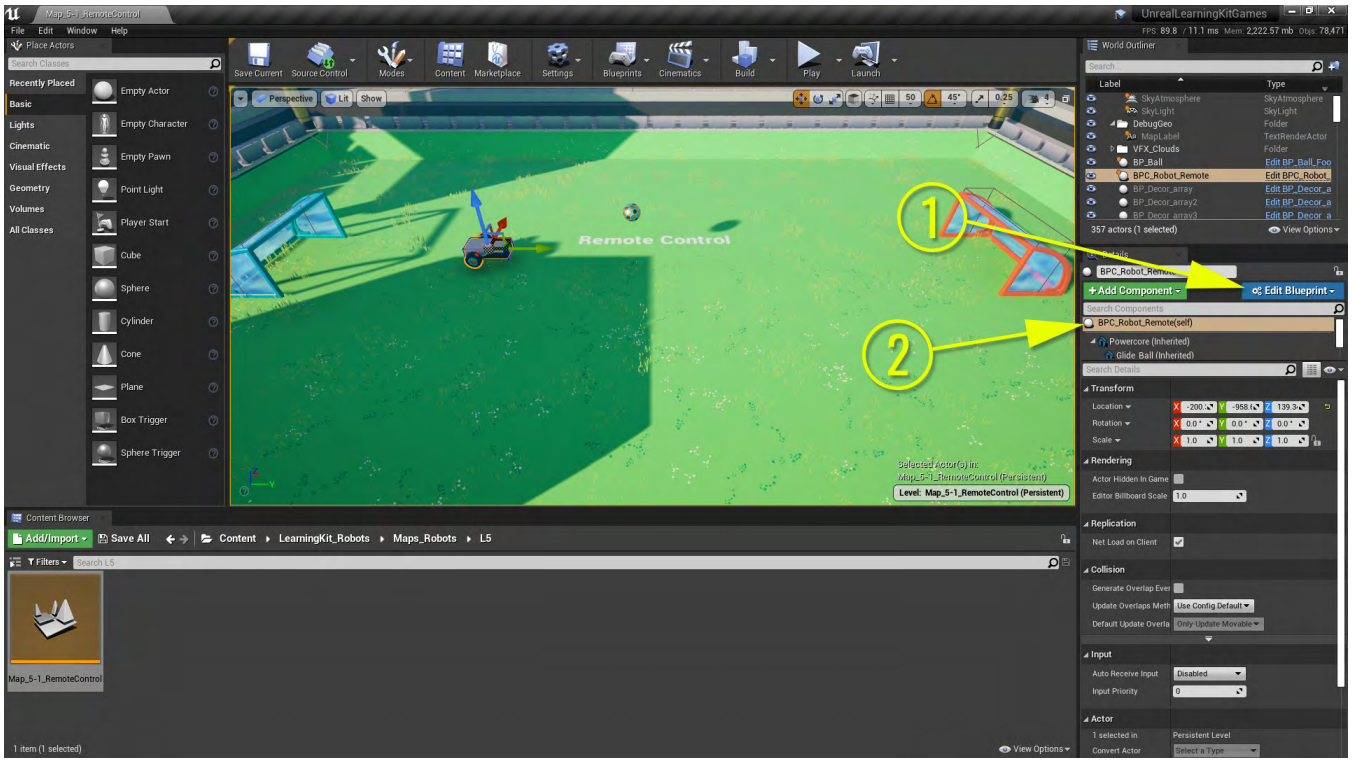


図 5-3

- [Components] パネルで **[Add component]** をクリックします。
- 「bp antenna」と**入力**して **BP_Antenna** を検索し、BP Antenna エントリをクリックすると、それがコンポーネント一覧の一番下に追加されます。

図 5-5

- **ビューポートをクリックすると、**ロボットと、さきほどロボットに追加したアンテナが表示されます。
- **Move Gizmo** ツールを使用して、望ましい位置に**アンテナを配置します。**推奨される位置については。図 5-5 を参照してください。

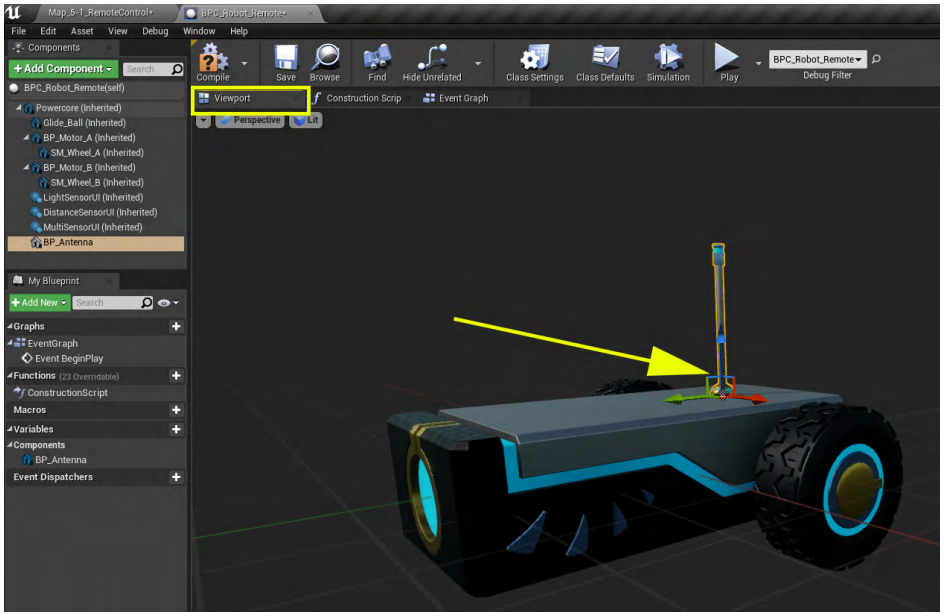


図 5-5

- **コンパイル**して**保存**します。

実習 3 – 遠隔操作キーをマッピングする

キーボード入力を受け取る

次に、キーボードからの入力を受け取るようにロボットをセットアップします。

- **イベント グラフ**をクリックしてコードに戻ります。
- 条件ボックスにある Event BeginPlay ノードで、出力ピンを**クリック**して、白色のワイヤーを右側に引き出します。
- 「**enable input**」と**入力**して Enable Input 関数を検索し、**Enable Input** を**クリック**すると、新しいノードが作成されます。

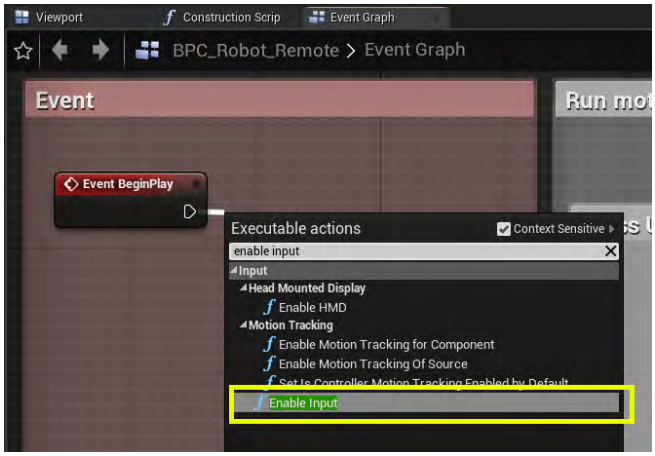


図 5-6

プレイヤー コントローラーへの参照をつなぐことで、キーボードから入力が入ってくるように設定します。

- **Player Controller** ピンを**クリック**し、青色のワイヤーを**左側に引き出します。**
- **「get player controller」と入力**して **Get Player Controller** 関数を**検索**して選択すると、そのノードが条件ボックスに追加されます。

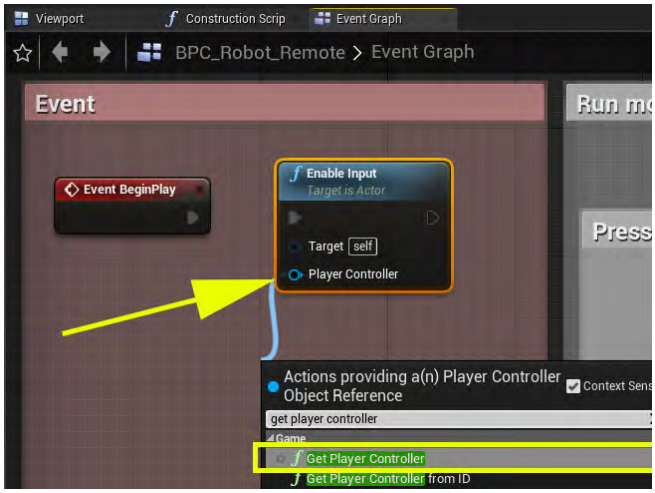


図 5-7

- コードは図 5-8 のようになっているはずです。

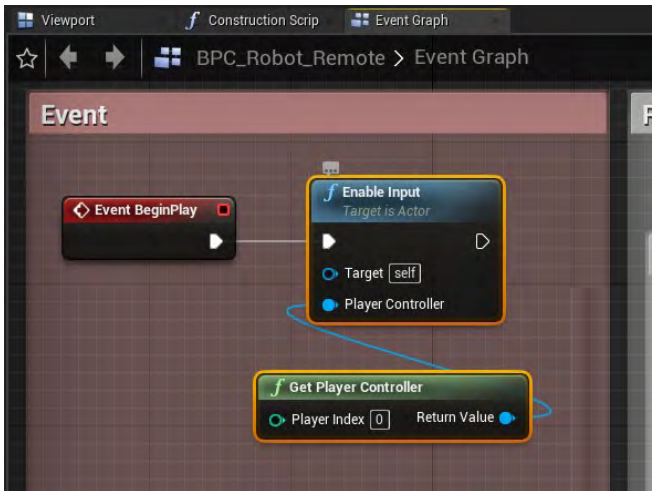
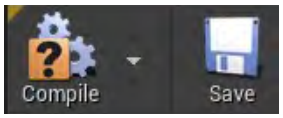


図 5-8

- **コンパイル**して**保存**します。



キーボードによる前進の操作

ロボットを操作するには、前進、後退、左旋回、右旋回のそれぞれの移動指示に対してどのようなモーター制御値が最適であるかを最初に決めておくことが重要です。

アクティビティ

次の例では、プレイヤーが上向きの矢印キーを押したときにロボットが前進し、そのキーを放すと前進をやめます。

なお、ロボットを動かす関数は **BP_Antenna** に組み込まれています。移動するようにロボットに指示するときに設定する必要があるのは Speed 変数だけです。

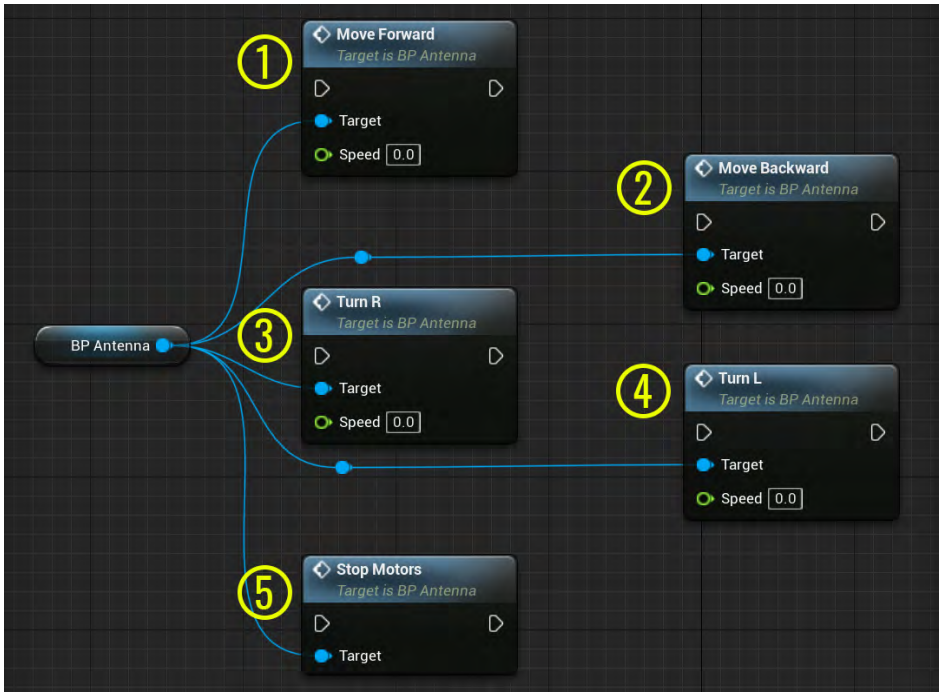


図 5-9

Speed フィールドでどのような値を使用すべきか

以前のアクティビティを思い出すと、正の値と負の値を使用して移動の向きを制御していました。カスタム関数は移動の向きに基づいて作成されているので、これらのノードの Speed の値は、速度を上げるには正の数値、速度を下げるには負の数値である必要があります。

ロボットを動かすために使用するキーを決めます。このアクティビティでは、矢印キーを使ってロボットを操作します。

ブループリントの Run motors セクションでコーディングを始めましょう。

「**Press Up to move Forward**」というラベルが付いているコメント ボックスで、**上向き矢印** キーが押されたときにロボットを前進させるようにコーディングします。

イベントを追加する

- コメント ボックス内で**右クリック**すると、[All Actions for the Blueprint] メニューが表示されます。
- **「up」と入力**して検索し、キーボード イベントにある **Up** を選択します。そうすると、キー入力イベントが追加されます。

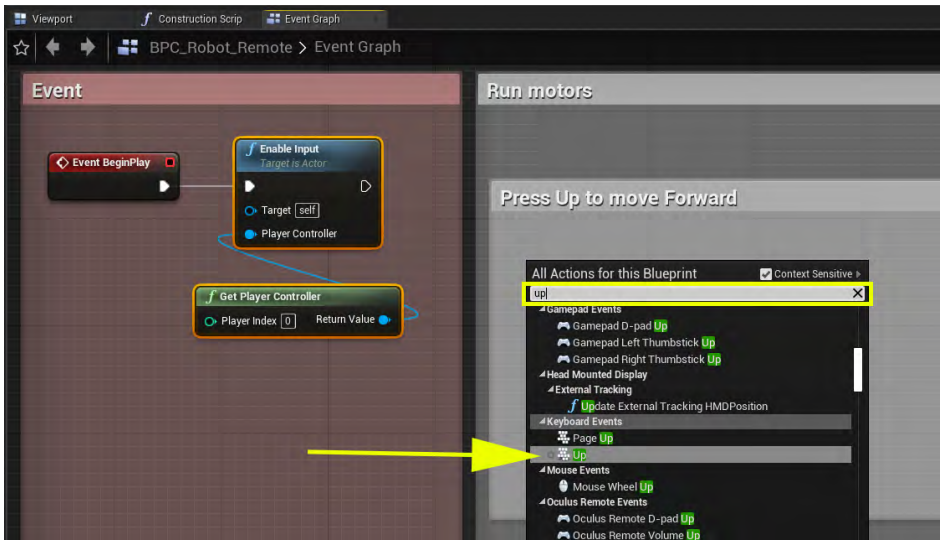


図 5-10

- [Components] パネルで、コンポーネント一覧にある **BP_Antenna** を**クリック**して「Press Up to move Forward」コメント ボックスにドラッグします
- (図 5-11 を参照)。

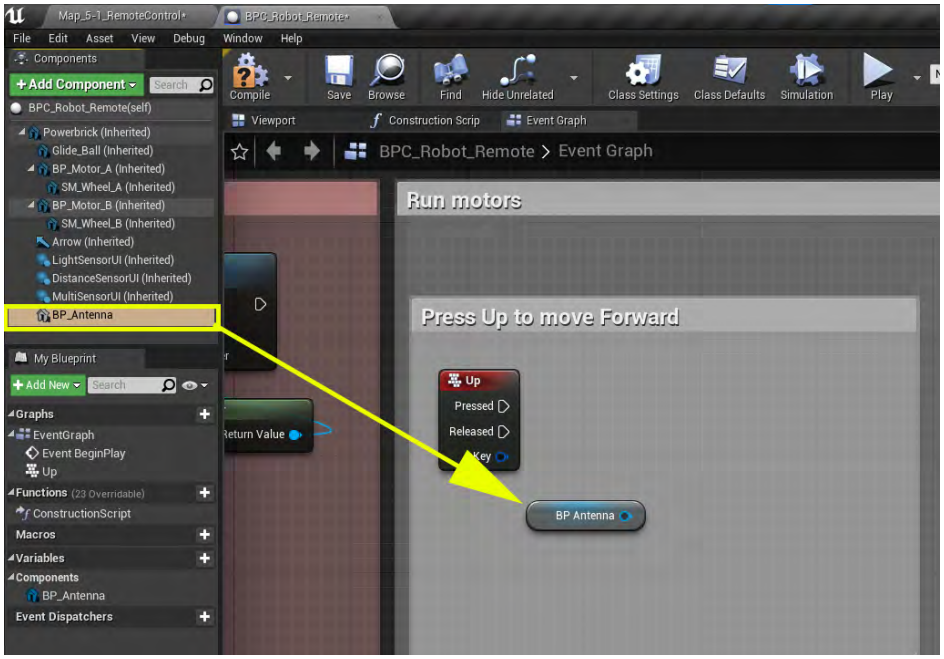


図 5-11

- **BP_Antenna のピンをクリックして引き出し、「move forward」と入力**します。
- **クリック**するか **Enter** キーを押して、**Move Forward** ノードを選択します。

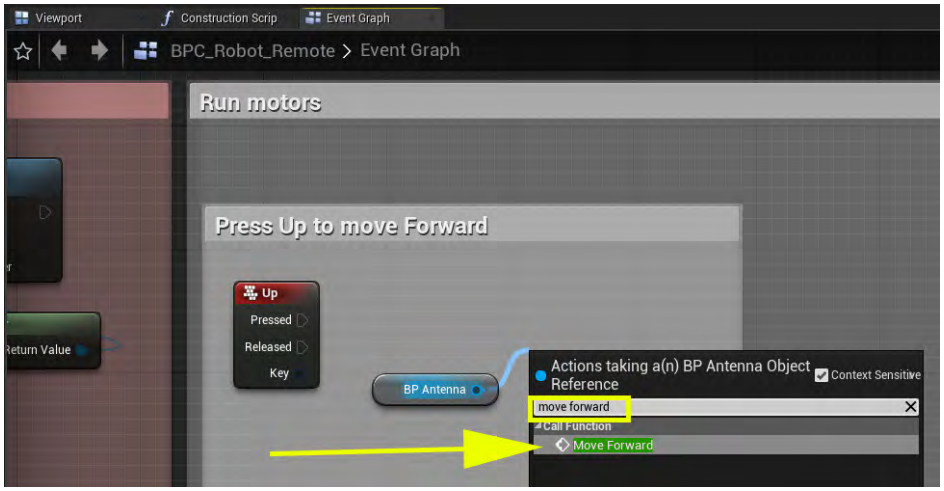


図 5-12

- BP Antenna ノードから Move Forward ノードの Target ピンに、青色のピンがつながれます。

Move Forward ノードが 2 つ必要です。それは、**Up** キーが押されたときにモーターに Speed を加算し、Up キーが放しされたときにその値を減算するからです。そうすると、モーターの回転が止まります。使用する速度値を決めるためにテストする必要があります。この値は -100 ～ 100 の範囲の値を取ります。最初は 30 以下の速度にしておくことをお勧めします。

- Move Forward ノードをもう 1 つ追加します (前の手順を繰り返すか、またはコピーして貼り付けて、2 つ目のノードをボックスに配置する)。
- **BP Antenna** ピンを**クリックして引き出し**、2 つ目の Move Forward ノードの **Target ピン**につなぎます。
- コードは図 5-13 のようになっているはずです。

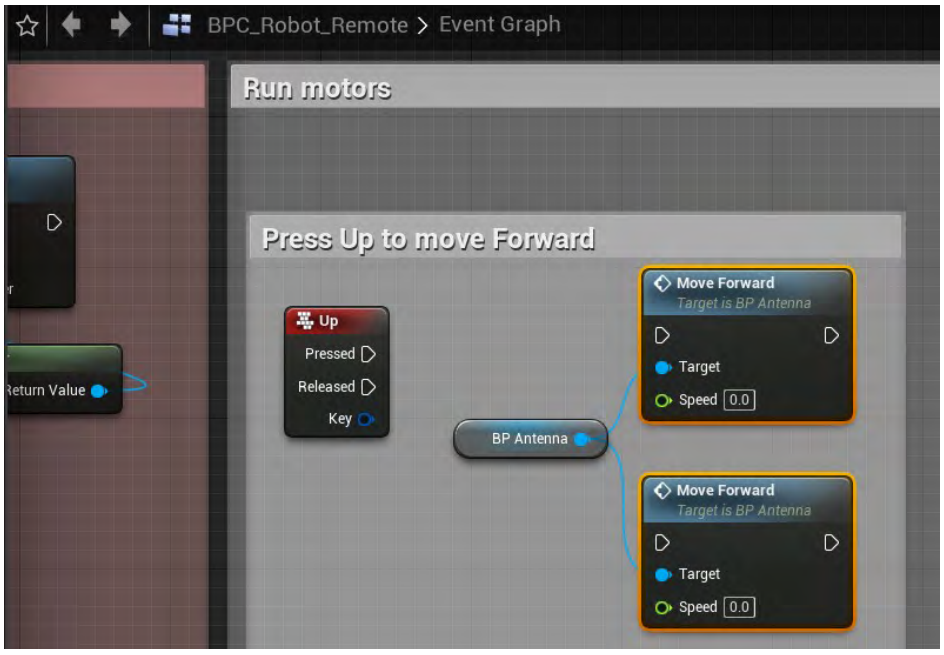


図 5-13

次に、Up ノードから Move Forward ノードにピンをつなぎます。

- **Up** イベントの **Pressed ピン**を**クリックし、白色のワイヤー**を引き出して、**1 つ目の Move Forward** ノードの入力
  につなぎます。
- **Up** イベントの **Released ピン**を**クリック**し、**白色のワイヤー**を **2 つ目の Run Motors** ノードの入力ピンにつなぎます。

それぞれの Move Forward ノードで、**Speed 値**を -100 ～ 100 の範囲内の好きな値に**編集**します。デフォルト値の 0 のままにはしないでください。最初は 30 以下の速度にしておくことをお勧めします。

- コードは図 5-14 のようになっているはずです。ただし、Speed は独自の値です。

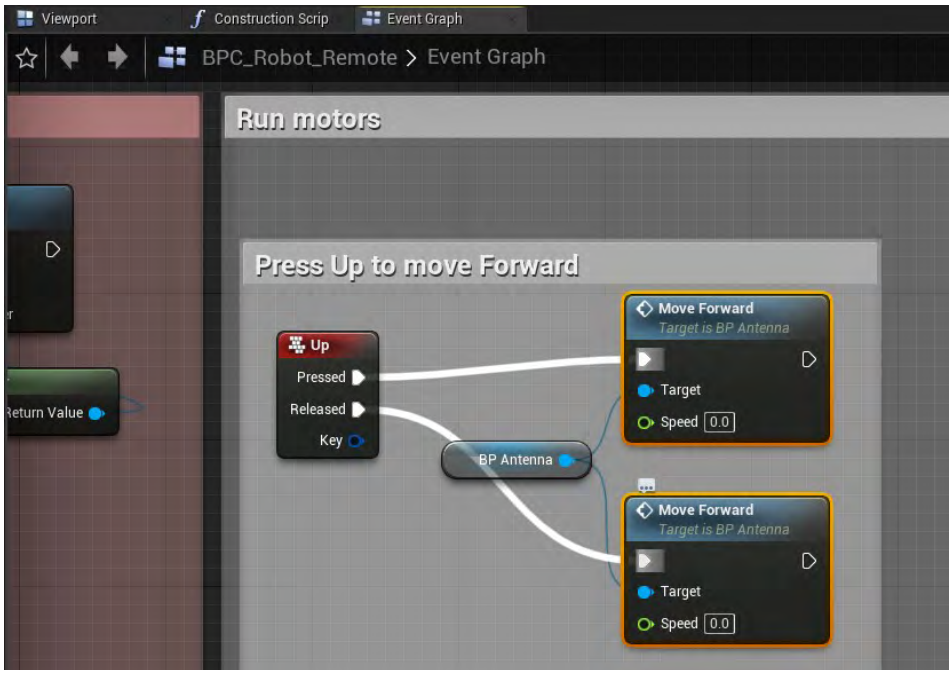


図 5-14

- **コンパイル**して**保存**します。

次のようにして、コードが機能しているかどうかを**テスト**します。

- **Map_5-1_RemoteControl** タブをクリックします。
- ゲームを**プレイ**します。
- **ビューポート**内で**クリック**します (遠隔操作キーを押す前に、ビューポートに移動しておくことは非常に重要です。[Play] を押した後にビューポート内でクリックしていないと、遠隔操作キーはアクティブ化されず、コードが正しくてもロボットは動きません)。
- キーボードの**上向き矢印キー**を**押したままにして**、ロボットが前進するかどうかを観察します。
- キーボードの**上向き矢印キー**を**放して**、ロボットが前進を止めるかどうかを観察します。
- **Esc キー**を押すとプログラムを停止できることを**思い出してください**

デバッグに関する注記:

- ロボットが動かない場合は、**[Play] を押した後にビューポート**内で**クリック**していて、キーボードのキー押下でロボットの動きを操作しながら、実際にゲーム フィールド内でプレイできることを確認します。
- コードを調べて、それぞれの Move Forward ノードで Speed に値を入力していることを確認します。

キーボードによる後退の操作

さきほど作成したコードをたたき台として使用して、ロボットを後退させることができます。

- **Up** と **BP_Antenna** ノードを複製します (それらを**選択**した状態で **Ctrl + W キーを押す**)。
- それらを**ドラッグ**して **Press Down to move Backward** コメント ボックス内に配置します。

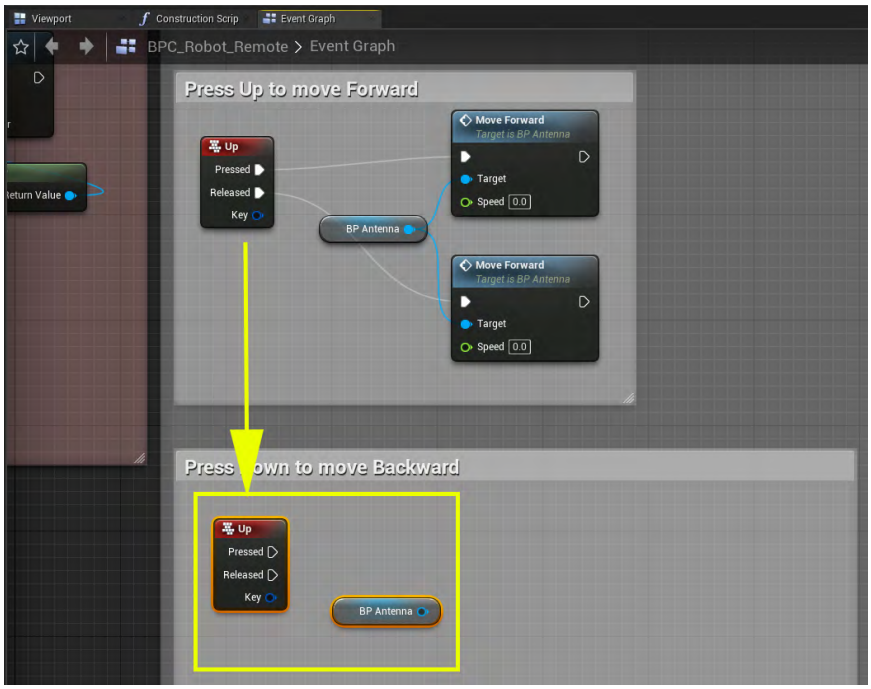


図 5-15

複製した **Up** イベントは、非常に簡単に **Down** イベントに変更できます。

- 新しい **Up ノードをクリック**して選択し、[Details] パネル (右側) を見ます。
- キーボードのキーの集合のように見えるボタンがあります (図 5-16 を参照)。
- この**ボタン**を**クリック**してから、キーボードの**下向き矢印キー**を**押す**と、イベントに割り当てられます。

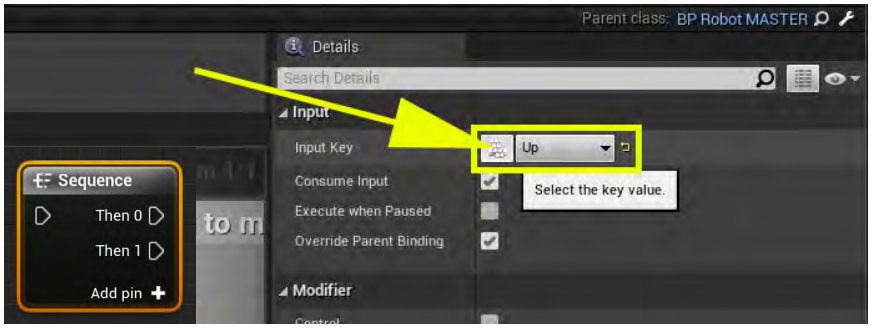


図 5-16

- そうすることで、押されたキーがイベントに「バインド」されます。
- このアクティビティでは、下向き矢印キーをこのイベントにバインドしています。

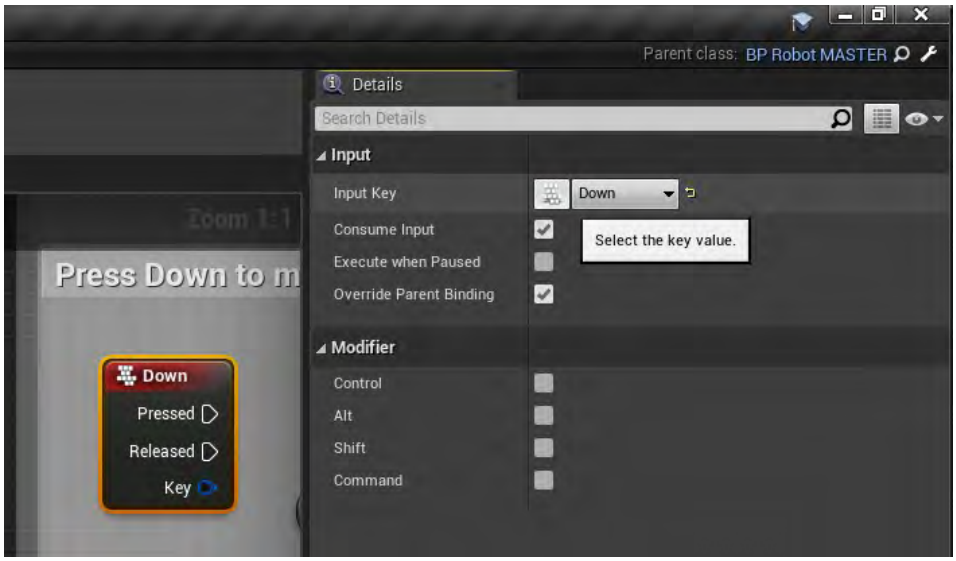


図 5-17

次に、Move Backward ノードを追加します。

- **BP Antenna** の**出力ピン**を**クリックし、青色のワイヤー**を**右側に引き出します。**
- [Actions taking a(n) Antenna Object] メニューで、「**move backward**」と**入力**します。
- **Move Backward をクリック**して選択します。

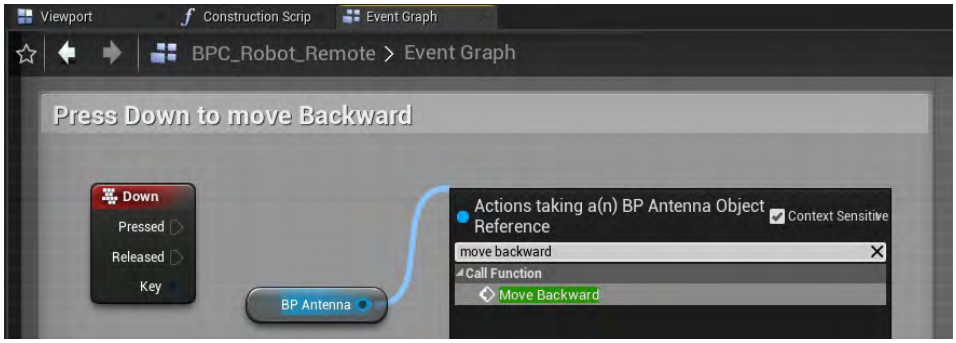


図 5-18

- Move Backward ノードを**選択した状態で Ctrl + W キーを押す**ことで、このノードを複製します。
- 2 つ目の Move Backward ノードを 1 つ目のノードの上方に**配置**します
- (図 5-19 を参照)。

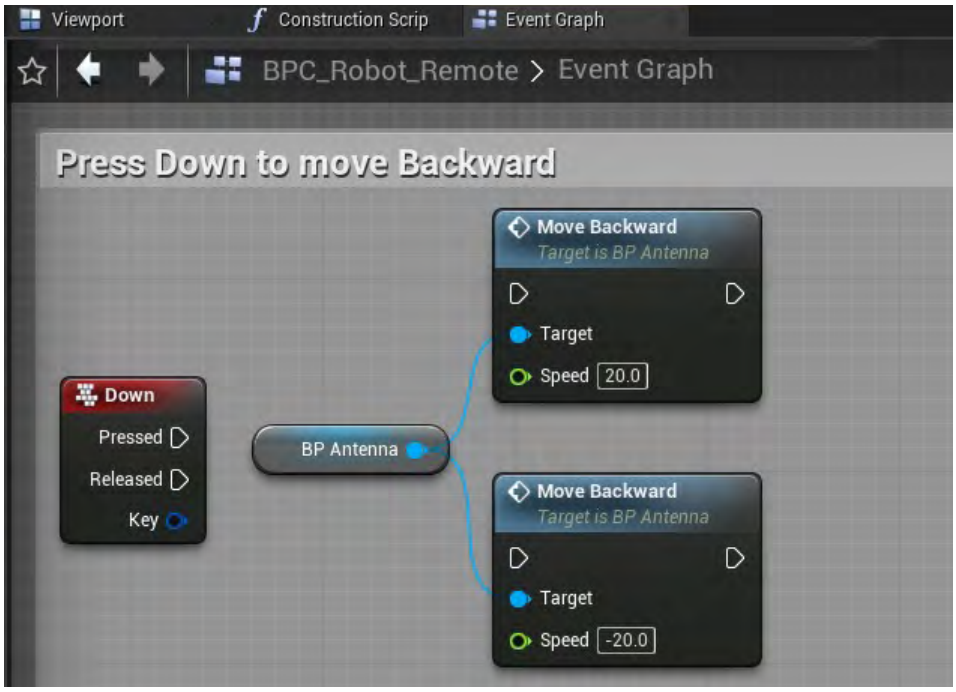


図 5-19

2 つの Move Backward ノードを Down イベントにつなぎます。

- Down イベントの **Pressed ピンをクリックして白色のワイヤー**を引き出し、上部にある **Move Backward ノードの入力ピン**につなぎます。
- Down イベントの **Released ピン**を**クリック**して**白色のワイヤー**を引き出し、**Move Backward ノードの入力ピン**につなぎます
- (図 5-20 を参照)。

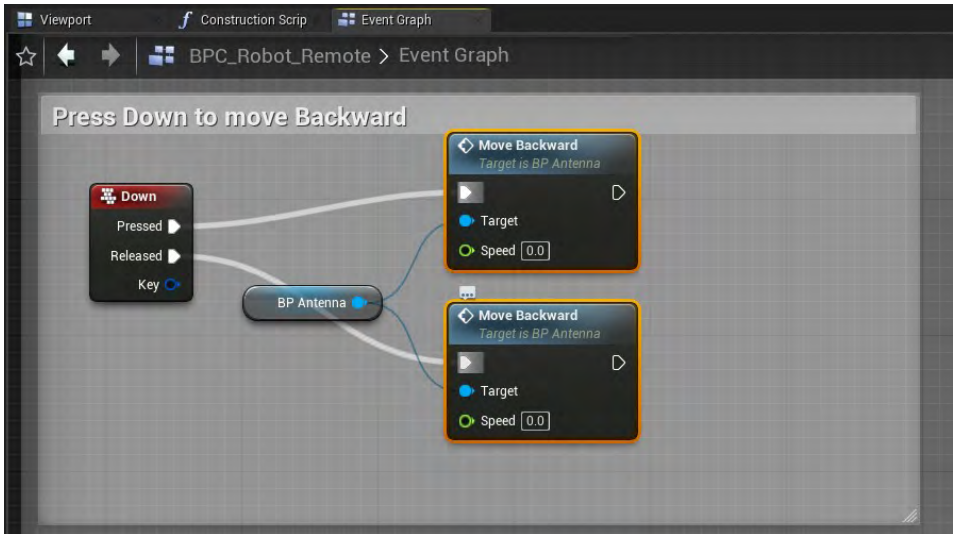


図 5-20

BP_Antenna 内にある **Polarity** 変数も設定する必要があります。そうすることで、ロボットを後退させるときにモーターが正しく回るようになります。

Set Polarity ノードを追加しましょう。

- BP Antenna の**出力ピンをクリック**して、**青色のワイヤー**を**右側に引き出します。**
- 「**set polarity**」と**入力**し、**Enter キーを押し**て選択し、それを配置します。

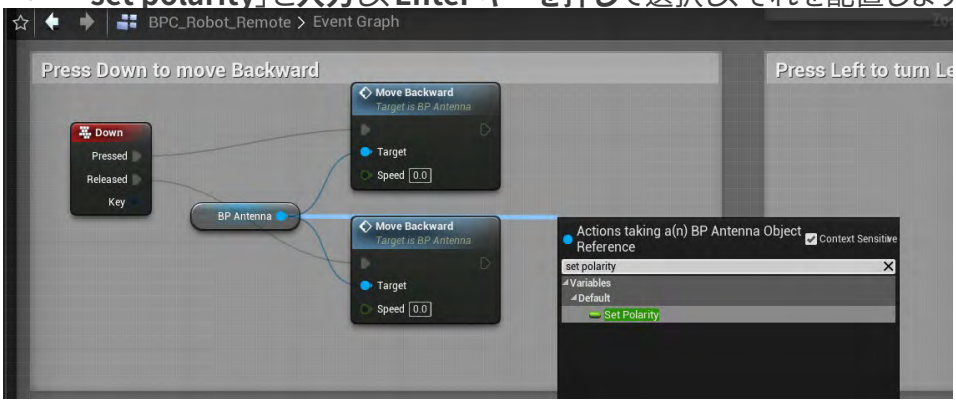


図 5-21

- **2 つ目の Move Backward の出力ピンをクリックして白色のワイヤーを引き出し、Set Polarity の In ピンにつなぎます。**
- Polarity テキスト ボックスで**「1」と入力**して、その値を 1 に変更します。
- **Move Backward Speed の値**を、上方にある **Move Forward Speed** と同じ値になるように**編集**します。
- **コンパイル**して**保存**します。
- **Map_5-1_RemoteControl タブ**を**クリックし、[Play] をクリック**してから、**ビューポート**内で**クリック**して、前進と後退をテストします。
- **Esc キー**を押すと、ゲームを停止してロボットの動きを止めることができることを**思い出してください。**

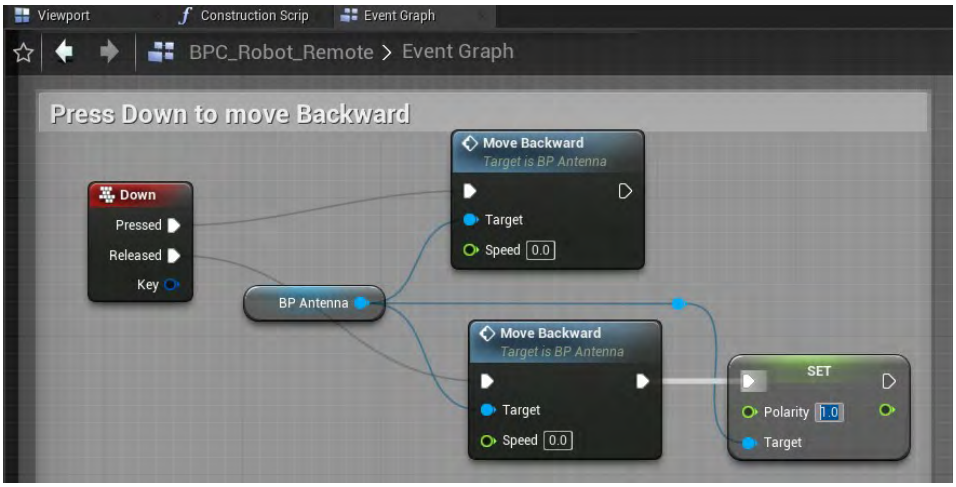


図 5-22

キーボードによる左旋回と右旋回の操作

次に、右向きと左向きに旋回するようにロボットをセットアップします。

- **Down イベント**と **BP_Antenna** を複製します (両方を選択した状態で **Ctrl + W キー**を押す)。
- コピーした新しいノードを**ドラッグ**して、**Press Right to turn Right** コメント ボックス内に配置します。
- もう一度 **Ctrl + W キーを押し**て、もう一つの新しいコピーを **Press Left to turn Left** コメント ボックスに配置します。

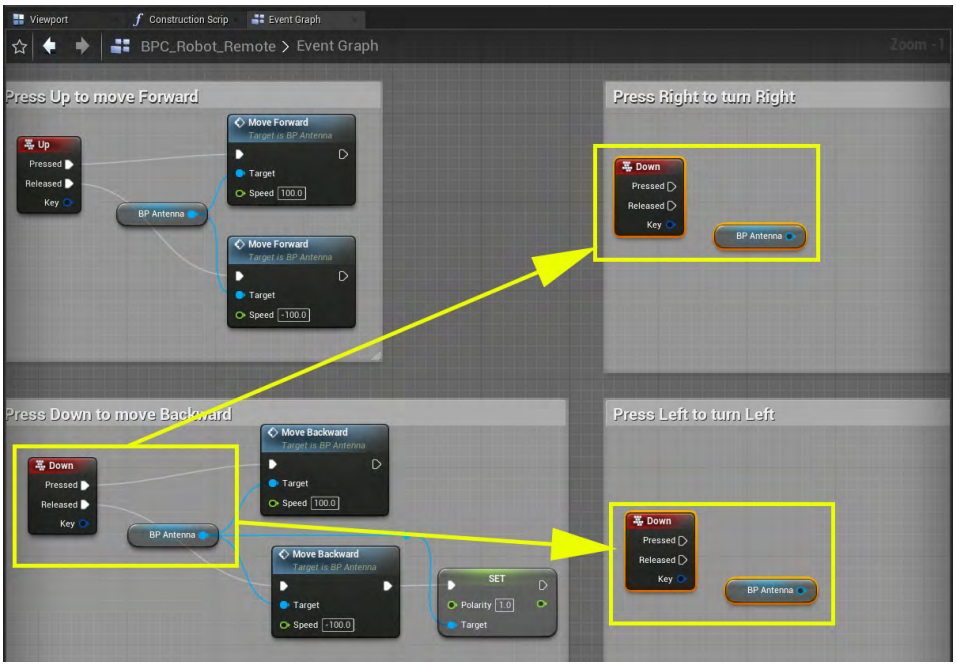


図 5-23

まず、右旋回をセットアップします。前にやったのと同様に、**Down** イベントを **Right** イベントに変換します。

- **Press Right to turn Right** コメント ボックスにある新しい **Down ノード**を**クリック**して選択します。
- [Details] パネル (右側) を見て、キーボードのキーの形状のボタンをクリックします (図 5-16 を参照)。
- キーボードの**右向き矢印キー**を**押す**と、それがイベントに割り当てられます。そうすることで、右向き矢印キーがこのイベントにバインドされます
- (図 5-24 を参照)。

図 5-24

- **BP_Motor_B** ノードを**クリック**して**青色のワイヤー**を右側に引き出します。
- [Actions taking a(n) Antenna Object] メニューで、「**turn r**」と入力して **Turn R** ノードを検索します。
- **Enter キーを押す**とそのノードが追加されます。
- 上記の手順を繰り返して、2 つ目の **Turn R** ノードを追加します。
- **それぞれのノードをドラッグ**して、**Press Right to turn Right** コメント ボックス内に収まるように配置します。

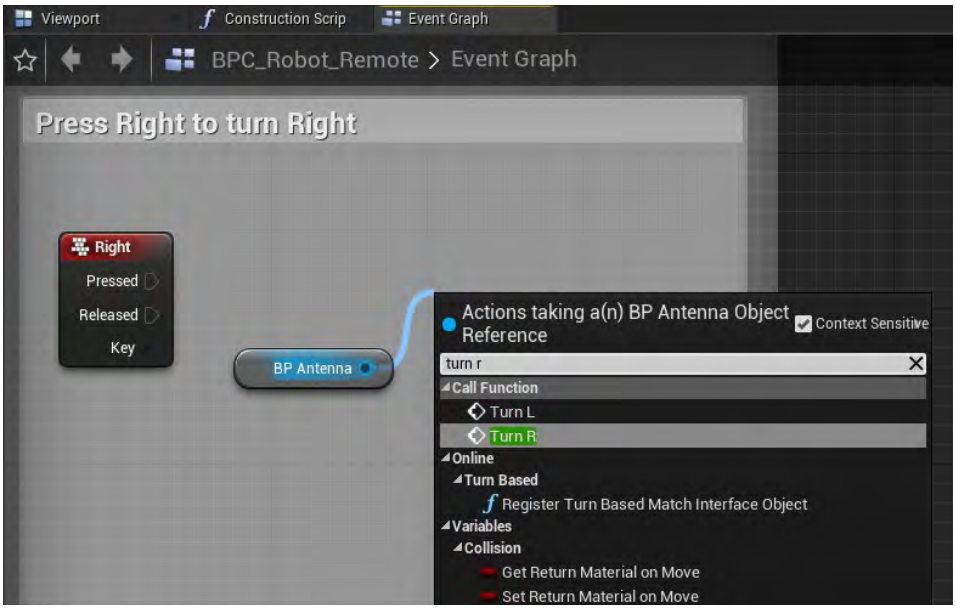


図 5-25

それらを **Right** イベントにつなぎます。

- Right イベントの **Pressed** ピンを**クリックして白色のワイヤー**を引き出し、上部にある **Turn Right** ノードの**入力**ピンにつなぎます。
- Right イベントの **Released** ピンを**クリックして白色のワイヤーを引き出し**、下部にある **Turn Right** ノードの**入力**ピンにつなぎます
- (図 5-26 を参照)。

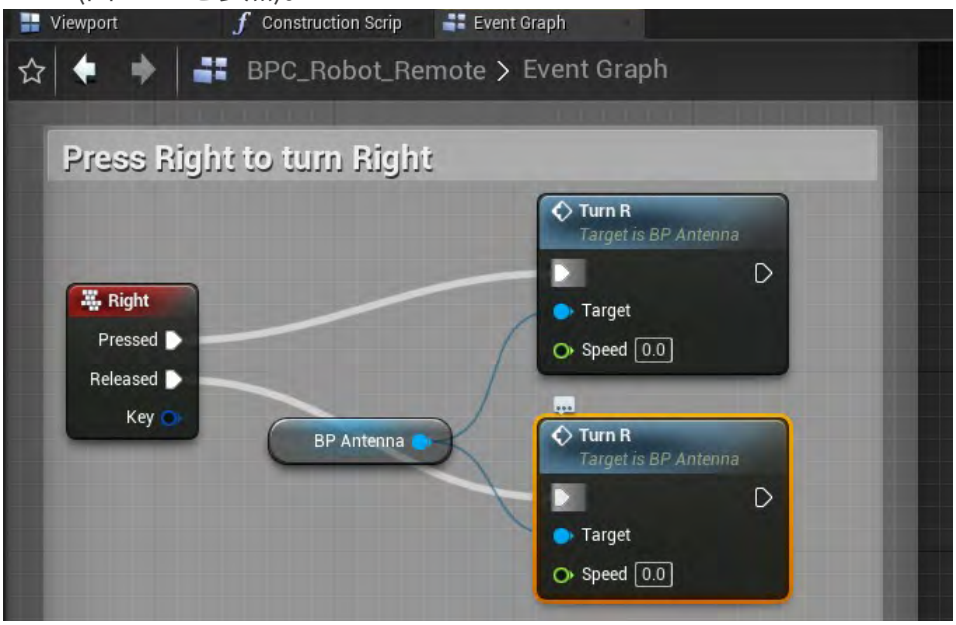


図 5-26

- それぞれのノードの Speed 変数の値を**編集**します。
- **コンパイル**して**保存**します。
- 次のようにしてテストします。**Map_5-1_RemoteControl タブ**を**クリックし、[Play] をクリック**し、**ビューポート**内で**クリック** してから、**右向き矢印キー**を**押し**て、結果をテストします。
- **Esc キー**を押すと、ゲームを停止してロボットの動きを止めることができることを思い出してください。
- 必要であれば、旋回速度を編集して調整します。そして、**コンパイル**して**保存**することを忘れないでください。

次に、左向きに旋回できるようにロボットをセットアップします。Left イベントになるようにイベントを変更します。

- **Press Left to turn Left** コメント ボックスにある新しい **Down ノード**を**クリック**して選択します。
- [Details] パネル (右側) を見て、**キーボードのキーの形状のボタン**を**クリック**します (図 5-16 を参照)。
- キーボードの**左向き矢印キー**を**押す**と、それがイベントに割り当てられます。そうすることで、左向き矢印キーがこのイベントにバインドされます

図 5-27

- (図 5-27 を参照)。
- **BP_Motor_B** ノードを**クリック**して**青色のワイヤー**を右側に引き出します。
- [Actions taking a(n) Antenna Object] メニューで、「**turn l**」と入力して **Turn L** ノードを検索します。
- **Enter キーを押す**とそのノードが追加されます。
- 上記の手順を繰り返して、2 つ目の **Turn L** ノードを追加します。
- **それぞれのノードをドラッグ**して、**Press Left to turn Left** コメント ボックス内に収まるように配置します
- (図 5-28 を参照)。

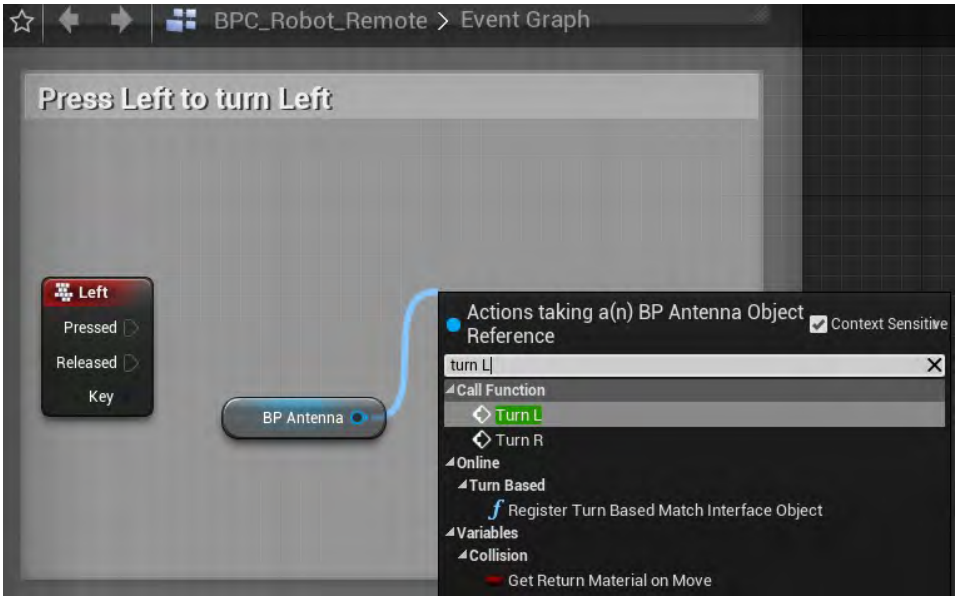


図 5-28

それらを **Left** イベント ノードにつなぎ、**Turn R** でやったのと同様に、**Speed** 値を設定します。

- **Left** イベントの **Pressed** ピンを**クリックして白色のワイヤー**を引き出し、上部にある **Turn L** ノードの**入力**ピンにつなぎます。
- **Left** イベントの **Released** ピンを**クリックして白色のワイヤーを引き出し**、下部にある **Turn L** ノードの**入力**ピンにつなぎます。
- それぞれのノードの Speed 変数の値を**編集**します。

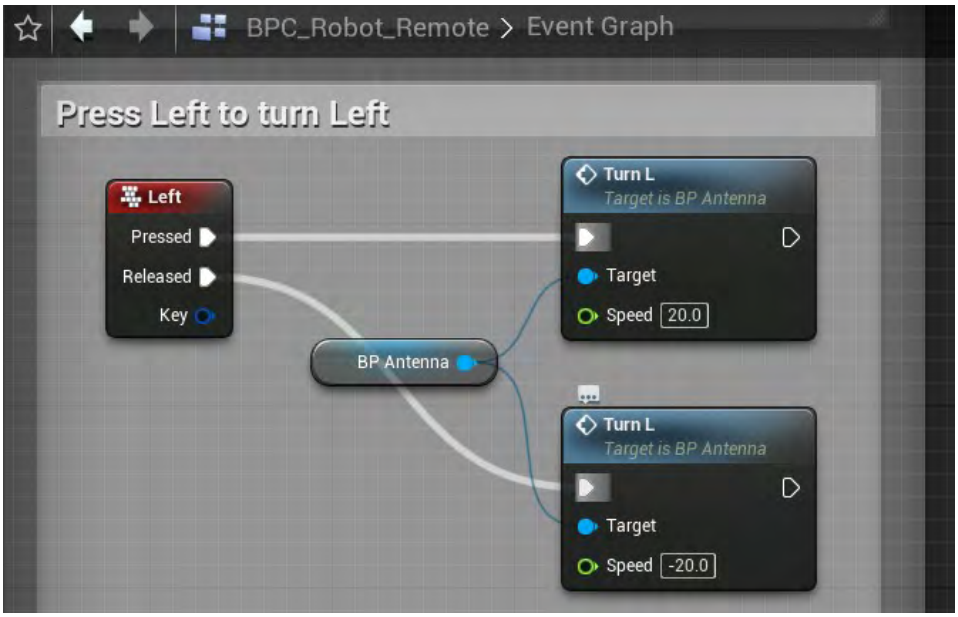


図 5-29

- **コンパイルして保存**します。

次のようにして、コードが機能しているかどうかを**テスト**します。

- **Map_5-1_RemoteControl タブをクリックします。**
- ゲームを**プレイします。**
- **ビューポート**内で**クリックします。**
- キーボードの**左向き矢印キー**を**押したままにして**、ロボットが左向きに進むかどうかを観察します。
- キーボードの**左向き矢印キー**を**放して**、ロボットが動きを止めるかどうかを観察します。
- **Esc キー**を押すとプレイを停止し、いつでもキーを使用してロボットが動くのを停止でき、マウスを動かしたときにカメラが動くのを停止できることを**思い出してください。**

デバッグに関するヒント：

- 既存のイベントを複製した後に、キー押下を正しい矢印キーにマッピングしていることを確認します。
- ノードで Speed の値を必ず設定します。
- 「Pressed」イベントは Speed 値が正であるノードにつながれていて、「Released」イベントは Speed 値が負であるノードにつながれている必要があります。
- スクリーンショットを参照して、すべてのワイヤーが正しくつながれていることを確認します。
- [Play] をクリックしてビューポートでユーザー入力をアクティブ化した後に、ビューポート内でクリックすることを忘れないでください。

キーボードによる移動操作がないことのチェックを追加する

場合によっては、ユーザーがあまりに多くのキーのキー押下とキー解放を素早く行うと、いずれかのイベントがスキップされることがあります。イベントがスキップされると、ユーザーがすべてのキーを放した後でも、ロボットが走行し続ける可能性があります。どのキーも押されていない場合に車両を停止させる、特殊な関数を作成します。

- 「Stop Motors and Reset Speed Vars」コメント ボックス内で**右クリック**すると、[All Actions for the Blueprint] メニューが表示されます。
- **「event tick」と入力**して検索し、メニュー オプションにある **Event Tick** を選択します。そうすると、イベント ティックが **Stop Motors and Reset Speed Vars** コメント ボックスに追加されます。
- このコードは、割り当てられているキーが押されていない場合に、すべての動きの Speed 値をリセットします。

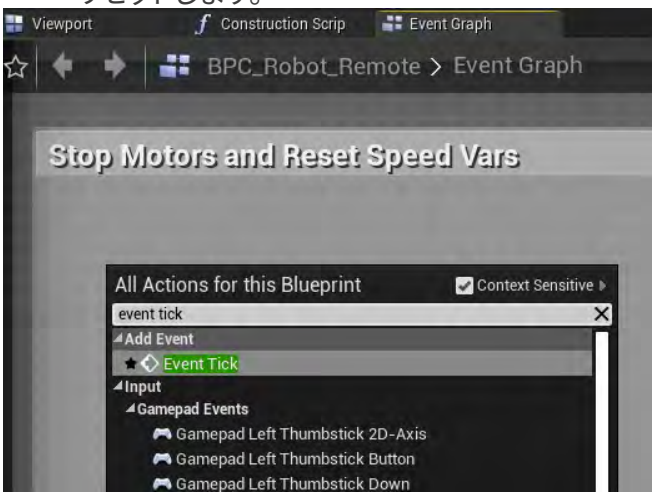


図 5-30

- **Event Tick** の実行ピンを**クリック**して**白色のワイヤー**、を**右側に引き出します。**
- 「**branch**」と入力して **Enter キー**を押すと、**Branch** ノードが追加されます。

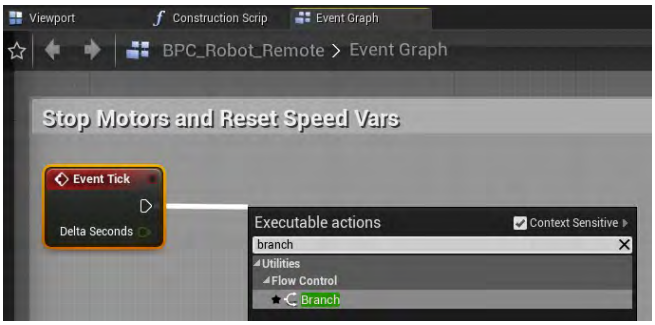


図 5-31

- 「Stop Motors and Reset Speed Vars」コメント ボックス内で**右クリック**すると、[All Actions for the Blueprint] メニューが表示されます。
- **「get player controller」と入力**して **Enter キー**を押すか**クリックして選択**すると、**Get Player Controller** イベントが追加されます。

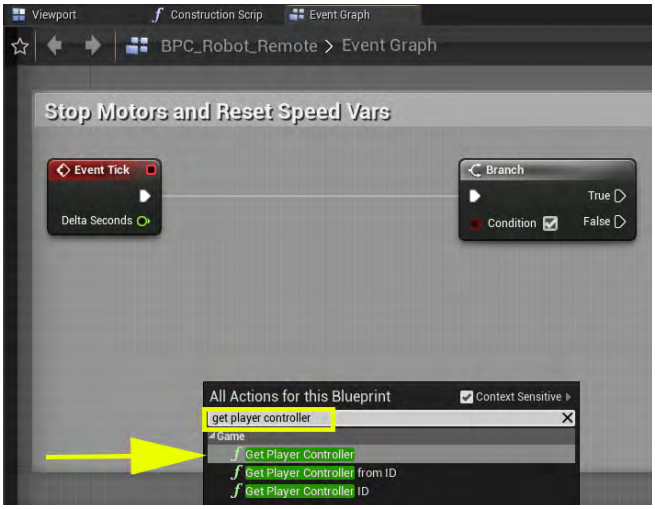


図 5-32

- Get Player Controller イベントの **Return Value ピンをクリック**して**青色のワイヤー**を**右側に引き出します。**
- **「is input key down」と入力**して **Enter キー**を押すか**クリックして選択**すると、**Is Input Key Down** イベントが追加されます
- (図 5-33 を参照)。

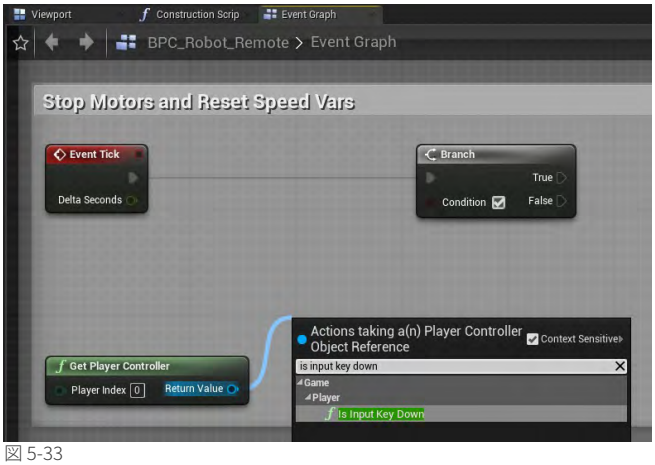


図 5-33

- **Key drop down** を**クリック**します。
- 「**any key**」と**入力**し、**Enter キー**を押すと、それが Any Key に設定されます。

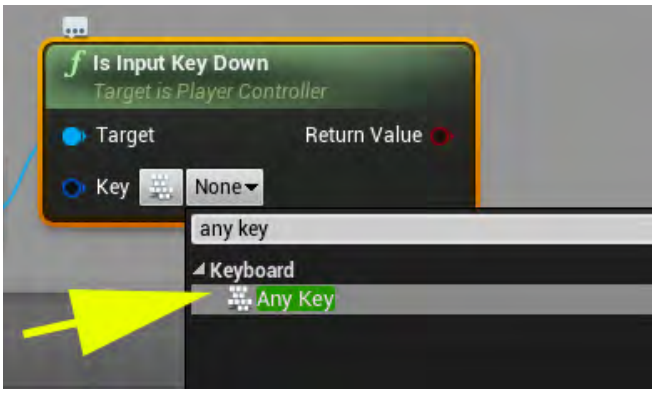


図 5-34

- **Is Input Key Down** イベントの **Return Value** ピンを**クリック**して**ワイヤー**を**右側に引き出します**。
- 「**not boolean**」と入力して**選択**すると、**Not Boolean** ノードが追加されます。

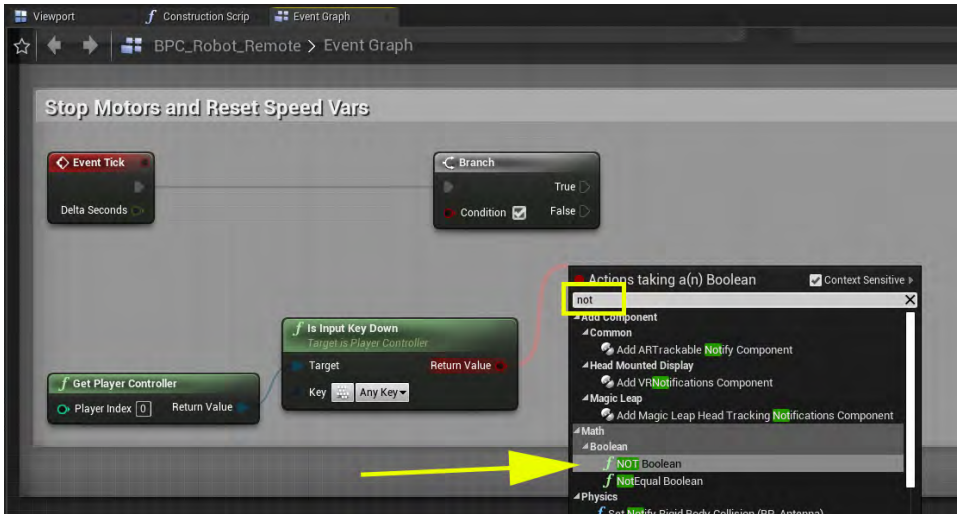


図 5-35

それを **Branch ノードの Condition ピン**につなぎます

- **Branch ノード**をさらに**右側にドラッグして、**NOT ノードの上方に配置します。
- **NOT Boolean のピン**を**クリック**して**赤色のワイヤーを引き出し、Branch ノードの Condition ピン**につなぎます。
- コードは図 5-36 のようになっているはずです。

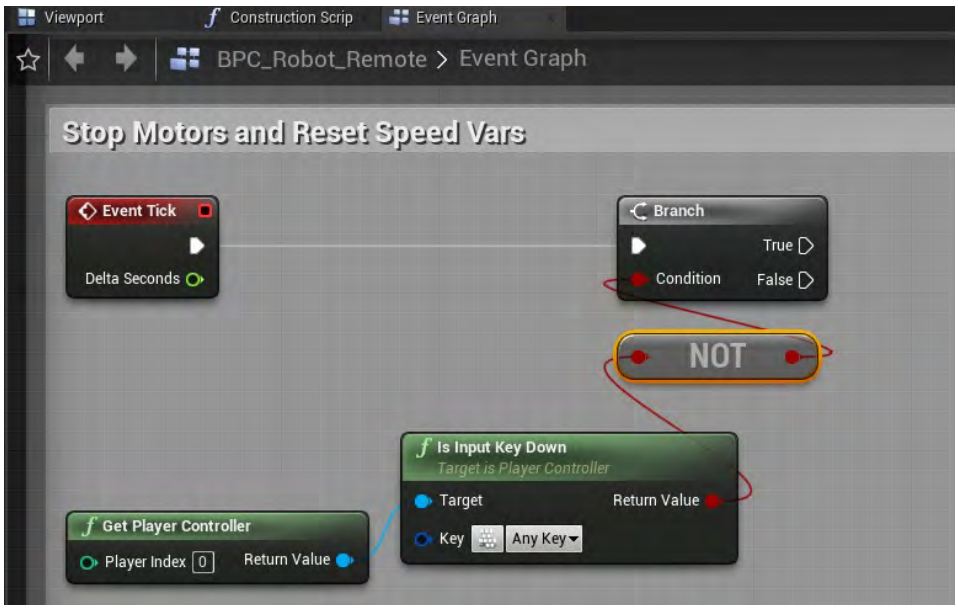


図 5-36

- **[Components] パネル**で、**BP_Antenna** をクリックしてイベント グラフに**ドラッグ**すると、**NOT Boolean** ノードの右側にノードが追加されます (図 5-37 を参照)。

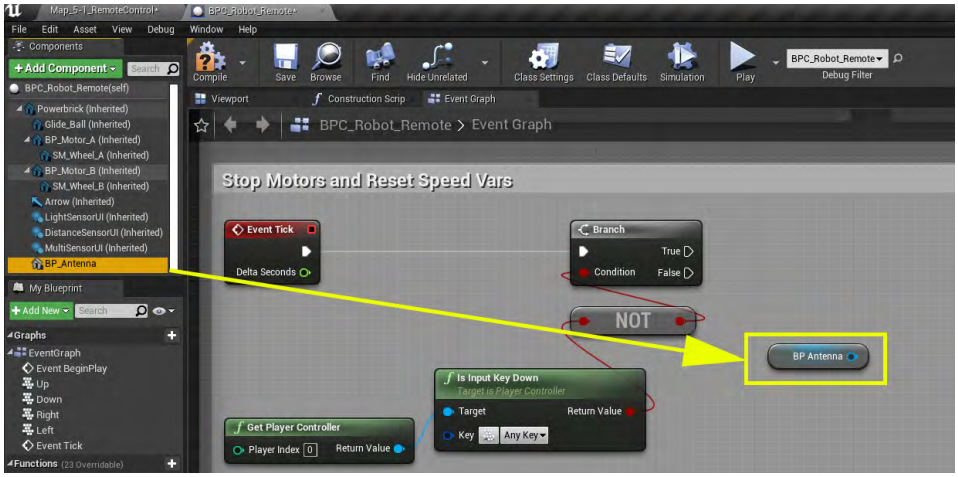


図 5-37

- **BP_Antenna** ノードのピンを**クリック**して**青色のワイヤー**を**右側に引き出します。**
- **「Stop Motors」と入力**してそれを**選択**します。
- そうすると、**Stop Motors** ノードが **BP_Antenna** ノードからつながれます。

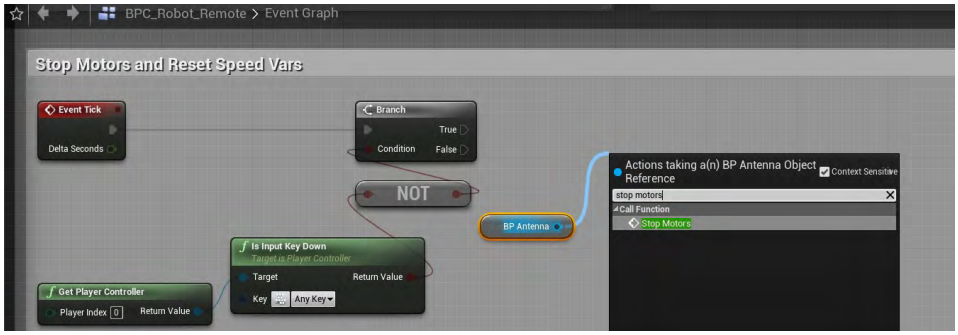


図 5-38

- **BP_Antenna** ノードのピンを**クリック**して**青色のワイヤー**を**右側に引き出します**。
- **「set speed」と入力**して **Set Speed A** を選択すると、ノードが配置されます。
- **BP_Antenna ノードのピン**を**クリックして青色のワイヤー**を**右側に**引き出します。
- **「set speed」と入力**して **Set Speed B** を選択すると、ノードが配置されます。
- 両方のノードを図 5-39 のように並べ替えます。

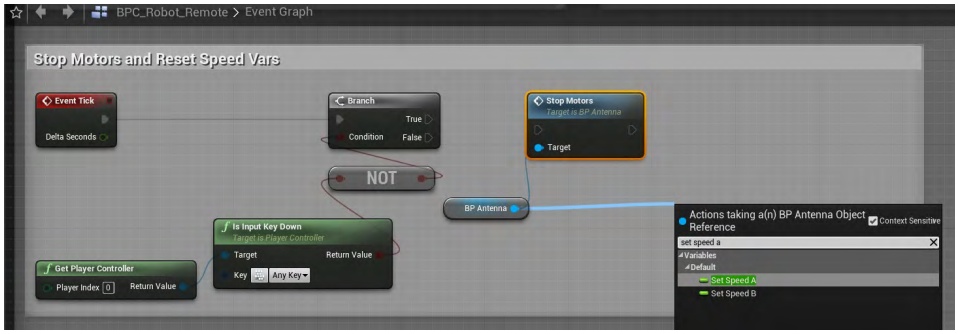


図 5-39

図 5-40 に示されているように、Branch ノードと他のノードの間で実行ピンをつなぐ必要があります。

- Branch ノードの **True ピン**を**クリックして白色のワイヤーを引き出し**、Stop Motors の **Input ピン**につなぎます。
- Stop Motors の**出力ピン**を**クリック**して**白色のワイヤーを引き出し**、Set Speed A の**入力ピン**につなぎます。
- Set Speed A の**出力ピン**を**クリック**して**白色のワイヤーを引き出し**、Set Speed B の**入力ピン**につなぎます。
- 必要であれば、いくつかのノードをドラッグして、ボックス内にすべてのノードが収まるようにスペースを空けます。

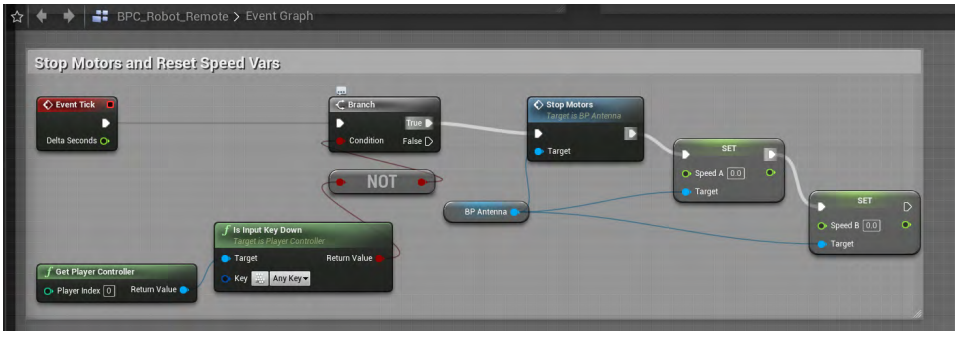


図 5-40

- コード全体は図 5-41 のようになっているはずです。

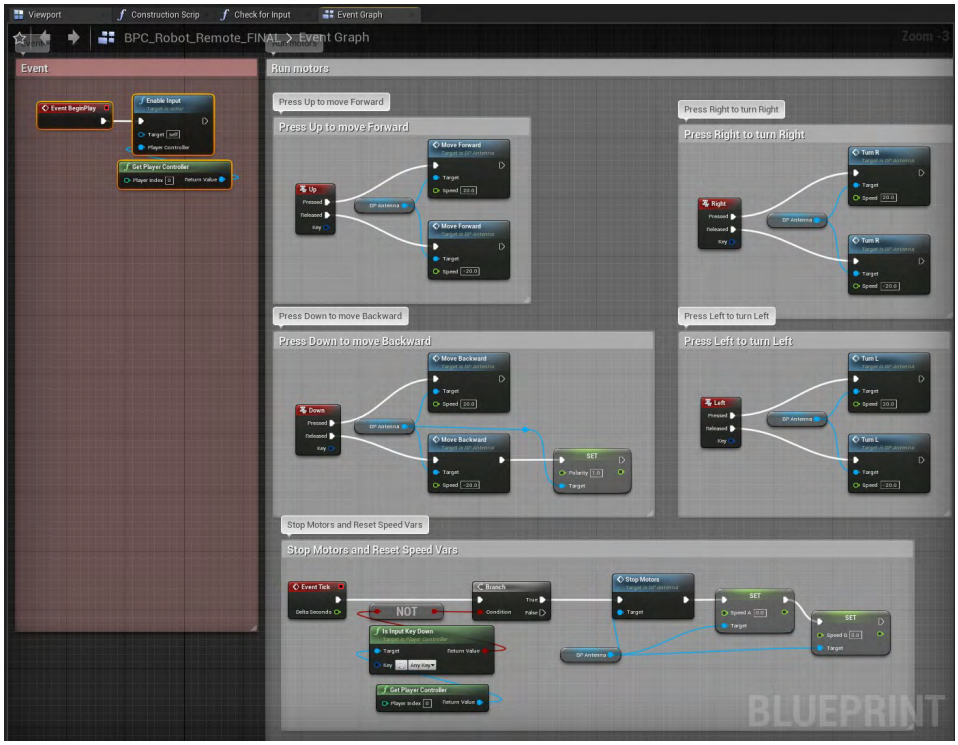


図 5-41

- **コンパイルして保存**します。

次のようにして、新しいコードをテストします。

- **Map_5-1_RemoteControl** タブをクリックします。
- ゲームを**プレイします。**
- ビューポート内で**クリック**します。
- キーボードの 1 つまたは複数の**矢印キー**を**押します。**
- 指示した向きにロボットが動くかどうかを**観察**します。
- キーボードの**矢印キーを放す**と、ロボットが動きを止めるかどうかを観察します。
- **Esc キー**を押すとプレイを停止し、いつでもキーを使用してロボットが動くのを停止でき、マウスを動かしたときにカメラが動くのを停止できることを**思い出してください。**
- W/A/S/D キーを使用してカメラをあちこち動かし、ロボットのビューおよびサッカー場での配置を改善します。
- 観察結果をメモします。

実習 4：移動のためのモーター値をテストして微調整する

通常は、ロボットの速度を上げるほど、正確さや一貫性の問題が大きくなります。構成した移動操作を評価します。ロボットの速度が速すぎたり、遅すぎたりしていませんか。

- サッカーの試合をプレイし、サッカー ボールをゴールに数回入れてみます。
- キーボードからの指示に対する行動とロボットの反応を観察します。
- 必要に応じて、コードで調整を行います。
- 編集してゲームをプレイし直すときは毎回、コンパイルして保存することを忘れないでください。

アクティビティ：前進または右旋回だけでなく右向きの前進のように、2 つの向きを組み合わせることができます。前進と右旋回が独立して機能し、両方のキーが同時に押されている場合はそれらが組み合わされて機能します。

さまざまな向きの動きを組み合わせて、サッカー ロボットを走行させてみます。

課題：2 人目のプレイヤー用のロボットを追加する

好みの制御値が見つかったら、ロボットを複製し、そのロボットが異なるキー押下で動作するように設定できます。独立して操作される 2 つのロボットを使用して、1 対 1 のサッカーの試合をすることができます。

目的

実際に機能するロボットを複製し、2 つ目のロボットの操作をキーボードの別のキーにマッピングします。ロボット間でコードをコピーする方法については、リソースを参照してください。

プレイして反復する

ゲームの作成時には、プレイテストを何度も行います。どのような改良を行うと、ロボットが成功を収めることが多くなりますか。

拡張アクティビティ

- 制御不能にならない程度にロボットの速度を上げてみる。
- サッカー ボールをもっとうまく扱えるように、ロボットの設計を変更する。
  ヒント：*すでに用意されているアクセサリーを使用してロボットを変更できる。*
- モーターを増やして特殊な制御を与え、その機能をキー押下にマッピングする。
- カメラの動きを操作できるようにゲーム コントローラーをマッピングする。

リソース

観察ログ – 空白のログと、拡張アクティビティ用のログ

評価
基準

概念	抜群	熟練	適格	未熟
ロボット設計の概念	ロボット制御のキー マッピングとそれがどのように機能するかを他の人に説明できる。プロジェクトで、レッスンでのキー マッピングの使用およびロボットでの最適な活用法を示している。デモまたは生徒のドキュメントで、コンポーネントの定義が示されている。	ロボットでのキー マッピングの活用法を示している。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、講師に示されている。	ロボットをキー マッピングに対応させることはできるが、キー マッピングについて意義のある知識を提供していない。ハードウェア コンポーネントについての基本的な知識を持っていることが示されている。	ロボットのキー マッピングその機能を理解している根拠がない。ハードウェア コンポーネントについての知識を持っていることが示されていない。
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。キー マッピングの使用、キー マッピングの機能、および結果としてのロボットの望ましい行動に必要なコマンド 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 グループがプレゼンテーションまたはドキュメントで示されている。	キー マッピングの使用、キー マッピングの機能、および結果としてのロボットの望ましい行動に必要なコマンド グループについて理解している。複数のコマンドまたはコマンド グループについて生徒のプレゼンテーションで触れられている。	促されれば、キー マッピングの使用、キー マッピングの機能、および結果としてのロボットの望ましい行動に必要なコマンド グループについて基本的な知識を持っているが、デモでは触れていない。	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、イベント、条件付きコマンドについての知識が示されている。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 つ以上を参照している。	促されれば、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、そのこと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に 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 (遠隔操作できるサッカー ロボット)	遠隔操作できるサッカー ロボットを正常に完了した。取り組み、試行、成功したことをドキュメントに記載している。ゲームに勝つサッカー ロボットに必要とされる革新的なパーツや機能を追加することを示している。	カスタマイズした遠隔操作でサッカー ロボットを走行させることに成功した。サッカー ゲームでのすべての動きを行うために必要なキー押下をコーディングした。	ロボットを走行させるための遠隔操作の最初の部分を示している。必要なすべての動きを使用してサッカー コースをやり終えることに成功していない。	遠隔操作できるサッカー ロボットを作成しなかった。キー押下をマッピングできることを示していない。

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 5：遠隔操作できるサッカー ロボット