01 起動方法

Java 同梱版の場合：展開したフォルダ内の「起動.bat」をダブルクリック

単独版の場合：展開したフォルダ内の「launcher.jar」をダブルクリック

モード説明

プレイモード

実際にロボットを動かすモードです。新たにワールドを作成する「開始(ホスト)」と、すでに起動しているワールドに参加する「参加（ゲスト)」があります。

エディットモード

機体を組み立てるモードです。

プログラムモード

機体の制御プログラムを作成するモードです。

02 操作方法

ランチャー画面

マウスで操作します。

ゲームのプレイ

---

1. 機体一覧から select ボタンを押し、機体を選択
2. ゲームを開始

   (ア) ローカルプレイの場合：開始(ホスト)を選択

   (イ) オンラインプレイ(ゲスト)の場合

   ① 接続先アドレスにホストのアドレスを入力

   ② 参加(ゲスト)を選択

   (ウ) オンラインプレイ(ホスト)の場合

   ① 参加者から 51225 ポートがアクセス可能な状態にする

   ② 開始(ホスト)を選択

機体を作成

---

1. 機体作成ボタンの左のフォームに作成する機体の名前を入力
2. 機体作成ボタンを押す
3. edit ボタンでエディットモードを起動し、機体を作成
4. program ボタンでプログラムエディタを起動し、制御プログラムを作成

02 操作方法

プレイモード

マウス移動	視点切り替え
マウスホイール	ズーム調整
F キー	復帰
Enter キー	初期地点に移動
その他キー	機体の操作(機体制御プログラムによる)
Esc キー	マウスカーソルの表示・非表示

エディットモード

マウス移動	視点切り替え
左クリック	ブロックを移動、設置
右クリック	ブロックを削除
ホイールクリック、Q キー	作業パーツの切り替え
WASD、Space、Shift キー	視点の移動
矢印、カンマ、ピリオドキー	ブロックの移動
Ctrl+矢印、カンマ、ピリオドキー	機体全体の移動
E+カンマ、ピリオドキー	ブロックの回転
E+矢印キー	ブロックの種類変更
Z キー	取り消し
Y キー	やり直し
C キー	パーツの複製
Tab キー	パーツ編集とブロック編集の切り替え
F1 キー	データの保存
Esc キー	マウスカーソルの表示・非表示

03 ゲーム概要

ロボットを作成して遊ぶ 3D 物理シミュレーションゲームです。

できること

- 車を作って走らせる
- 光センサーを利用してライントレースする
- ブロックを拾って運ぶ
- 歩く

など、アイデア次第で様々なことができます。

オンラインプレイについて

TCP による通信でオンラインプレイができます。標準では 51225 ポートを利用します。

オンラインプレイでホストになる際にはポート開放などが必要となります。

04 機体作成の方法

機体の構成

機体は複数のパーツから構成されます。

パーツは複数のブロックから構成されます。パーツ同士はモーターブロックなどで接続され、親子関係を持ちます。

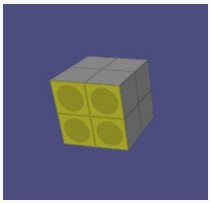
ブロックの種類

基本ブロック

基本となるブロックです。様々な大きさがあります。

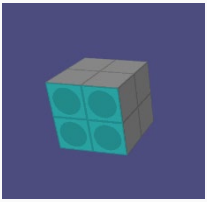
モーターブロック

接続したパーツ同士を回転させるブロックです。黄色い面に接続された子パーツを回転させます。



光センサーブロック

明度を判定するブロックです。水色の面の先にあるブロックの明度を判定します。ブロックまでの距離によって値は変動します。このため、距離を調べることにも利用できます。



05 制御プログラム

機体の制御プログラムはブロックを組み合わせることで作成します。

プログラムは「プログラムが呼ばれたとき」というブロックから始まり、上から下へと実行されていくことが基本です。

使用可能な文法

- float 型変数
- if 文、if-else 文
- while 文
- 関数定義・呼び出し
- 1 次元配列（テキスト編集でのみ）
- //や/* */によるコメント（テキスト編集でのみ）

定義済みの変数

ブロックのアドレス

ブロックの固有名が、制御プログラム内では変数名として扱われ、値はそのブロックを表すアドレスとなります。ブロックの固有名はエディットモードで確認できます。

例：MOTOR_0、SENSOR_1

キー番号

ブロックのアドレスと同様に KEY_<キー名>の形で変数が定義されています。

例：KEY_W、KEY_UP 、KEY_LEFT_SHIFT

組み込み関数

float get(float addr)

与えられたアドレスの表すブロックが

- モーターブロックなら、現在の角度(-π から π)
- 光センサーブロックなら、明度

を返します。

float set(float addr, float value)

与えられたアドレスの表すブロックに値を設定します。

モーターブロックに対しては電圧が設定できます。

float key(float number)

与えられた番号に対応するキーが押されているかを調べます。

戻り値：押されているなら 1，でなければ 0

float keyDown(float number)

与えられた番号に対応するキーが押されたかを調べます。

key 関数と違い、押されたフレームのみ 1 を返します。

戻り値：押されたなら 1，でなければ 0

float print(float...args)

任意の個数の引数を取り、すべてをカンマ区切りで出力します。

float sin(float rad)、float cos(float rad)、float tan(float rad)

ラジアンで表された角度から三角関数を計算し、その値を返します。

float asin(float v)、float acos(float v)、float atan(float v)

逆三角関数の値を返します。

範囲は

- asin と atan：$-\pi/2$ から $\pi/2$
- acos：0 から π

float sqrt(float v)

平方根を返します。

float exp(float v)

オイラー数 e の累乗を返します。

最低要件

OS：Windows(Windows 10 Pro 20H2 で動作確認)、Linux(Ubuntu 18.0.4 で動作確認)、mac(動作未確認)

CPU: x64 の CPU

備考：Java8(64-Bit)かそれ以降が必要

推奨環境

シングルプレイまたはマルチプレイでホストをする場合

CPU：ちょっと良いもの(Intel Core i5-9600 にて動作確認)

07 困ったときは

実行ができない場合

1. Java(64-Bit)はインストールされていますか?
   されていない場合はインストールしましょう
2. インストールされている Java は 64-Bit 版ですか?
   異なる場合は 64-Bit 版のインストールをしましょう
3. zip ファイルは展開しましたか?
   ダウンロードした zip ファイルを展開してから起動してみましょう
4. 機体は選択できていますか?
   機体一覧から機体を選択してみてください
5. これらに当てはまらない場合
   Twitter や配布サイトの質問フォームなどから連絡をください

動作が重い場合

複雑な機体は動作が重くなります。ブロックの数を減らせないか考えてみましょう。たくさんの小さなブロックよりも大きなブロックを利用した方が処理は軽いです。また、ブロック同士がめり込んだ場合は非常に処理が重くなります。

機体が暴走する場合

現状では仕様です。複雑な機体や大きな機体は不安定になる場合が多いです。

アップデートにより改善できればと考えています。