Wiiリモコンプログラミング

ソフトウェア実験

（ユーザインタフェース3）

担当：西出　俊

Wiiリモコンとは

1．一般的なゲームコントローラと
　同様の入力装置
　（十字キーとボタン）

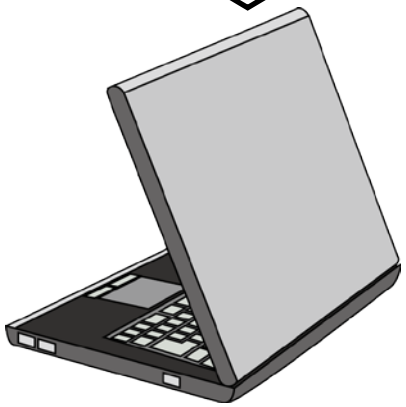
前面

2．内蔵された加速度センサを
　利用した入力装置
　（人間の動作と連動した操作等）

背面

Wiiリモコンとコンピュータの接続

Bluetooth



デバイスの検索

①と②ボタンを同時押し
または
背面のSYNCボタンを押す

```
$ hcitool scan
Scanning …
        00:22:D7:AA:14:D9   Nintendo …
```

識別ID（リモコンにも貼ってある）

識別IDを用いて実行ファイルで
Wiiリモコンを使用する

Wiiリモコンプログラミングライブラリ

Wiiリモコンプログラミングライブラリとは？

種類：libwiiremote, cwwid, libwiimoteなど

電算室のPCにインストール済み

libwiimoteで定義されている関数を
利用してプログラミングをする

ヘッダファイルのインクルード

```
#include <libcwiimote/wiimote.h>
#include <libcwiimote/wiimote_api.h>
```

wiimote.hとwiimote_api.hで定義されている
関数を使用することが可能になる

Wiiリモコンの変数

宣言

```
wiimote_t wiimote = WIIMOTE_INIT;
```

wiimote_t構造体の中身（よく使うもの）

wiimote.keys : 指定したボタンの状態
wiimote.mode : wiiリモコンのモード
wiimote.axis : x, y, z軸それぞれの加速度
wiimote.tilt : x, y, z軸方向の傾き
wiimote.force : x, y, z軸に対する重力加速度
wiimote.ir1:赤外線センサによって検知した座標

プログラミングの手順

1. ヘッダファイルのインクルード
   wiimote.h, wiimote_api.h
2. Wiiリモコンに接続
   （例）wiimote_connect(&wiimote, argv[1]);
3. Wiiリモコンの操作記述
   wiiリモコンとSDLとの連動など
4. Wiiリモコンの接続解除
   （例）wiimote_disconnect(&wiimote);
5. プログラム終了

コンパイル方法

```
gcc test.c –D_ENABLE_TILT –D_ENABLE_FORCE –L/usr/lib
```

十字キー、ボタンの操作

（例） 上ボタンが押された

if(wiimote.keys.up){操作内容}

wiimote.keys.up != 0の略

上（up）、下（down）、左（left）、右（right）
それぞれに対応した操作を操作内容に記述

（例） Aボタンが押された

if(wiimote.keys.a) {操作内容}

対応するボタンをwiimote.keys.ボタンに入れる

センサ使用

センサを使用するモード

wiimote.mode.acc = 1;

Wiiリモコンの傾き、加速度、重力加速度の大きさ

wiimote.axis, wiimote.tilt, wiimote.force

どの程度の値になるか直感的には分からない・・・

printf(“%03d　%03d　%03d¥n”, wiimote.axis.x,
wiimote.axis.y, wiimote.axis.z);

実行してprintfで値を出力し、その値を見ながらプログラムを組んでいく

振る操作

どうやって振る操作を判別するか？

1フレーム（時刻）だけの
センサ値からは判別できない・・・

数フレーム分のセンサ値を評価して、条件文を作る

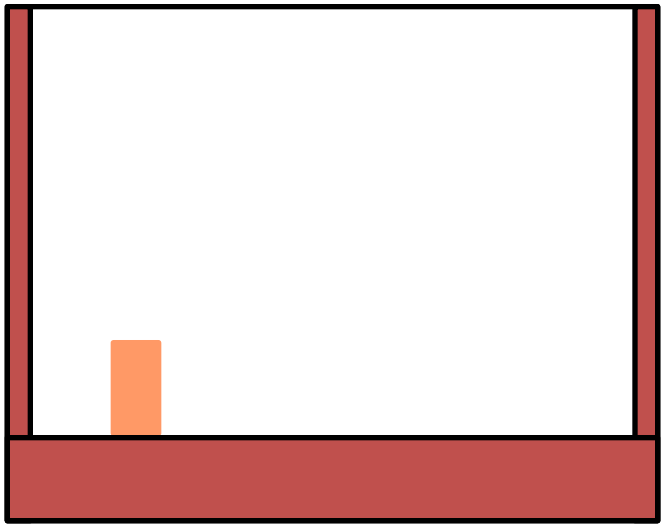
実際に振った時のセンサ値をprintfで出力して
振っていない時と振っている時のセンサ値の
差から振る操作の条件文を作る

SDLとの連動

SDL

Wiiリモコン

入力



Wiiリモコンの入力待ち

入力が来ると対応した処理をする

実際のプログラミングは？

サンプルプログラムwii_sample.c、
wii_kadai.cを改良していく。

wii_sample.c　：　wiiリモコンの各ボタンに対応する
変数やその値を調べる時に用いる
wii_kadai.c　：　UI3の課題のベースとなるプログラム
（wii_kadai.cから不要なものを削除）

試しに実行してみよう

そのままコンパイルして実行する

コンパイル方法

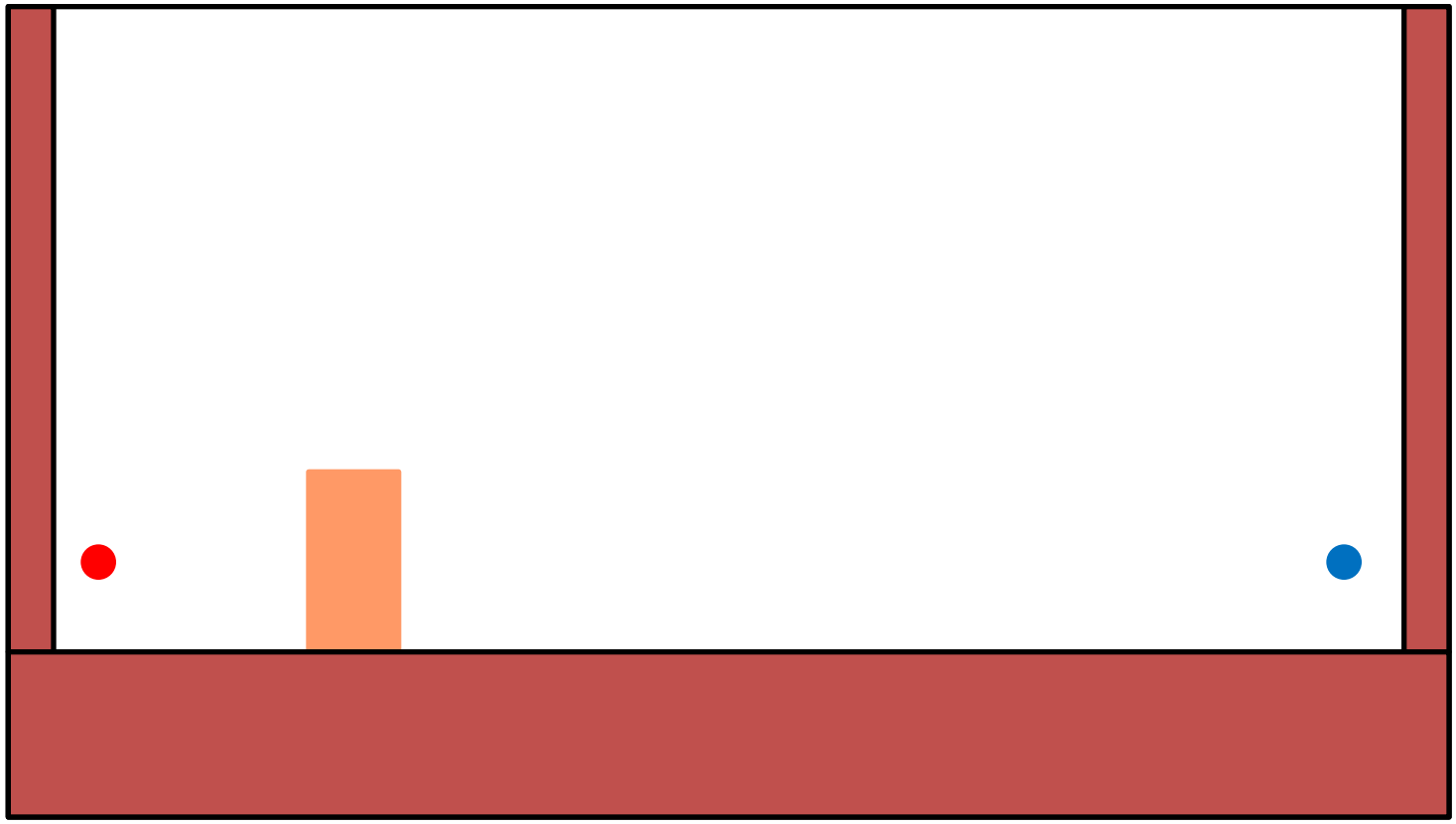
```
gcc -o wii_kadai wii_kadai.c  -lm -lSDL2 -lSDL2_gfx
 -lcwiimote -D_ENABLE_TILT -D_ENABLE_FORCE -L/usr/lib
```

バッチファイルmakeを実行してもコンパイルできる

実行方法

./wii_kadai 識別ID
（例）./wii_kadai 00:22:D7:AA:14:D9

実行画面



様々な初期化　→　whileのところをループ

今は図形描画処理のところをループして実行

簡単な動作を入れてみよう

whileのループの中に次を記述

```
 // 以下に処理を記述していく
if (wiimote.keys.down){
    chara.x += 2;
}

if (wiimote.keys.up){
    chara.x -= 2;
}
```

十字キー（右）

キャラのx座標を+2

十字キー（左）

キャラのx座標を-2

さらに入れてみよう

```
if(wiimote.keys.one){
    chara.w = 40;
}
else{
    chara.w = 20;
}
```

1ボタンが押された

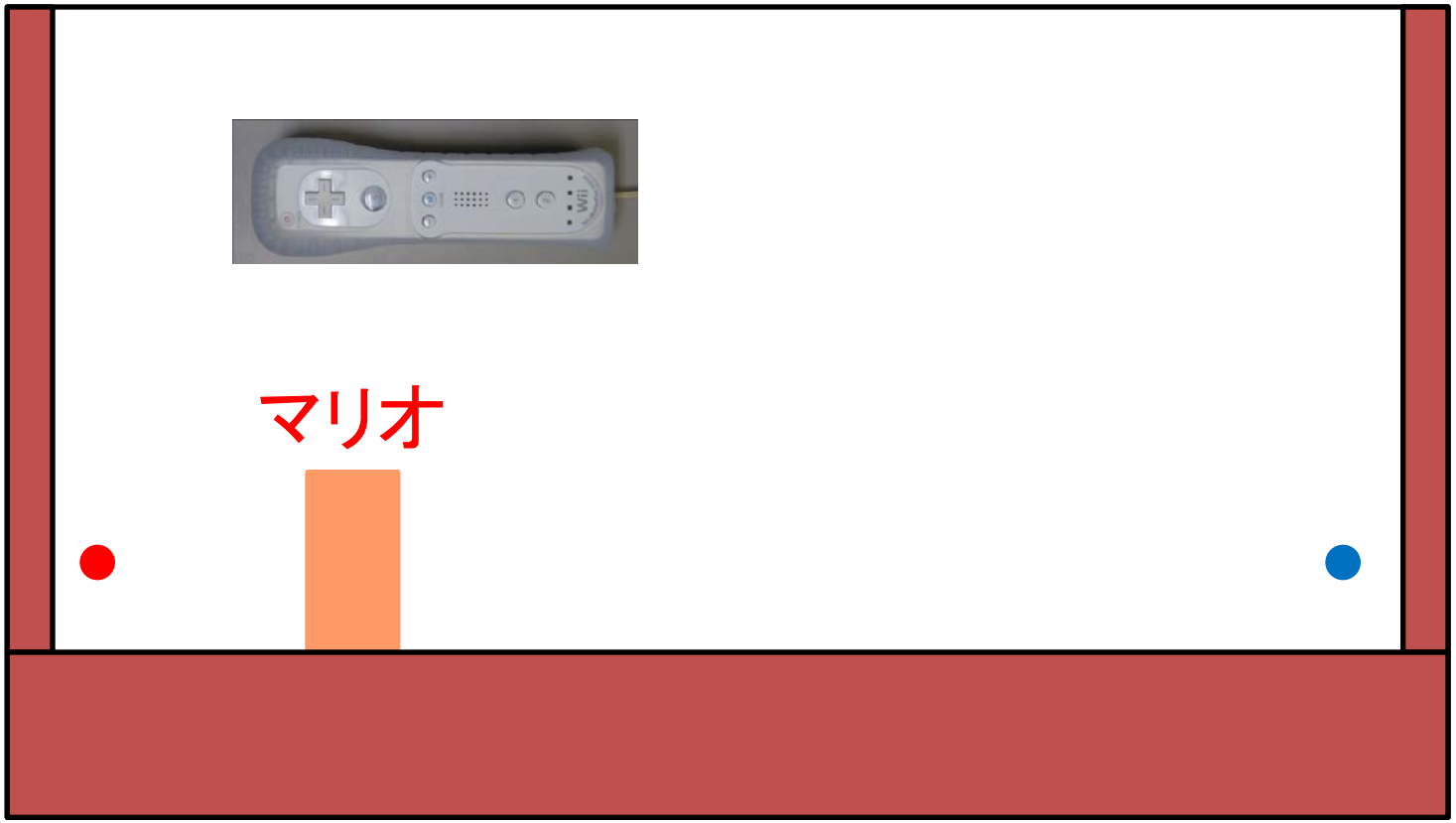
キャラの横幅を40にする

そうでない時

キャラの横幅を20にする

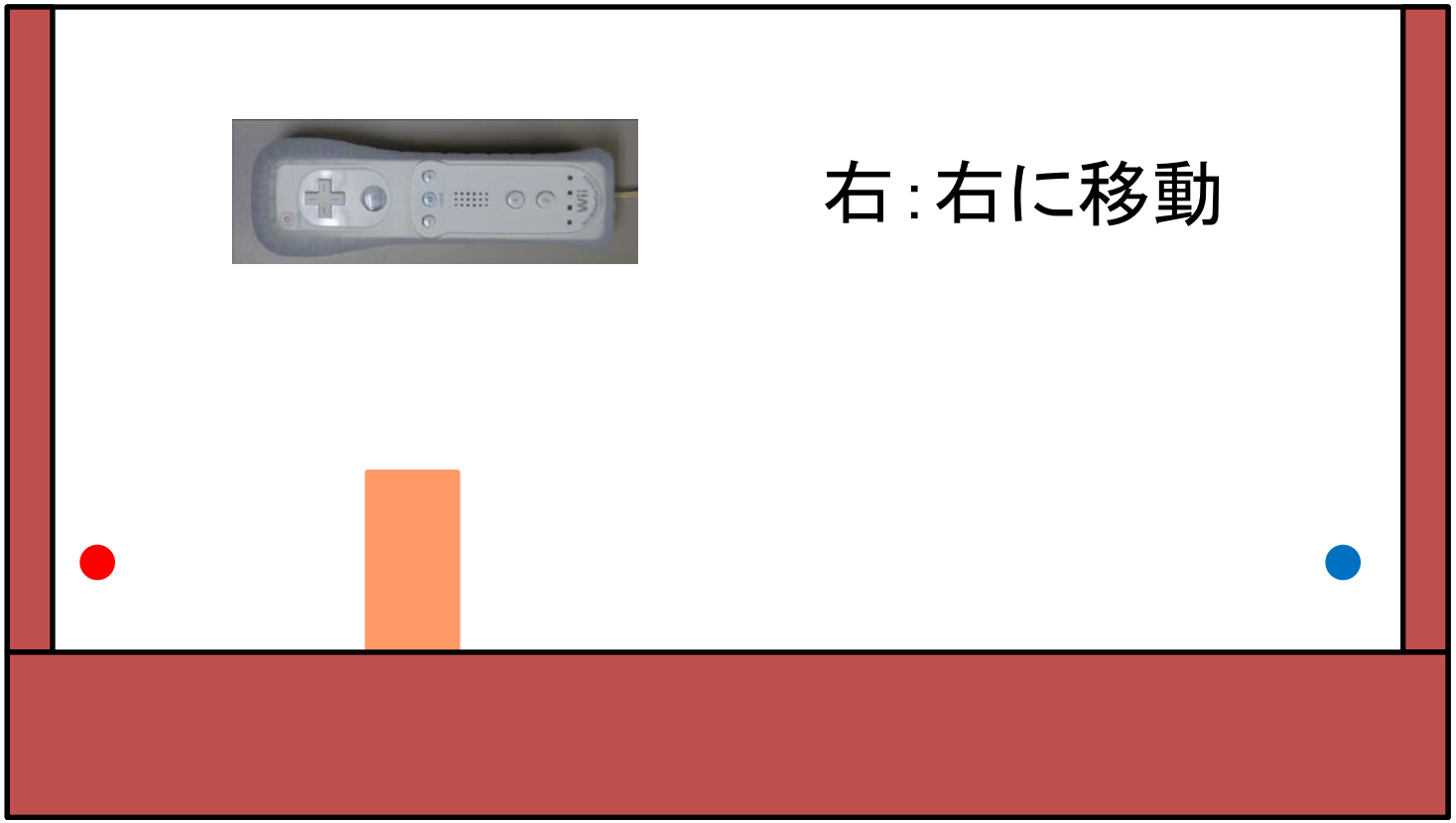
課題

Wiiマリオと同じような動きを作ろう！



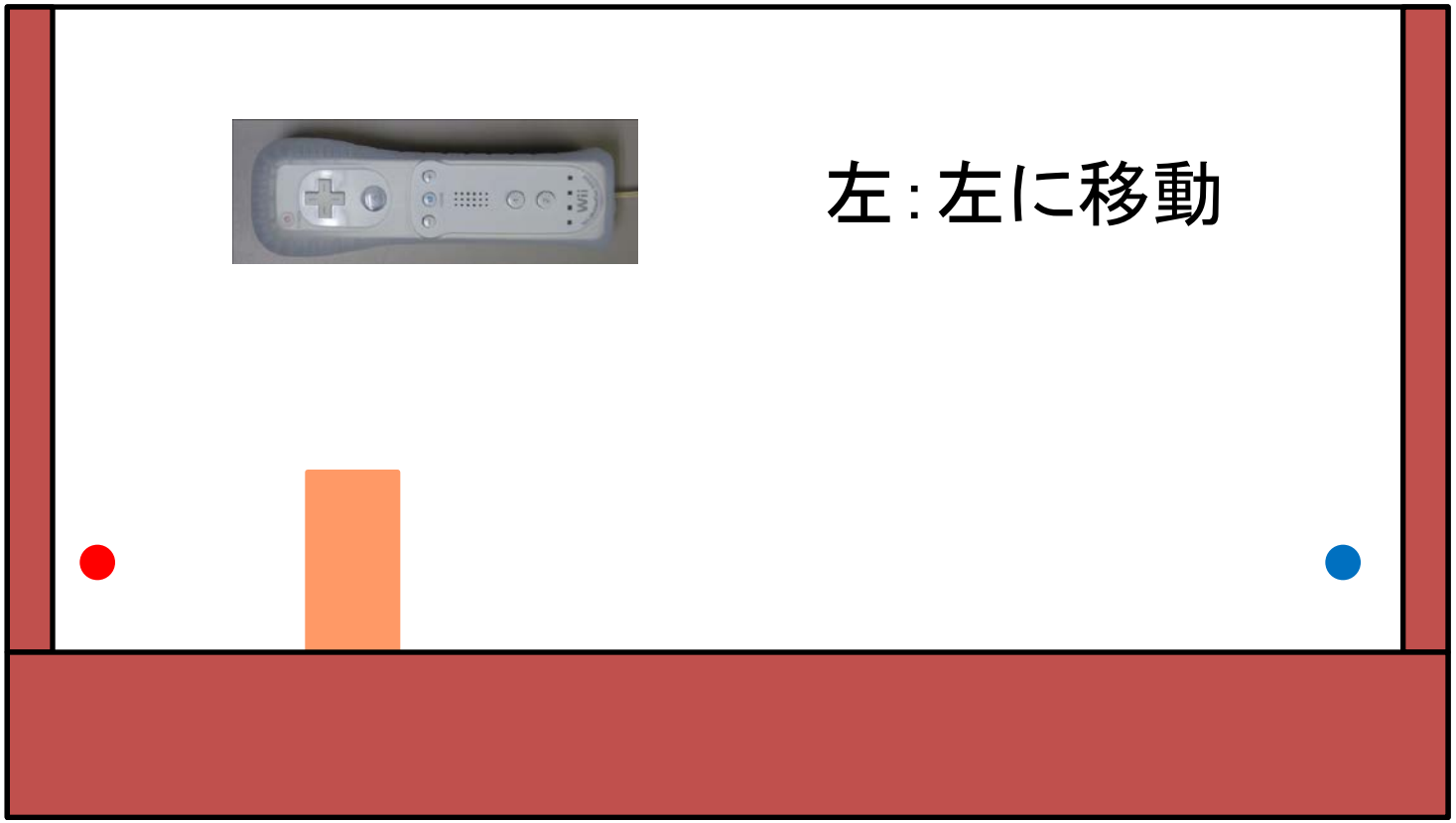
課題

Wiiマリオと同じような動きを作ろう！



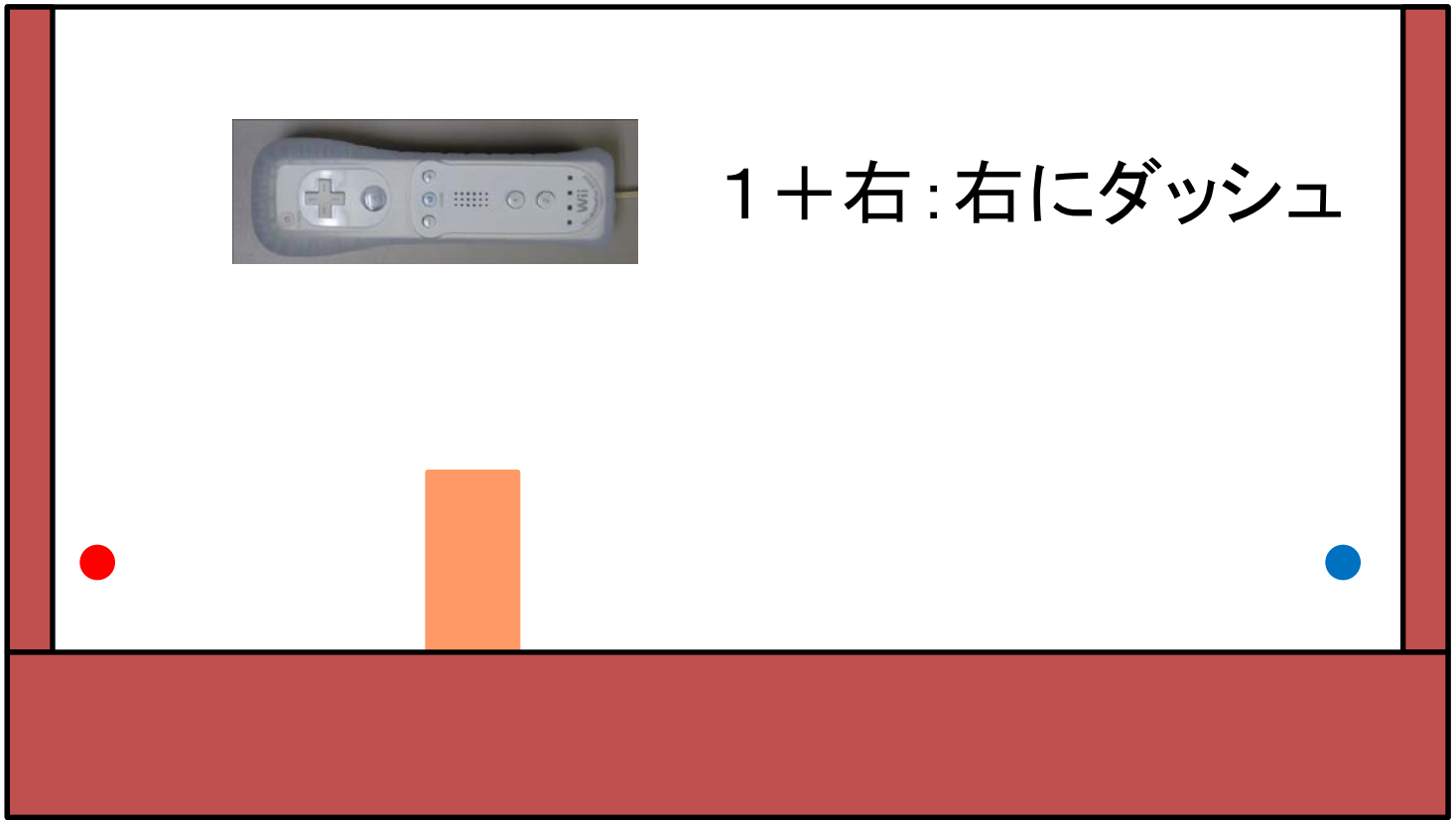
課題

Wiiマリオと同じような動きを作ろう！



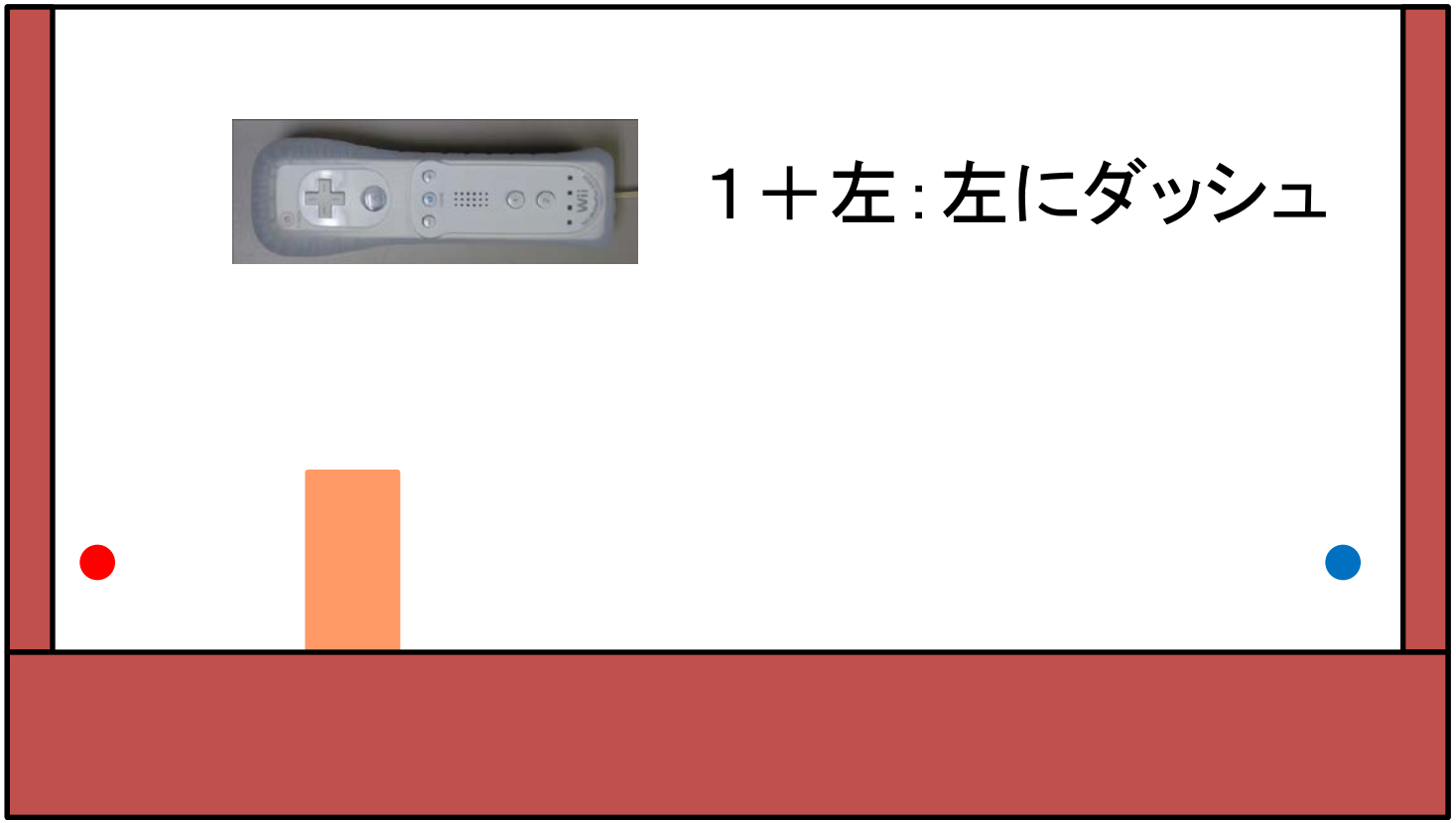
課題

Wiiマリオと同じような動きを作ろう！



課題

Wiiマリオと同じような動きを作ろう！



課題

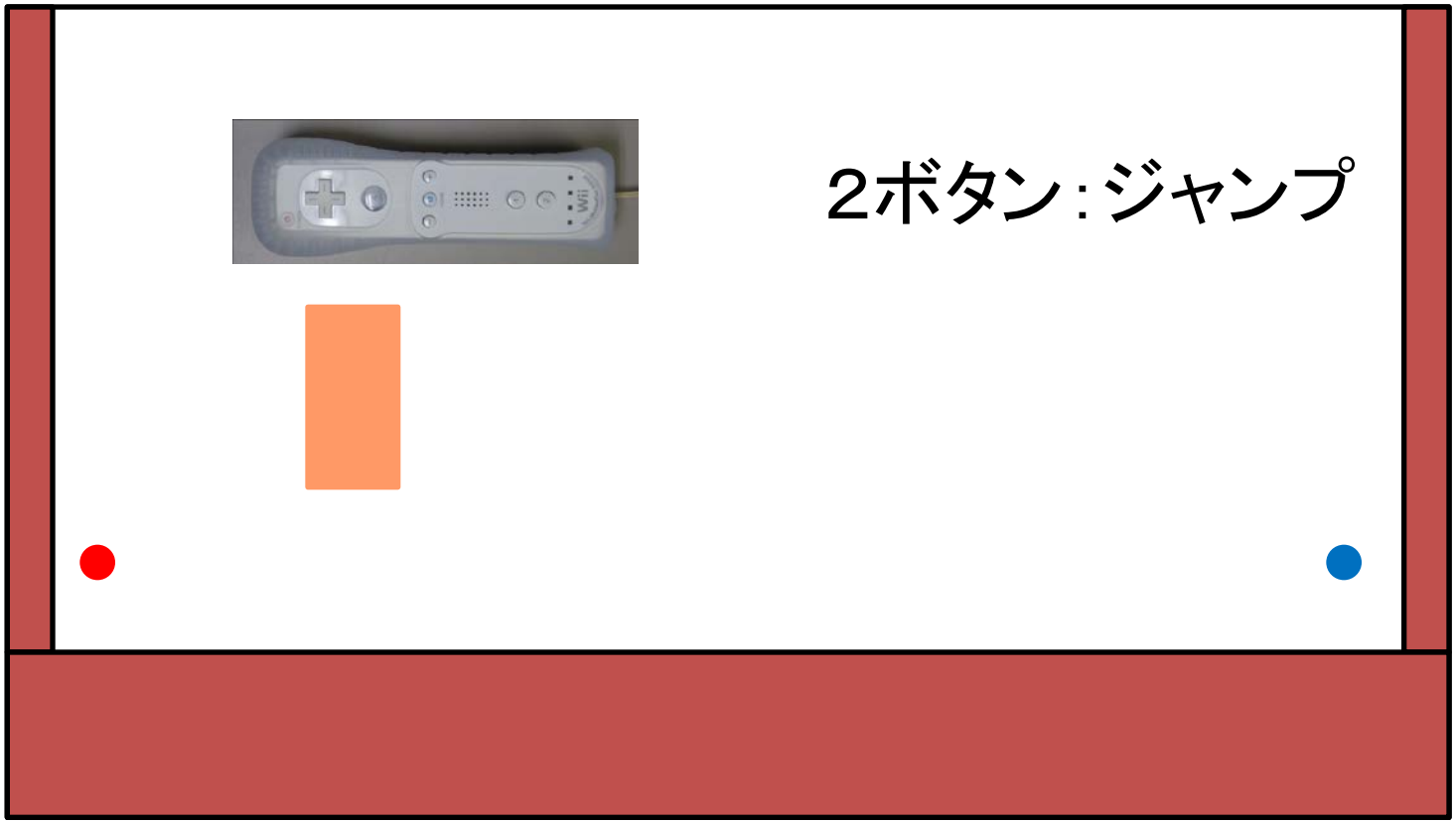
Wiiマリオと同じような動きを作ろう！

下：しゃがむ

縦方向の長さを半分に

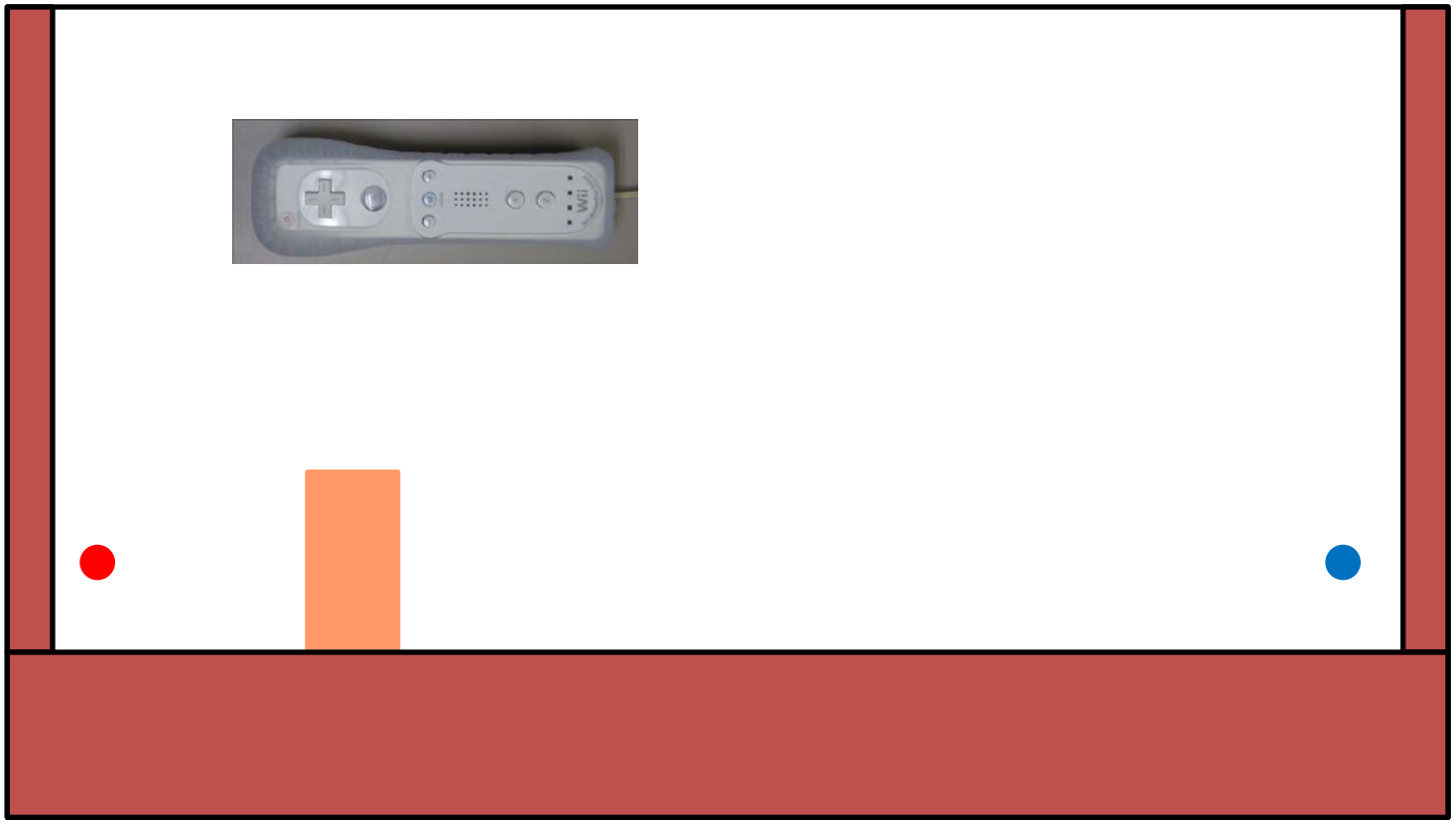
課題

Wiiマリオと同じような動きを作ろう！



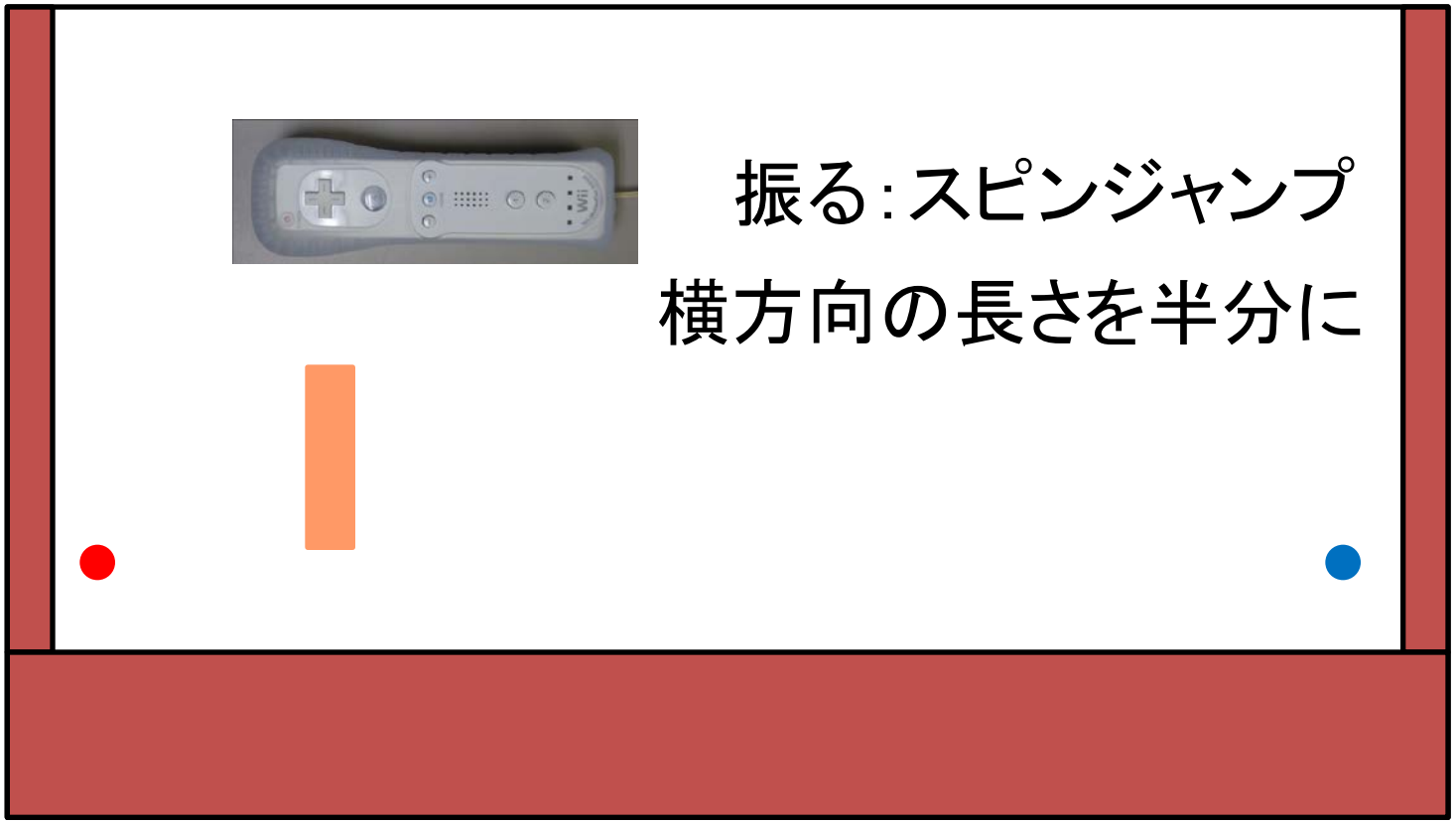
課題

Wiiマリオと同じような動きを作ろう！



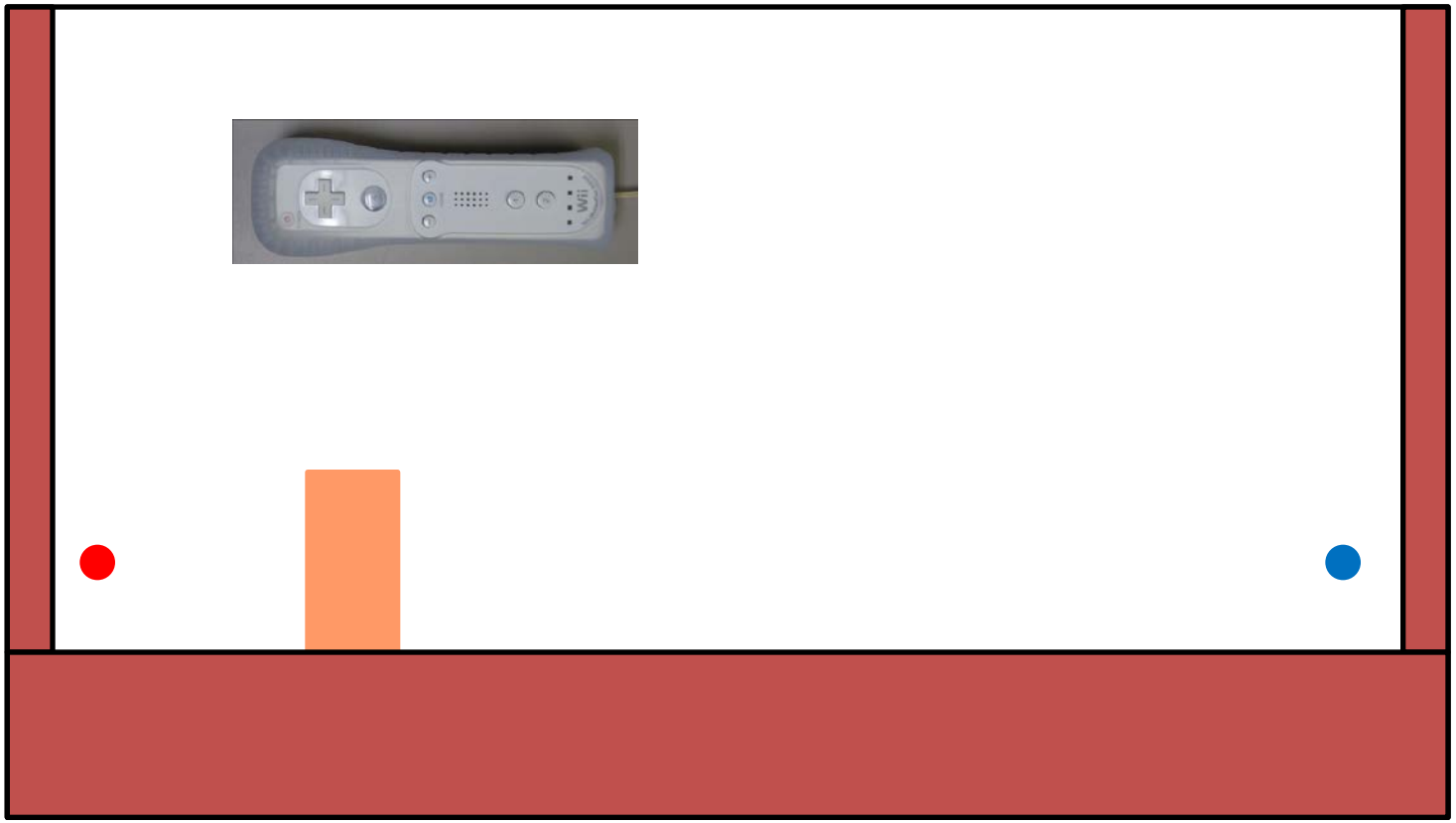
課題

Wiiマリオと同じような動きを作ろう！



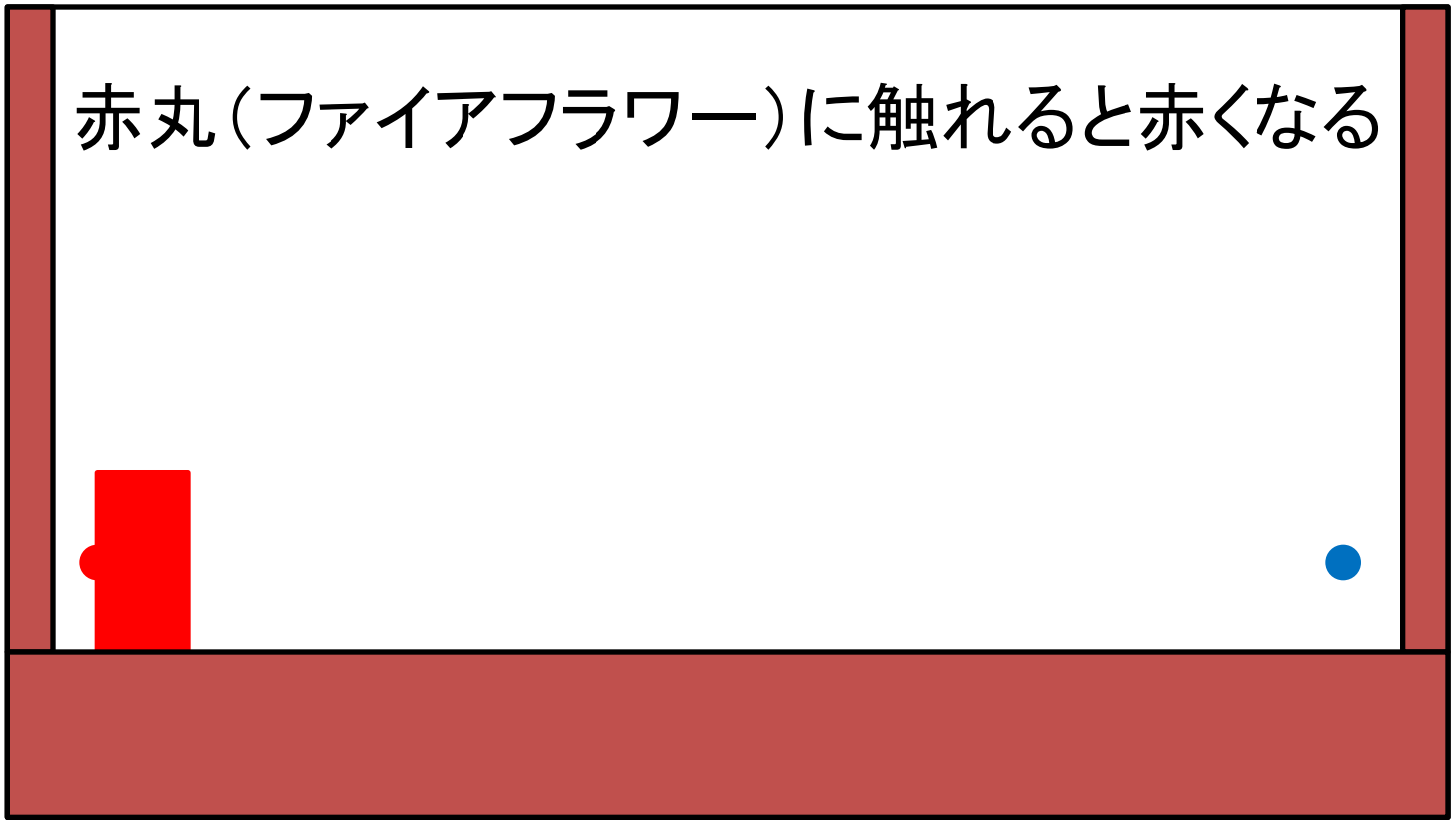
課題

Wiiマリオと同じような動きを作ろう！



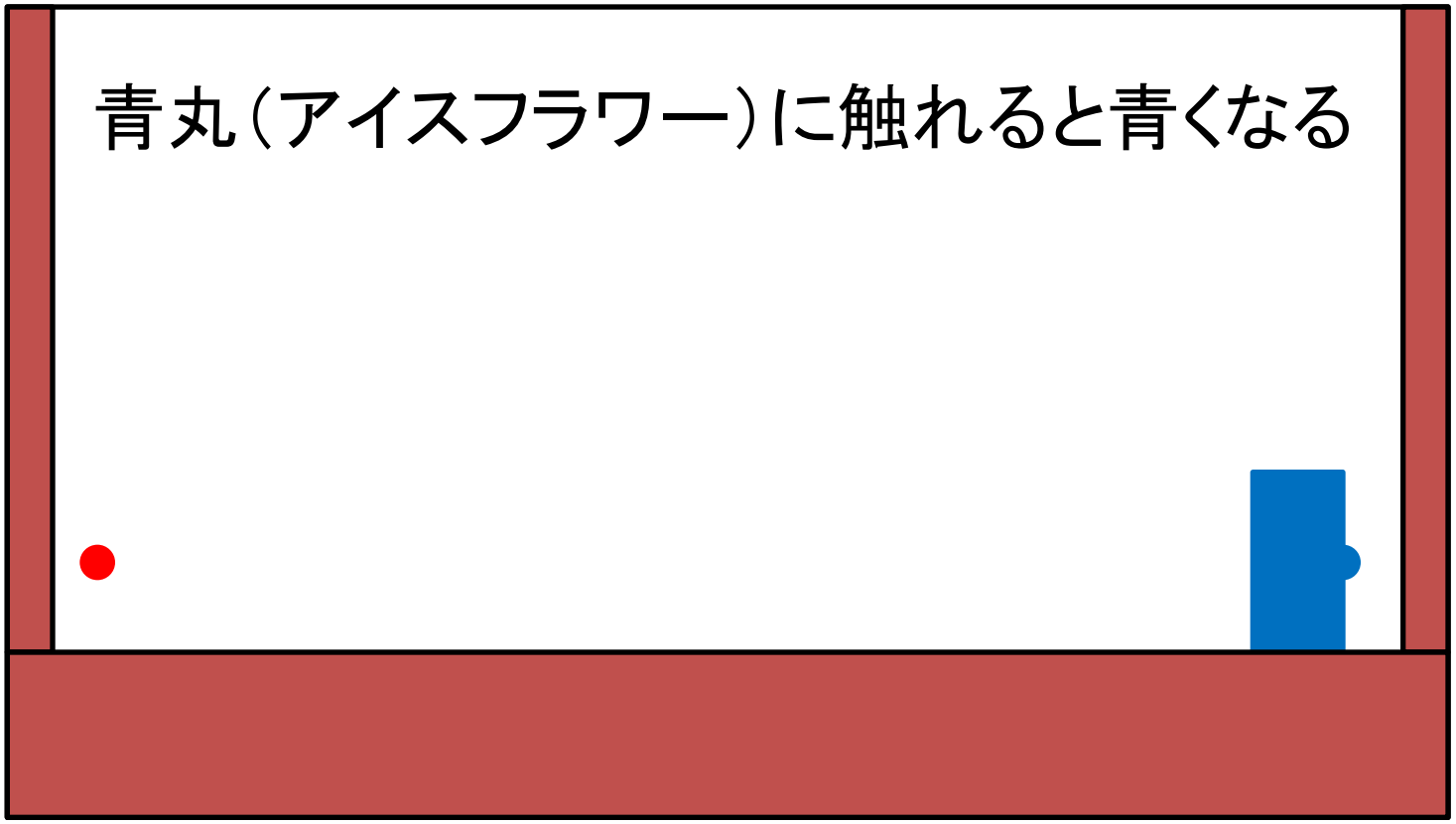
課題

Wiiマリオと同じような動きを作ろう！



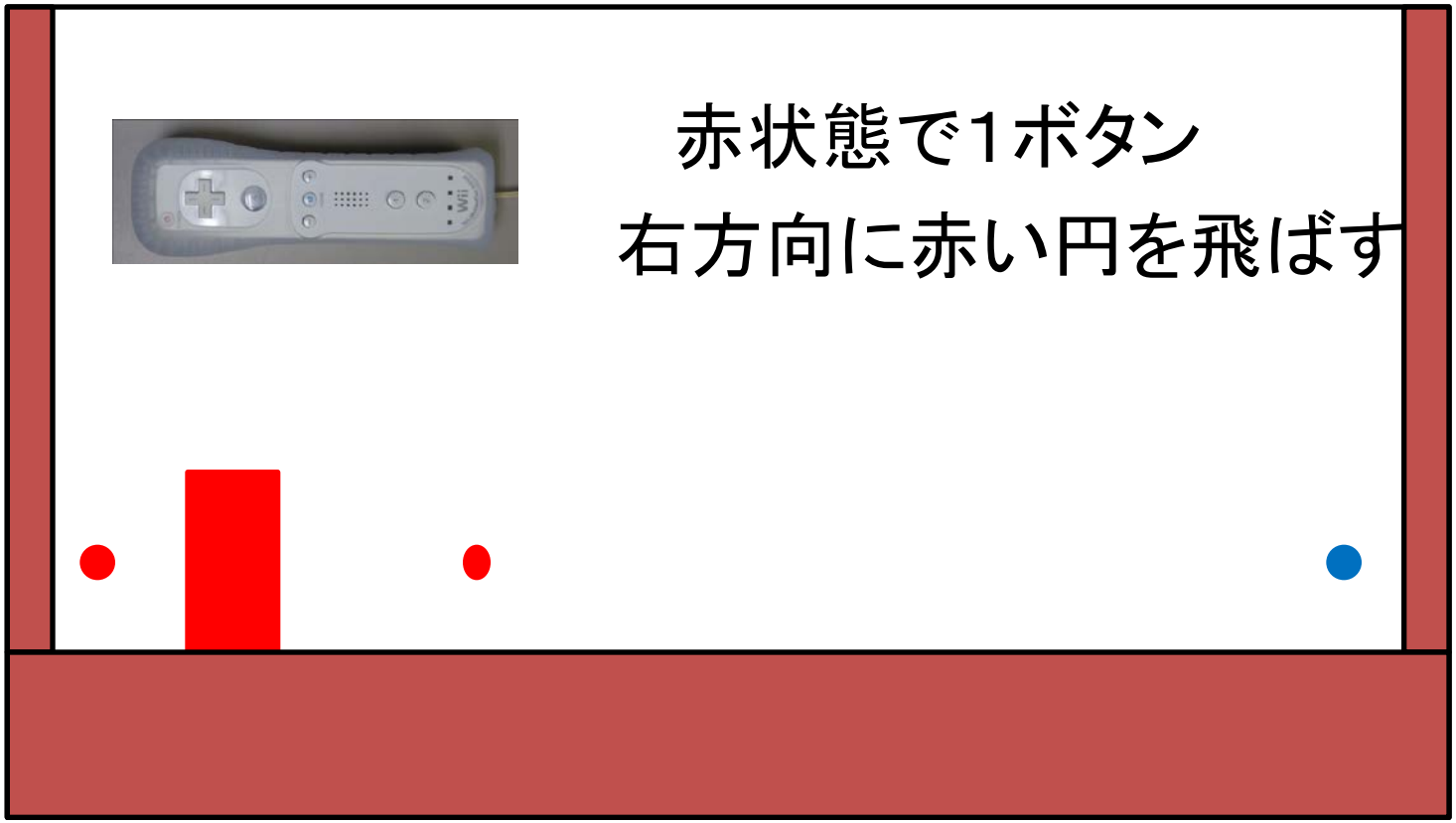
課題

Wiiマリオと同じような動きを作ろう！



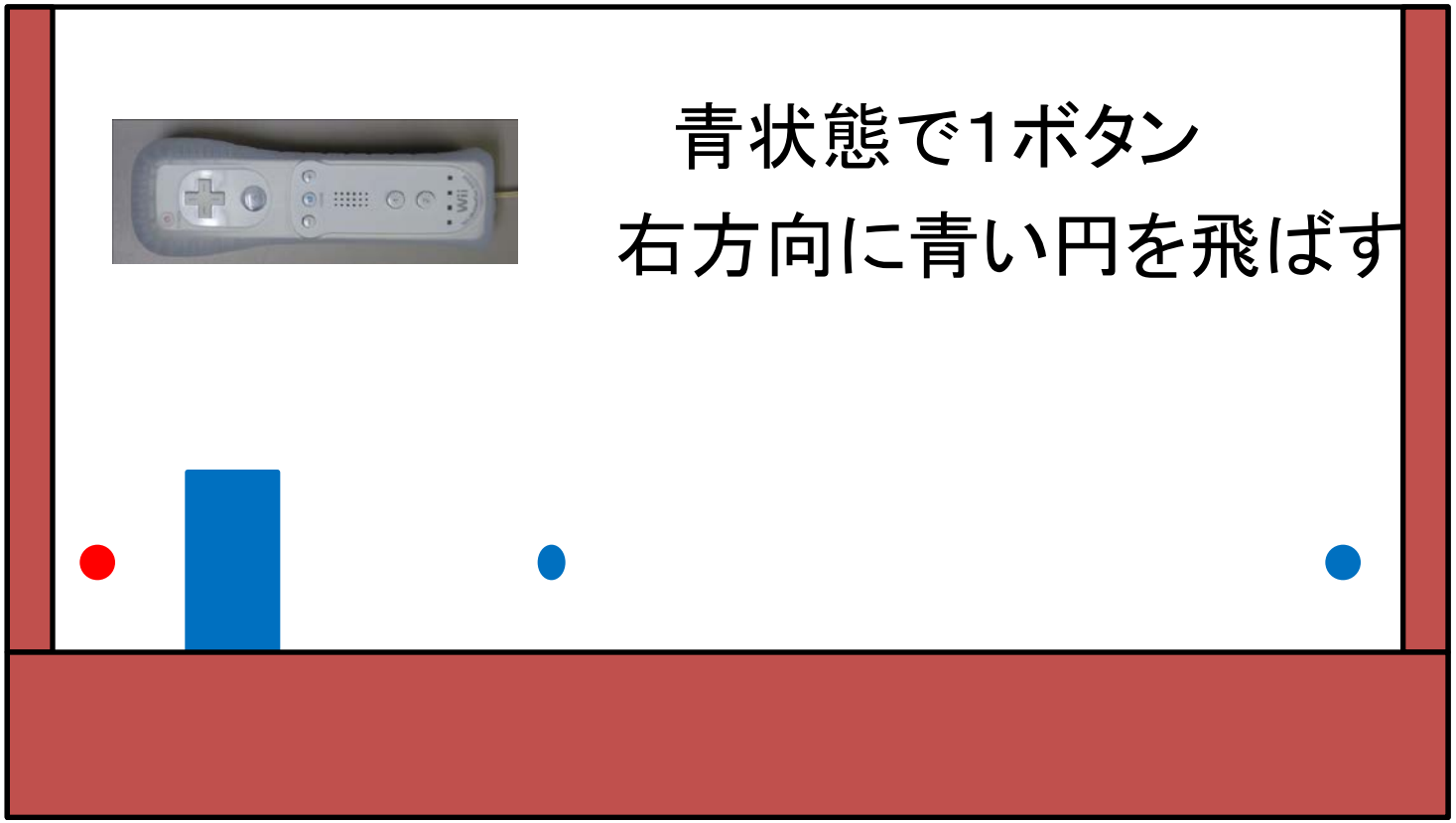
課題

Wiiマリオと同じような動きを作ろう！



課題

Wiiマリオと同じような動きを作ろう！



基本課題

Wiiマリオと同じ動きをSDLと連携して作る

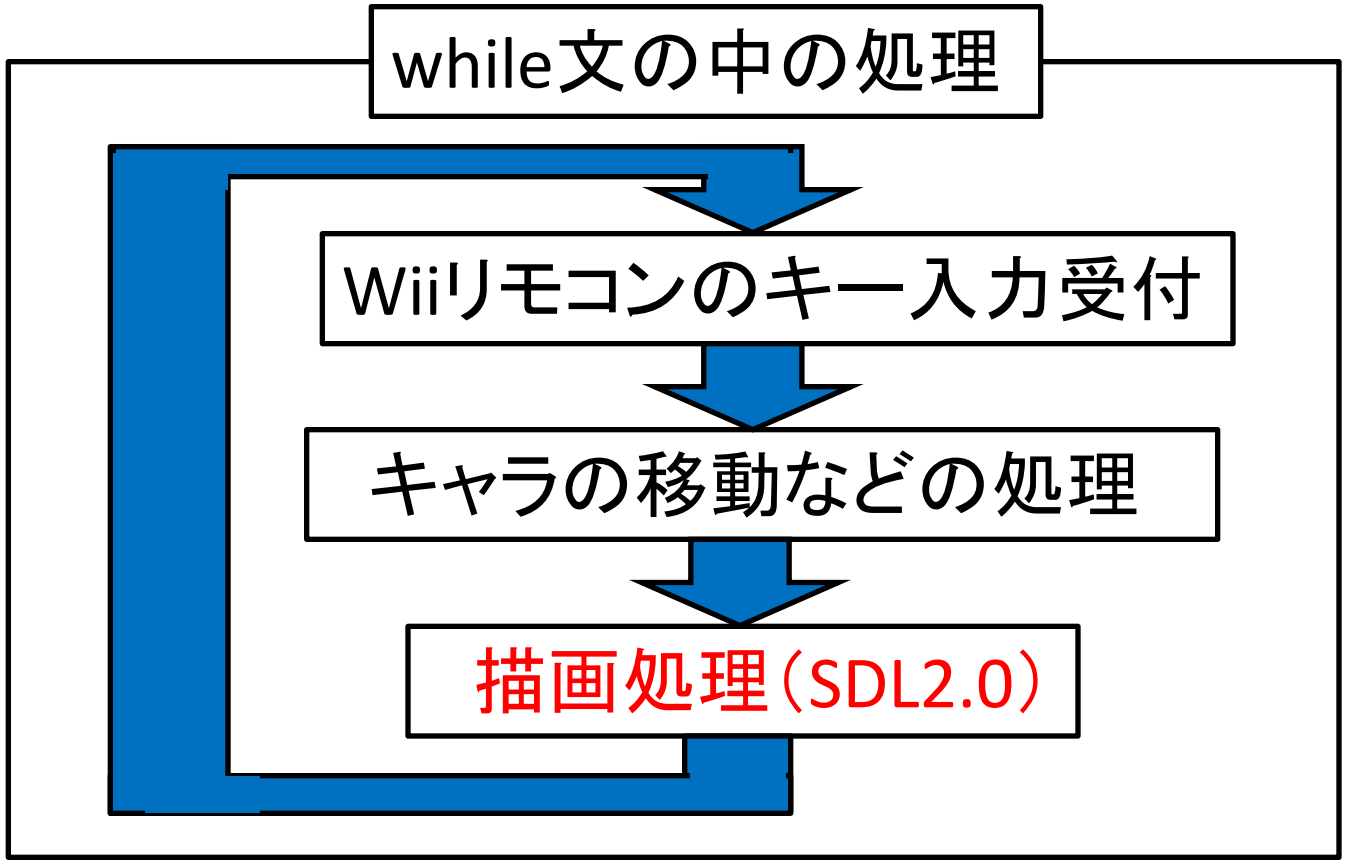
1. 走る（左右を押すと押した方向に走る）
2. ダッシュ（左右＋1ボタンで速く走る）
3. しゃがむ（下を押すとマリオの縦の長さを半分に）
4. ジャンプ（2ボタンを押すと上に移動し、ある高さに到達すると下に降りてくるようにする）
5. スピンジャンプ（振るとマリオの横の長さを半分にし、ジャンプする）
6. フラワー（画面の左右端に赤と青の円を配置）
7. ショット（フラワーに触れるとマリオの色を変え、その状態で1ボタンを押すと丸いショットが右方向に飛ぶようにする）

課題をやる時のポイント

1. 終了する時はhomeボタンを押して終了する。
   そうしないとSDLが残ってしまい、強制終了
   しないといけなくなる。

2. 基本課題の全ての機能を一度に入れようと
   するのではなく、一つの機能を入れると
   プログラムを実行して、望み通りの動作に
   なっているか確認していく。

現在のプログラムの問題点



描画処理が重いと全体的に処理が遅くなる！

解決方法は次回のマルチスレッドで！

自由課題と提出物

自分のアイディアで動作を作成する。

（例） １．走り出しと止まる時のなめらかさ
２．放物線上にジャンプ
３．炎、氷のショットを向いてる方向に飛ばす
４．ショットが地面で跳ねるようにする

提出物：

１．作成したプログラムの流れを説明し、工夫した点、アピールしたい点をまとめる。自由課題について特に詳しく説明すること。（ファイルはpdf形式で提出すること）
２．コメントを入れたプログラムソース

１．と２．を.tar.gzに圧縮して提出