ソフトウェア設計及び実験
第7回レポート

6119019056 山口力也

2019/06/04 日提出

1　プログラム概要

以下に作成したプログラムの流れを説明する.

1.1　走る動作

まず走る動作については以下のソースコード 1 に示す.

ソースコード 1: 走る動作

```
1 if(wiimote.keys.down){ //右ボタン
2 chara.x += 2;
3 }
4 if(wiimote.keys.up){ //左ボタン
5 chara.x -= 2;
6 }
```

キャラの位置情報を更新して走る動作を実現している.

1.2　ダッシュ動作

次に, ダッシュ動作について以下ソースコード 2 に示す.

ソースコード 2: ダッシュ動作

```
1 if(wiimote.keys.one && wiimote.keys.down){ //1ボタン+右ボタン
2 chara.x += 3;
3 }
4 if(wiimote.keys.one && wiimote.keys.up){ //1ボタン+左ボタン
5 chara.x -= 3;
6 }
```

ここでは if 文 1 ボタンと左右のボタンがどちらも押された時という条件を論理和で実現している.

1.3 しゃがみ動作

しゃがみ動作について以下ソースコード 3 に示す.

ソースコード 3: しゃがみ動作

```
1 if(wiimote.keys.left !=0 && flag_jump == false){ //下ボタン
2 chara.y = 370;
3 chara.h = 20; //しゃがむ
4 }
5 else if (wiimote.keys.left == 0 && flag_jump == false){
6 chara.y = 350;
7 chara.h = 40;
8 }
```

後述するジャンプ動作中にしゃがみ動作が機能しないように論理和で条件付けを行っている. 基本的には下ボタンが押されるとキャラの位置情報の y を変えることで実現している. ここで,SDL のウィンドウの y 軸は上方向が 0 で, キャラの位置情報が左上端からはじまっているので, しゃがむ場合はキャラの位置情報の始点の y の値を増やす必要がある.

1.4 ジャンプ動作

ジャンプ動作について以下ソースコード 4 に示す.

ソースコード 4: ジャンプ動作

```
1 if(flag_jump == true){ //flag がたっているなら
2         chara_temp.y = chara.y; //現在の値を一旦格納
3         chara.y += (chara.y - chara_prev.y)+1; //現在の値 -
              過去の値 + 1(上昇中は負の値,下降中は正の値)
4         chara_prev.y = chara_temp.y; //格納していた
              y の値を前回の値として格納
5         //デバッグ用
6         printf("chara-prev = %d\n",chara.y-chara_prev.y);
7         printf("chara.y= %d\n",chara.y);
8         if(chara.y == 350){ //元の地面についたら
9                 flag_jump = false; //終了.flag 初期化
10        }
11 }
```

```
12 if(wiimote.keys.two != 0 && flag_jump == false){ //2ボタンが
       押された時かつflag が立っていない時
13         flag_jump = true; //flag をたてる
14         chara_prev.y = chara.y; //現在の
               y の値を過去の値として格納
15         chara.y -=20; //最初に引く値
16 }
```

ここで,flag_jump は bool 型の大域変数で初期値は false で,chara_prev と chara_temp は chara 構造体と全く同じ構造体である.flag が true になると y の値に現在の y の値と前回の y の値の差に+1 した値を足していく. この値は上昇中は負の値になり, 下降中は正の値になる. これによりジャンプの頂点に近くなればなるほど値が 0 に近くなることで重力らしさを実現している.

1.5 スピンジャンプ

スピンジャンプについて以下ソースコード 5 に示す.

ソースコード 5: スピンジャンプ

```
1  if ( abs(126 - wiimote.axis.x) > 10 && abs(130 - wiimote.
       axis.y) > 10 && abs(153 - wiimote.axis.z) > 50 &&
       flag_spin == false ){ //事前に
       printf した値との差の絶対値があるしきい値を超えたら
2          printf("SPIN JUMP\n"); //デバッグ用
3          flag_spin = true; //フラグを立てる
4          chara_prev.y = chara.y; //現在の
               y の値を過去の値として格納
5          chara.y -=20; //最初に引く値
6  }
7  if ( flag_spin == true ){ //flag が立っているなら
8          chara.w = 10; //キャラの横幅を半分に
9          chara_temp.y = chara.y; //現在の値を一旦格納
10         chara.y += (chara.y - chara_prev.y)+1; //現在の値 -
               過去の値 + 1
11         chara_prev.y = chara_temp.y; //格納していた現在の値を
               前回の値として格納
12         //デバッグ用
13         printf("chara-prev = %d\n",chara.y-chara_prev.y);
14         printf("chara.y= %d\n",chara.y);
15         if(chara.y == 350){ //地面についたら
16                 flag_spin = false;
17                 chara.w = 20; //キャラの横幅を戻す
18         }
19 }
```

基本的な動作はジャンプ動作と同じだが, 異なる部分はその条件式である. 事前に printf しておいた値からある程度の予想をつけて論理和で振る動作をしたときの加速度センサの値を見て, 振っているかそうでないか判断した.

1.6 フラワー

フラワーについて以下ソースコード 6 に示す.

ソースコード 6: フラワー

```
1  if(0 <= chara.x && chara.x <= 30 && 325 <= chara.y && chara
       .y <= 430 ){
2          flag_red = true;
3          flag_blue = false;
4  }
5  /*キャラ横幅 20,青丸の横幅 5で青丸の
       x 座標が 485 なので x 座標は 460<=x<=510で当たり判定*/
6  /*キャラの縦幅 50,青丸の縦幅 5で青丸の
       y 座標が 380 なので y 座標は 325<=y<=430で当たり判定*/
7  if(460 <= chara.x && chara.x <= 510 && 325 <= chara.y &&
       chara.y <= 430 ){
8          flag_red = false;
9          flag_blue = true;
10 }
11 /**************中略****************/
12 /************描画部分**************/
13 filledCircleColor(renderer, 25, 380, 5, 0xff0000ff); // 左に
       赤丸アイテム
14 filledCircleColor(renderer, 485, 380, 5, 0xffff0000); // 右
       に青丸アイテム
15 if(flag_blue == true){ //青丸に触れた時
16         SDL_SetRenderDrawColor(renderer, 0, 0, 255, 0);
17         SDL_RenderFillRect(renderer, &chara); // キャラの描画
18 }
19 else if(flag_red == true){ //赤丸に触れた時
20         SDL_SetRenderDrawColor(renderer, 255, 0, 0, 0);
21         SDL_RenderFillRect(renderer, &chara); // キャラの描画
22 }
23 else if(flag_blue == false && flag_red == false){
24         SDL_SetRenderDrawColor(renderer, 255, 150, 150, 0);
25         SDL_RenderFillRect(renderer, &chara); // キャラの描画
26 }
27 SDL_RenderPresent(renderer); // 描画データを表示
```

処理の部分で, 当たり判定を行う.flag_red と flag_blue は bool 型の大域変数で初期値は false である. 描画部分では, キャラの色を変えるかどうかを処理部分の flag を見て決めている.

1.7　ショット動作

次のショット動作については時間が足りずに完成しなかった.

2　自由課題

自由課題としてはジャンプ動作の部分で重力を実装したことと, 動作には関係ないが,BGM を入れて見たことである.