14班企画書

細野虎太郎　小嶋樹　早道広峻　谷口隼輔

令和2年 10月 18日 提出

1　概略

このゲームは最初, プレイヤーが同じ場所から同時に一斉にスタートし, 同じゴールを目指す. ゴールまでの道のりは必ずしも平たんとは限らない. 例えば谷が存在した場合には橋をかけたり, 丘が存在した場合, プロックを使って乗り越えるか, 迂回するかを自分で決め, プレイヤーはその道を進んでゴールを目指す. 表 1 にゲームの仕様とルールを示す.

表 1: ゲームのおおまかな仕様とルール

ゲームジャンル	レースアクションゲーム
プレイヤー人数	2 〜 5 人
ゲームに対するルール	プレイヤー全員がゴールした時点で終了する

2　コンセプト

このゲームのコンセプトは『自分の道は自分で決めろ！』である. 指先だけでなく体重移動まで使った特徴的な操作や, 自由なブロックの設置を通じて, プレイヤーの数だけ正しいルートがある全く新しいレースゲームである.

3　ゲーム画面

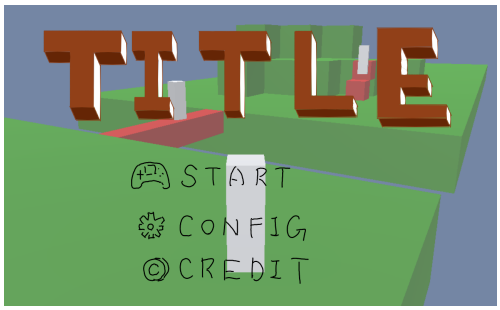


図 1: タイトル画面

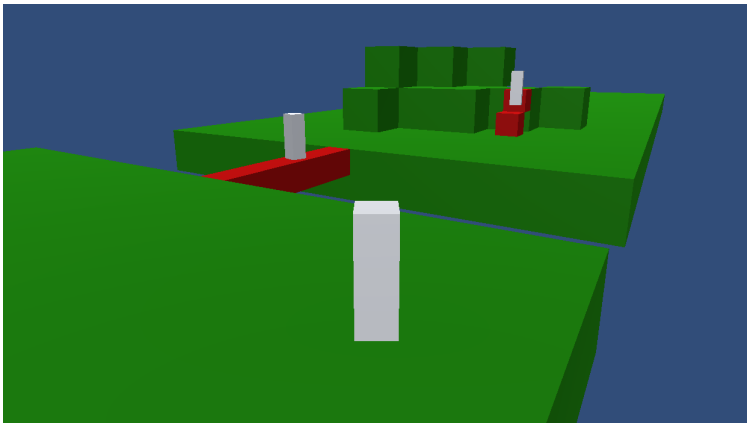


図 2: ゲーム画面

このイメージ画像は Unity を用いて作成したものである.
画面中央の白い直方体が自分の操作するキャラクターを示しており，緑色のブロックが初期からあるブロックである.
また，赤色のブロックがプレイヤーが配置したブロックである.

カメラは常に自身の操作するキャラクターの後方に位置しており, プレイヤーの回転に合わせてカメラ自身もキャラクターの進行方向を向く.
このイメージ画像では画面奥にゴールがあるという想定であるため，自分以外のプレイヤもゴールに向かって進んでいる.

4 操作方法

ゲームの操作は以下の入力機器のいずれかを用いる.

- キーボード
- wii リモコン
- バランス wii ボード&wii リモコン

操作の種類は以下の通りである.

表 2: 操作方法 1

操作	キーボード	wii リモコン
加速	W	2
左右旋回	AD	<->
ジャンプ	Space	振る or 1
ブロックの設置	Enter	A
ゲーム終了	Escape	+

表 3: 操作方法 2

操作	バランス wii ボード&wii リモコン
加速	2
左右旋回	体重移動
ジャンプ	屈伸 or 振る or 1
ブロックの設置	A
ゲーム終了	+

5 世界観の設定

西暦 20xx 年．Hayamich 社と Taniguch 社によって共同開発された「ホバリングボード (通称 セグウェイ)」と「ブロック生成デバイス」を用いた新たな娯楽がひそかに人気を集め始めていた．(背景：食料革命によって引き起こされた人口爆発は世界を手狭にさせた．住む場所が足りなくなった人類は空中に地面を生成することに成功し，無限ともいえる居住区を手に入れた．Taniguch 社はこの技術を一般市民でも購入できる安価な機器として再開発した．さらに, Hayamich 社はセグウェイを開発し空中の居住区間移動を従来からは考えられないほど劇的に容易なものとした．スポーツマンをはじめとした人々はこのボード (セグウェイ) とデバイスを用いて観客を楽しませている．)

6 ゲームデータの構造

以下にゲームデータの構造を示す

表 4: クライアントのデータ構造

データ型	変数名	内容
char	`name[MAX_LEN_NAME]`	クライアントの名前
FloatCube	pos	マップ上の場所
int	rank	順位
bool	goal	ゴールしているか

表 5: ネットワークモジュール用のクライアントの情報

データ型	変数名	内容
int	connect	クライアントがサーバーに接続しているか
int	sock	使用するソケット
struct sockaddr_in	addr	ソケットの設定

表 6: クライアントの座標

データ型	変数名	内容
float	x	x 座標
float	y	y 座標
float	z	z 座標

表 7: 直方体形を定義する構造体

データ型	変数名	内容
float	x	x 座標
float	y	y 座標
float	z	z 座標
float	w	x 方向の長さ
float	h	y 方向の長さ
float	d	z 方向の長さ

表 8: マップ用のデータ

データ型	変数名	内容
int	`_TerrainData [MAP_SIZE_W][MAP_SIZE_H][MAP_SIZE_D]`	マップデータ
`vector<PlaceData>`	_ObjectDatas	オブジェクトデータ
int	_MapW	マップの横幅
int	_MapH	マップの横縦
int	_MapD	マップのサイズ

7 モジュール

7.1 サーバー

以下に各モジュールの外部関数を説明する

1. ネットワークモジュール

関数名	void SetupServer(int numCl, u_short port)
機能	サーバーの初期設定を行う
引数	numCl —— クライアント数
	port —— ポート番号
返り値	無し

関数名	int ControlRequests()
機能	クライアントからのリクエストに対応する
引数	無し
返り値	1:通信継続/0:通信終了

関数名	void RunCommand(int id, char com)
機能	コマンドの実行
引数	id —— 送信先のクライアント ID
	com —— 送信するコマンド
返り値	無し

関数名	void TerminateServer()
機能	サーバーの終了処理を行う
引数	無し
返り値	無し

2. マップモジュール

関数名	void LoadMapData(char* fileName)
機能	マップデータの読み込みを行う
引数	fileName —— マップデータファイル名
返り値	無し

7.2 クライアント

以下に各モジュールの外部関数を説明する

1. ネットワークモジュール

関数名	void SetupClient(char *serverName, u_short port)
機能	クライアントの初期設定を行う
引数	serverName—— サーバー名
	port —— ポート番号
返り値	無し

関数名	int ControlRequests()
機能	クライアントの制御を行う
引数	無し
返り値	1:通信継続/0:通信終了

関数名	int InCommand(char com)
機能	コマンドに対する処理を行う
引数	com —— コマンド
返り値	1:通信継続/0:通信終了

関数名	void TerminateClient()
機能	クライアントの終了処理を行う
引数	無し
返り値	無し

2. システムモジュール

関数名	const PlayerData* GetPlayerData()
機能	プレイヤーデータ配列の先頭ポインタを返す
引数	無し
返り値	プレイヤーデータ配列の先頭ポインタ

3. マップモジュール

関数名	SetMapData(int mapW,int mapH, int mapD, int terrainData[MAP_SIZE_W][MAP_SIZE_H][MAP_SIZE_D])
機能	マップデータの初期設定を行う
引数	mapW —— マップの幅
	mapH —— マップの高さ
	mapD —— マップの奥行
	terrainData —— マップの地形データ
返り値	無し

4. 入力モジュール

関数名	virtual void GetInput(SDL_Event event)
機能	ユーザー入力を受け取る
引数	event —— SDL のイベント
返り値	無し

関数名	InputType GetInputType()
機能	現在の入力情報を返す
引数	無し
返り値	現在の入力情報

5. グラフィックモジュール

関数名	void InitGraphic()
機能	グラフィックモジュールの初期設定を行う
引数	無し
返り値	無し

関数名	void Disp()
機能	グラフィックの表示を行う
引数	無し
返り値	無し

8 コマンドプロトコル

1. サーバー→クライアントのプロトコル

コマンド	M(PlayerData[])
機能	移動処理
引数	全プレイヤーデータ
対象	全クライアント
タイミング	毎フレーム

コマンド	P(PlaceData)
機能	ブロックの設置
引数	設置されたブロックのデータ
対象	全クライアント
タイミング	設置時

コマンド	D
機能	設置不可能
引数	無し
対象	クライアント
タイミング	設置時

コマンド	G
機能	ゴールしたことを通知
引数	なし
対象	クライアント
タイミング	ゴール時

コマンド	E(char[])
機能	エラーを通知
引数	エラー内容
対象	クライアント
タイミング	エラー発生時

コマンド	F(PlayerData[])
機能	ゲームの終了を通知
引数	全プレイヤーのデータ
対象	全クライアント
タイミング	全プレイヤーゴール時

コマンド	Q
機能	プログラムの強制終了
引数	無し
対象	全クライアント
タイミング	任意

2. クライアント→サーバーのプロトコル

コマンド	M(FloatPosition)
機能	移動処理
引数	自身の移動後の座標
対象	サーバー
タイミング	毎フレーム

コマンド	P(PlaceData)
機能	ブロックの設置
引数	設置するブロックのデータ
対象	サーバー
タイミング	設置時

コマンド	Q
機能	プログラムの強制終了
引数	無し
対象	サーバー
タイミング	任意

9　スケジュール

図 3: スケジュール

No	親タスク	子タスク	担当	開始日	終了日	所要日数
1	クライアント	初期化処理	小嶋	10月1日	10月25日	24
2	クライアント	入力切替	小嶋	10月25日	11月7日	13
3	クライアント	画面変更	小嶋	11月7日	11月17日	10
4	サーバー	マップの初期化処理	細野	10月1日	10月22日	21
5	サーバー	設置情報の処理	細野	10月22日	10月31日	9
6	サーバー	キャラ移動の処理	細野	10月31日	11月11日	11
7	サーバー	キャラ-マップの当たり判定	細野	11月11日	11月23日	12
8	サーバー	キャラ-設置物の当たり判定	細野	11月23日	11月30日	7
9	サーバー	キャラ-キャラの当たり判定	細野	12月1日	12月10日	9
10	ネットワーク	サーバー側構築	早道	10月1日	10月15日	14
11	ネットワーク	クライアント側構築	早道	10月15日	10月22日	7
12	ネットワーク	コマンドの対応	早道	10月22日	10月31日	9
13	入力	キーボード入力	早道	10月1日	10月31日	30
14	入力	Wii入力	早道	11月1日	11月14日	13
15	入力	バランスボード入力	早道	11月14日	12月15日	31
16	グラフィック	マップの表示	谷口	10月1日	10月19日	18
17	グラフィック	プレイヤーの表示	谷口	10月19日	10月31日	12
18	グラフィック	プレイヤーの回転	谷口	11月1日	11月12日	11
19	グラフィック	カメラの位置、回転	谷口	11月12日	11月21日	9
20	グラフィック	オブジェクトの表示	谷口	11月21日	11月30日	9
21	グラフィック	テクスチャの貼り付け	谷口	12月1日	12月9日	8
22	音楽	BGM作成	小嶋	11月17日	11月26日	9
23	音楽	SE作成	小嶋	11月26日	12月5日	9
24	UI作成	時間	小嶋	12月5日	12月12日	7
25	UI作成	順位	小嶋	12月12日	12月19日	7
26	クライアント	タイトル画面	細野	12月10日	12月19日	9
27	クライアント	リザルト作成	早道	12月15日	12月24日	9

図 4: ガントチャート 1

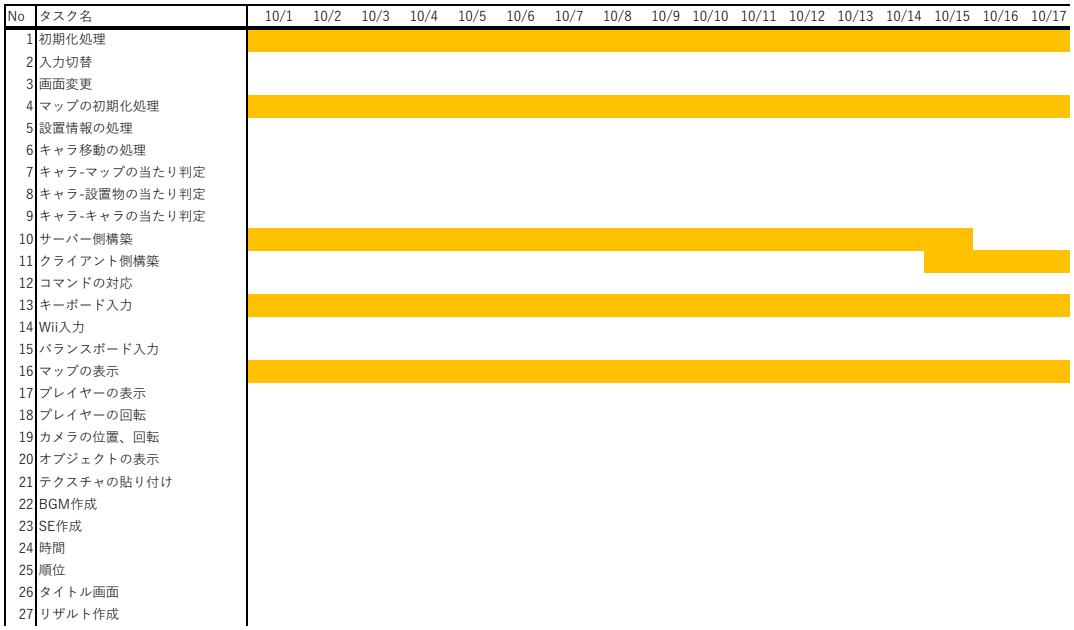


図 5: ガントチャート 2

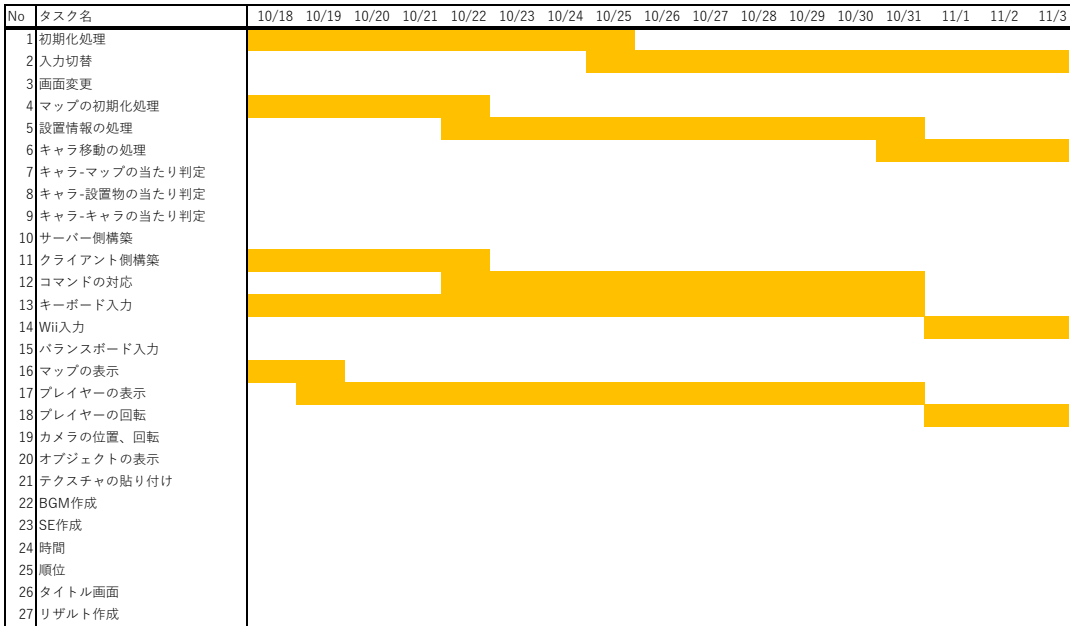


図 6: ガントチャート 3

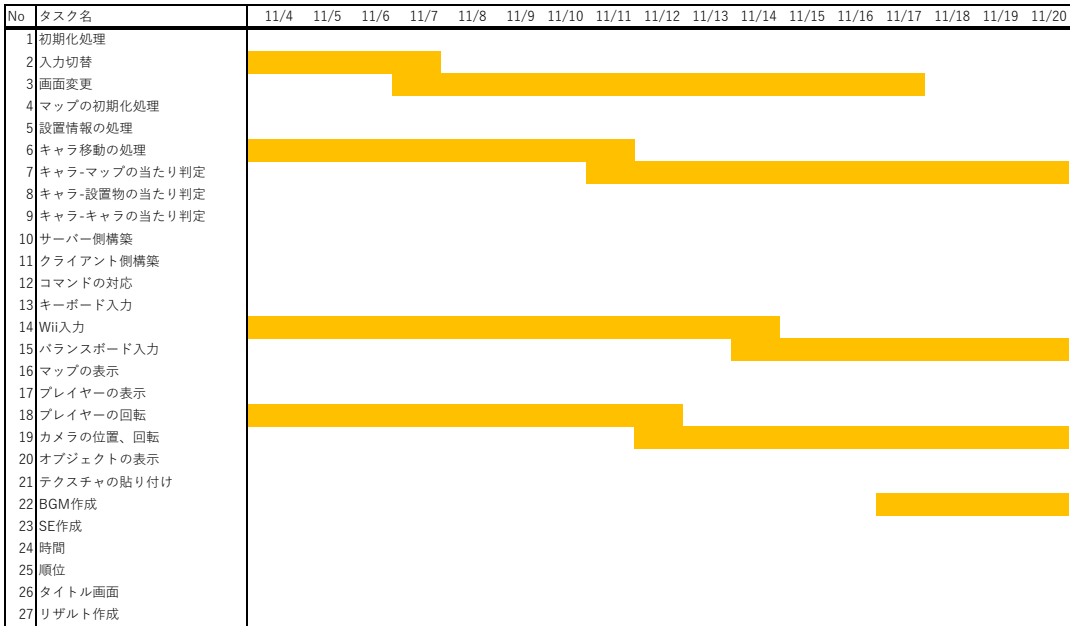


図 7: ガントチャート 4

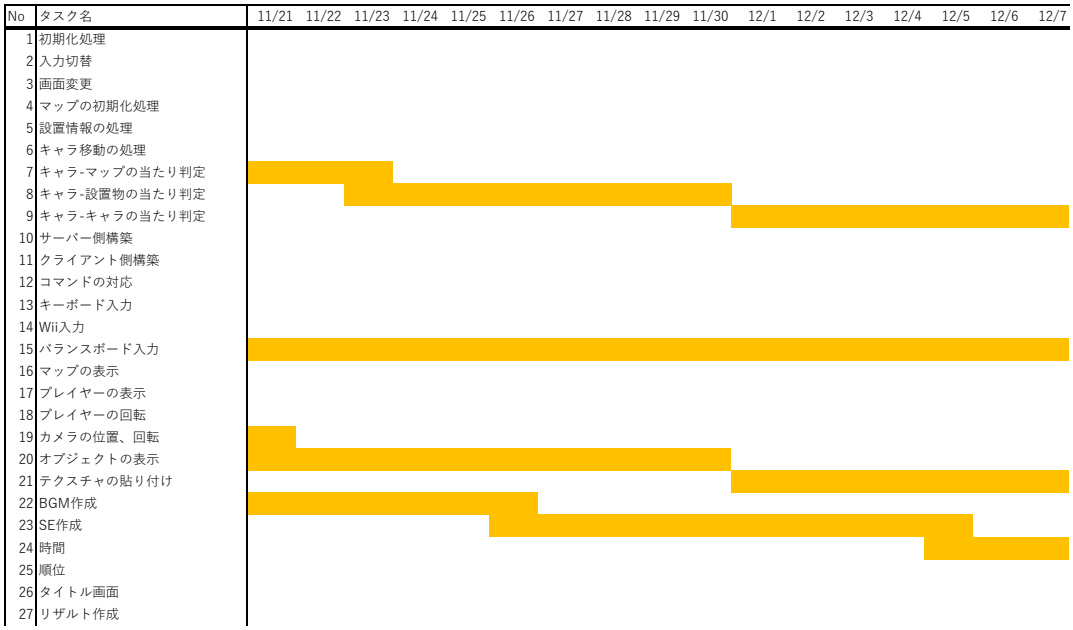
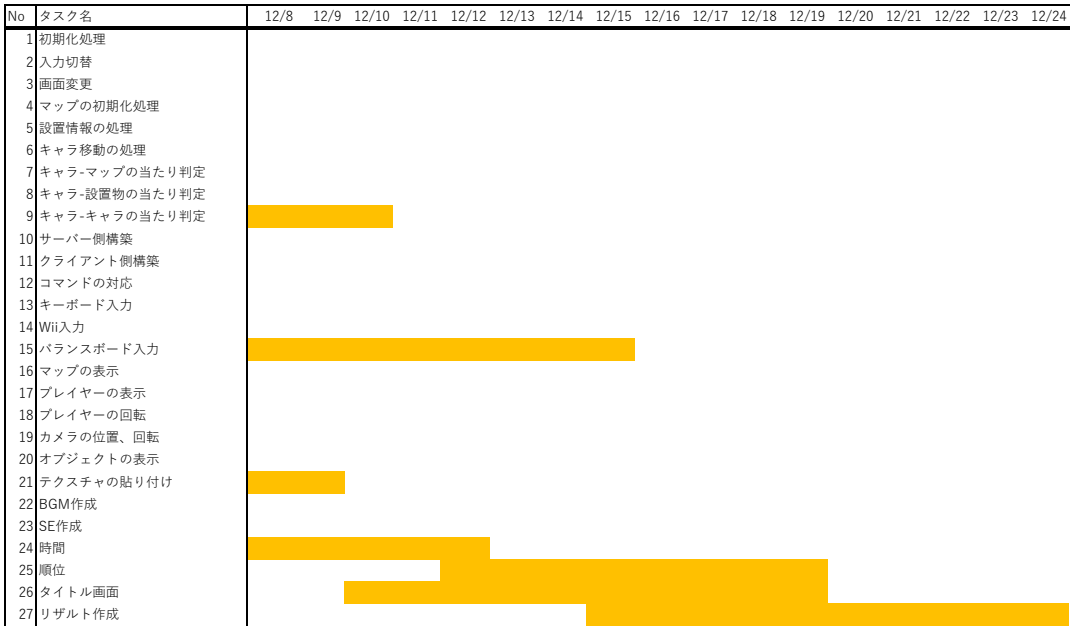


図 8: ガントチャート 5


No
タスク名
12/8
12/9
12/10
12/11
12/12
12/13
12/14
12/15
12/16
12/17
12/18
12/19
12/20
12/21
12/22
12/23
12/24
1 初期化処理
2 入力切替
3 画面変更
4 マップの初期化処理
5 設置情報の処理
6 キャラ移動の処理
7 キャラ-マップの当たり判定
8 キャラ-設置物の当たり判定
9 キャラ-キャラの当たり判定
10 サーバー側構築
11 クライアント側構築
12 コマンドの対応
13 キーボード入力
14 Wii入力
15 バランスボード入力
16 マップの表示
17 プレイヤーの表示
18 プレイヤーの回転
19 カメラの位置、回転
20 オブジェクトの表示
21 テクスチャの貼り付け
22 BGM作成
23 SE作成
24 時間
25 順位
26 タイトル画面
27 リザルト作成