デジタルコンテンツ 1：最終レポート

5418049 上田 春河　5418015 金 智源　5418076 福嶋 稜

2020 年 8 月 25 日

概要

本稿はデジタルコンテンツ 1 の最終レポートである. 私たちのチームが作成したゲームに関しての総まとめを記載する.

1　作品の狙いやアイデア

今回私たちのチームの作成するゲームは高校生や大学生を対象とした手軽なシューティングゲームである. コンセプトは簡単, 短時間で他人と競えるエンタテイメントである. シューティングゲームをプレイしたことのない人でもできる手軽さと絶妙な難易度による戦略性を持たせる.

2　プログラムの仕様

- スタート画面
  スタート画面では以下のような仕様とプログラムが実装される
  - 4 種類の難易度を 1 から 4 の数字キーで選択
  - 対応するキーを押すことで 3 秒カウント画面に移行
  - 3 秒カウントされ次第、選択された難易度のゲームがスタート
  - ゲームの進行を管理する GameManager クラスでタイトル画面やその他の画面への遷移を管理する。
    また、ゲーム自体は Game クラスで作成しており、引数として敵の出現率、タイプの違う敵が出る確率を渡すことで難易度を変更できる。
- ゲーム画面
  スタート画面では以下のような仕様とプログラムが実装される
  - スタート画面で難易度選択をしたあとすぐにゲームが開始する
  - ランダムに敵が上からまたは左右から出現し、自機を狙った弾や、まっすぐ打つ弾、全方位弾など様々な弾幕を展開する。
  - 難易度ごとに敵の出現率を管理する。また、敵の得点も難易度ごとに管理できるするようにする。
  - 画面左部分がゲーム画面となっており、右部分は残り時間、現在のスコアを表示する。
  - 自機の操作は、キーボードの上下左右キーで上下左右に移動し、移動しながら SHIFT キーを押すことで低速移動が可能となる。また、z キーを押すことで自機が弾を発射する。
  - 制限時間を過ぎたらゲームを強制終了し、スコア画面に移る。

– 作成クラス

* Drawable クラス
  画面上に描画されるオブジェクト（自機、弾、敵）の親クラスで抽象クラス。
  - 変数
    PVector loc; //位置
    float velX, velY; //移動速度
    boolean isDead; //消失判定
  - 抽象メソッド
    abstract void display();
    abstract void update();
* Player クラス
  自機の描画やキー入力に対する行動を決めるクラス。
  - 変数
    float size; //自機のサイズ
    float hitSize; //当たり判定サイズ
    int coolingTime; //次弾発射までの間隔
    int coolMax = 5; //発射間隔を決める値
  - メソッド
    void display() //表示メソッド
    void update() //座標移動など、更新メソッド
    void moveByKey() //キー入力によって行動を決めるメソッド
    など
* Enemy クラス
  敵の描画や行動などを管理するクラス。
  - 変数
    float size; //敵のサイズ
    float hitSize; //当たり判定サイズ
    int coolingTime; //次弾発射までの間隔
    float angle;　　//弾のアングル（おそらく必要なし）
    int score; //敵個体が持つスコア
    int max_interval; //敵の弾発射間隔の最大値
  - メソッド
    void display() //表示メソッド
    void update() //座標移動など、更新メソッド
    int getScore() //個体のスコアを返すメソッド
    private void hitOthers() //自機の弾などと接触したかを判定するメソッド
    など
* Bullet クラス
  このクラスは自機、敵の弾を管理するクラスである。

· 変数

float sbx, sby, sb; //sbx:敵と自機の x 座標の差 sby:敵と自機の y 座標の差 sb:敵と自機の直線距離

float vel; //移動速度ズ

float diffusion_angle = PI/3 * 2; //拡散角

float base_angle, change_angle, edge_angle; //中心角度、間隔角度、端の角度）

boolean isMine; //自機の弾かどうか

など

· メソッド

コンストラクタ内で弾の挙動を初期化

void display() //表示メソッド

void update() //座標移動など、更新メソッド

- スコア表示画面

スタート画面では以下のような仕様とプログラムが実装される

  - ゲーム画面が終了したらこの画面を自動的に表示
  - ゲームでとったスコアを一番下に表示
  - 自分のスコアがランキング内に入ったらその順位と得点の色をわかりやすくする
  - 右下のボタンを押すとスタート画面に戻りもう一度ゲームを始められる
  - 作成クラス
    * Score クラス

      このクラスはゲーム終了後のスコアを表示する動きを管理するクラスである. プレイヤーが今回のゲームで獲得したスコアと今まで記録し最多たスコア降順を表示, 比較, 獲得スコアが記録スコアより大きかったらその順位を入れ替えてソートして表示を行う.

      コンストラクタには引数としてゲーム画面で獲得したスコアを入手して変数 score に格納する.

      メソッド readScores は別ファイルに保存してある今までの今までのスコアを高い順に記録しているを読み込みその内容を配列 scoreRanking に保存する.

      メソッド sortRanking でコンストラクタで入手した獲得スコアを記録スコアと比較, もしスコアの大きさによって必要があれば配列 scoreRanking をソートをする.

      メソッド displayScores で score 変数と scoreRanking 配列を画面に表示する. またこのメソッドの中でクリックするとスタート画面に戻ってもう一度ゲームを開始するボタンを実装する.

3　プログラム実行の様子

1. スタート画面

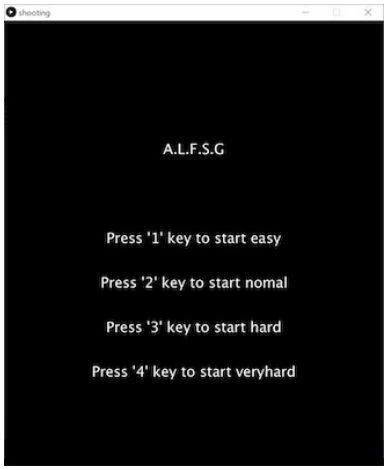

図 1　スタート画面

ゲームを実行すると図 1 のような画面が表示されて難易度を選択させる. キーボードの数字キーを押すと対応した難易度のゲームが始まる. 難易度を選択後 3 秒の時間経過の後にゲーム画面に遷移してゲームが開始する。

2. ゲーム画面

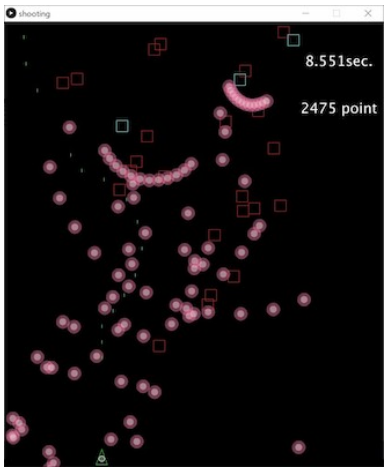

図 2　ゲーム画面

ゲームを開始さすると図 2 のようなゲーム画面に遷移して指定した難易度に合わせた数の敵がランダム

に出現し, 攻撃を仕掛けてくる. キーボードの十字ボタンで画面下の緑色の自機を操作しながら攻撃を掻い潜り, キーボードの z キーを押すことでこちらも攻撃を行い敵に当たれば倒すことがでいる. 以上のゲームが 30 秒続き 30 秒経過で強制的に操作受付を終了して次のスコア表示を行っている.

3. スコア表示

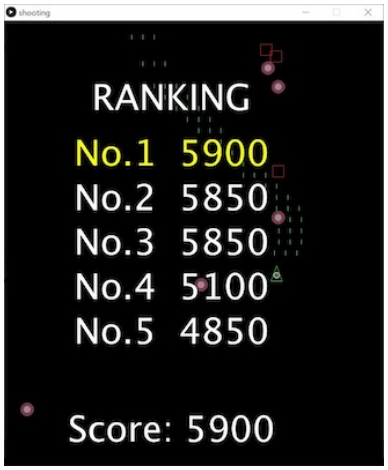

図 3　スコア表示画面

ゲーム画面で 30 秒たったら強制的に図 3 のようにスコアが表示される. 一番下に今回のゲームで獲得した自分のスコア, 上 5 つのスコアは自分が以前に獲得したスコアの上位 5 つが保存されておりそれが表示される仕組みとなっている. さらに自分が今回獲得したスコアが今までの上位 5 つのスコア以上ならばそれらをソートして今回のスコアを挿入した後のスコアが表示黄色で目立つように表示される.

4　仕様結果

私たちの作品は、短い時間内により多くのスコアを稼ぎ他の人とも競い合えるゲームを目指して作成したので、メンバー全員の作成した難易度別のゲーム進行、ゲーム自体、スコア管理のプログラムはその目標を達成できるものであったと考える。

5　発表時に受けたコメント

- 敵ごとに得られる得点に差はあるのだろうか?
  差はなく全てのてきは倒すと難易度ごとに設定されて一定のスコアを加算するものである.
- 難易度が難しすぎるのではないか?
  その調整を行ったものがシューティングゲーム経験者であったため確かに難易度 nomal でもそこそこの難易度がある.
- パワーアップアイテムはある？

自機は攻撃を受けるたびに攻撃力がなくなる。そのため回復手段はあるのか？という質問であったが、30 秒という制限時間内にどれほど得点を稼げるかというゲームのため、そのような回復手段は用意しなくてもよいと判断した。

- 難易度ごとに敵の出現率は変わるなら、難易度が高いほうが稼げてしまうのでは？

  敵のスコア自体を難易度で変更するように作成していなかったため、スコア合計を計算するクラスで、難易度に応じて加算されるスコアに対し重みを掛けることでどの難易度でもおおむね同じような点数が取れるように調整を行った。
- 敵がランダムに出ているようだが、プレイするたびに敵の数が変わると運ゲーとなってしまうのでは？

  敵はなるべくランダムに出現させたかったため、ランダムを使用しているが、easy と normal についてはランダムシードが 1 種、それ以上は 4 種とほとんど数が変わらないようになっているので、完全な運ゲーとならないように調整している。
- シューティングには覚えゲーとなるものもあるが、これは覚えゲーか？

  先ほど同様 easy と normal については、ランダムシードを 0 として敵の出現パターンは固定になっているが、それ以上の難易度は 4 種類ほどのランダムシードをランダムに渡す仕様となっているので、完全な覚えゲーにならないようになっている。

6　今後の課題

今後の課題を箇条書きで以下にまとめる.

- 適切な難易度調整

  発表時に受けたコメントでもあるとおり, 初心者に対しては難易度が下でも一定の難しさがあることがわかったため初心者でもできるようにゲームを易しくする工夫を導入していく必要があるとわかった.
- 終了カウントダウン

  現在のプログラム使用では 30 秒が立つと前触れもなく強制終了するのでラスト 5 秒前くらいでカウントダウンの表示があるとユーザにより親切なゲームプログラムになるのではないかと思う.