Scheme+OpenGL でゲームを作る

Freak32768

Table of Contents

1. 概要

こんにちは、ふりーく(Freak32768)です。ここでは Colour についての解説や、 製作中の気付きについて書いていきます。

2. 前提知識

2.1. Scheme

- 括弧だらけのプログラミング言語 Lisp の方言(意味は日本語の"方言"とだいたい同じです)。 仕様がすごくシンプルである。
- "Scheme"という単語はプログラミング言語の仕様を指す。 Scheme のプログラムを動かすには、後述の Gauche のようなソフトウェアが必要。

2.2. Gauche

- Scheme のプログラムを動かすソフトウェア。製作者は Shiro Kawai さん、日本人である。
- 実は Scheme はそのままだとシンプルすぎてほぼ何もできないため、いくつか機能が追加されている。
- ちなみに読み方はゴーシュらしい。

2.3. OpenGL

- 3D グラフィックを扱うためのツール。類似のものに DirectX があるがこちらが Windows, Xbox 専用なのに対し OpenGL はそれ以外の OS でも使うことができる。
- CG の理論を忠実に反映した設計であり、またシンプルである。

2.4. GLUT

- OpenGL は(これまた)シンプルすぎて扱いづらいため、それを扱いやすくするためのツール。これがないとやってられない。

2.5. gauche-gl

- 先ほど紹介した Gauche というソフトウェアから OpenGL, GLUT を使えるようにしたもの。

2.6. Colour

- 私が作ったゲーム。この記事で解説していくが、正直クオリティは低い。

2.7. オブジェクト

- ゲーム内のキャラクターや構造物など、描画されるモノのことを示す単語。 プログラミング経験者の方は紛らわしいと感じるかもしれないが、どうかご容赦いただきたい。

3. 解説

- 解説における括弧の中身はプログラムを実際に読んでみたい方向けです。ざっくり理解するだけならお気になさらず。

3.1. プログラムの大まかな流れについて

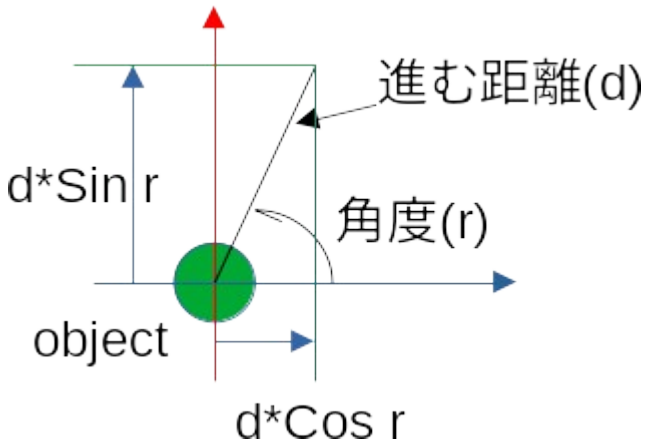
- このプログラムが起動されるとまず、main という部分(main.scm L439-L451)が実行されます。 ここではプレイヤーとか諸々を表示するウィンドウを作り、それから必要な設定などをします。

- それが終わると、init という部分(main.scm L406-L417)が実行されます。ここでは OpenGL を初期化します。 内容は光源の設定、プログラムの処理方法について、などです。
- 次は game-init(main.scm L63-L126)という部分が実行されます。ここではゲーム内のキャラクターや構造物などを初期化します。
- ゲームの中心となるのは display-main-scene という部分(main.scm L154-L160)です。ここでゲームの諸々を描画しています。
- しかしこれをご覧の皆さんはプログラムが短すぎて、本当にキャラクターや構造物が描画されているのか疑問に思うかもしれません。
- じつは display-main-scene では display-canvas という部分(main.scm L162-L171) を呼び出しています。さらに display-canvas では各オブジェクトを描画する部分を呼び出しています。 全部一箇所にまとめて書くと長くなってしまうので、複数の部分に分割しているのです。

3.2. ゲームプログラミングのテクニック

3.2.1. 斜め移動

- "オブジェクトを向いている方向に移動させる"というのはゲームプログラミングにおいてほぼ必須の技術ですが、意外と面倒です。 その理由は、グラフィックの世界が数学のグラフのように複数の座標軸からなっているからです。 そのためオブジェクトを向いている方向に動かすには、x 軸方向にどれくらい、y 軸方向にどれくらい動かす、 というように命令する必要があります。 そこで使うのが三角関数。sin, cos をうまく使うと、下図のように軸ごとの移動距離を求められます。



3.2.2. 当たり判定

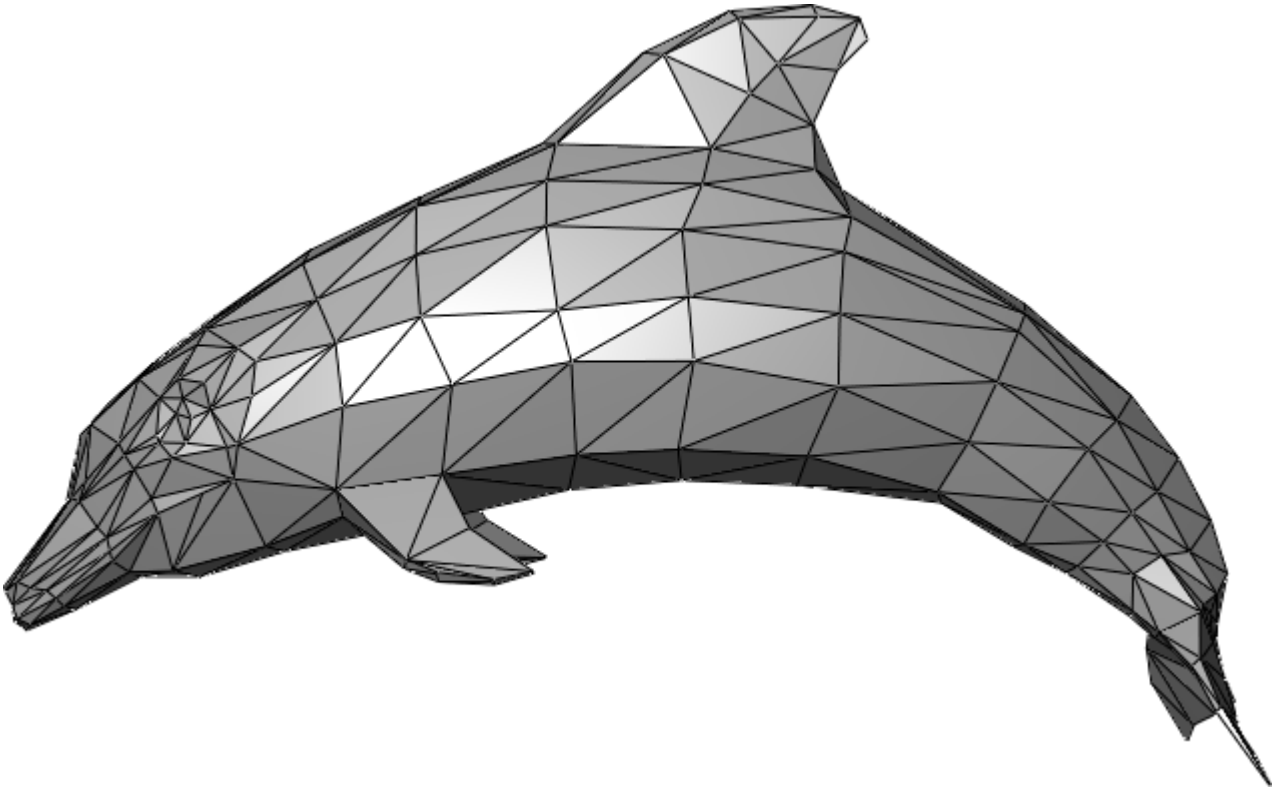
- これまたゲームプログラミングにおいてほぼ必須の当たり判定ですが、これまた意外と面倒なのです。 先述の通りグラフィックの世界は複数の座標軸からなっていて、斜め方向の距離などという概念はありません。 そこで使うのが三平方の定理です。

下図のように三角形を作ると、斜め方向の距離を求めることができます。

オブジェクト

3.2.3. 3D モデルの描画

- ある意味でこのゲーム最大の目玉()である 3D ですが、OpenGL の 3D 機能はとても貧弱です。 なんといっても、OpenGL では基本的な図形しか書くことができないのです。 ではどうするか、というと、3D モデルを下図のように一旦複数の三角形の面に分割します。それから三角形の面を 1 つずつ描画していきます。



(https://en.wikipedia.org/wiki/File:Dolphin_triangle_mesh.png, by Chrschn)

- しかし今回はこの仕組みを自前で用意しなければなりませんでした。libs/3d-object.scm にその実装があります。 3D モデルを描画するシステムの実装は、この作

品における最大の難所でした。

4. 気づき

4.1. プログラムの分割

- 今回の Colour のようなちっぽけなゲームでも、ゲームとして完成させるためにはコードが数百行になります。 1 画面に収まりきらない量なので、これを 1 つのファイルにすべてまとめると、後で見返したときに大変なことになります、 というか製作中になりました。
- プログラムを書くときは 1 つのファイルが大きくなりすぎないよう、適切に分割する必要があります。

4.2. 言語の選び方

- 今回 Colour を作るにあたって私が Scheme というマイナーな言語を選択した理由は、Scheme、 というか Lisp の構文が美しいと感じていたからです。そしてそれは今も同じです。 当時はむしろ"Scheme で書く > ゲームを作る" という有様でした。 しかし言語がマイナーであるが故に情報がほぼ公式ドキュメント(英語)しかなく、欲しい情報を探すのが非常に大変でした。 また音声を扱う gauche-openal に至ってはドキュメントが見つからず、ゲームの効果音がないという事態になってしまいました。 この一件で私は、構文だけではなく、その言語の便利なツールの数や情報の数もまた言語の価値だと気づきました。
- プログラミング言語は目的から選ぶのが良いでしょう。そしてある程度メジャーで、日本語の情報が多いものを選ぶとなお良いです。

5. 感想

- 実は私はこれまで普段遣いのちょっとしたツールしか作ったことがなく、そこそこちゃんとしたプログラムを作るのは初めてでした。 今回のプロジェクトは私にいくつもの新しいことを教えてくれました。特に"気づき - 言語の選び方"の経験で、私は "ゲームプログラミング初心者だけどマイナー言語でも余裕だろう"と自身が意気がっていたことに気付かされました。 次回のプロジェクトは今回の反省を活かし、よりゲームらしいものにする予定です。乞うご期待

6. License