Ballistics: 2D, 3D & NetGame

Overview

Create games that use a simple ballistic projectile simulation as the central interaction model for game play. In general this type “artillery game” is typically turn-based, however real-time asynchronous play is also popular.

Introduction

Player success in an artillery genre game requires the selection (or adaptation) of appropriate parameters that influence a projectiles trajectory in the game space. The minimum parameters are direction (angle) and initial force. Other possible parameters include mass and type of the projectile (and modifiers to the damage) and economic models that require strategic approaches to consistently win the game.

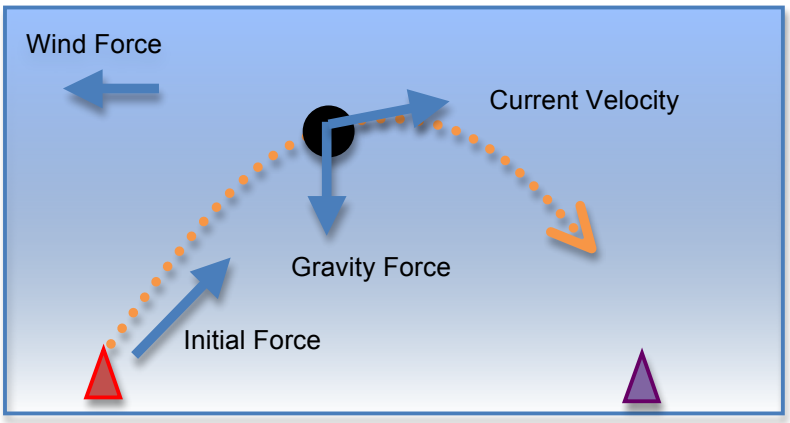

Figure 1. A simple representation of the forces influencing the projectile mass and the resultant velocity.

If force and velocity are represented as vectors, we can describe the change in position of the projectile with respect to time with the following equations where ***F*** = force (vector), *m* = mass (scalar) and ***a*** = acceleration (vector), and as a result ***v*** = velocity (vector), ***p*** = position (point represented as a vector) and *dt* = time (delta) step used. (Bold is used to indicate a vector.)

$$\boldsymbol{F} = m * \boldsymbol{a}$$
$$\boldsymbol{a} = \boldsymbol{F} / m$$
$$\boldsymbol{v}_{new} = \boldsymbol{v}_{old} + \boldsymbol{a} * dt$$
$$\boldsymbol{p}_{new} = \boldsymbol{p}_{old} + \boldsymbol{v}_{new} * dt$$

The force vector ***F*** is the net force of all vectors acting upon the mass (the projectile). In this case that means that at the start there is the initial force + gravity + wind, but for the duration of the flight only gravity + wind continue to alter the acceleration (and hence alter the current velocity and new position per time step update).

This specification document breaks the game into three different versions of increasing feature complexity; a **2D version**, a **3D version** and a network **NetGame** gameplay model. However the gameplay for all versions is, in essence, based on a very simple physics model: adjusting parameters for direction (angle) and initial velocity for a projectile that will influence the path of a projectile.

- Version 1: 2D Ballistics
- Version 2: 3D Ballistics
- Version 3: NetGame Ballistics (2D or 3D network enabled multiplayer game)

Assumptions

The details provided for each game version are very brief. It is assumed that you will create a detailed design based on the tools available to you and the decisions you make. For example, the game states (menus) required have not been specified, entity (token) identification and interactions, nor requirement for input handling or sound.

Version 1: 2D Ballistics

Required features

- Simple two player gameplay (no computer player)
- Turn-based gameplay
- Simple “block-world” objects for the terrain (see classic Gorillas game as a reference) and simple
- Allow each player to enter their “name” before the game begins
- Simple representation of players (tanks, domes, square… whatever)
- Allow user to interact with game to enter (text input?) or GUI adjust (sliders?) the angle and initial force parameters
- Each game should have a random wind force value (constant during the game) that is indicated to the players in some way (text+arrow for example).
- Each game should be a random terrain created using blocks
- When “fired” each players’ projectile should be displayed as it travels though the 2D space.
- Projectiles that travel outside of the window should be allowed (with reasonable time limits)
- There must be reasonable limits to the force and angle parameters
- Collisions between the projectile and the environment (blocks), and the projectile with both players must be supported. It is possible for a player to kill themselves if their own projectile hits themselves.

A number of game modes could be supported;

- sudden death (first kill wins),
- best of 3 or more.

You decide.

Optional features

- *Animated sprites for the projectiles during flight and for block-world collision or player collisions*
- *Sounds for projectile launch (ascending / descending) and collisions (with world or with players).*
- *Time limits (to stop players from “square counting” their way to a first shot kill).*

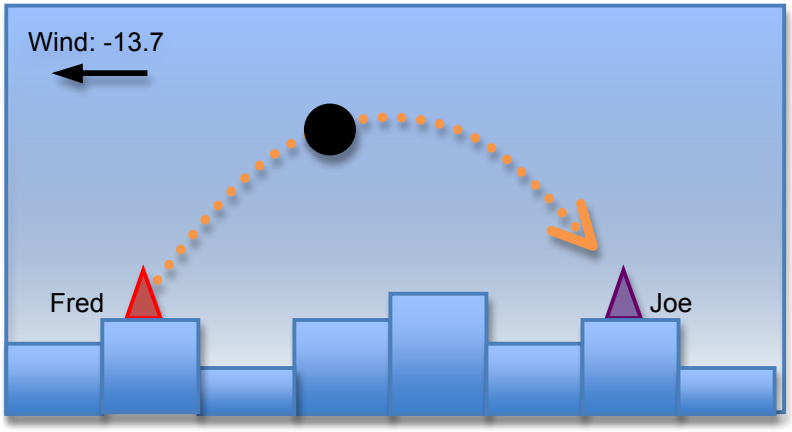

Figure 2. Use simple blocks (rectangles) to construct the world environment and test for collisions.

Version 2: 3D Ballistics

Essentially this version needs the same features as 2D version, however the controls require both an elevation angle and a direction. The 3D environment should be constructed with simple “boxes” (which makes for simple projectile collision detection with bounding boxes) and the camera angle should move to a position behind the current player who’s turn it is (in line with their current angle).

Optional: Use a terrain surface (mesh) instead of basic blocks, and this might be quite easy if the 3D engine or tools you have selected explicitly support this.

Version 3: NetGame Ballistics

A network enabled version of either the 2D or 3D game version. The choice of network model (server or peer) is not defined. There are different data requirements depending on the mode of gameplay selected (synchronous turn-based or asynchronous real-time play). Additional game states (menus) will be required to enable the setup and configuration of a network game. This might include the launching of a server, or the polling of a game server for available games.

Optional: Consider a server based hall-of-fame for players to record their kill ratios and other stats.

Reference Examples

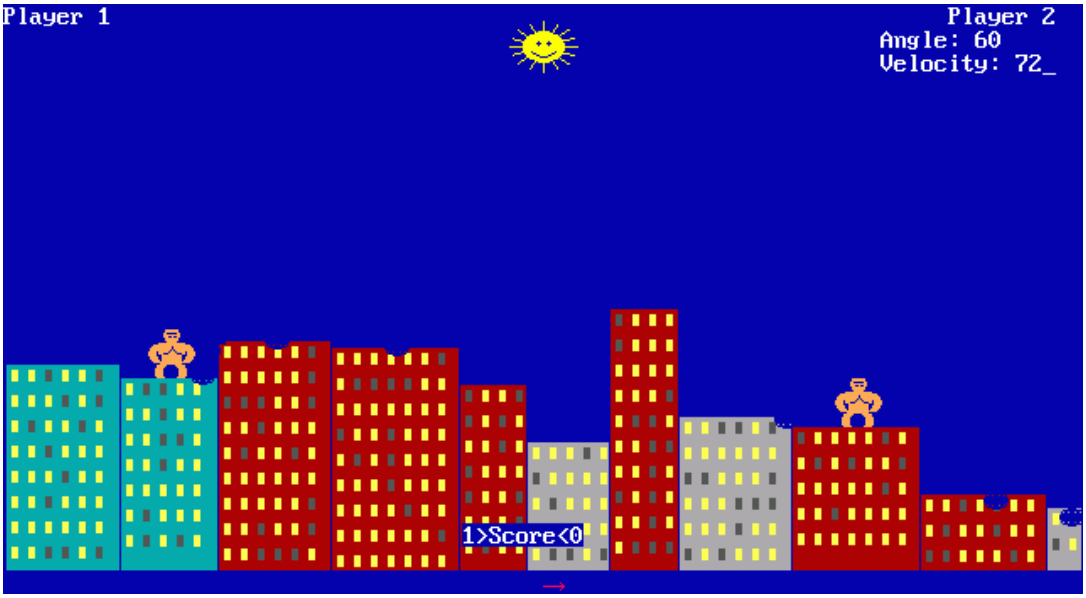

Figure 3. Screenshot from the Gorillas (gorillas.pas) BASIC game distributed with MS DOS 5. Each player directs their gorilla character to throws banana projectiles at opposing player in a turn-based model. Parameters are specified using a simple text input mechanism.

Figure 4. Screenshot of the Scored 3D game. This popular modern reincarnation of the artillery genre includes a wide variety of projectile types, economy system, network play for two or more players and optional asynchronous (real-time non-turn-based play) features. Parameters are adjusted using GUI widgets controls for each value as well as camera position controls.

Warning (joke): Don't play Scorched3D during the exam period; although some claim destruction activities have therapeutic effects, the addictive gameplay qualities may prove detrimental to your unit results.