Multi-Player Pong

©. Copyright 2009, Ray C Horn, All Rights Reserved.

The playing field is split for each set of N/2 players using the following progression.

1-Player

700x500 field

2-Player

Each player gets their own 700x500 field.

3-Player

Each player gets their own 700x500 field. The 4th player is a wall that rebounds.

4-Player

Each player gets their own 700x500 field.

Paddle Position

Paddle position reflects the relative position of each player in relation to the other players.

Vector Graphics

Simple easy to code vector physics allows the puck to be moved anywhere within the gaming field using minimal coding overhead because a direction vector is nothing more than a single Point object that describes a slope of a line and thus the puck follows a line from collision to collision. One can easily manipulate the direction vector at collision time to determine how the puck will behave after each collision.

Surface Texture Simulation

Surface texture simulation is achieved by manipulating the direction vector at collision time.

A smooth glassy surface is achieved by flipping the sign of the x,y coord based on the surface the puck hits; y for the top and bottom and x for the

©. Copyright 2009, Ray C Horn, All Rights Reserved.

left and right. Collision with the left wall without striking the paddle means the puck was missed and a miss is recorded by the gaming engine. Collision with the right wall means the puck's x value is sign flipped to send the puck sailing back from whence it came. **Hey kids, no reason to use any trig functions here.**

A rough stucco surface is achieved by treating the direction vector to some kind of random manipulations to either increase or decrease the x,y coord values based on the surface that was hit. This adds to the fun factor of the game by making the behavior less predictable.

A rubberized surface is achieved by increasing the x,y coord values at collision time to cause the puck to speed-up after a collision.

A sticky surface is achieved by decreasing the x,y coord values at collision time to cause the puck to slow-down after a collision.

Expanding the game

As each additional player is added to the game an additional playing field is also added along with rebounding walls that fill-in for the missing players.

Modifying the game

Game mods can be achieved by placing obstacles onto the gaming field at random intervals. Each obstacle can be made of a different type of material thus causing the vector physics model to behave differently for each obstacle object. Movie clips are used for each obstacle object to leverage the Flash API 's ability to process the collisions.

Pong Golf

A golf game can be produced from Pong by placing holes and sand traps onto the gaming field. The paddle becomes a golf club. The puck becomes a golf ball. The vector physics model allows the golf ball to be struck with differing forces to simulate the effect of striking a golf ball with an 8 iron versus having used a Wedge or Wood. The puck's color can be changed as well as the dimensions of the puck. The state machine changes for a golf game versus a Pong game but the rest of the gaming engine remains largely the same.

Various golf courses can be achieved by building some metadata that describes each course one hole at a time.

Pong Billiards

©. Copyright 2009, Ray C Horn, All Rights Reserved.

A billiards game can be produced from Pong by placing pockets around the gaming field; make the gaming field into a rectangle; modify the color of the puck; make the paddle into a cue and change the state machine a bit.

Gravity Pong

Gravity Pong is achieved by causing the direction vector to change over time. The puck slows and drops towards the bottom wall via simple manipulations on the direction vector.

Zero-G Pong

Zero-G Pong is achieved by causing the direction vector to change over time. The puck slows and floats towards the top wall via simple manipulations on the direction vector.

Pong Invaders

Pong Invaders is achieved by placing some animated movie clips on the gaming field that slowly crawl across the playing field from right to left and top to bottom using rows of crawling creatures and such.

Metadata can be used to describe variations on this theme to allow numerous Pong Invader games to be created from a single gaming engine.

Power-Ups

Puck and Player power-ups can be achieved by causing the direction vector to be changed for short periods of time.

Game Capacity

As many as 12 players can all play at the same time on a single 4x4 playing grid with each grid having 3 players connected by a central playing field that connects all the other fields.

Game Server

XML/RPC Gaming Server connects each of 4 separate XML/RPC Servers to which each Human player is connected.

Each Player's XML/RPC Server handles the physics for that player and tracks the movement of the puck on that player's field of view but only while the puck is on that player's field of view.

©. Copyright 2009, Ray C Horn, All Rights Reserved.

As the puck moves from each player's field of view a message is sent to the central gaming server and that server tracks the movement of the puck but only while the puck is on the central field of view.

As the puck moves off the central gaming field or field of view a message is sent to the receiving player's gaming server and the process of tracking the puck continues.

Each Flex client polls for data in the background then handles the vector physics to track the puck through the current player's gaming field.

Game Play

Each Player begins with 0 points.

Missed returns cost -1 point.

When the game reaches a limit of 10 or 20 scores the game stops and the high scores are tracked in a central database connected to the central gaming server.

Game Variations

Multi-Pucks

Each player gets a single puck that leaves the paddle in a direction determined by the position of the paddle to the left of right of center.

Each puck gets a different color.

There can be many pucks within the same gaming field at a time and this adds to the fun.

Social Networking

Connections with Twitter, FaceBook and real-time chats via the distributed Game Servers.

©. Copyright 2009, Ray C Horn, All Rights Reserved.

Change Log

08-04-2009

Initial version.

08-05-2009

Fixed the collision detection between the paddle and puck using a built-in function from the Flash API called hitTestObject(). This function can be used to determine whenever any two movie clips have come into contact with each other. This is yet another reason to use movie clips to encapsulate behaviors in Flash as doing so leverages the API for game development.

Added a more randomized model for the creation and placement of the puck on the playing field and made the direction vector more random and less predictable.

Added some ideas on Game Modifications and Variations on games using Pong as the base gaming model.

©. Copyright 2009, Ray C Horn, All Rights Reserved.