**Ping-Pong Game Report**

**Abstract:**

Yeah sure… abstract… Uhh…. We wrote pong. It’s garbage but we wrote it.

**Background:**

We did your assignment!

**Design:**

An oversimplified model was used where the first client’s game state is used as a “Master” for the game. Player 2’s game state is synced to player 1, with the exception of his paddle.

**Implementation:**

The server sends “commands” to the clients to control their execution of the game. The clients are held in lock-step with the server and every communication requires an ACK. This works fine for localhost, however on the real internet (with ping > 8 ms) this would quickly become unplayable.

**Key Features (Server):**

* Game initialization and configuration for two players.
* Continuous game loop for updating and synchronizing game state.
* Syncing player 1’s game state to player 2.
* Managing scores and determining the winner.
* Communication through JSON-encoded messages over sockets.
* Acknowledgment mechanism to ensure synchronization.

**Key Features (Client):**

* Pygame is used for graphical rendering and user input.
* Connection to the server using sockets.
* Sending paddle movements to the server.
* Performing game logic.
* Receiving and processing game state updates.
* Displaying scores, paddles, and the ball on the screen.
* Acknowledgment mechanism for synchronization.

**Synchronization Mechanism:**

To ensure synchronization between clients and the server, acknowledgment messages are used. Both the server and clients send acknowledgment messages after critical events, such as the start of the game, paddle movements, and score updates. This helps maintain a consistent game state across all participants.

**Challenges:**

Very little. The implementation is extremely straightforward. The server and clients are held in lockstep. We had to do this because the json decoder we use doesn’t support multiple objects in a single string. I couldn’t be bothered to write a proper packets implementation. The code is also broken if a partial json object is received.

**Lessons Learned:**

This project was pretty much a joke and I didn’t learn much at all, the implementation was shoddy and I think we could have done much better. It would be impractical to execute this on anything other than a LAN due to the requirement for the server to receive an ACK for every single communication, for every single frame. Latency quickly makes this game unplayable and if a player disconnects during a match, the other player and the server break.

**Conclusions:**

<Nothing to conclude>