2D multiplayer car game - Specification

Overview

The concept of this game is a multiplayer 2D car game. It will focus on a drifting system which is used to determine the players “point score”. There will also be a timer that allows players to try and achieve the fastest time, or achieve the highest point score, for each map. After the first player has crossed the finish line; The players who are still on the track after will have their point scores deducted from incrementally until they finish the race.

The players should be able to login to the client, see what cars they have, what friends they have online and create lobby’s to join up with other players; They should also be able to initiate and play games with the players in their lobby. The servers should be authoritative and hosted officially rather than by the users; This is to try and keep the games secure and to assure rewards and high-scores earned during those games are legitimate.

There should be a secure high-score's web app I can go to and see what players have achieved the best times and scores for each map and to handle/create their user account for the game.

Gameplay Overview

The driving/drifting should feel responsive and have a high skill-cap.

There will be a timer that displays the elapsed race time and a score counter that displays your current drift score.

The cars can have different properties which affect how they drive (Acceleration, Traction, Top Speed, etc.).

Drifting will occur when the horizontal component of the car’s velocity relative to its forward direction (the amount it’s slipping sideways) goes over a certain “drift threshold”.

The score from each drift will be calculated from the time in the drift and how “sideways” you’re slipping.

Cars should be able to collide with each other and the edges of the track and have realistic reactions, movement and rotation, to those impacts. Crashing into the edge of the track whilst drifting should invalidate that drift’s “score”.

This project will be split into four main parts:

- The Client (Unity game)
- The Game Server(s) (C# Console application)
- The Match-making Server (C# Console application)
- The High-scores (MVC web application)

The client will be responsible for:

- Connecting to the match-making server,
- Displaying information on (Lobby's, Car's, Highscores, Friendslist's, and Acccounts)
- Requesting Creation/Joining/Deletion of lobby's
- Requesting initiation of a game from the match-making server
- Connecting to the requested game-server once its created
- Displaying of the game graphics
- Handling of the users inputs
- Predicting player and opponent positions based on (past positions, velocities and user inputs)
- Communicating player inputs to the server so it can update the game-state
- Reconciling and interpolating player(s) based on the server's game-state
- Handling disconnection from the game server and/or match-making server,

The match-making server will be responsible for:

- Handling incoming connections from any client
- Authenticating the users and linking the clients to the accounts
- Handling any pre-game configurations ie. (Car type, Max acceptable ping,)
- Allowing create,/joining/delete for lobby's so players can play with their friends
- Communicating with the logon and user information database to get player information to serve to the client or game server.
- Communicating with a hosting platform to provision a new game server instance when a lobby of players want to play a game
- Sending data to the players clients to allow them to join the game-server, and re-connecting the players after a game
- Updating of the player information database based on what the game-server tell it happened in the game

The game server will be responsible for:

- Awaiting connection from the players that the match-makings server has said are in the game
- Handling removal of disconnected users.
- Initiating the game once all the players have connected or after an amount of time
- Keeping track of game state and handling game logic
- Receiving the requests to change the game state as “user input”
- Validating user input and updating the game-state
- Sending back the data the client needs to update/validate it’s game-state
- Sending of game information securely to the high-scores application and the match-making server

The high-scores application will be responsible for:

- Allowing secure POSTing and GETing of the high-score data
- Serving “High-Scores” web page that proves a list of best times and scores for different maps
- Providing account creation and management functionality
- Securely communicating with the login and player information databases
- Updating relevant databases when users edit their accounts.