

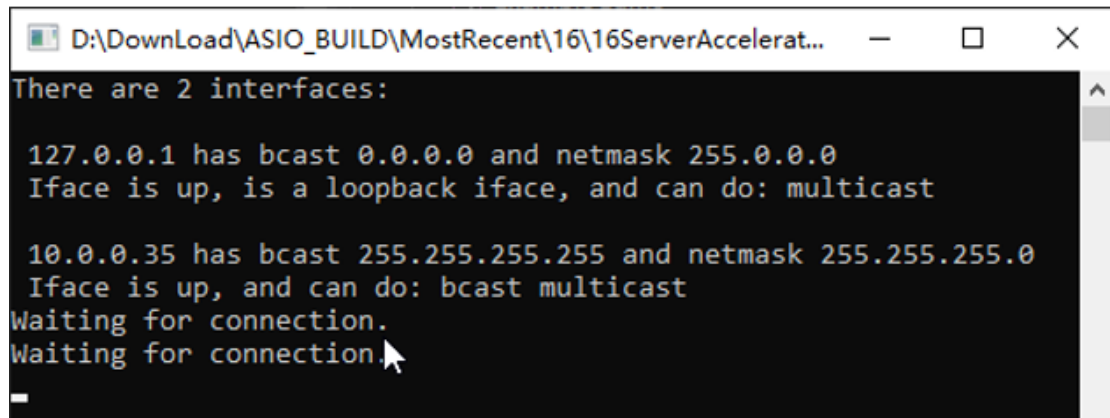
Final project write up

What this game is about.

I have made a multiplayer bumper car game, where each player controls a car. Player could crash into others to bump them outside of boundary and players outside of the boundary will be reset automatically at the center of the map. For now this game provides supports to up to three players.

How to play this game

One of the player will run a server application. Which looks like this.

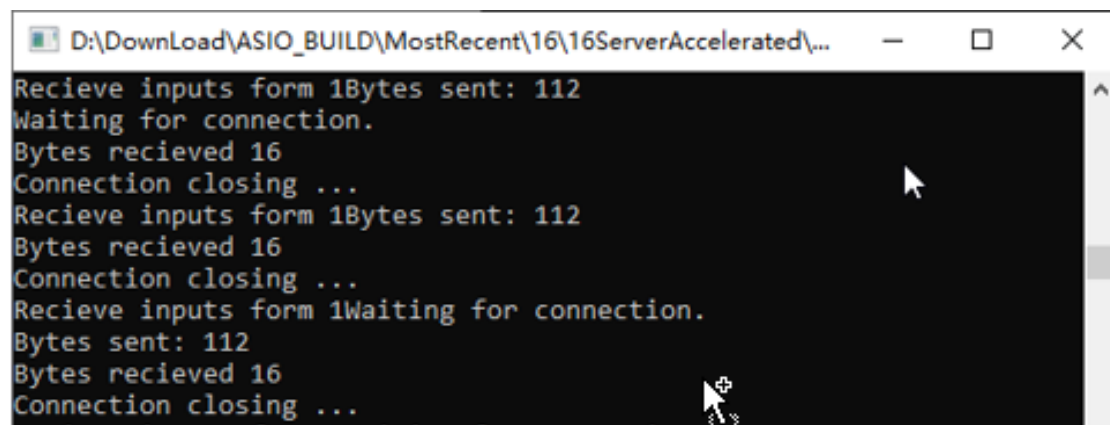


```
D:\Download\ASIO_BUILD\MostRecent\16\16ServerAccelerat...
There are 2 interfaces:

127.0.0.1 has bcast 0.0.0.0 and netmask 255.0.0.0
Iface is up, is a loopback iface, and can do: multicast

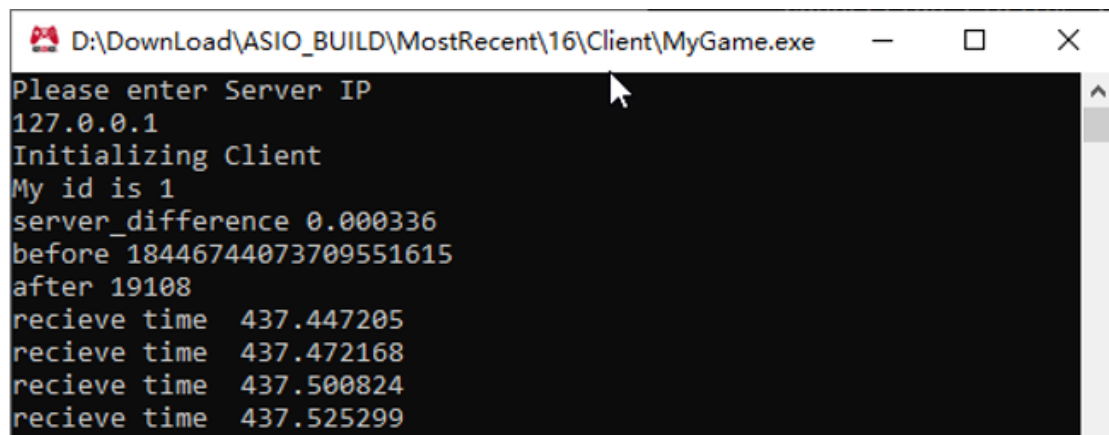
10.0.0.35 has bcast 255.255.255.255 and netmask 255.255.255.0
Iface is up, and can do: bcast multicast
Waiting for connection.
Waiting for connection.
```

Then the player will want to share the IP address other than the local address “127.0.0.1” one to other players. There could be multiple IP address, cause different routers would assign different IP addresses to the same computer, and the solution is to let other players try to connect to them all. A running server.



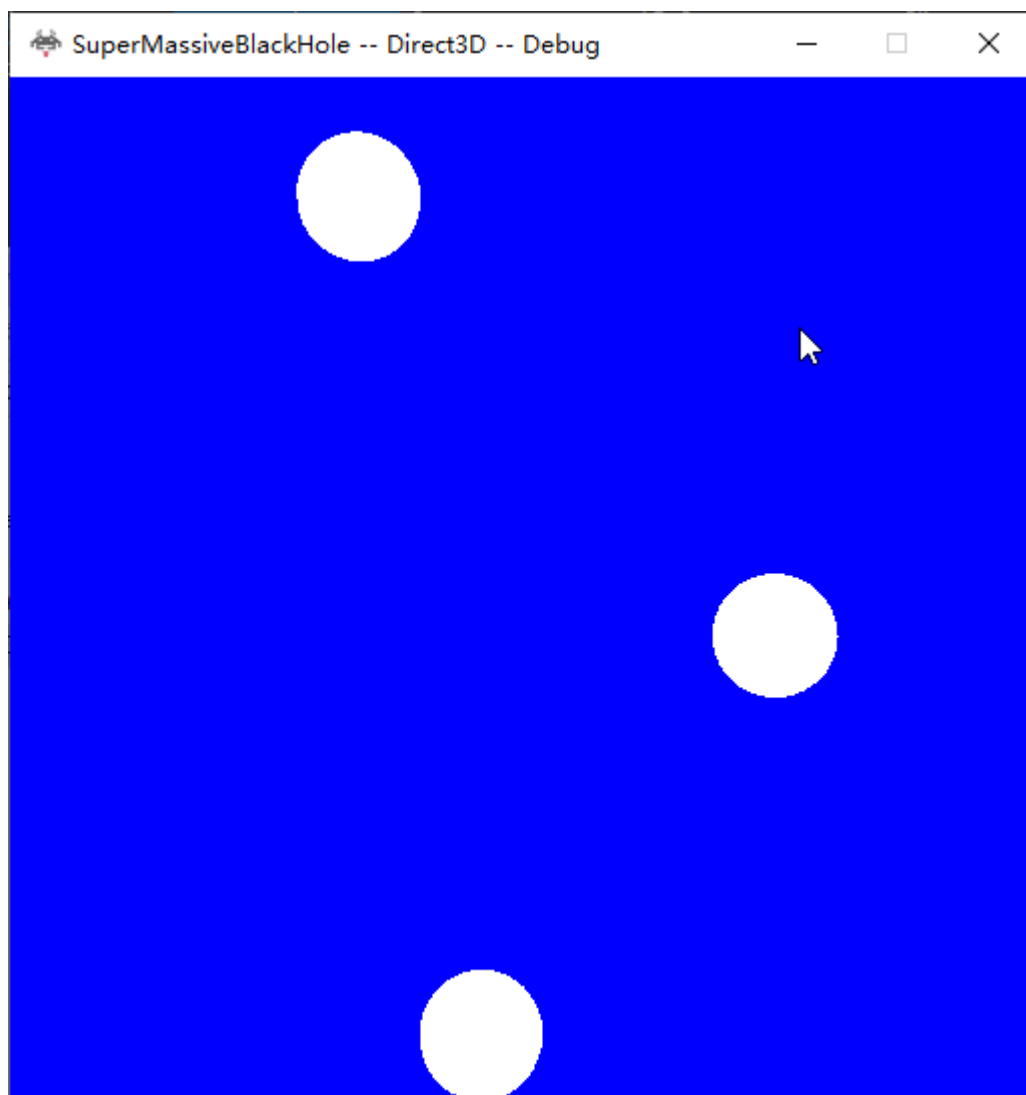
```
D:\Download\ASIO_BUILD\MostRecent\16\16ServerAccelerated\...
Recieve inputs form 1Bytes sent: 112
Waiting for connection.
Bytes recieved 16
Connection closing ...
Recieve inputs form 1Bytes sent: 112
Bytes recieved 16
Connection closing ...
Recieve inputs form 1Waiting for connection.
Bytes sent: 112
Bytes recieved 16
Connection closing ...
```

The player runs the server could directly connect to the local address



```
D:\Download\ASIO_BUILD\MostRecent\16\Client\MyGame.exe
Please enter Server IP
127.0.0.1
Initializing Client
My id is 1
server_difference 0.000336
before 18446744073709551615
after 19108
recieve time 437.447205
recieve time 437.472168
recieve time 437.500824
recieve time 437.525299
```

When you see this screen you are connected and ready to go.



Engine systems used to make this game

Used my network connection system.

<https://aliensmith.github.io/eae6320/NetworkSystem.html>

Used this audio system provided by Srija Kambhampati.

<https://thedarkmiko.itch.io/audio-engine-system>

A simple collision detection and collision response routine is implemented to make this game.

Implementation details, Challenges, Interesting bugs and so on

Well, I did not run into any problems with integrating the audio system into my game. Please visit their websites for further details of the audio system.

When you have a hammer, everything looks like a nail. Back then when I was implementing the network system. I have just went through the first three chapter of *C++ Concurrency in Action: Practical Multithreading*. Hence, as a challenge I implemented the network system using only blocking functions of winsock, and multithreading. It make more scenes to use non-blocking functions, because the server for games usually works in a parallel or concurrent manner by communicating with multiple clients at the same time. To further understand multithreading, and for fun I implemented the network system as how it is now, which leads to an interesting bugs I run into.

When it comes to bugs and multithreading, the first two things I had in mind are problematic racing condition and deadlock, and the one I run into looks very much like a racing condition. In short, after running a while the data in the a slot of buffer suddenly becomes inconsistent and wrong with the rest of the data in that buffer. I used a double buffer and swap pattern to manage the shared the data between threads. As long as two threads runs with their own data, they are safe to do anything with their data. In my specific case the shared data flow from one thread to the other, and the synchronization or the maintaining of the invariance between two buffers happens when the source thread is submitting the data to a buffer and swapping of the buffers, which is also the blocking points of the two threads. Theoretically, there should not be a problematic racing condition problem. Turns out it is an overflow bugs, due to my indexing convention the data in one array belonging to one buffer overflowed into the same array in another buffer.

The other challenge is the synchronization of time between multiple clients and servers. Usually, due to latency between network connections and other issues, The server would run at a lower update frequency than the game running on the clients. To make things looks smooth, extrapolations or interpolations are used. In my game extrapolation of position based on speed are used at each frame. Extrapolations or Interpolations would require a change of time since last updates, which requires the clients and servers being synchronized in time. There are two routines running on the server, one will collects all inputs, update the game and distributes the results to clients. The other will manage the id of each clients and synchronized clients with server. When initialize connection with the server, the clients will attempts to synchronized with the

server four times, with 20ms gaps in between. And the synchronization results with the smallest latency will be adopted.