Repos:

https://github.com/effypelayotran/Grav-Nav-RL-Server

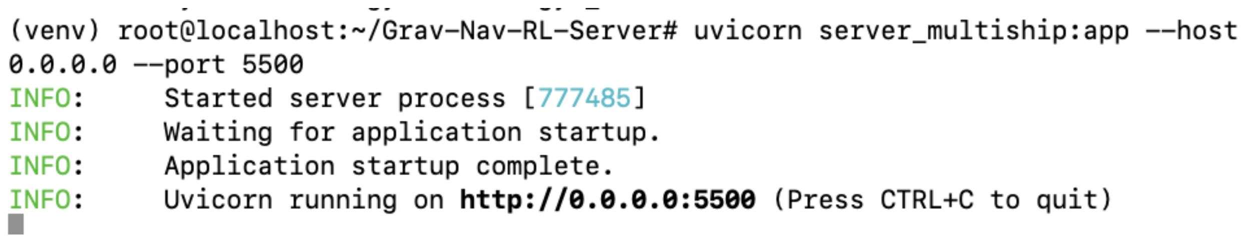
https://github.com/effypelayotran/Grav-Nav-RL-Frontend

Original RL Repo from Sofia and Ben:
https://github.com/BrownParticleAstro/Grav-Nav-RL.git

How to Start the Server on Linode and Connect Client - Current Version using Fast API

On the server side, we're going to be running `python3 server.py` on a Linode SSH connection

1. Make sure the in the repo, the server address is ("0.0.0.0", 5500) and the client_menu address is ("*Public IP Address for your Linode*", 5500)
2. **Open a Terminal on your computer.**
3. **Run** ssh root@45.79.130.19 (or ssh root@ whatever **Public IP Address** is listed for your Linode.)
4. Enter password **brownsmli#12025!**
5. **Make sure the Linode is up to date by running "**apt update && apt upgrade" -y and "apt install python3 python3-pip python3-venv git -y" in the terminal.
6. **Git clone the repo**: "git clone https://github.com/effypelayotran/Grav-Nav-RL-Server" in the terminal
7. **Cd** into the repo directory
8. **Run** `python3 -m venv venv`
9. **Run** `source venv/bin/activate`
10. **Run** `pip install -r requirements.txt`
11. **If you want to run the server in the foreground to quickly try it out, run** `uvicorn server_multiship:app --host 0.0.0.0 --port 5500`. On success, the terminal should look like this.



```
(venv) root@localhost:~/Grav-Nav-RL-Server# uvicorn server_multiship:app --host
0.0.0.0 --port 5500
INFO:     Started server process [777485]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:5500 (Press CTRL+C to quit)
```

12. **If you want to run your server in the background so that you have your command-line of the Linode free to do other tasks run:** `nohup uvicorn server_multiship:app --host 0.0.0.0 --port 5500 > log.txt 2>&1 &` instead ot the uvicorn command in the step above. You can cat log.txt to read the logs. **To kill the background server, run** `ps aux |`

`grep uvicorn`, find the session number listed, and do `kill -9 777485` where 777485 is the session number that was listed. Or `pkill -f "uvicorn server_multiship:app"` to kill quickly.

13. **If you want to more reliably and rock-solidly create a Systemmd Service to run the server in the background, instead of just using nohup, do the following.**
    a. This setup will ensure your server_multiship.py app stays up even after reboot and automatically restarts on crashes — no need for nohup or screen. Let me know if you'd like it to auto-pull updates from Git or deploy via GitHub Actions.
    b. `sudo nano /etc/systemd/system/orbital-server.service`
    c. Paste this configuration:

```
[Unit]
Description=Orbital Multi-Ship Server
After=network.target

[Service]
User=root
WorkingDirectory=/root/Grav-Nav-RL-Server
ExecStart=/root/Grav-Nav-RL-Server/venv/bin/uvicorn server_multiship:app --host
0.0.0.0 --port 5500
Restart=always
RestartSec=5
Environment=PYTHONUNBUFFERED=1

[Install]
WantedBy=multi-user.target
```

    d. Enable and start the systemmd service by pasting all of these comamnds:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable orbital-server
sudo systemctl start orbital-server
```

    e. Check if its running by doing `sudo systemctl status orbital-server`. You should see: Active: active (running)
    f. View logs in real-time with `journalctl -u orbital-server -f`
    g. Restart or stop the systemmd service server with `sudo systemctl restart orbital-server` or `sudo systemctl stop orbital-server`

14. On the client side, make sure the ws address is: new WebSocket("ws://your-linode-ip:5500/ws");

How to Start the Server on Linode - Older Python Sockets Server

On the server side, we're going to be running `python3 server.py` on a Linode SSH connection

1. Make sure the in the repo, the server address is ("0.0.0.0", 5500) and the client_menu address is ("*Public IP Address for your Linode*", 5500)
2. Open a Terminal on your computer.
3. ssh root@45.79.130.19 (or ssh root@ whatever **Public IP Address** is listed for your Linode.
4. Enter password **brownsmli#12025!**
5. **Git clone the playble_phys_game_multiplayer** repo: git close https://github.com/effypelayotran/playable_phys_game_multiplayer/
6. Run `screen -ls` to see any currently active screens
7. Run `screen -S Asteroids`
8. Run `cd the playable_phys_game_multiplayer`. Then, run `python3 server.py` . Await the 'Server started'Mmessage
9. `Ctrl A+D` if you want to close the screen but not to detach the screen.
10. Run `screen -S 3672 -X quit` if you want to actually to kill server(but 3672 is replaced with actual ID you see in screen -ls)

How to get an ssh root@xx.xx.xx.xx address

1. Sign up for Linode
2. Click Create+ (blue button) on Linode with the $5/mo Shared CPU plan
3. Once created, you'll find the address listed under **Public IP Addresses.**

How to Connect as a Client - Older Python Sockets Server

On the client side of https://github.com/effypelayotran/playable_phys_game_multiplayer , make the executable of client.py

pyinstaller --onefile --windowed --add-data "assets/Hyperspace.otf" client.py

or run `python3 client.py` in the terminal

Future Development Notes from 6.12.25

1. **Press `r` or a simple keyboard shortcut to Restart Game on the Server-Side.**
2. **Automatically have the game restart after 10 minutes, or after a Win/Loss state has been reached.**
3. **Add Win/Loss states.**
4. **The second, third, fourth, etc. player that joins can click those Number of Player? Growing Blackhole? Options Buttons but they currently don't do anything in terms of changing the game that the 1st player that joined and made the game did. Somehow update so that the second, third, fourth etc. player that joins can join an existing game immediately without seeing those options buttons OR start a new game.**
5. **On the same note as above, handle Multiple games on the same Server. Add 'Waiting Room' of softs.**

6. **Add RL Ship (use the get_state as the model input side, model output is either increaseThrust, turnLeft, or turnRight).**

Getting Asteroids on the Web Notes from 6.12.25 - Achieved with the Current Version!

1. If we want to get the client to connect to the pygame via a website instead of opening this python executable, you'll need to use **pygbag** to convert the client-side version of the pygame into web assembly. (Or use some other new library that can do this?) (Or potentially rewrite the game in Javascript, but this options does not make as much sense given that all the RL training is done in python, and we want to keep the RL and the game compatible.)