Lab07\_AIGame

**Fufillment**

* I finished all the AI objectives!
  + We discussed not having to make a game if I further investigated optimizations, which I did and got decent results
    - Up to 90 fps in debug (from 22) with the same number of npcs doing the same AI techniques

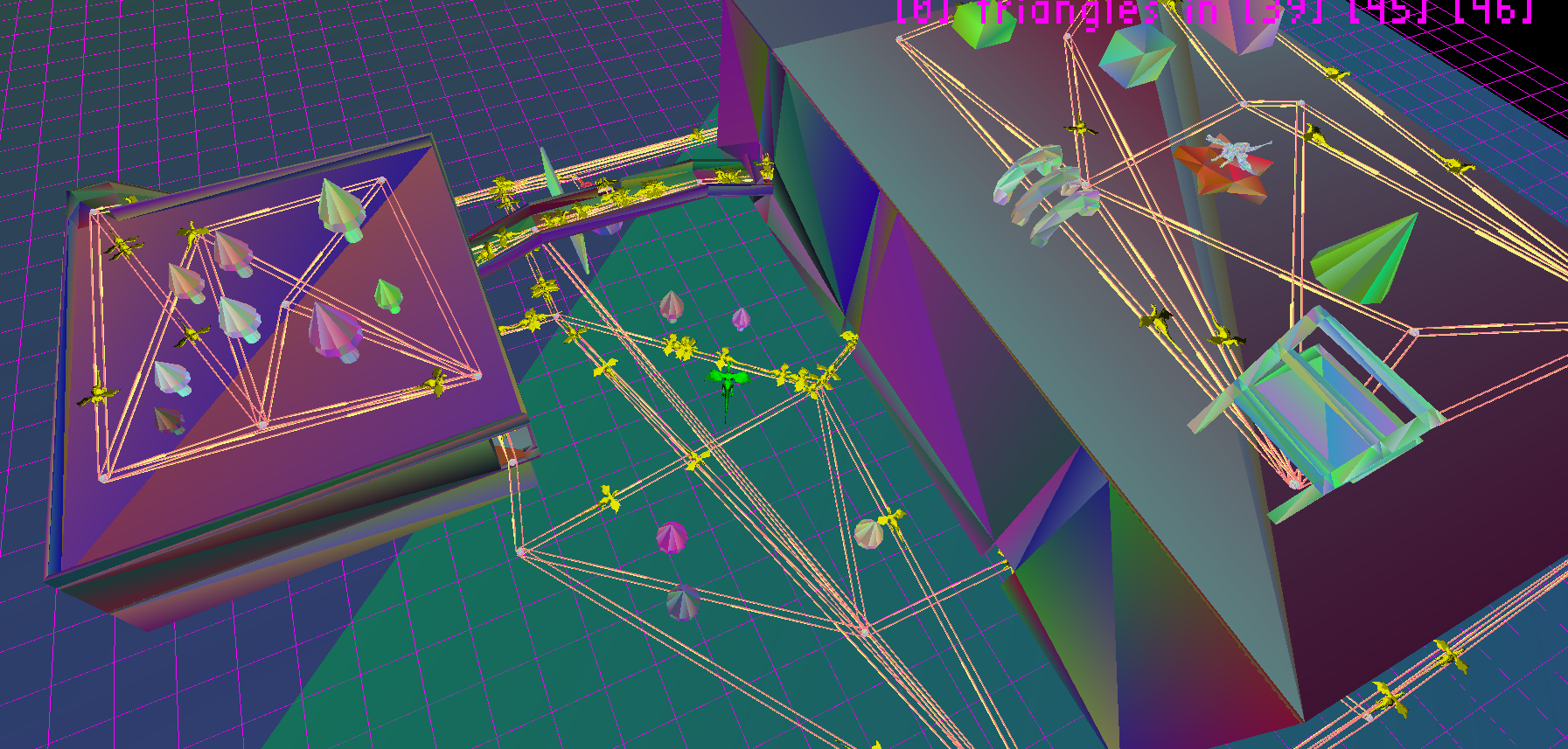
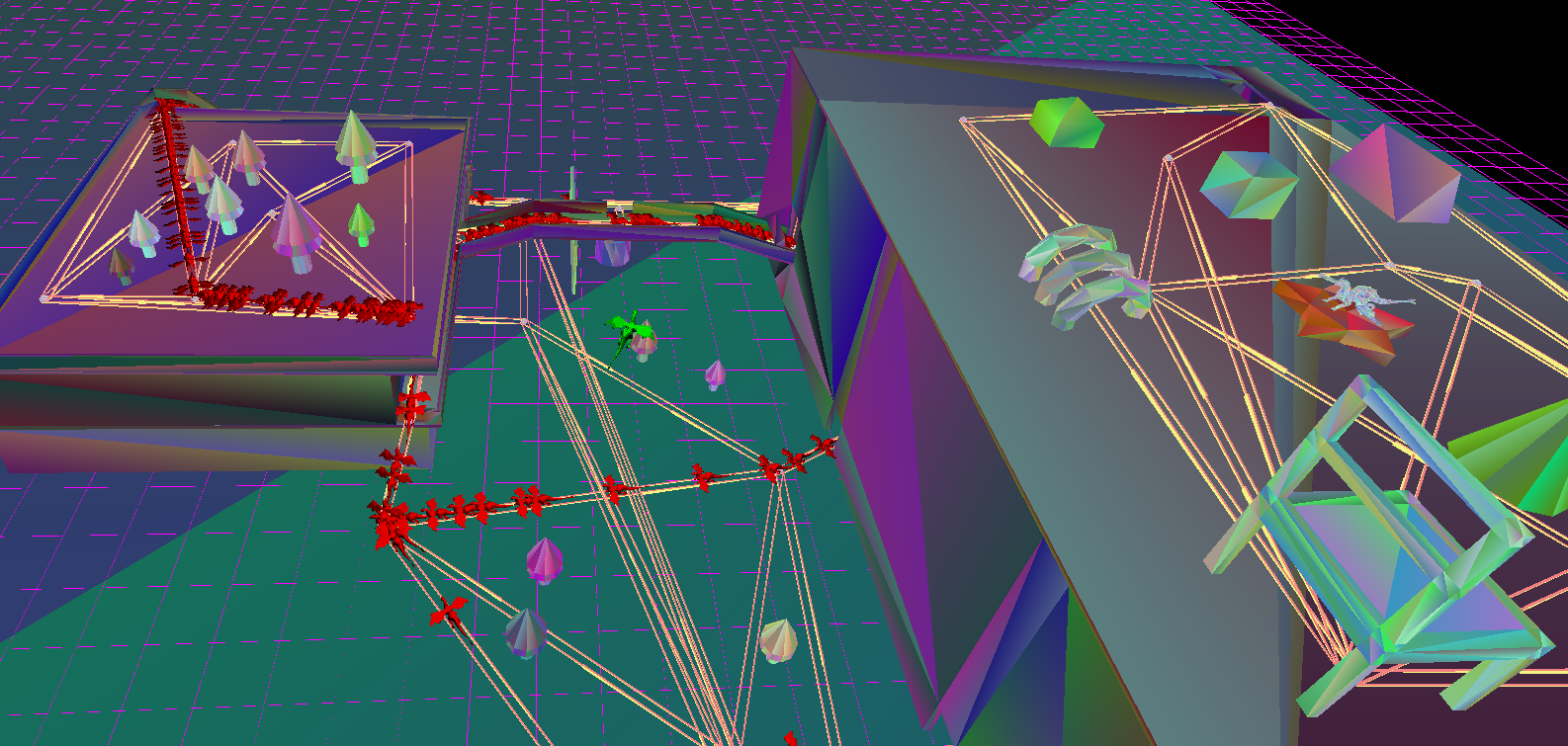
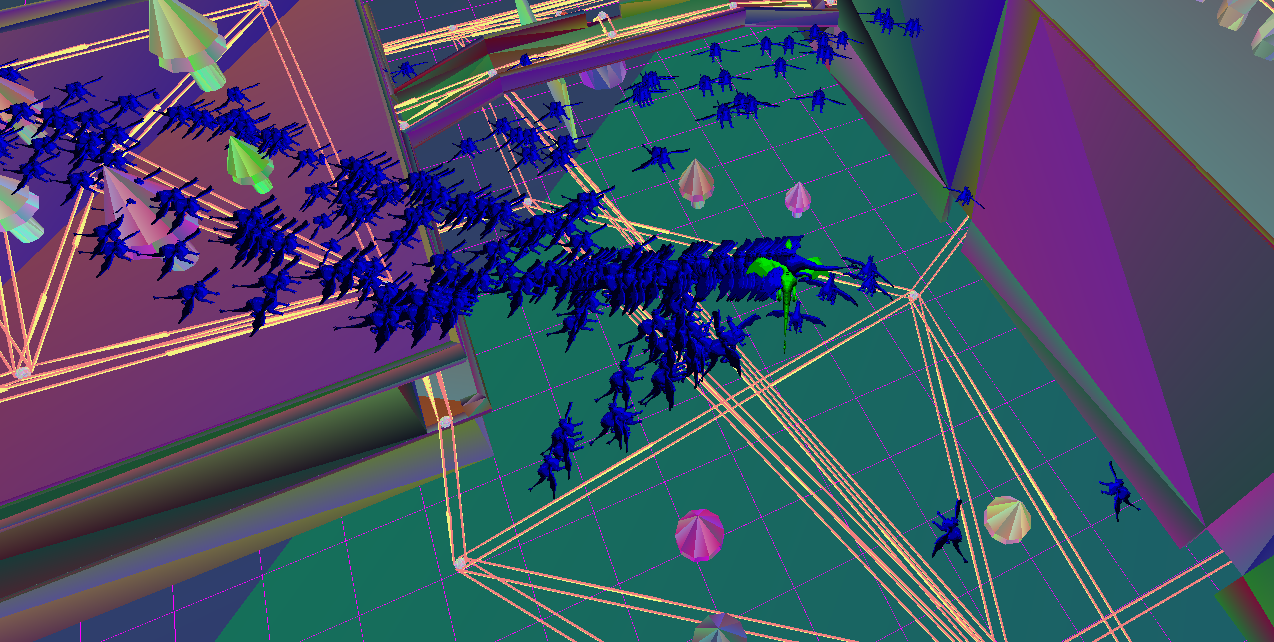
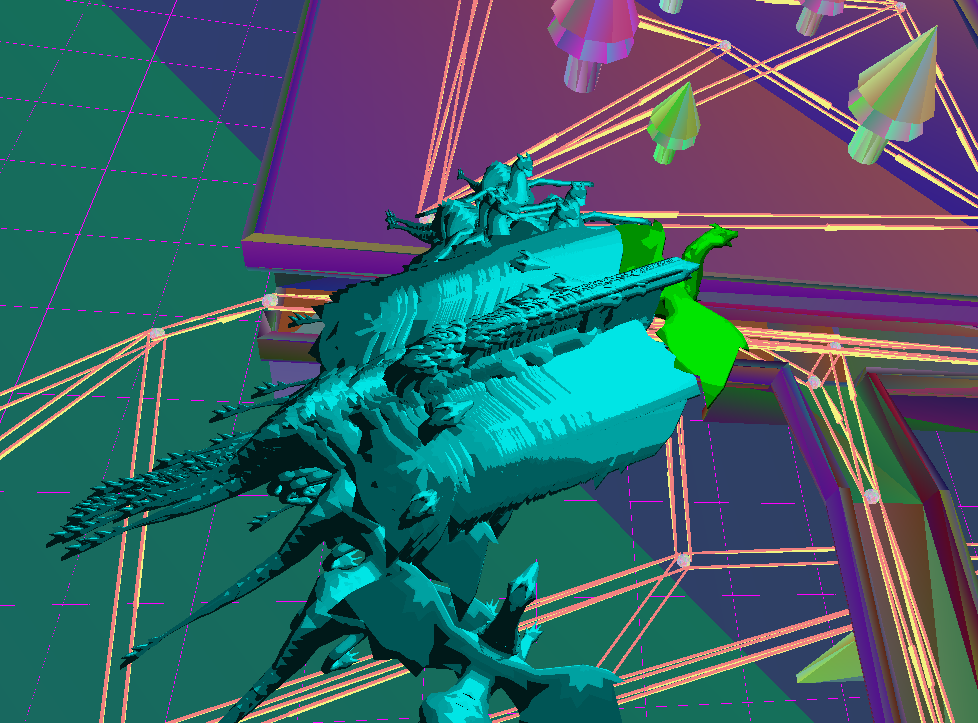
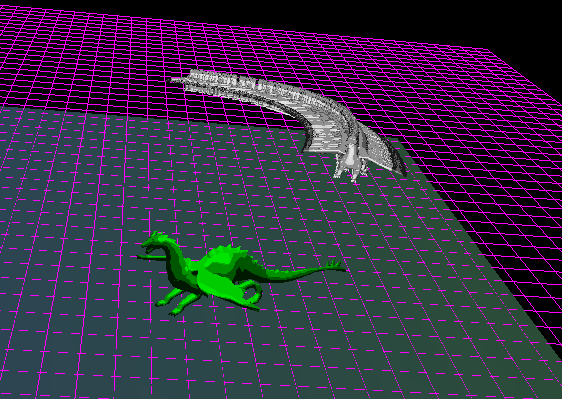
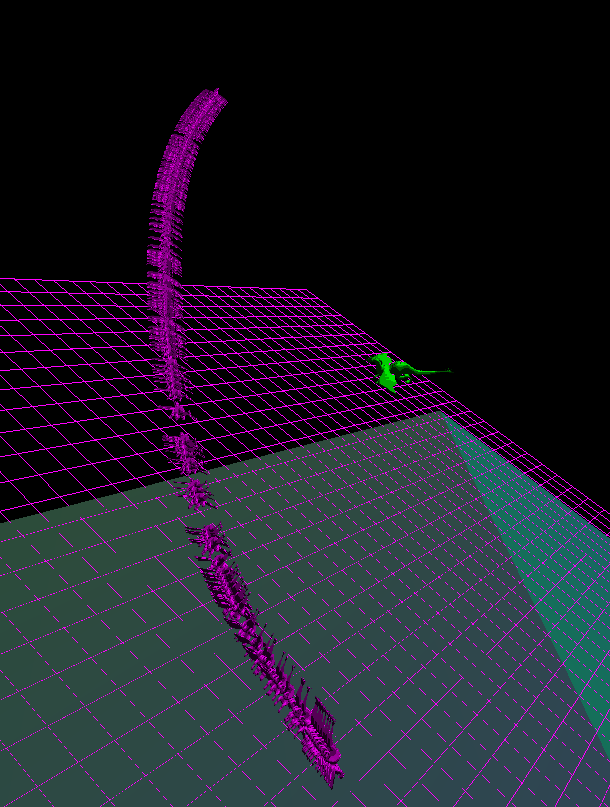
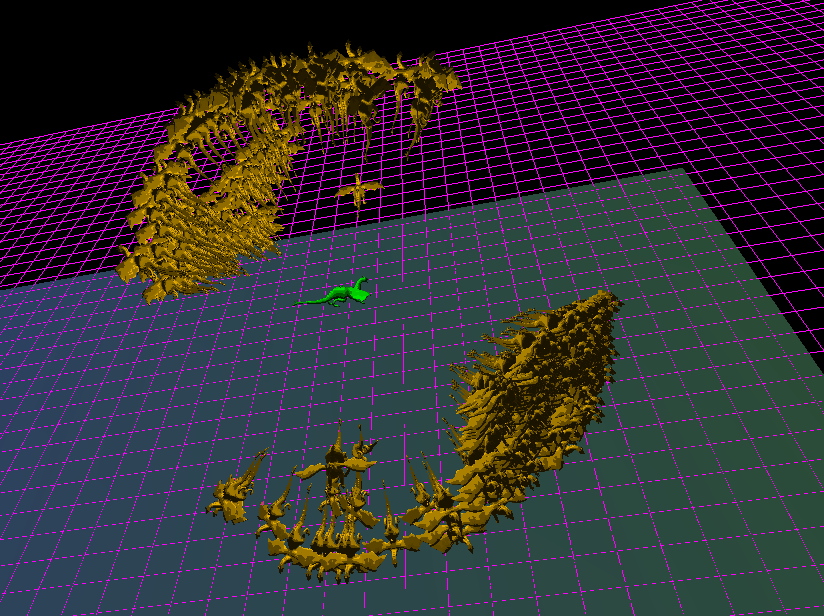
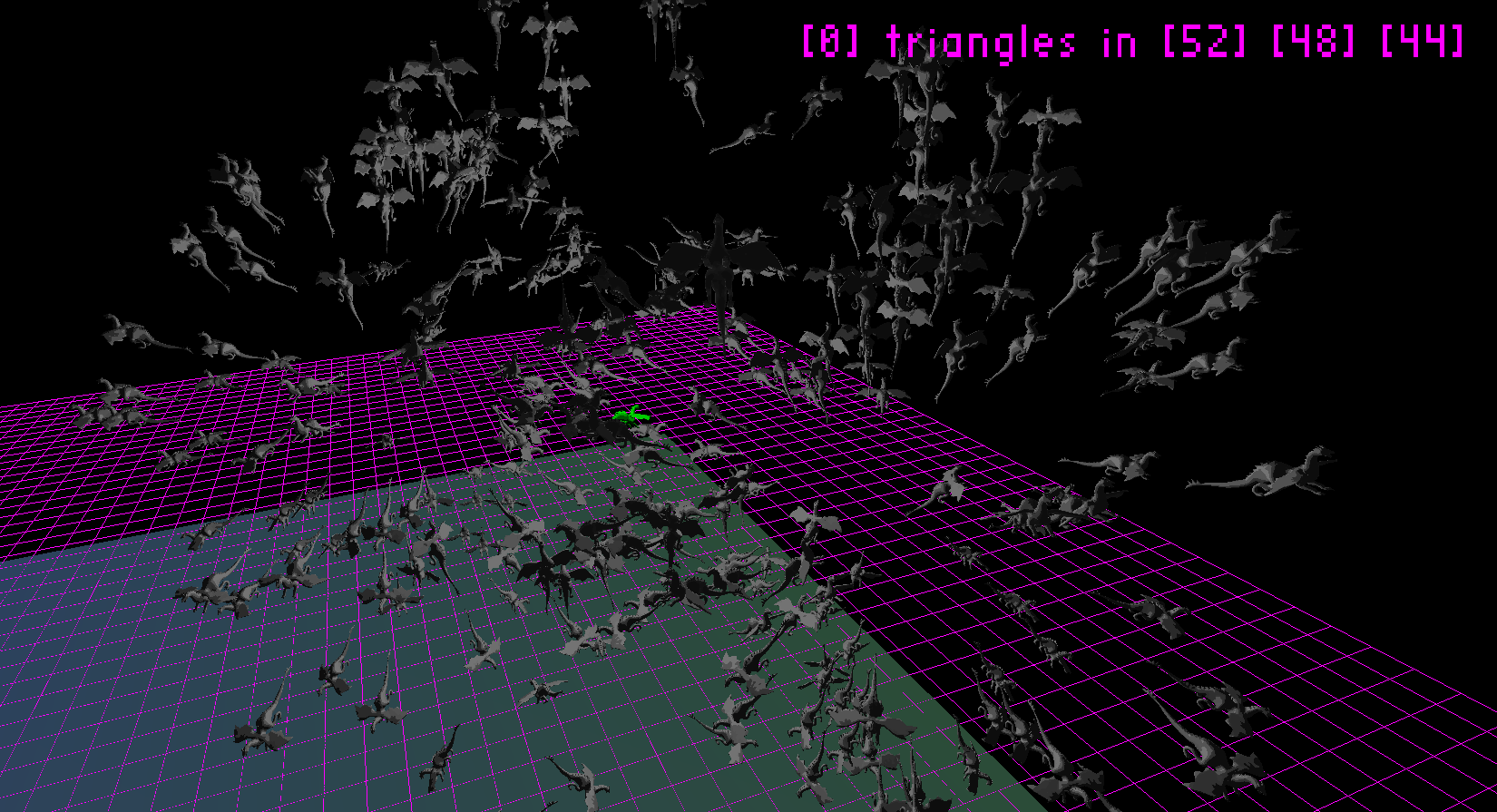
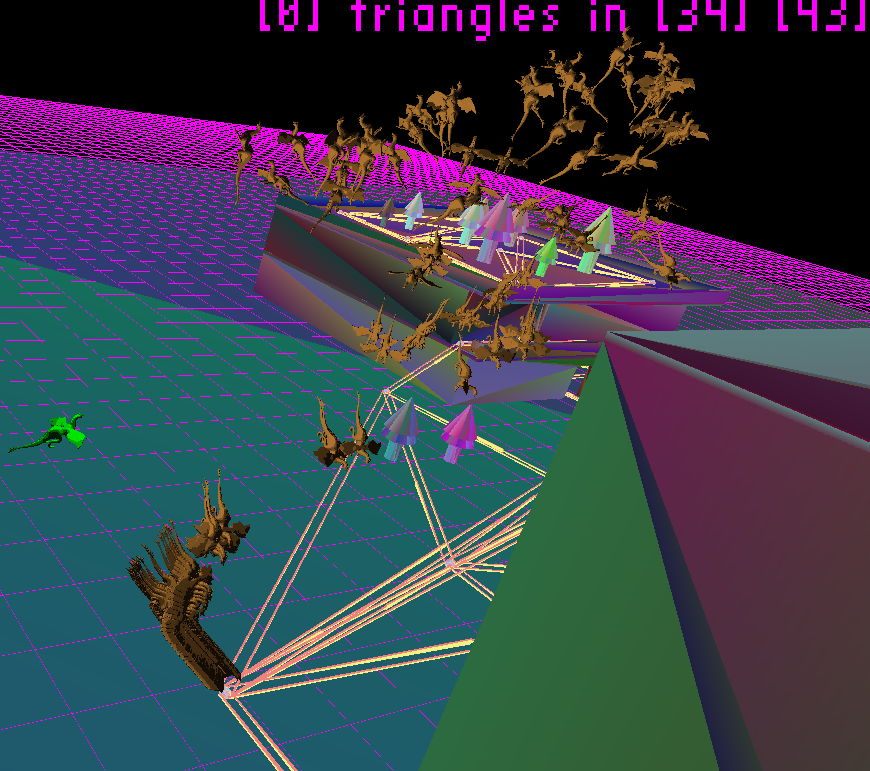
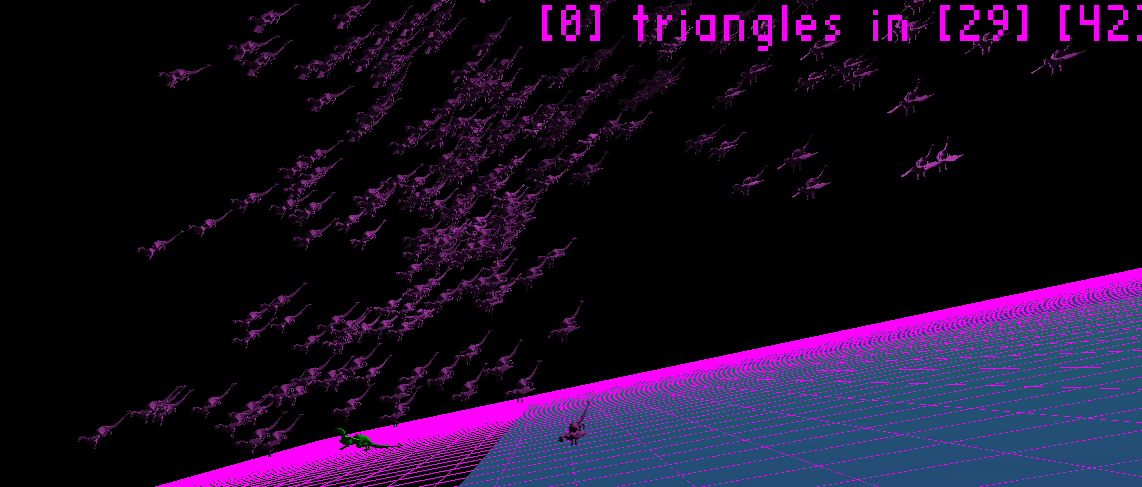
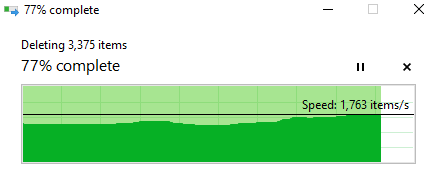
**Execution**

* I do not believe there should be any unexpected requirements for running

**Controls (Game)**

* Pressing the X key will close the application
* WASDQE Space – Same Dargon controls as the last multiple quarters…
  + W/S – Pitch
  + A/D – Yaw
  + Q/E – Roll
  + Space move forward relative to BetterDargon
  + Note controls are inverted while upside down
* Shift + 0-9 toggle layer visibility and intractability for spatial partitioning
  + Shift + U will make all layers visible
  + Shift + 2 toggle off node connections and nodes, and toggle on the hideout
  + Shift + 3 will toggle on nodes, and toggle off connections and the hideout
  + Shift + 4 will toggle on node connections, and toggle off nodes and the hideout
* K will spawn another Red Enemy Dargon if the cap of 100 has not yet been reached
  + One spawns every 0.1 seconds otherwise
* G will toggle the 3D spatial grid graphical object’s visibility
* 0-9, Y & T Will PUSH an AI state to the demo dargon’s finite state machine
  + 0 will push RandomAStar to their brain
    - Should be equivalent to previous Random A Star Lab
  + 1 Will push FollorAStar
    - Should be equivalent to “angry mode” from previous A star lab
  + 2 will push Seek
    - They will move towards the player, over shoot, and turn back repeatedly
  + 3 will push Arrive
    - They will move towards the player, until within the slowdown radius, where they will still move towards the player, slower and slower, until reaching a complete stop upon the player
  + 4 will push OffsetPursuit
    - The entities will surround the player in the shape of a larger dargon
  + 5 will push Pursue
    - The entities will go towards where the player is moving to, rather than to the player’s exact position
      * Works nicer for moving objects
  + 6 Will push Flee
    - The entities will go directly away from the player’s position
  + 7 will push Evade
    - The entities will go away from where the player is moving to, rather than from the player’s exact position
      * Works nicer for moving objects
  + 8 will push wander
    - The entities will move randomly (controlled to not be super jerky)
  + 9 will push forage
    - The entities will wander until something edible is within a distance of them, and they will pursue the closest edible thing until arriving at it and eating it
  + Y will push flock
    - The dargons will flock
  + T will push random
    - The dargons will each get one of the above pushed randomly onto their brain, so you can see them all acting differently
* V will pop the top state from the dargon’s brain
  + Whatever they are doing will be removed and whatever is under it will be done instead

**Screenshots**

* + **Random A-Star pathing Dargons. Notice they are going from node to node and equally distributed as would be indicative of random A-Star**
  + 
  + **Follow A-Star – notice how they are all taking the short path to the closest node to the player. The closest node can be seen on top of the building where some of them have stopped. The rest are going to that point.**
  + 
  + **Seek – notice how they are moving towards the player (these dargons always face the direction they are moving, an this remains true for later screenshots – a dargon can be assumed to be moving the direction it is facing (or still))**
  + 
  + **Arrive. The dargons are trying to arrive at the player’s location. Notice how they do not over-shoot the player and turn around, but rather remain facing the direction they came from.**
  + 
  + **Offset pursuit. Dargons assemble! Take form of META-DARGON! And yes, I did mean recursion! (haha programmer humor)**
  + 
  + **Pursuit. They are moving towards the position I am going to be at rather than where I am currently.** 
  + **Flee. Move directly away from the green dargon. Do not pass go. Do not collect $200. (haha, monopoly jokes)**
  + 
  + **Evasion, like flee but moving away from where I am going, not from where I am. Note that they are in the shape of a sphere because I called them towards me before making them run.**
  + 
  + **Wander. They just kinda fly around the world and into stuff (or through stuff…) Gets them nice and spread out though.**
  + 
  + **Forage! Definitely in my top three favorites for this lab (flocking, offset pursuit, forage if you’re wondering ☺). They will wander until spotting a resource. You can see this in the screenshot as the ones on top are wandering (they don’t see anything) and the ones on the bottom are arriving at a resource (they spotted the node and are hungry enough to eat anything) (run in release please)**
  + 
  + **Flocking! These dargons flock. Of all of the AI techniques we implemented, this one is probably the least impressive in screenshot form. Highly recommended to view in real time via the demo. (Run in release please, reduce constant value for more small flocks)**
  + 
  + **Log files! Over 3k! Wow! I guess its not that surprising though because this was a multi-week assignment so you should expect more than usual.**
  + 

**Post-Mortem**

* **First things first. This lab in particular looks much better in real time than through screenshots. I strongly suggest actually running the application instead of reading the document if you aren’t impressed. It looks much better in real time. (at least in my opinion)**
* **I have no idea how long I spent on this lab because it was so spread out over multiple weeks.**
* **I really enjoyed this lab – maybe it was the freedom, maybe it was the fun deviations I did that enabled me to work on other parts of my engine, maybe it was just how cool it looked.**
* **Some of the AI techniques were not so awesome in my opinion, others were way cooler than I expected.**
* **My favorite things from this lab were foraging, offset pursuit and flocking**
  + **Not only do they look the coolest, they were all easier to implement than expected**
* **This lab in particular made me optimize things in my game engine**
  + **I tried looking at many, many things – I made dozens of changes I thought would help – here are the quite interesting results**
    - **I went through all my shaders and tried to optimize them – I used the website you linked me for reference and went through all of their techniques (use mix instead of lerping yourself, replace division with multiplication and addition, etc) and found that it made a whopping zerp-FPS difference!**
      * **It was neither better nor worse. I guess the GLSL compiler is really smart and found all the things and optimized it itself**
    - **I went through my engine and removed redundant outdated code**
      * **I had hoped not adding to unneeded counters, checking for mouse clicks, etc. would increase fps**
        + **I think it may have, but not enough to be confident. In my short single test I think it went from 91->92 so I’m not willing to make any statistical claims on that (could have been random variance)**
    - **I implemented instanced rendering for my dargon npcs**
      * **This one was surprising because it DID give an increase, but a much smaller one than I expected**
        + **For 250 DARGONS, it increased fps from 83 -> 96**

**This was in release, it also increased debug fps from 22->29**

* + - * + **I, and everyone I spoke to about this, expected it to be a much larger increase. We’re all still confused. We’re pretty sure it’s not the buffer updating, but we ran out of ideas and still aren’t sure why it was such a minimal increase.**
    - **I found I was querying opengl for errors on a PER-UNIFORM-PER-OBJECT-PER-FRAME basis**
      * **So, 250 objs, 13 uniforms per obj, means 3250 error checks per frame JUST FOR THE NPCS!!!**
        + **This made the difference between 29 and 70 fps in debug**
        + **It had absolutely zero effect in release (96->96)**
        + **I guess release optimized it away for me somehow!**
    - **I found that my spatial component was creating 4 matrices and multiplying them when it could create a single matrix**
      * **This made ZERO difference in release (96fps -> 96 fps)**
      * **This made a significant difference in debug (70 fps -> 96 fps)**
        + **I guess release optimized it away for me too somehow.**
    - **After hours of optimizing, I have managed to get my debug fps to match my release fps for most AI techniques**
      * **Release still has faster startup time, and I know release still makes certain things faster (like ray-casting) but it seems that when it comes to my general game and rendering loop, there is no significant difference anymore**
        + **They are both about 96 fps now, when they used to be very different**
      * **UPDATE: After more thorough testing, I have noticed that everything seems to perform nearly equal in release and debug EXCEPT flocking and foraging! Flocking and foraging need to be run in release to get hundreds of participants at a reasonable frame rate!**
  + **As a different comparison point, across all the updates my fullscreen fps went from 55 fps to 70 fps**
    - **Fullscreen is slower because fragment shaders**