

# Interim Demo

Anil Celik Maral

Erick Lorenzen Meneses

Matija Jajcinovic

Lorenzo Russo da Costa Auer



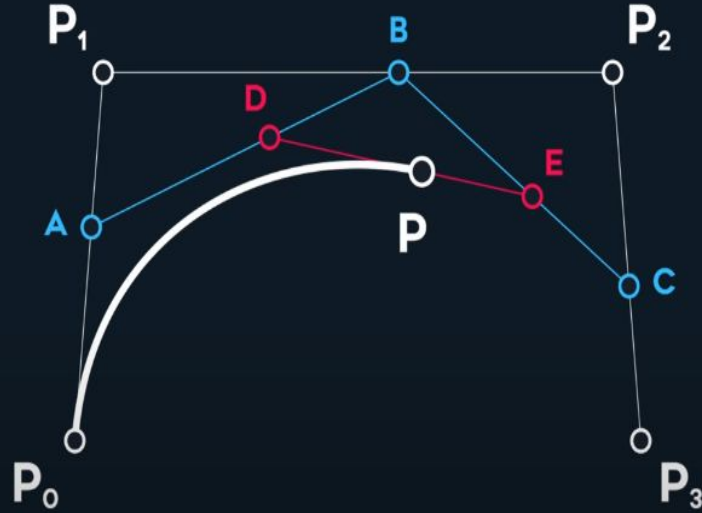
# Roller Coaster Movement - Ideas

1. Move the whole level on a roller coaster track, to get as realistic movement as possible
  - easier to design with visual feedback
  - closer to real physics
  - can be buggy or movement can be too boring
  
2. Apply force to all objects in the level at specific moments
  - easier to implement
  - more control over what happens
  - harder to design levels



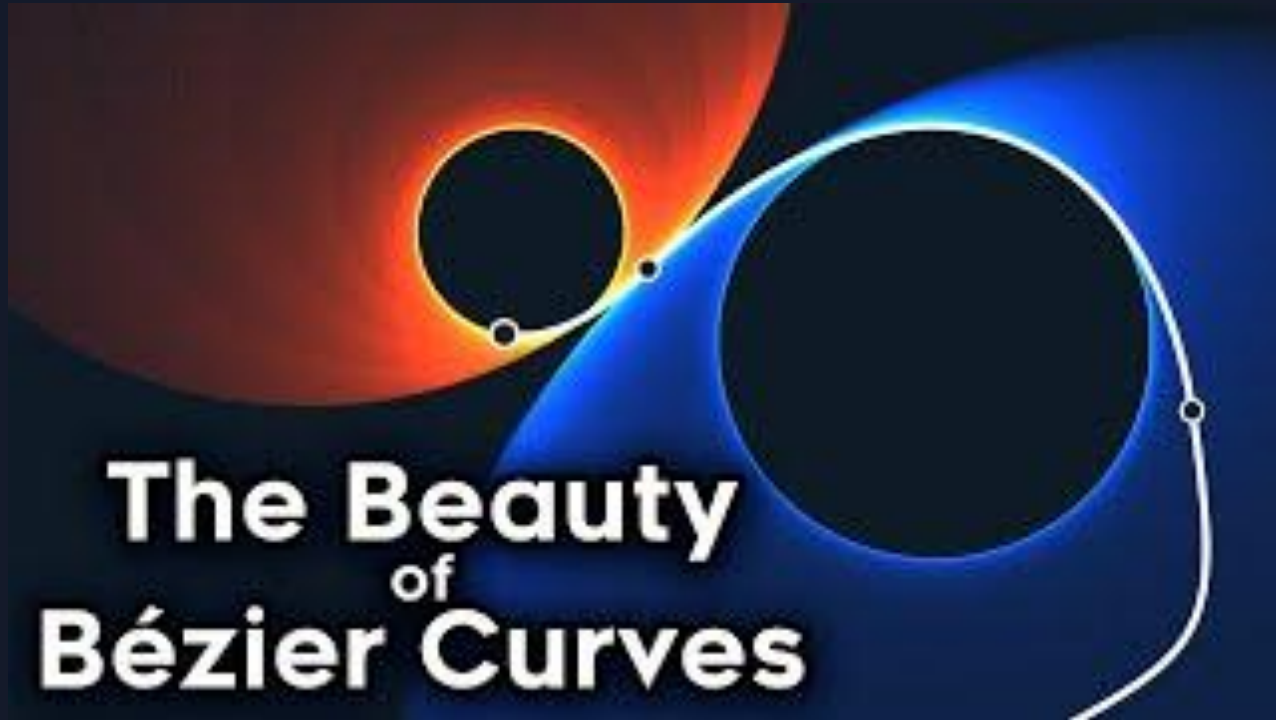
# Roller Coaster Movement - Bezier Curves

```
A = lerp( P0, P1, t )  
B = lerp( P1, P2, t )  
C = lerp( P2, P3, t )  
D = lerp( A, B, t )  
E = lerp( B, C, t )  
P = lerp( D, E, t )
```



Picture 1

# Roller Coaster Movement - Bezier Curves

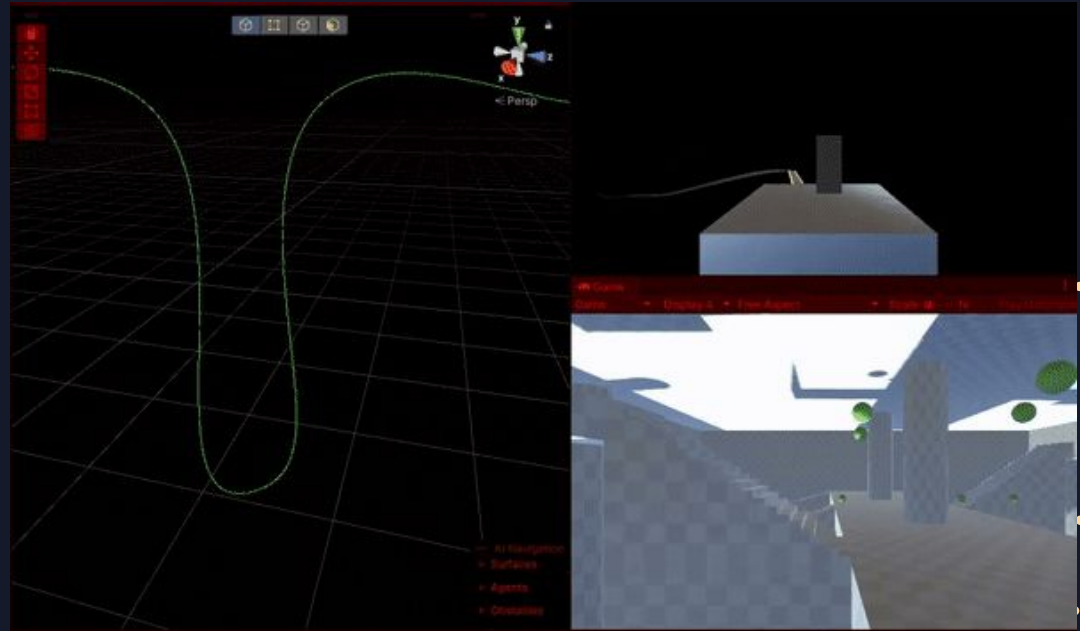


Video 1



# Roller Coaster Movement - Implementation

- Spline with tangent and normals for rotation
- Follower that moves along the path
- Level attached to the Follower



Video 2

# Level Design - Tools

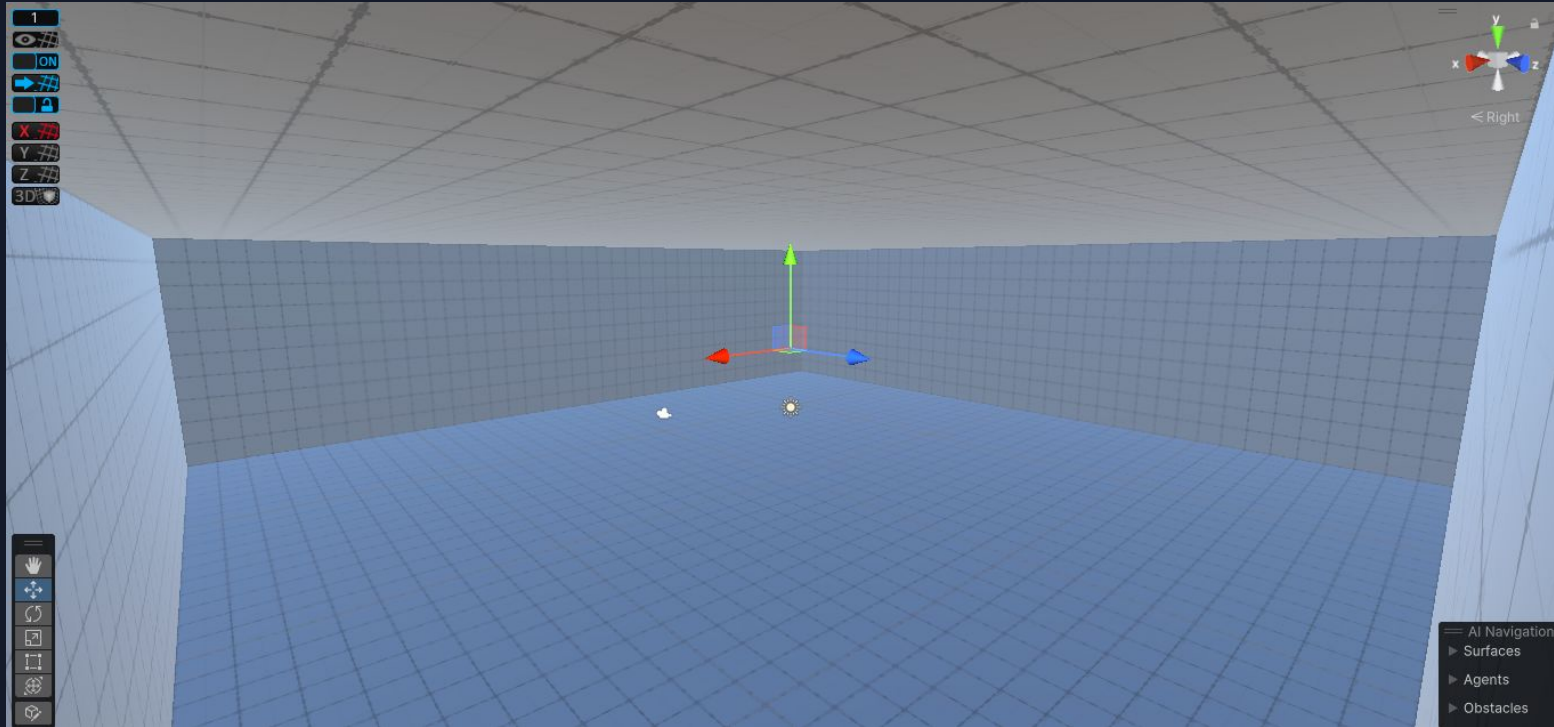
- ProBuilder
- PolyBrush
- ProGrids
- Blender



# Level Design – Challenges

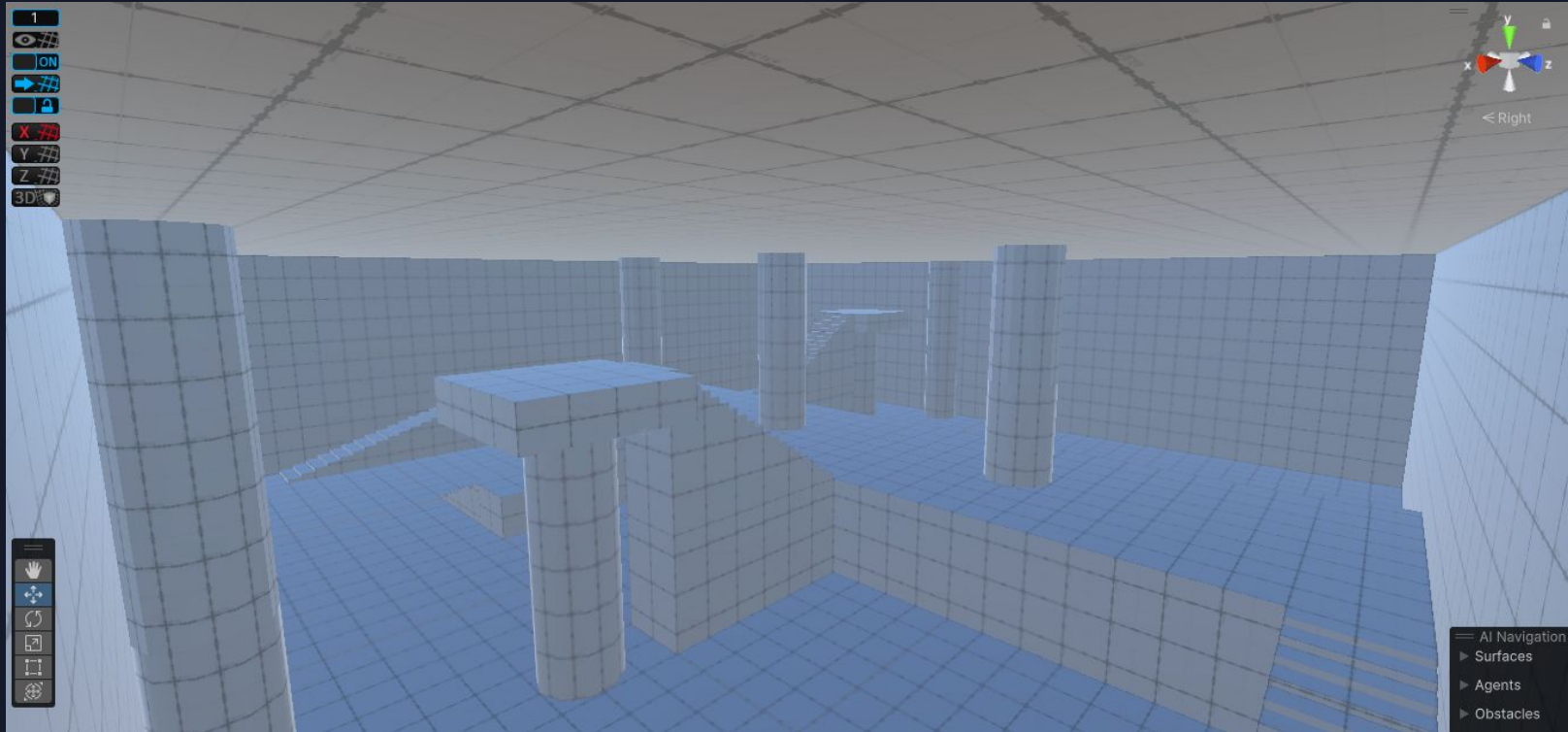
- Incompatibility between objects created using base Unity tools vs. objects created using ProBuilder
- ProGrids is no longer officially supported
- Precise and detailed structures and objects require a lot of hacks / steep learning curve
- Poor documentation for ProBuilder and PolyBrush
- Certain level of detail is impossible to achieve compared to tools such as Blender

# Level Design – Simple Level






# Level Design – Complex Level

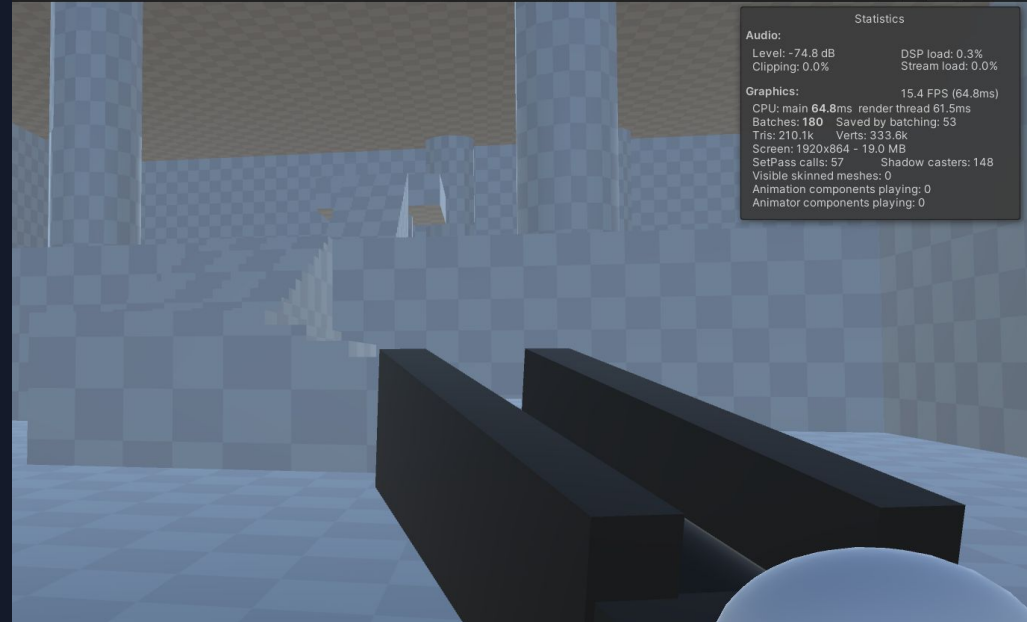


# Level Design – What's to Come

- Due to difficulties of using ProBuilder, use it mostly for rapid prototyping
  - Use Blender to flesh out the details and create complex objects such as chandeliers etc.
  - Enhance the level by incorporating moveable objects with rigidbody physics
  - Add the materials, textures etc. to polish the levels
- 

# Player Movement

- Physically Based
  - Natural fit for the game
- Basic Movement
  - Up, down, left, right
  - Jump
  - Camera control
  - Shooting
- Important To Get It Just Right
  - Highly chaotic environment
  - Easy to get frustrated



# Player Movement: Challenges

- Traditional Controllers do not work well
  - Assume a static world
  - Down direction remains constant
- Values need to be fine-tuned
  - Air mobility
  - max speed
  - drag, friction
- Remains difficult to control well
  - Jumping
  - Level movement difficult to determine



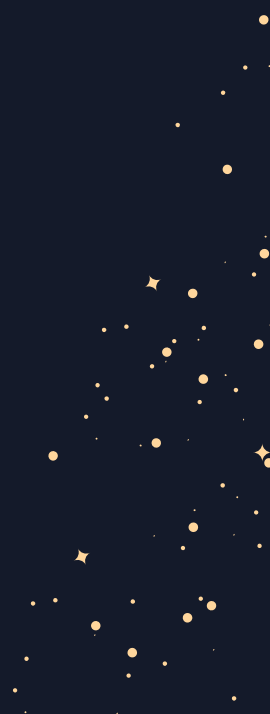
# Guns and Gunplay

## Main Weapon: Fabricator Rifle

- Shoots a high-speed projectile
- Capable of knocking enemies away
- Can be charged for more powerful shots
- High player knockback -> High mobility

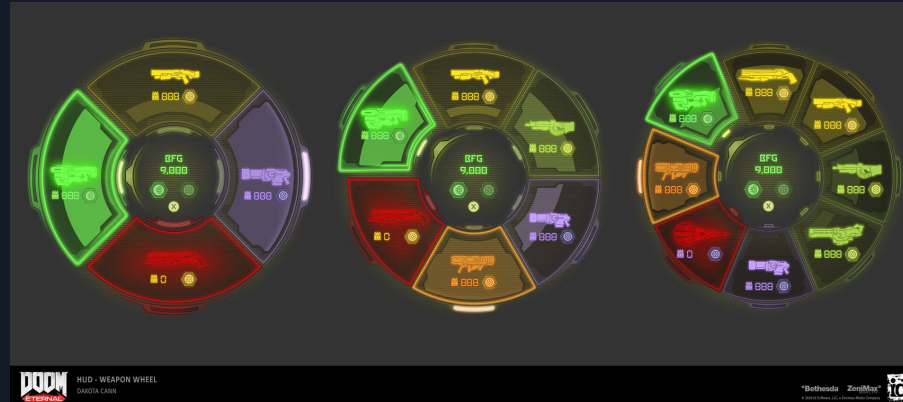
## Secondary Weapon: Pulse Shot

- Burst of 5 projectiles
- Attach to enemies or objects
- Explode after a short delay
- High damage potential



# Guns and Gunplay: Future Work

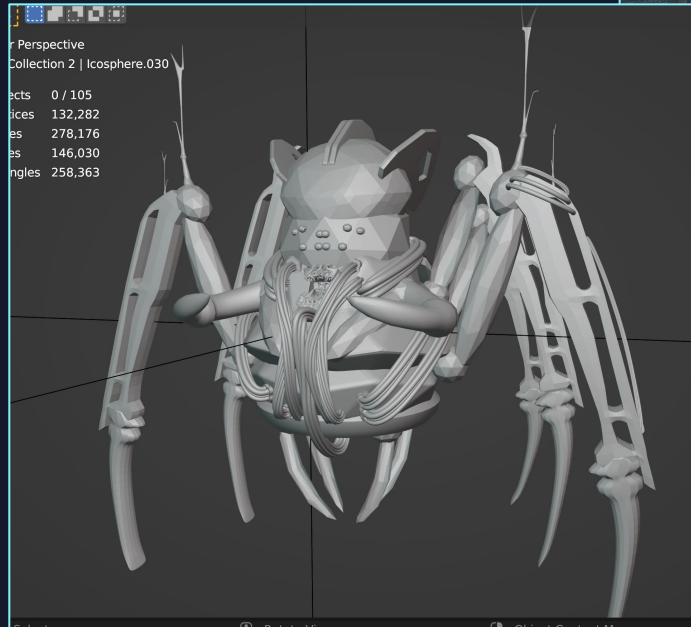
- Easy to add more weapons from a technical perspective
  - But each weapon requires a model and animations
  - Solution: one weapon model with different shooting modes
- Need to add a way to change weapons
- Main weapon does not need ammo
- Ammo for other weapons appears around the level
  - Spawns at the start of a wave
  - Encourage player to move around and collect it



Picture 2

# Enemies

- Starting easy:
  - 6 legs for easier movement
  - more stable to Roller Coaster disturbance
  - easier to model
- selfmade in blender
- 48 DoF



# Training the walking:

1. Add joints to blender model in Unity
2. Compile the game
3. Use ML-Agents to train the enemy to walk
4. Fix bugs and update hyper parameters
5. rinse repeat from step 2 until you have a good result





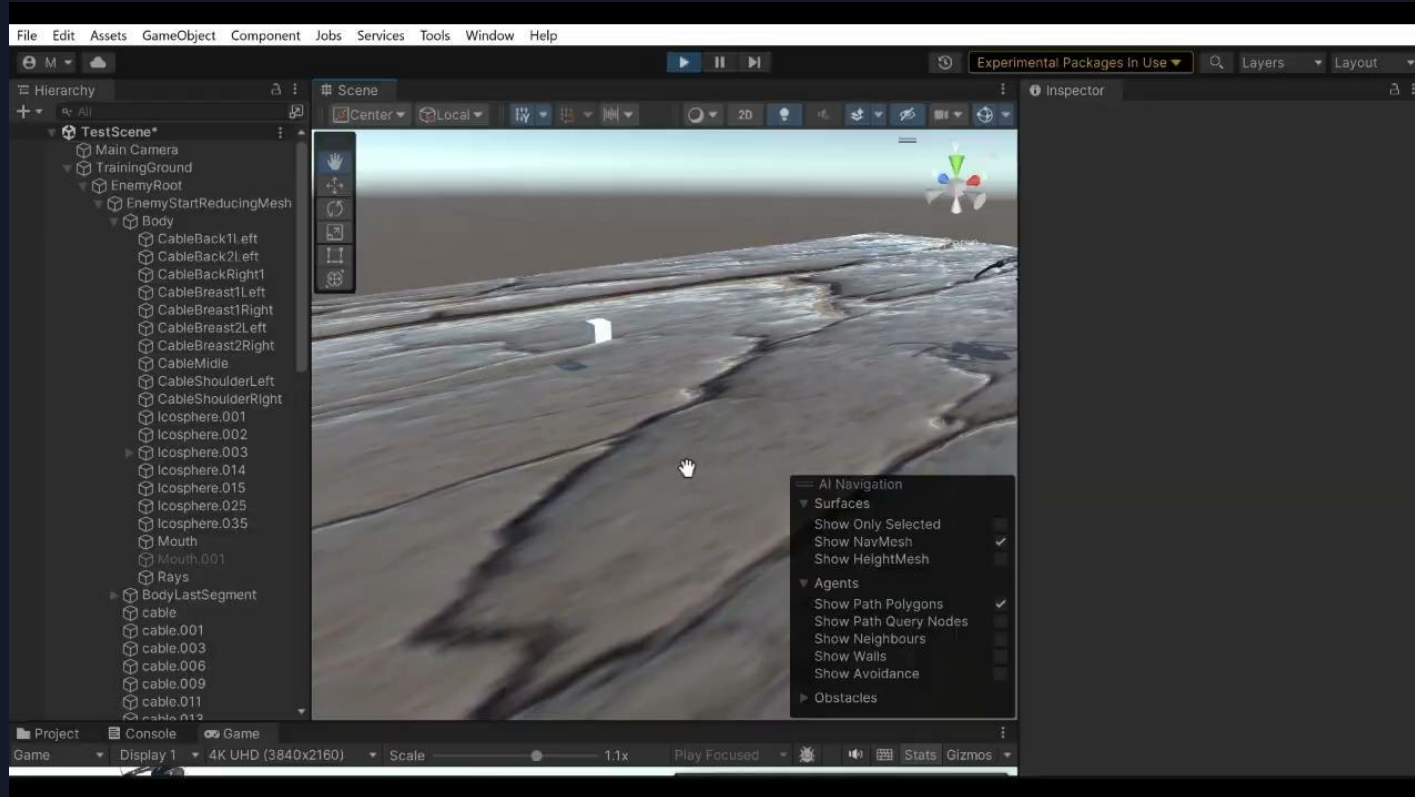
# Training Steps and Challenges

Some examples of failed training:

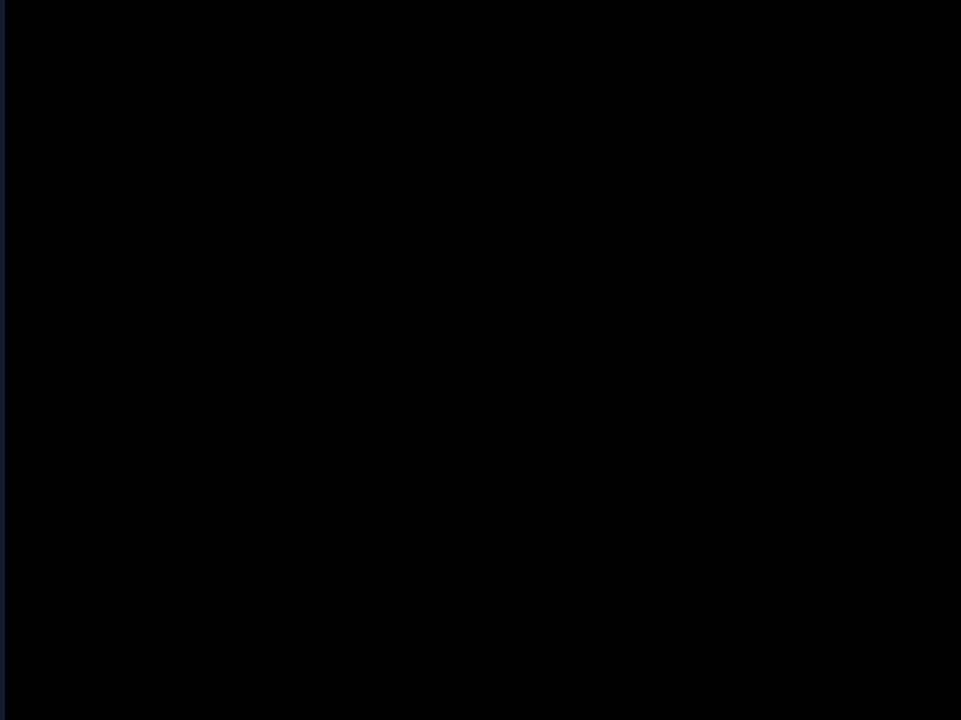
- Setting the touch Ground penalty too low makes the Spider rotate on its back and walk this way due to being more stable with this walking technique, while the gaze is still directed to the goal.
- Setting the Ground penalty too high made spider jump of the cliff/plane all the time due to the fall off penalty being too low.
- When accidentally setting the wrong forward vector it learnt to walk backwards
- When setting the reward to be the inverse distance to the goal (1 for being very close), the spider moved as near as it could to the goal but did not touch it to maximize the points. Setting the Reward of reaching the target to `Maxsteps - CurrentStep` solved that issue.

=> Ideas for implementing such walking behaviours to catch player off guards

# Video showing how the Agent optimizes in unexpected ways

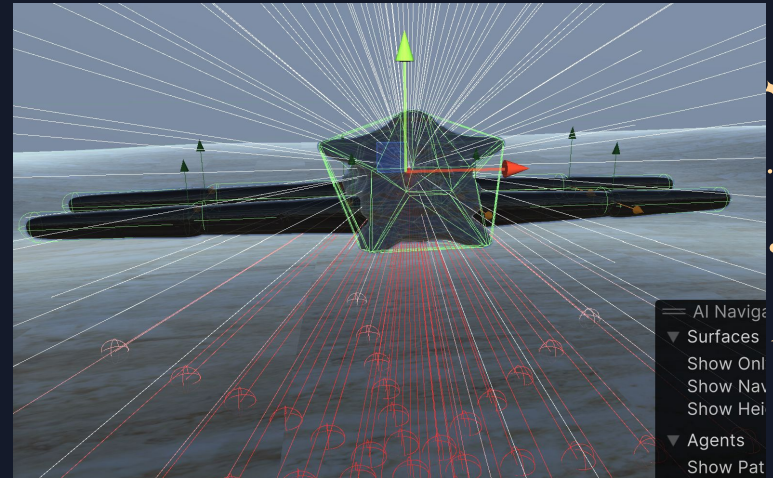
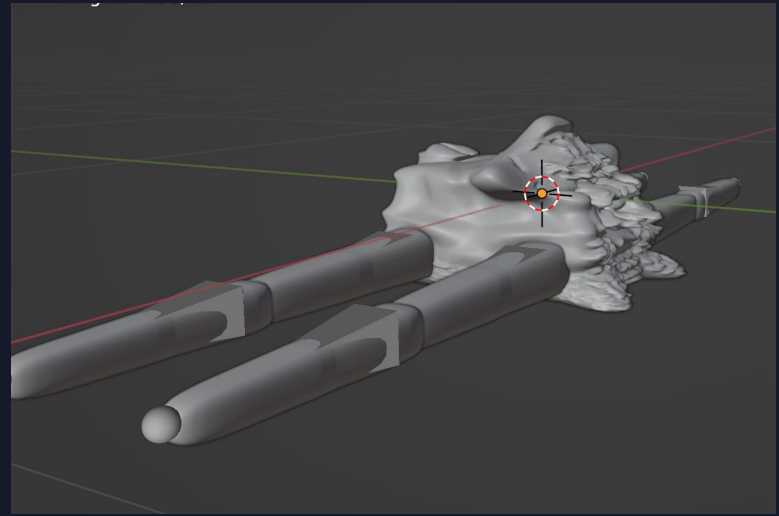


# Second Spider optimizing for stability, not reaching goal



# Enemies 2


- Starting even easier:
  - 4 legs for easier movement
- selfmade in blender
- 16 DoF
- Armature for nicer visualization



## What failed

- Make it work moving in complicated map
- Make it move fast and action packed

## Solution

- using Nav-Mesh or other classical pathfinding
  - Adding jump or boost to the Agent to use
  - Putting canons on spiders to be dangerous from the distance
  - making flying enemies
- 

DEMO



Thank you for your attention!

GG



# Citations and Sources

- Gladiator heads: <https://slidesgo.com/theme/league-of-champions-business-meeting#search-league+of+champions&position-1&results-1&rs=search>
- Slides Style: <https://slidesgo.com/theme/outer-space-and-galaxies#search-space&position-6&results-114&rs=search>
- Picture 1: Screenshot from Video: <https://www.youtube.com/watch?v=aVwxzDHniEw> at minute: 4:16
- Picture 2: <https://www.artstation.com/artwork/Pe8VJZ>
- Video 1: <https://www.youtube.com/watch?v=aVwxzDHniEw>
- Video 2: Screen Recording from our development