Parkour by Intelligent Agents in Synthetic 3D Environment

Team Members:

Manuj Babu Karki	**(THA075BCT028)**
Prasaya Acharya	**(THA075BCT032)**
Suprim Shrestha	**(THA075BCT045)**
Suven Pandey	**(THA075BCT046)**

Supervisor:

Er. Dinesh Baniya Kshatri

Department of Electronics and Computer Engineering

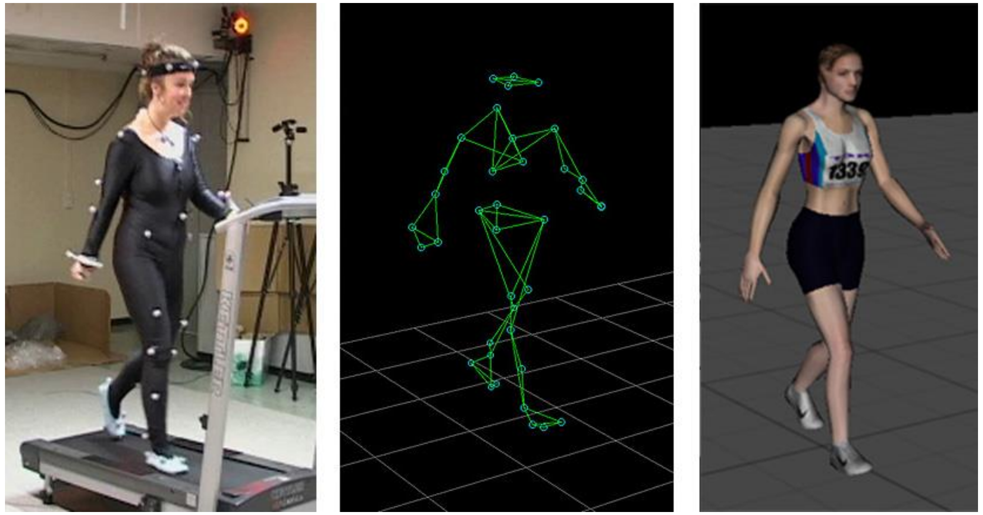
IOE Thapathali Campus

03/11/2023

Presentation Outline

Motivation

- Simulation better than repeatedly changing and experimenting in hardware
- Simulate robot movements before deployment in real world
- Alternative to motion capture for safety of stuntmen



Introduction

- Complex 3D scenes get created from a partial set of 2D images
- Obstacles are introduced in the scene to create a parkour course
- Artificial agents receive training to perform different actions on the environment such as walking and running
- Multiple agents placed in the same environment to navigate their own obstacle course

Problem Statement & Objectives

- Problem Statement
  - Animation requires painstaking manual labor
  - High risk to stuntmen and equipment alike during motion capture

- Objectives:
  - To synthesize a three-dimensional environment for intelligent agents to train in
  - To teach intelligent agents to complete an obstacle course via reinforcement and imitation learning.

Scope of Project

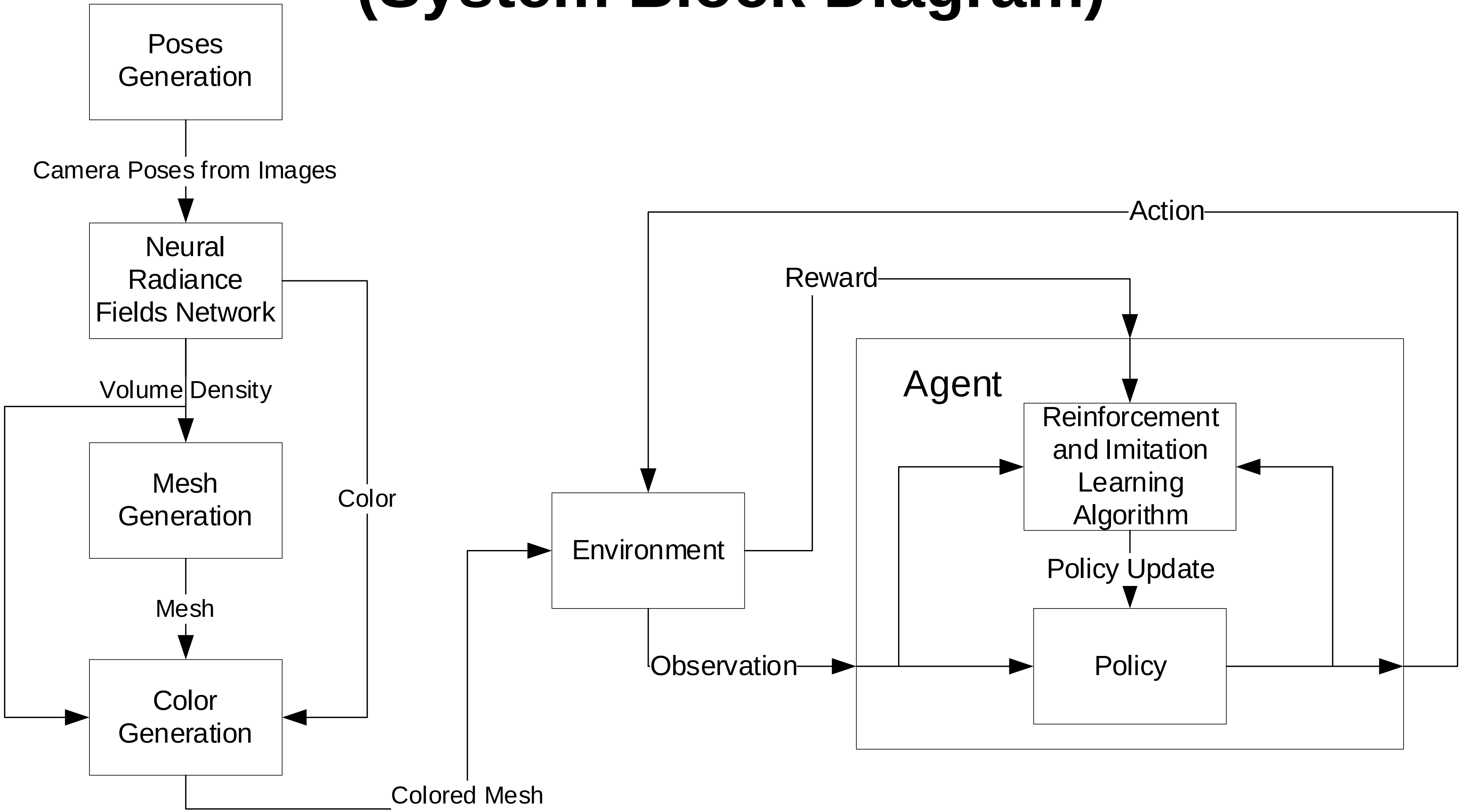
- Capabilities of project:
  - Generation of synthetic three-dimensional environment
  - Basic movements by agent like walking and running to avoid obstacles

- Limitations of project:
  - No sound and weather/special effects for the environment
  - Cannot learn natural movements for new skills without reference motion
  - Cannot perform advanced acrobatics to avoid obstacles

Project Applications

- Games
  - Create realistic character movement and motion
- TV and Cinema
  - Create realistic action sequences and human interactions
  - Minimize the physical risks involved with stuntmen
- Robotics
  - Accelerate the development of robots with more realistic movements
- Mixed Reality, Virtual Reality and Augmented Reality
  - Generate realistic scenes based on real world sites

Methodology
(System Block Diagram)

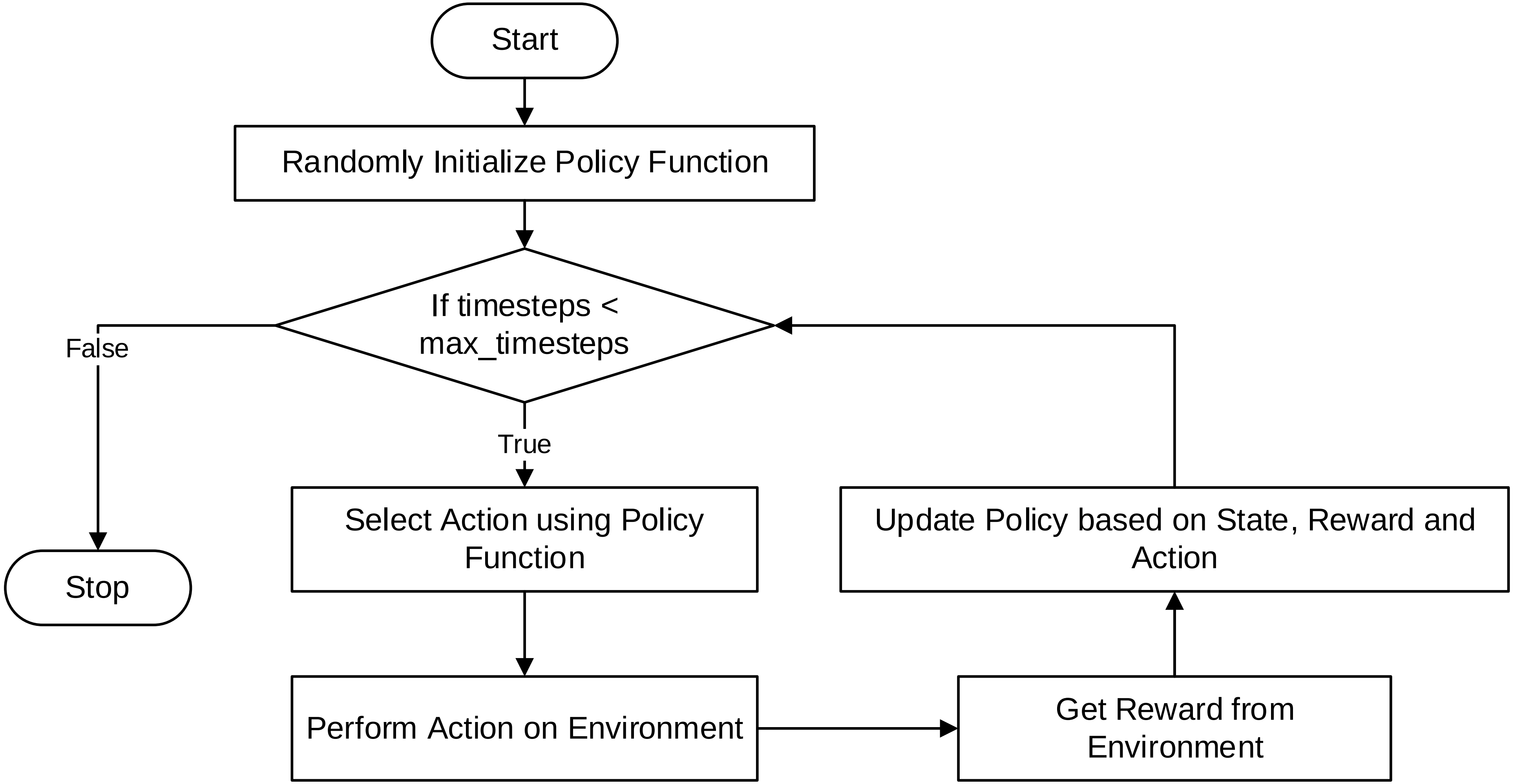


Methodology
(Reinforcement Learning)

- Agents achieve a goal in an environment through the notion of rewards and punishments

- Common terms:
  - **State**: Complete description of the world
  - **Observation**: Partial description of a state
  - **Action space**: Set of all valid actions in the given environment
  - **Policy**: Rule used by the agent to decide what action to take
  - **Trajectory**: Sequence of states and actions
  - **Value**: Expected return given a state-action pair under a given policy

Methodology
(Policy based Learning)



Methodology
(Proximal Policy Optimization)

- For long running episodes, a discount factor is used to make recent rewards less significant.

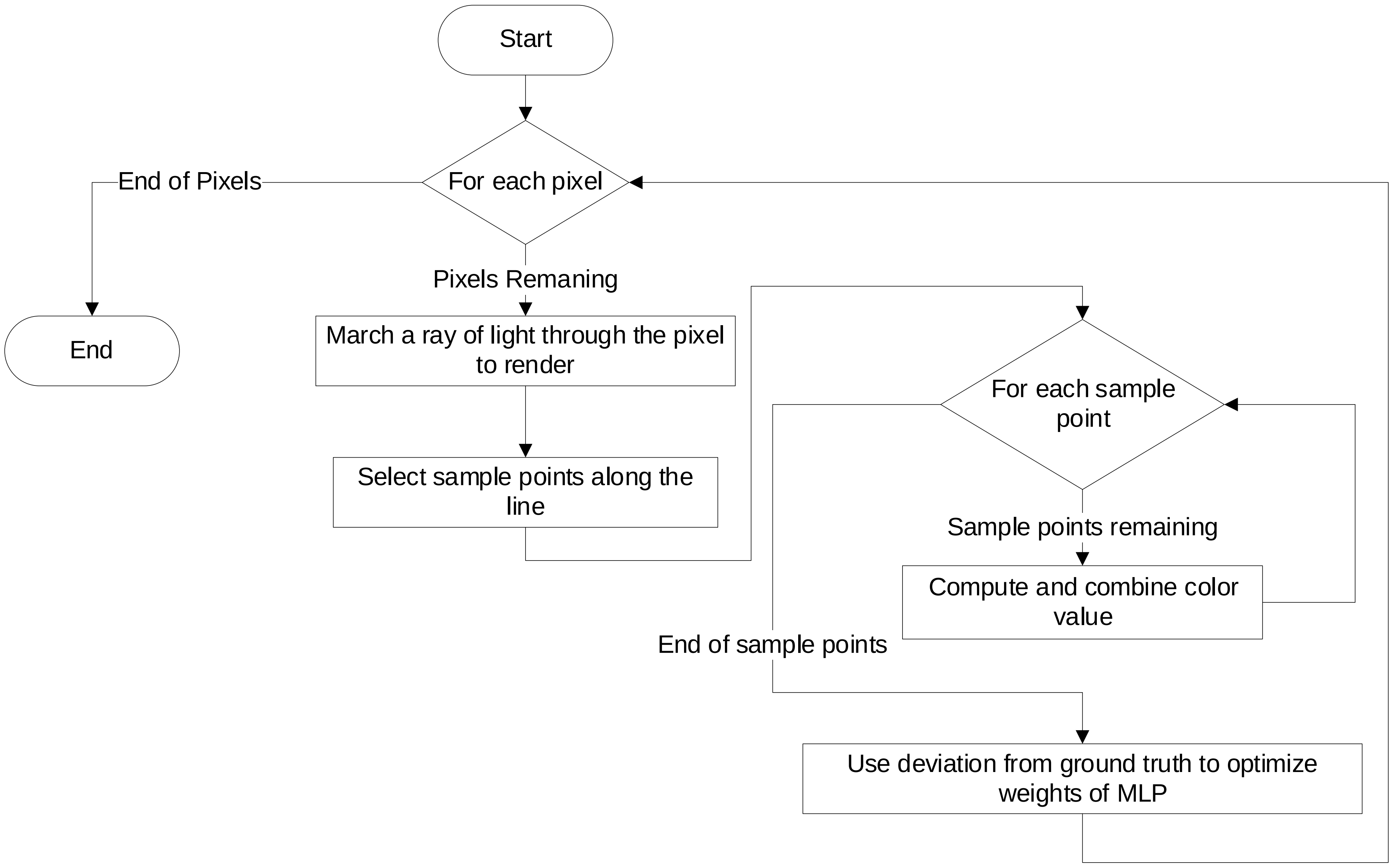
- PPO ensures that the updated policy is not too different from old policy.

- If the probability ratio between the new policy and old policy falls out of a certain range, the advantage function will be clipped.
    - Advantage function is a measure of how much is a certain action a good or bad decision given a certain state

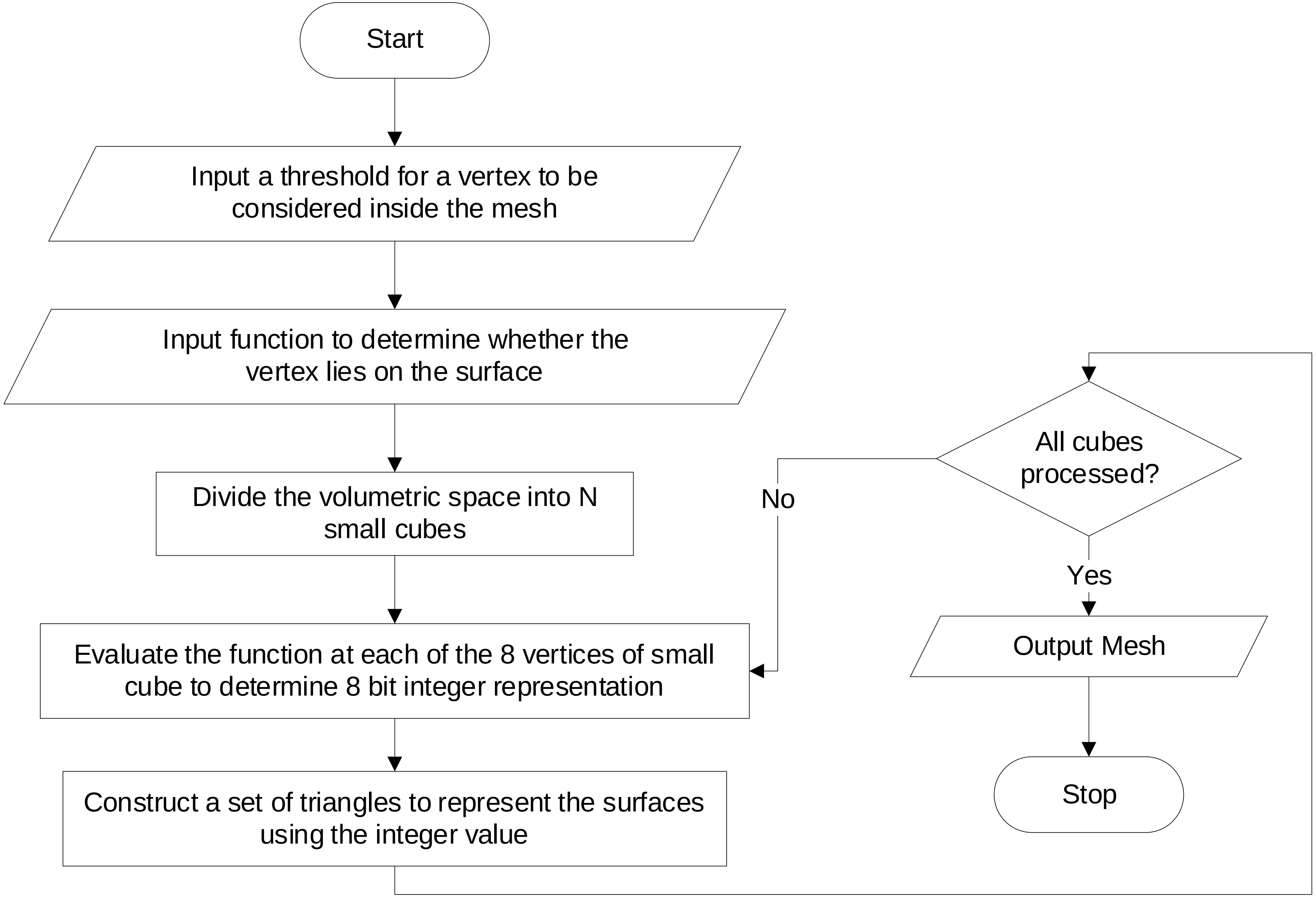
Methodology
(Imitation Learning)

- Agent imitates actions shown by reference trajectory
- Reference trajectory provided by MoCapAct dataset
  - MoCapAct includes processed trajectory from CMU MoCap Dataset
- Uses CoMic reward function for optimization
- Use of MoCapAct speeds up training time by bypassing training of low-level motions
- Can be used to synthesize higher level policies by using lower level motions

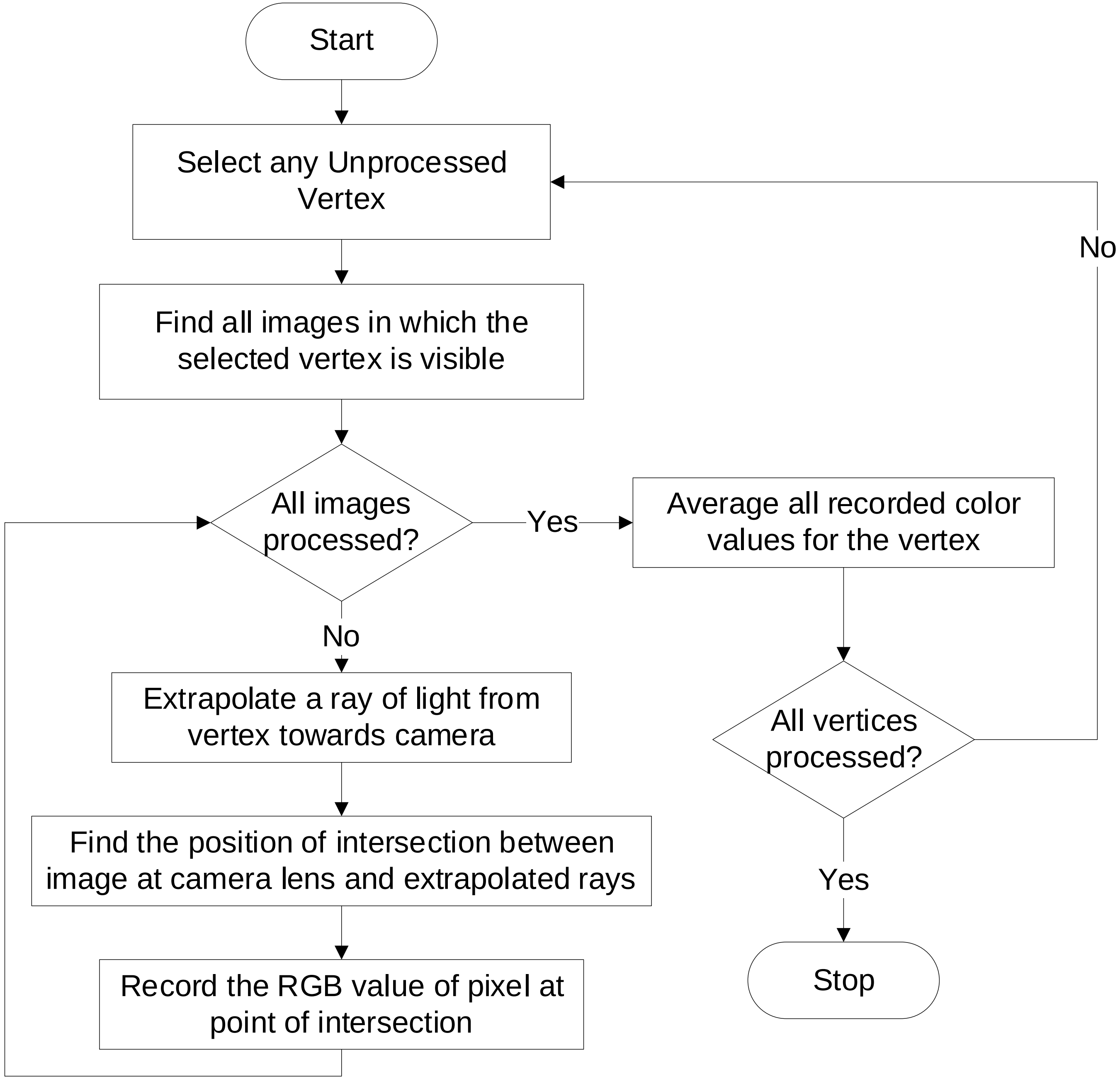
Methodology (Neural Radiance Fields)



Methodology (Marching Cubes Algorithm)



Methodology (Extraction of Color on Mesh)



Methodology
(Instrumentation)

- Hardware Used:
  - Cloud TPU v3
    - 335 GB of RAM
    - Intel(R) Xeon(R) CPU @ 2.00GHz with 96 cores
  - Nvidia Tesla T4 GPU
    - 12.7 GB of RAM
    - Dual Core Intel(R) Xeon(R) CPU @ 2.20GHz
- Software Used:
  - COLMAP to get camera poses from 2D images
  - JAX for hardware accelerated numerical processing
  - Flax for neural network models and utilities
  - Brax and MuJoCo for hardware accelerated physics simulation

Methodology
(Physics Engine Parameters)

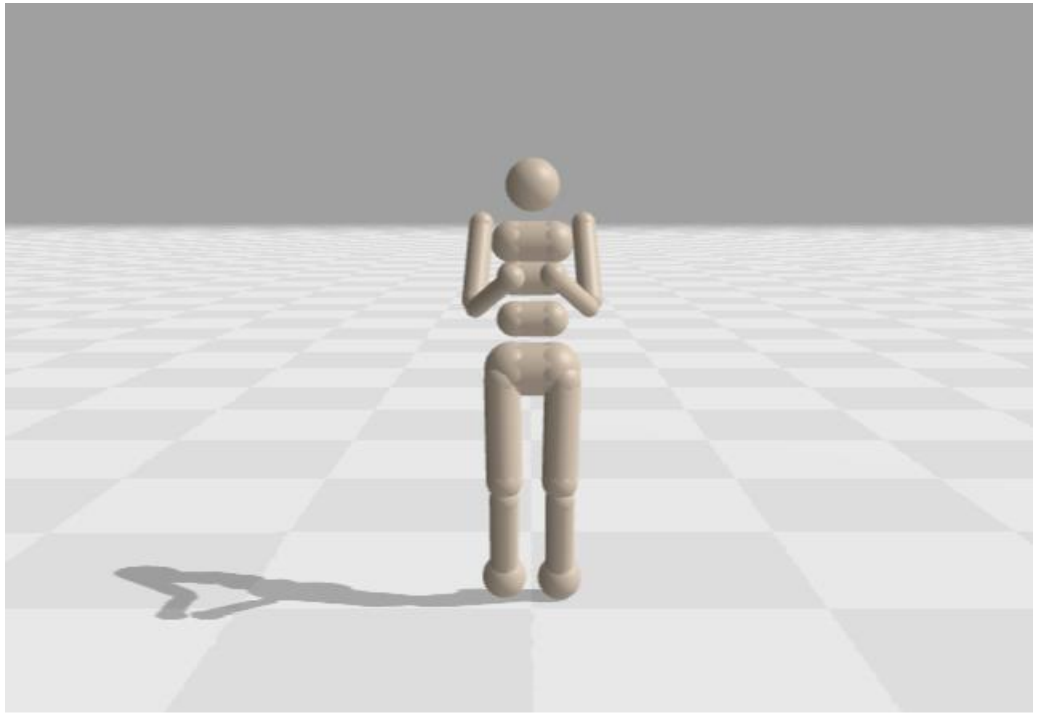
Brax Environment Parameters

S.N.	Parameter Name	Value	Remarks
1.	Friction	1.0	Friction between all bodies
2.	Gravity	{z: -9.81}	Gravity acts towards negative z-axis
3.	dt	0.01	Wall clock time corresponding to one timestep
4.	Sub-steps	8	Number of sub-steps in each timesteps for numerical stability
5.	Elasticity	0.1	Default bounciness for objects

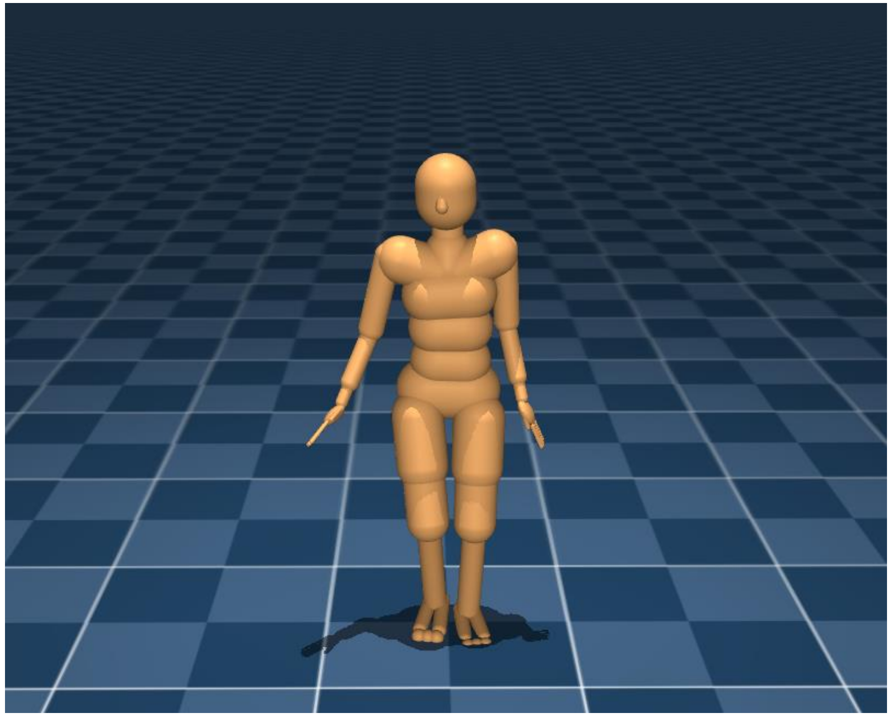
MuJoCo Environment Parameters

S.N.	Parameter Name	Value	Remarks
1.	Gravity	{z: -9.81}	Gravity acts towards negative z-axis
2.	Timestep	0.002	Simulation time step in seconds
3.	Sliding friction	0.7	Coefficient of friction when a body is sliding against another body
4.	Torsional friction	0.005	Coefficient of friction when a body tries to rotate against another body
5.	Rolling friction	0.001	Coefficient of friction when a body is rolling

Methodology
(Humanoids from Different Physics Engines)



Humanoid from Brax Environment



Humanoid from MuJoCo Environment

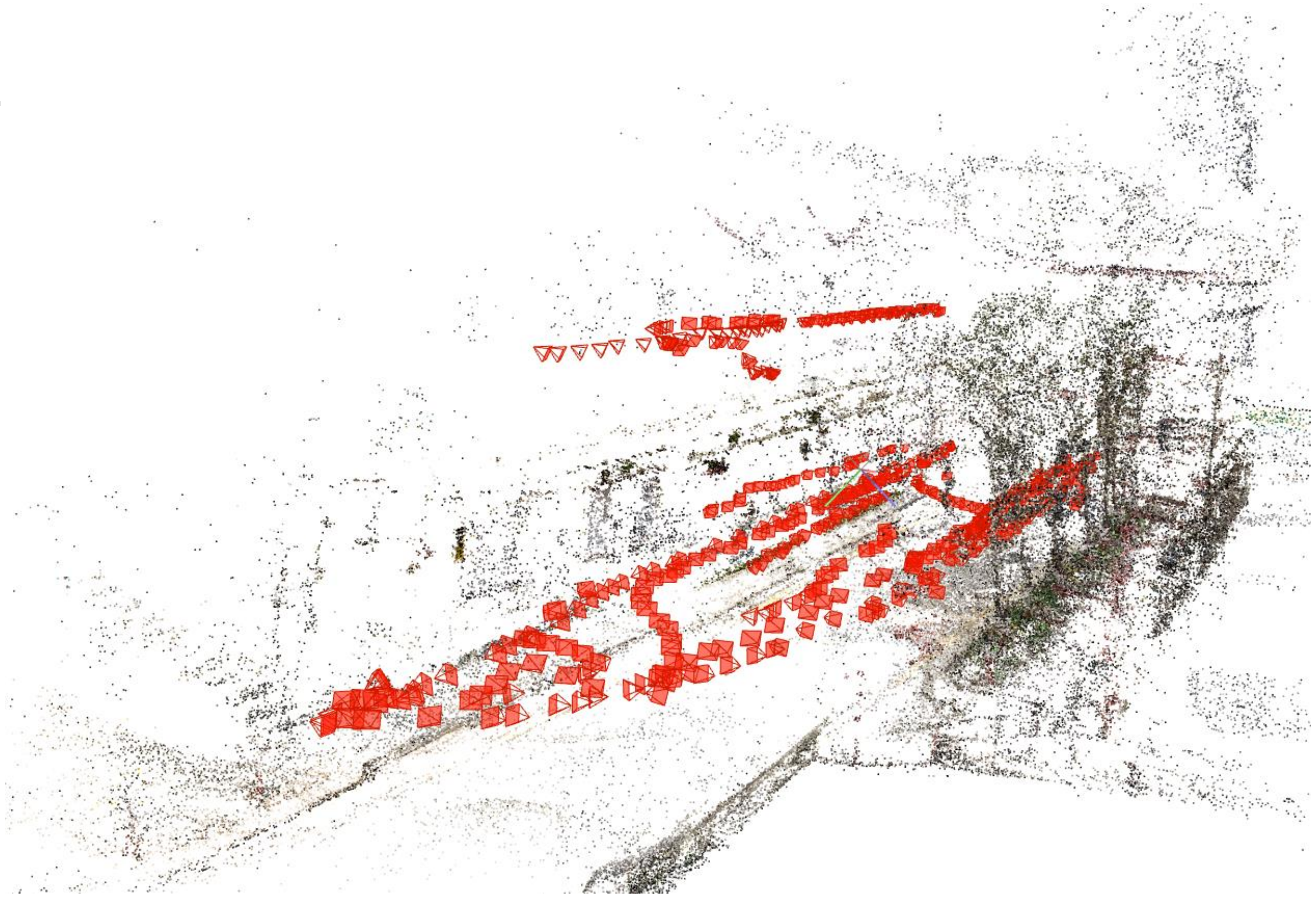
Results
(Camera poses from COLMAP for Outdoor Scene)

- Preprocessing step to train NeRF model

- Red symbols are camera positions

- Images were taken moving in a straight line more than once with different camera angles

- Sparse reconstruction of features



Results
(Feature Extraction and Matching)

- Red dots show different features from SIFT represented as 128D vectors
- Green lines represent match between objects

Pair of Images with Overlapping Objects

Results
(NeRF Result of Outdoor Scene)

- 499 images were taken to train the NeRF

Novel view from a height

Novel view from the ground

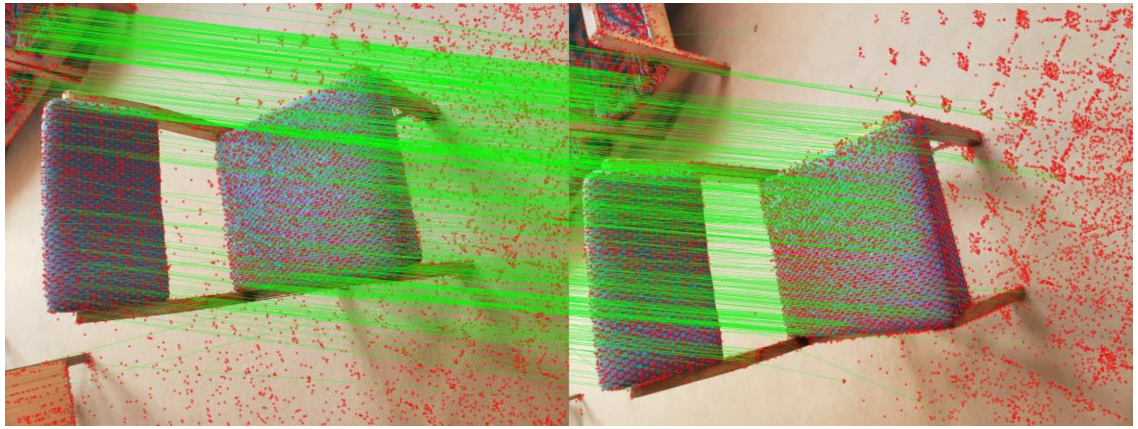
Results
(Comparison of Ground Truth and NeRF Output)

Ground Truth

NeRF Output

Results
(NeRF Result of Chair)

- NeRF of chair which will be used as one of the obstacles in agent's environment
- 68 images were taken to train the NeRF
- Chair is the main focus so is more recognizable while background is distorted

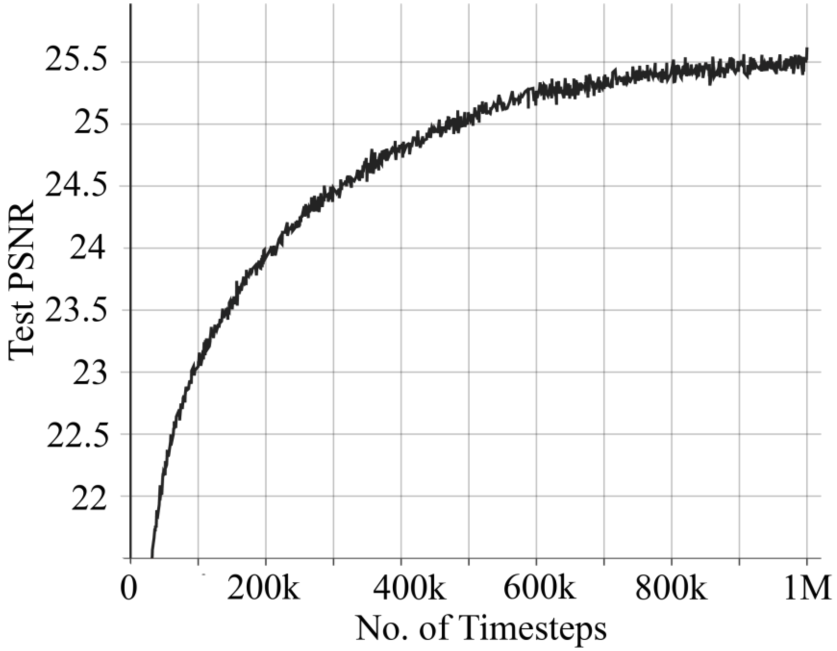


Features matched between overlapping images

NeRF Output of Chair

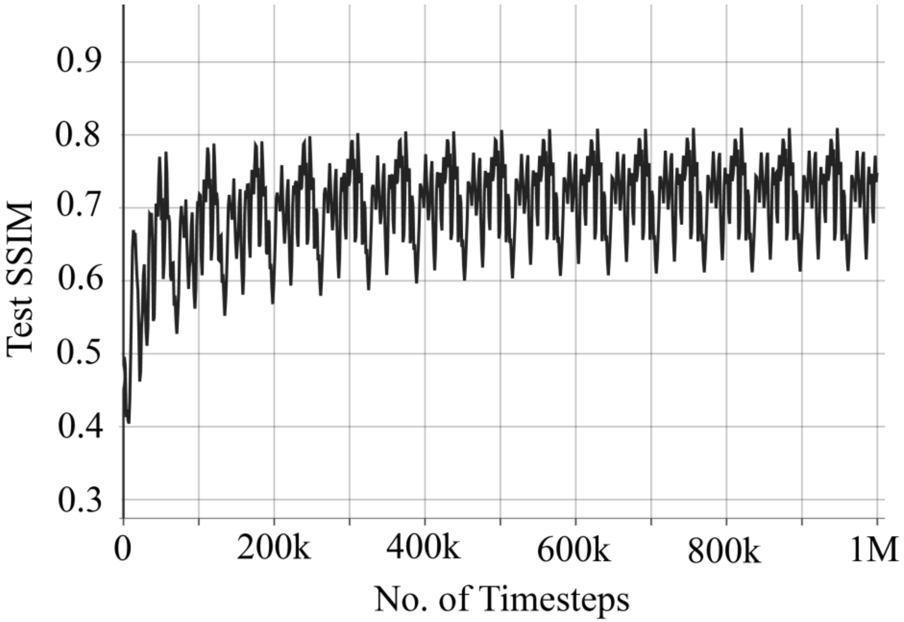
Results
(PSNR and SSIM Graphs)

- Ratio between maximum power of signal and noise



PSNR Graph

- Degree of perceived quality of image



SSIM Graph

Results
(Mesh of Chair Extracted from NeRF with Noise)

- The mesh extracted from NeRF contains a lot of noise
- To resolve this, the vertices are divided into connected clusters
- The cluster containing the object is selected, and all other clusters are removed

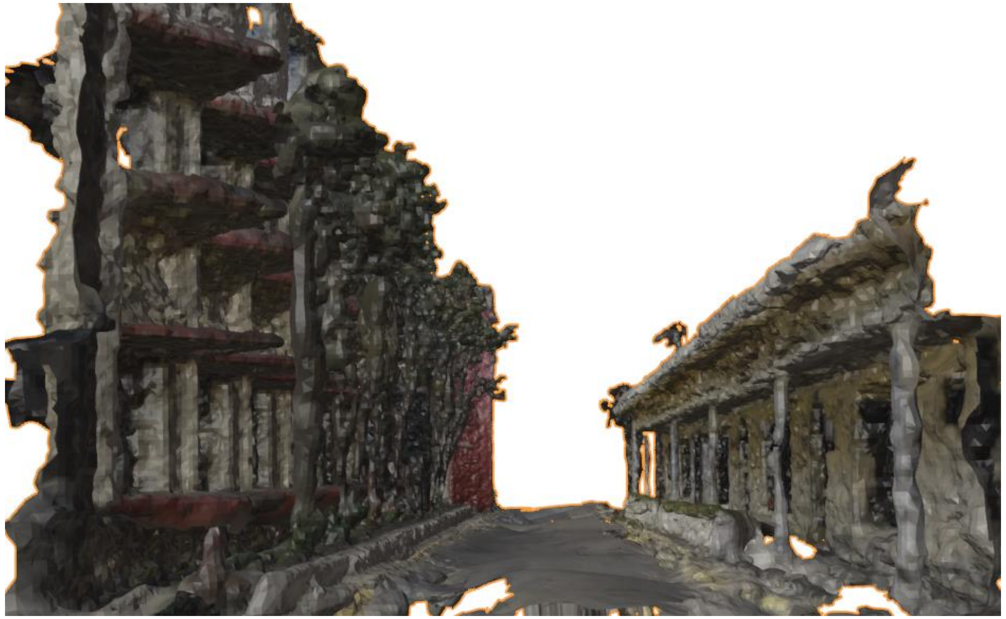
Results
(Mesh of Chair Extracted from NeRF without Noise)

Uncolored Mesh

Colored Mesh

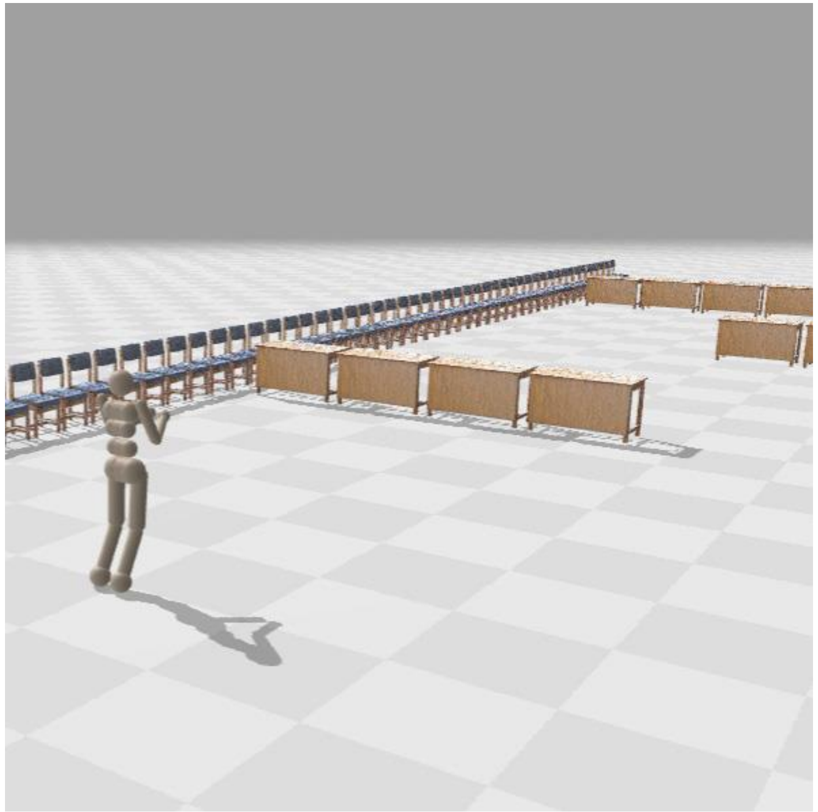
Results
(Mesh of Outdoor Scene)

Mesh with Noise

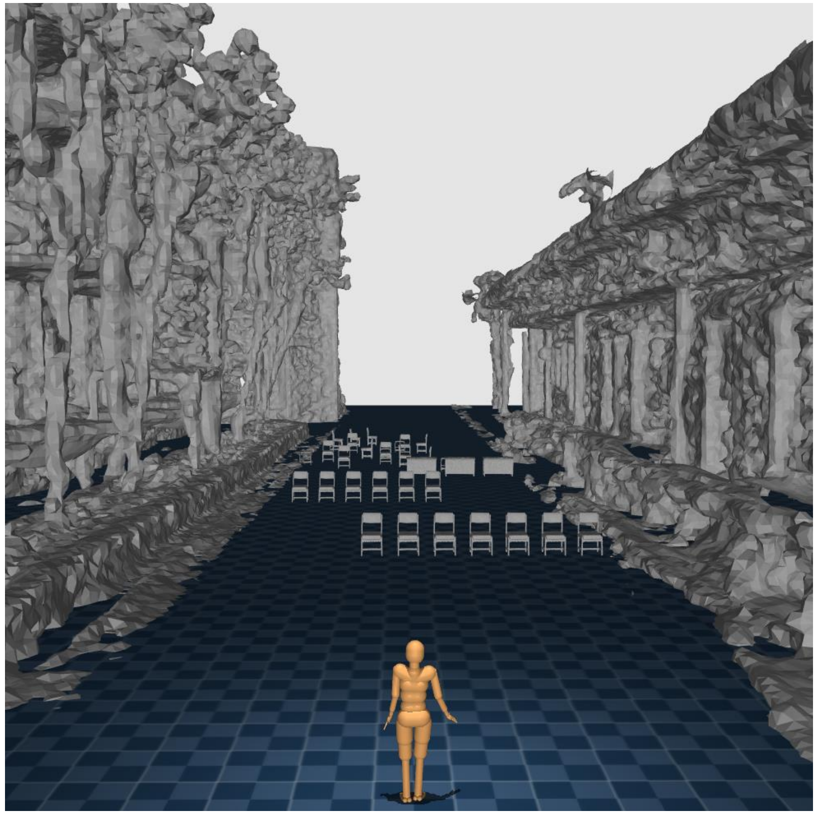


Mesh without Noise

Results
(RL Agent in Environment with Objects from NeRF)

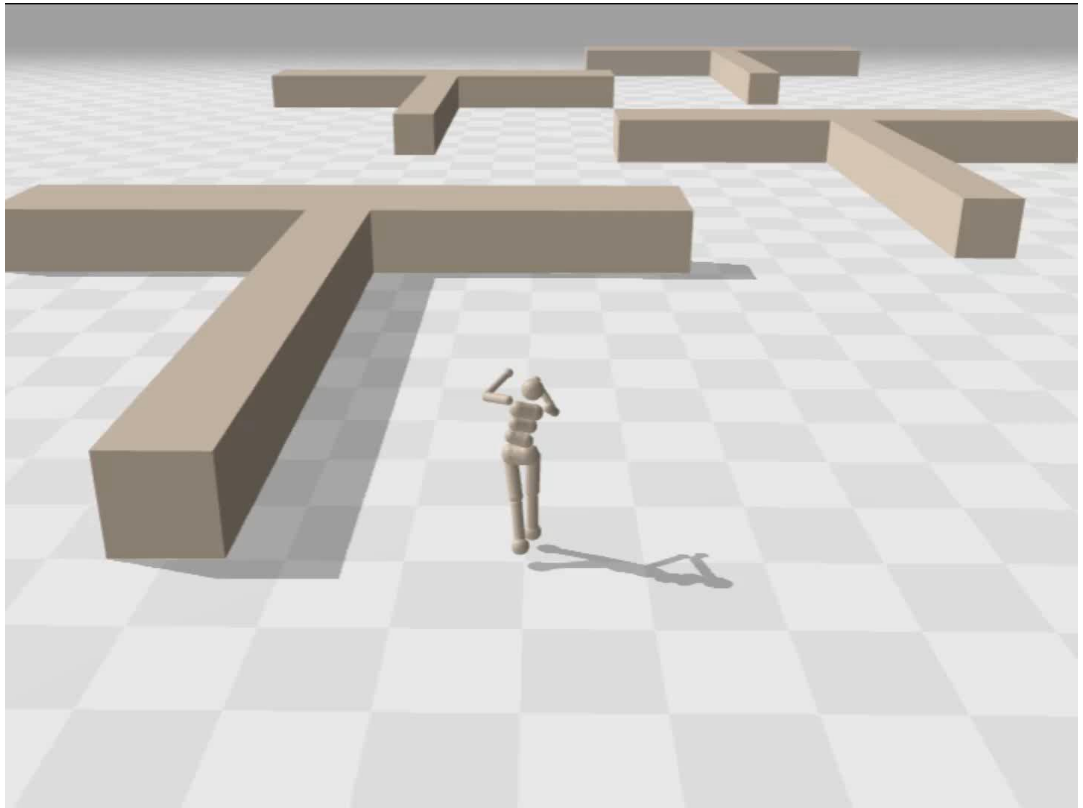


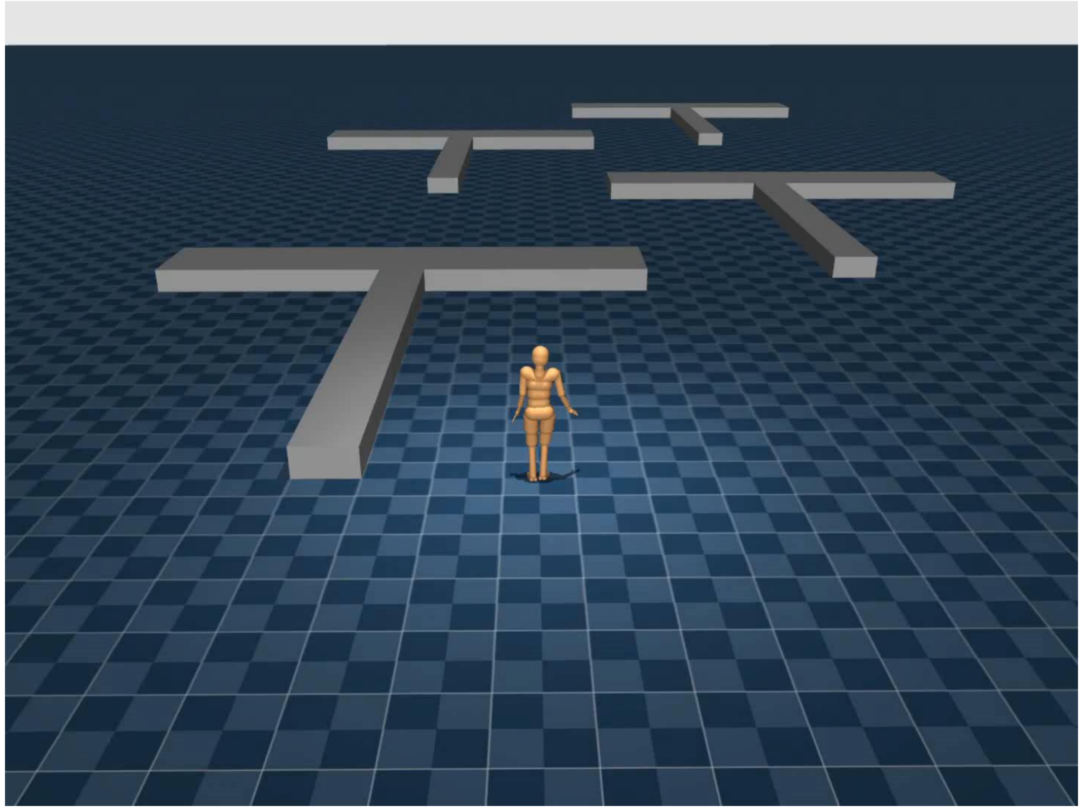
Mesh in Brax Environment



Mesh in MuJoCo Environment

Results
(RL Agent avoiding Walls)



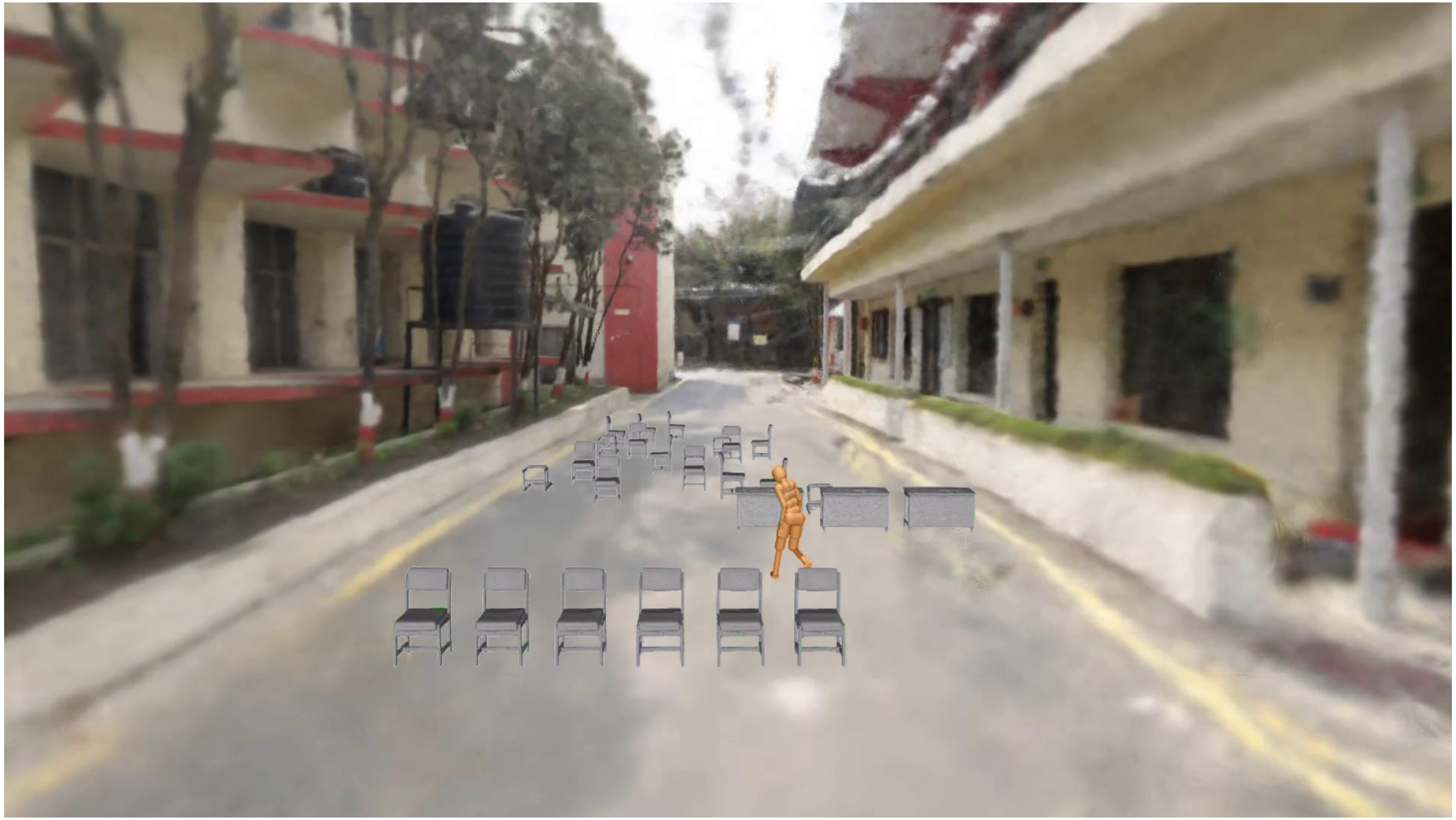


Without Imitation Learning

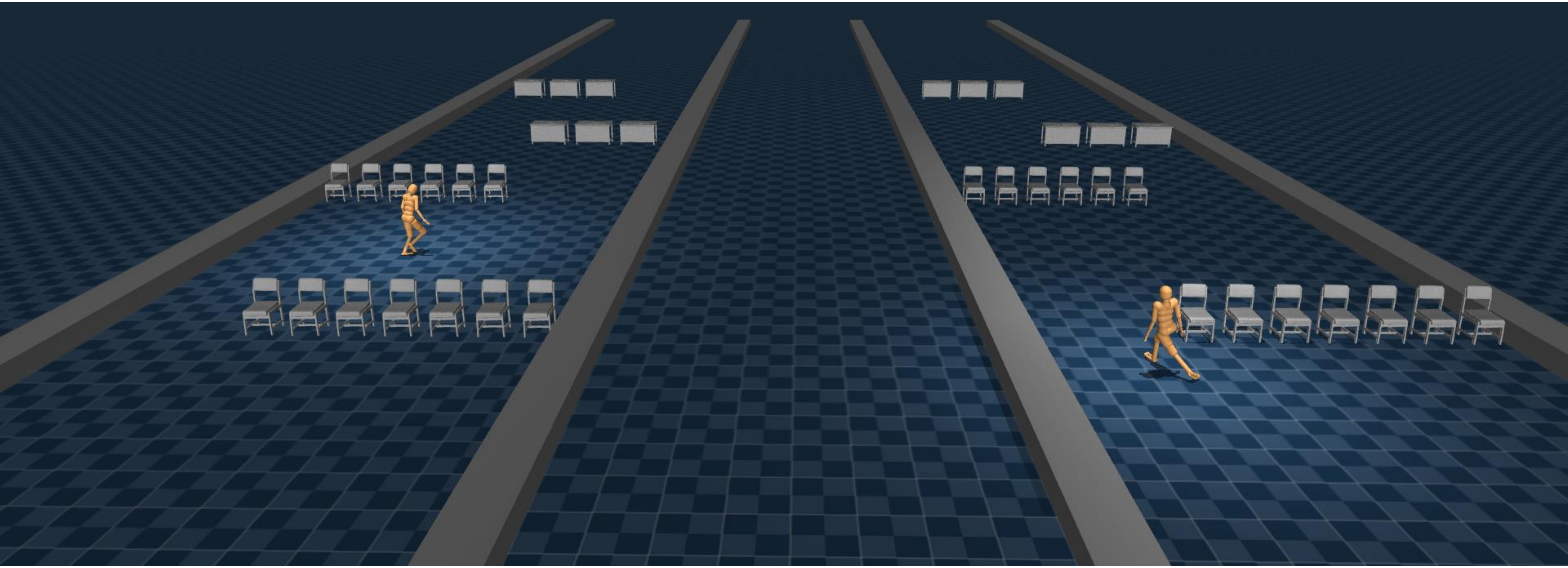
With Imitation Learning

Results
(RL Agent in Obstacle Course)

Results
(RL Agent in NeRF Environment)



Results
(Two Agents Competing in Same Environment)

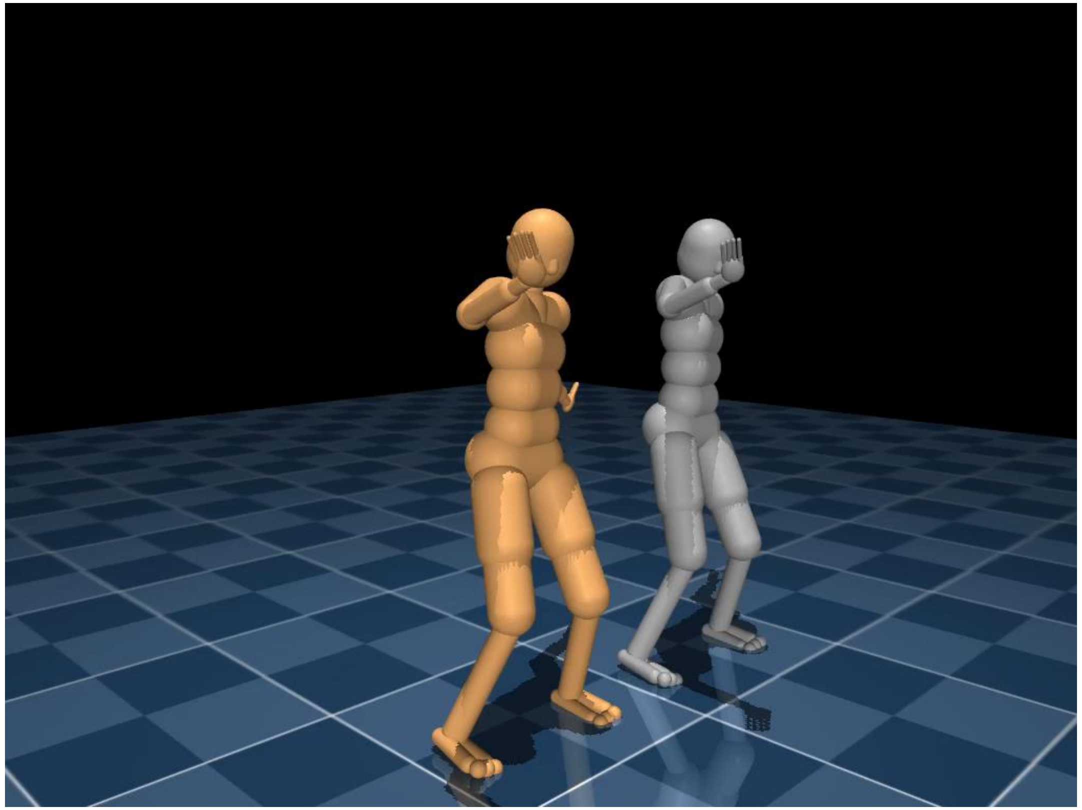


Results
(Agent performing Cartwheel from Reference Motion)

- Orange humanoid is the agent trying to learn from reference motion

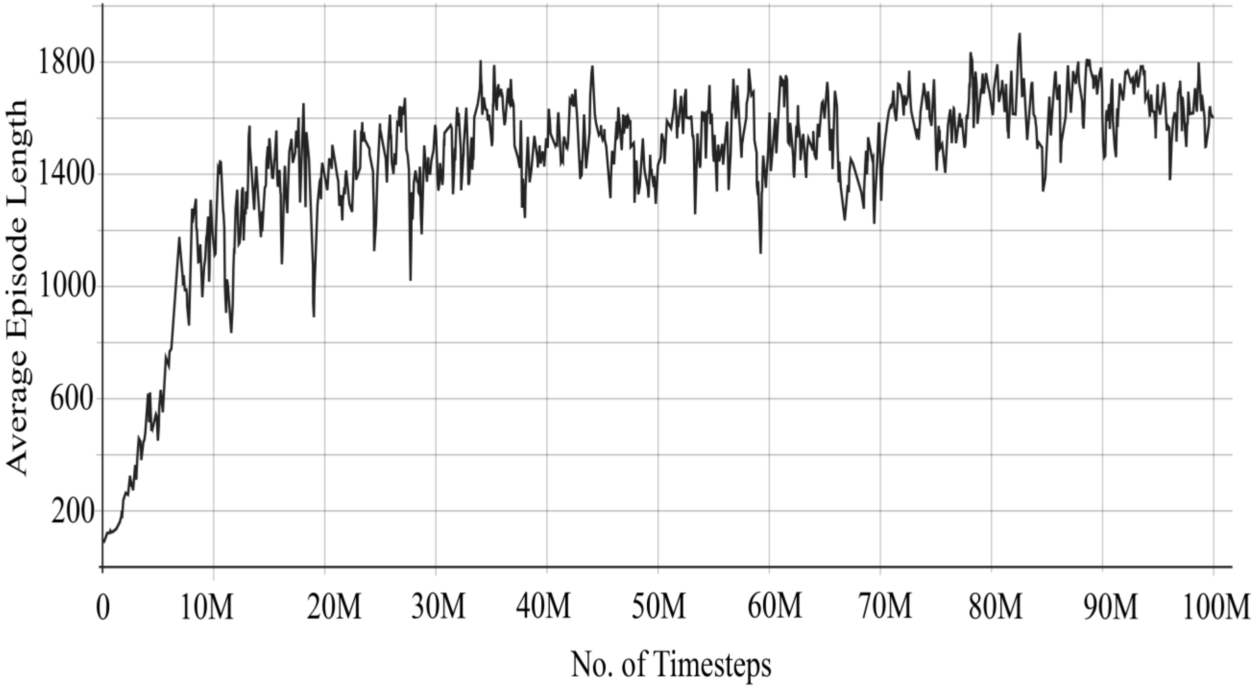
- Grey humanoid is the expert which performs the reference motion

- Motion no. 49_08 from CMU MoCap Dataset

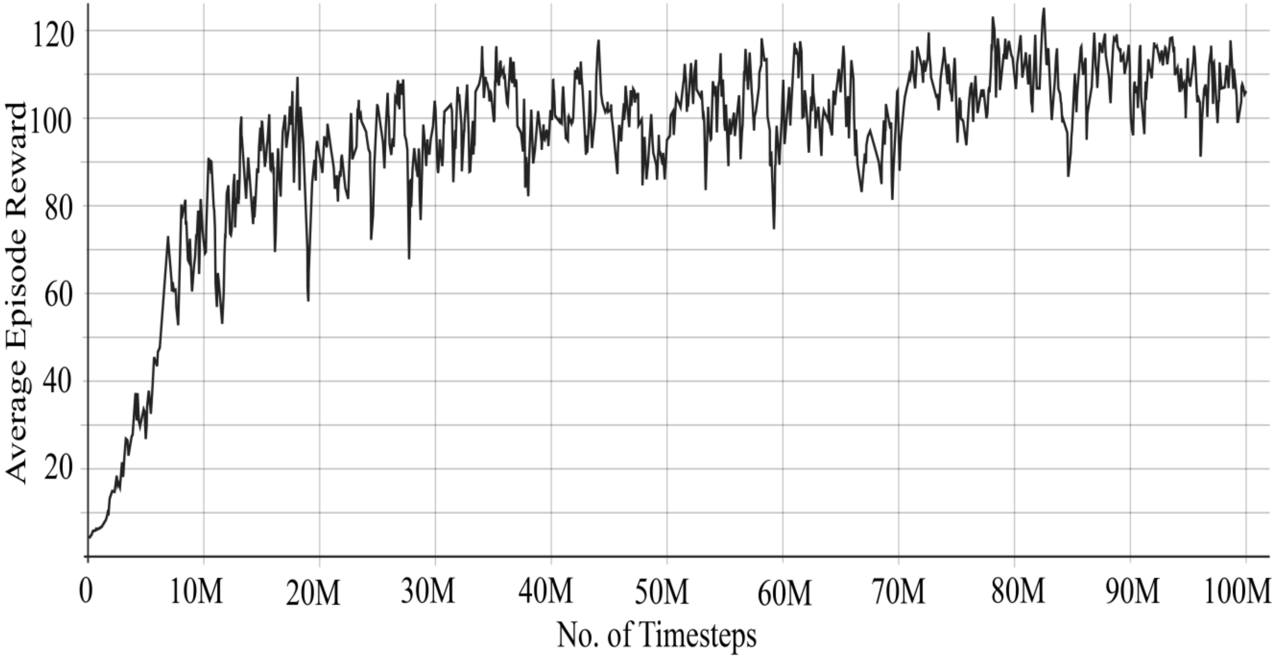


Results
(Episode Length and Total Reward Graphs)

- Total Reward and Episode Length increases as the agent learns to stay alive and perform actions



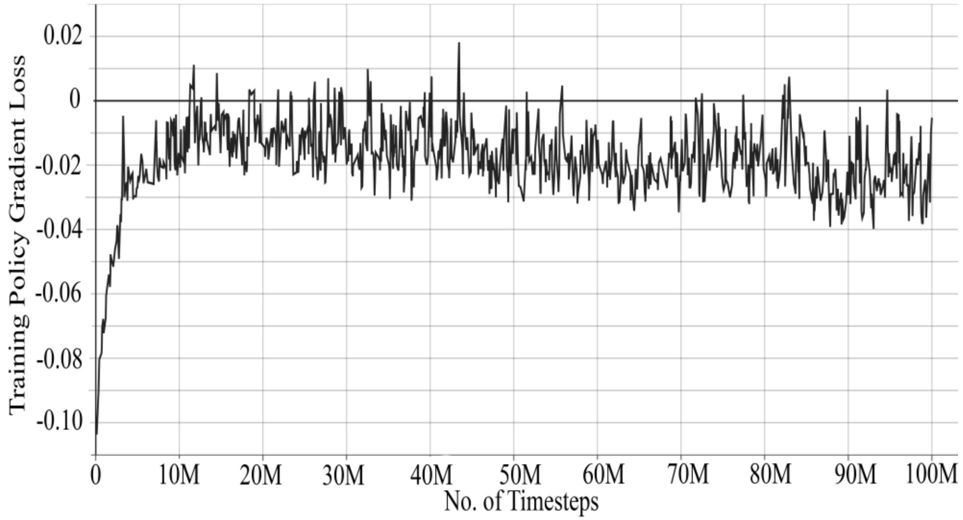
Episode Length vs Timesteps



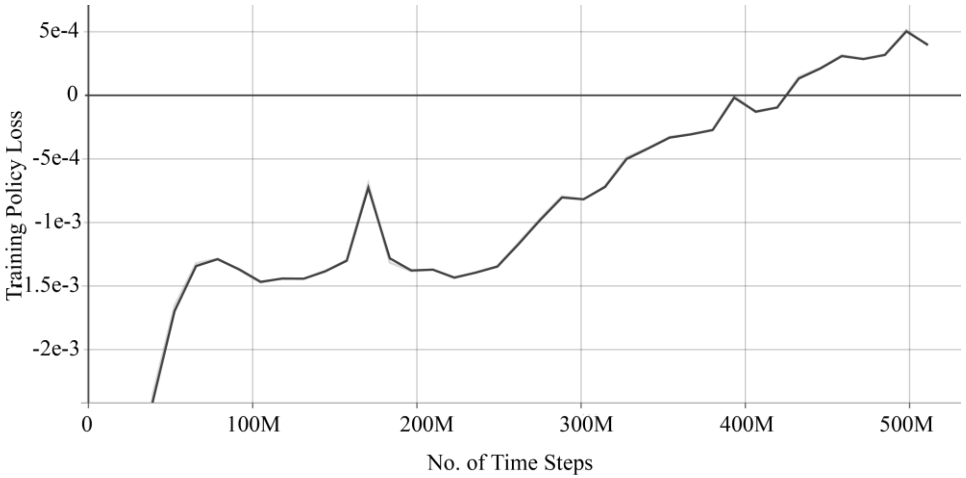
Total Rewards vs Timesteps

Results
(Policy Loss and Value Loss Graphs)

- Policy loss shows how much the policy is changing

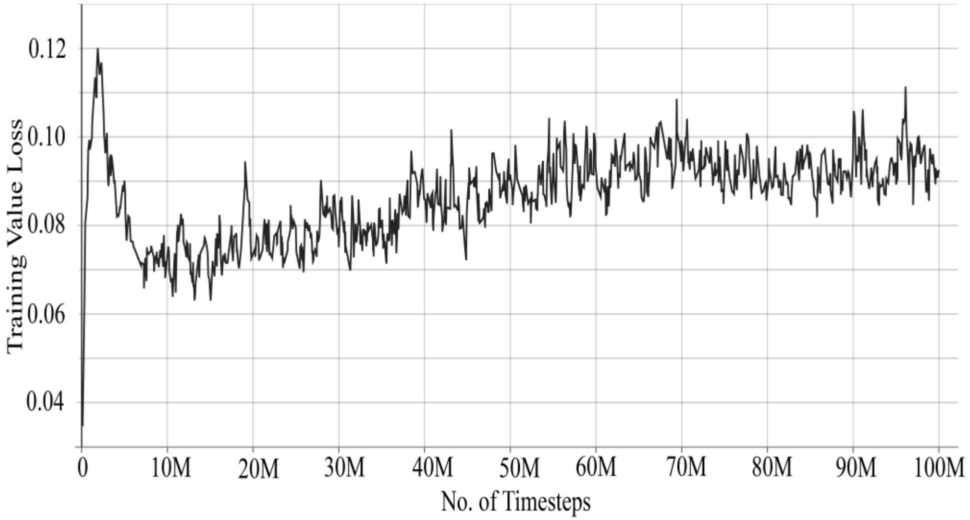


Policy Loss Graph for MuJoCo

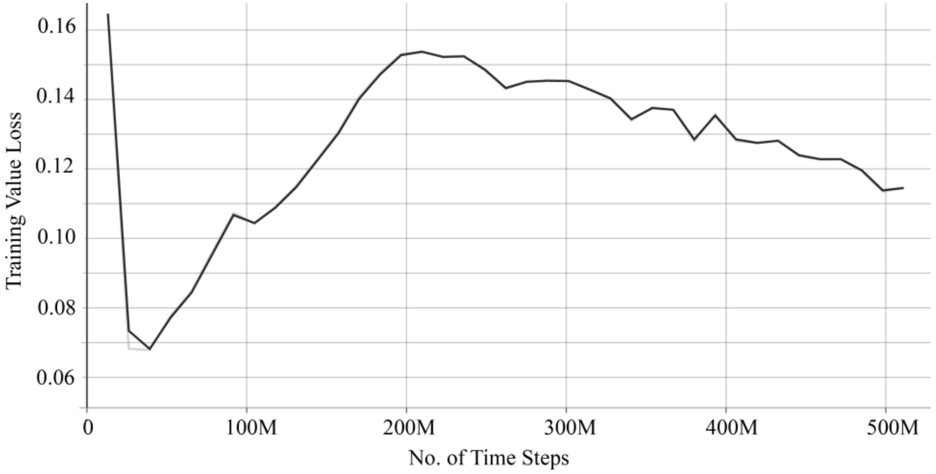


Policy Loss Graph for Brax

- Value loss shows how well critic network predicts value



Value Loss Graph for MuJoCo



Value Loss Graph for Brax

Analysis and Discussion
(NeRF)

- To create different viewpoints camera was translated and rotated.
- The images were preprocessed to generate poses using COLMAP
- The preprocessing step needs multiple overlapping features between objects
- Mesh was extracted from the trained NeRF using Marching Cubes Algorithm

Analysis and Discussion (Reinforcement and Imitation Learning)

- Reinforcement Learning
  - Agent avoids obstacles to survive longer and collect more rewards
  - Value loss represents how well the model predicts value of each state
  - Policy loss is the objective function

- Imitation Learning
  - Agent learns natural movements from MoCap data
  - Agent cannot integrate advanced movements due to lack of compute power

Further Enhancements

- Incorporate complex motions from the MoCap dataset
- Train for real-world robot instead of the humanoid agent
- Integrate NeRF output directly into the physics engine
- Implement path finding algorithm to navigate dynamic obstacles

Conclusion

- NeRF was used to model real-world scenes
- Mesh representation of objects from NeRF used as obstacles
- In Brax environment, agent completed obstacle course with unnatural movements
- In MuJoCo environment, agent completed obstacle course with natural motion using imitation learning
- Outputs of MuJoCo and NeRF merged using same camera trajectory

References - [1]

- X. B. Peng, P. Abbeel, S. Levine and M. van de Panne, "DeepMimic," *ACM Transactions on Graphics,* vol. 37, no. 4, pp. 1-14, 2018.
- N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. Riedmiller and D. Silver, "Emergence of Locomotion Behaviors in Rich Environments," 2017.
- C. Freeman, E. Frey , A. Raichuk, S. Girgin, I. Mordatch and O. Bachem, "Brax -- A Differentiable Physics Engine for Large Scale Rigid Body Simulation," 2021.
- B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," 2020.
- M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," *Advances in Neural Information Processing Systems,* vol. 33, pp. 7537-7547, 2020.

References – [2]

- P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron and P. Debevec, "Baking Neural Radiance Fields for Real-Time View Synthesis," in *Proceedings of IEEE/CVF International Conference on Computer Vision(ICCV)*, 2021.
- J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla and P. P. Srinivasan, "Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields," 2021.
- Y. A. LeCun, L. Bottou, G. B. Orr and K.-R. Müller, "Efficient BackProp," in *Neural Networks: Tricks of the Trade*, Berlin, Heidelberg, Springer, 2012, pp. 9-48.
- N. Max, "Optical Models for Direct Volume Rendering," *IEEE Transactions on Visualization and Computer Graphics,* vol. 1, no. 2, pp. 99-108, 1995.
- J. Schulman, F. Wolski, P. Dhariwal, A. Radford and O. Klimov, "Proximal Policy Optimization Algorithms," 2017.

THANK YOU!