Iowa State University
Electrical and Computer Engineering

Team: sdmay21-33
https://sdmay21-33.sd.ece.iastate.edu/

Reinforcement Learning with Graph Neural Networks for Drone Collision Avoidance

Summary

- A common challenge for drones is avoiding obstacles to prevent collisions. This is complex problem as obstacles can vary in shape and size
- We focused on applying Reinforcement Learning (RL) algorithms for a drone to learn how to avoid obstacles on its own in a simulated environment
- We developed deep learning architectures for comparison of convergence speed
  - Graph Neural Network (GNN) and
  - Convolutional Neural Network (CNN)
- Overall, we created several custom wrappers to interface between existing libraries and simplify the training process
- Our custom GNN policy is built inside existing tools for ease of use and correctness
- The use case is specifically for research

Design Requirements:

- Functional:
  - Drone can fly in simulation
- Non-functional:
  - Drone avoids collision
- Engineering Constraints:
  - GPU & simulation training speed
- Operating Environment:
  - Virtual simulated environment
- Standards:
  - This is a research-oriented project that includes many customizations and few standards that are applicable to RL

Simulation Environment:

- Tested and trained in Unreal Engine, a common video game engine with realtime physics
- Testing of the drone was performed in simulation for movement checks, vision check, and specific test cases with custom environments
- AirSim API receives commands from our Python code and moves the drone in the simulation while providing image feedback
- Created a custom wrapper to easily make AimSim calls as an OpenAI Gym environment
- The DQN with StableBaselines library could easily make calls to our Gym environment

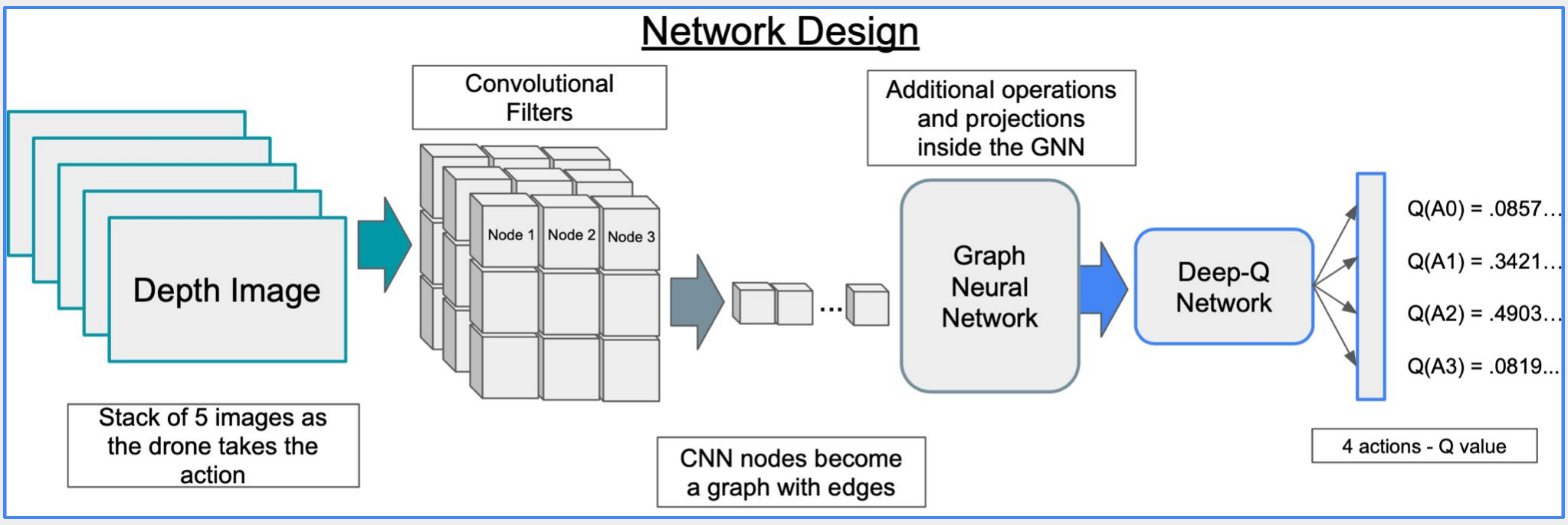
Containerizing the Setup:

- Problem that Unreal Engine, AirSim, and etc involve a very complex setup process
- We created a Docker container that already has everything installed and setup
- Docker container can run on multiple platforms, to solve issues between Windows and Ubuntu
- Our container interfaces with the GPU through WSL to allow for full functionality and training
- This container will save many hours of setup time and debugging in future use

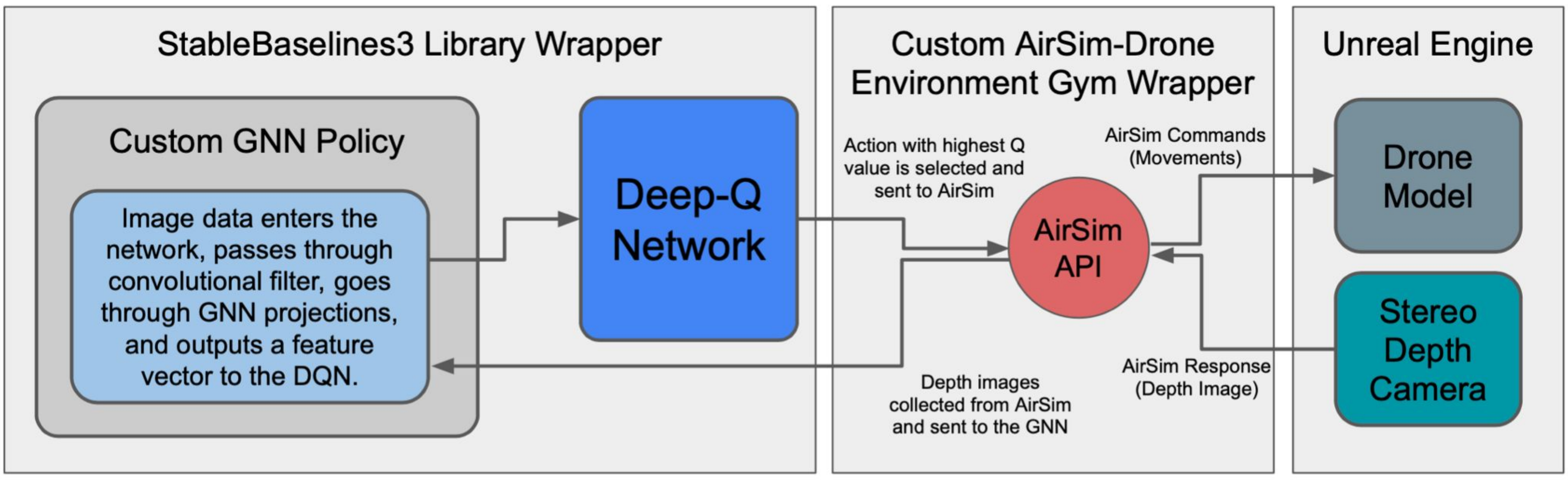
Technical Tools and Resources:

- Python3 and Anaconda manager
- AirSim API and Unreal Engine
- StableBaselines3, PyTorch, and Numpy
- Resources: RL/GNN textbooks and publications

Network Design



Training Process


StableBaselines3 Library Wrapper
Custom GNN Policy
Image data enters the network, passes through convolutional filter, goes through GNN projections, and outputs a feature vector to the DQN.
Deep-Q Network
Custom AirSim-Drone Environment Gym Wrapper
Action with highest Q value is selected and sent to AirSim
AirSim Commands (Movements)
AirSim API
Depth images collected from AirSim and sent to the GNN
AirSim Response (Depth Image)
Unreal Engine
Drone Model
Stereo Depth Camera

Client / Advisor
Deepak-George Thomas
Dr. Ali Jannesari

Team Members
Karter Krueger Matthew Phipps
Ryan Howe Zachary Mass
Rithvik Menon Thamir Al Harthy
Joshua Klayanapu

Acknowledgements
Thank you to Dr. Jannesari's SwAPP Lab for providing the GPU computer to train our model on.

Note: Additional specific resources can be found in corresponding final report