Master Thesis Presentation

Hussein LEZZAIK

Master Thesis Presentation
for the project entitled
Graph Neural Networks and Reinforcement Learning for Multi-Robot Motion Planning

Supervisors:
Gennaro Notomista
Claudio Pacchierotti

Jury:
Philippe XU
Joelle AL HAGE

September 3, 2021

Table of contents

Problem Formulation

- Centralized algorithms are computationally expensive, aren't scalable, and break easily
- Learning based-algorithms for decentralized models is a new approach to tackle this problem

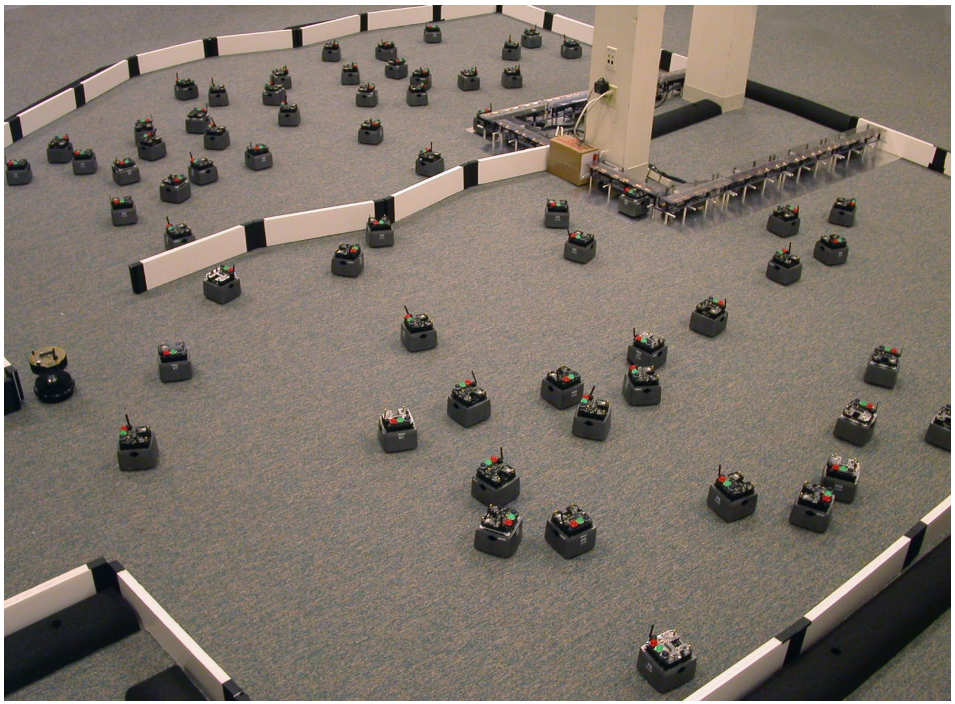
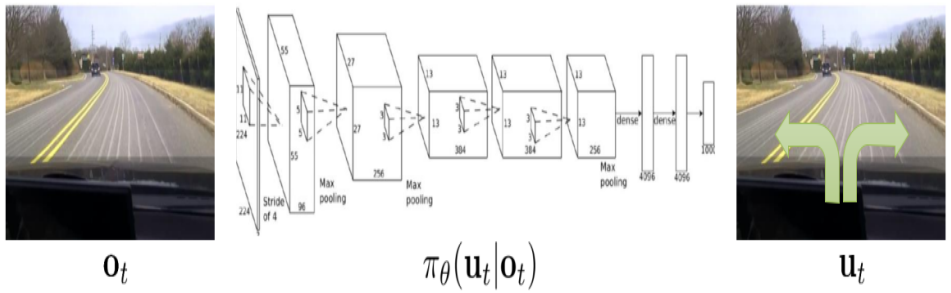


Figure: Swarm System

Imitation Learning - Behavioural Cloning

- Simplest form of imitation learning



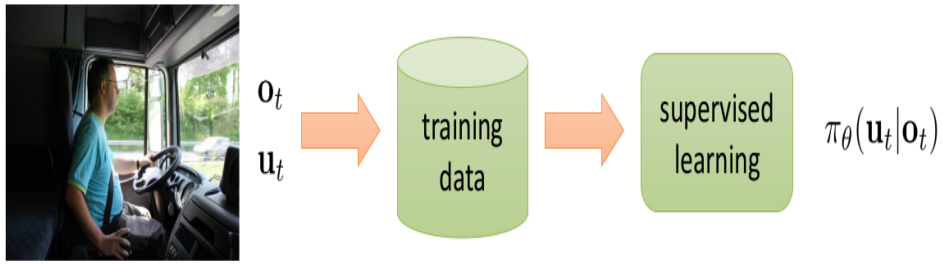


Figure: Behavioural Cloning, Nvidia 2016

GNN Model

- Regression MLP model to predict local control inputs for each robot
- Developed in PyTorch, decoupled inputs, decentralized, takes adjacency matrix

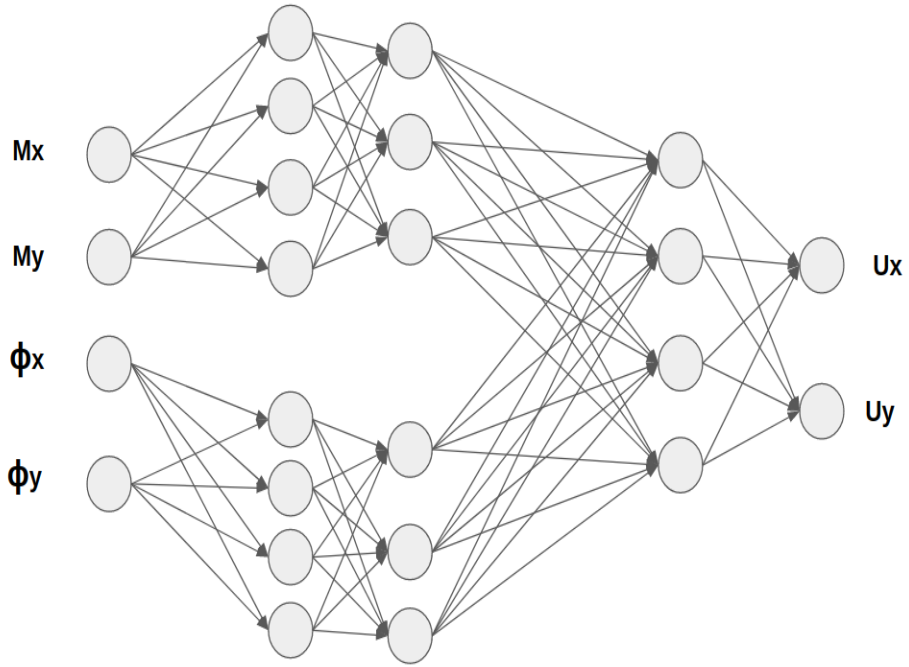


Figure: Ensembled GNN Model

Input Equations

$$Mx_i = \sum_{j \in N} \frac{A_{ij}(x_j - x_i)}{N-1} \tag{1}$$

$$My_i = \sum_{j \in N} \frac{A_{ij}(y_j - y_i)}{N-1} \tag{2}$$

$$\Phi_{xi} = \sum_{j \in N} \frac{A_{ij} Mx_i}{N-1} \tag{3}$$

$$\Phi_{yi} = \sum_{j \in N} \frac{A_{ij} My_j}{N-1} \tag{4}$$

CoppeliaSim Robotics Simulator

- V-Rep as a physics engine for simulation
- ROS to interact and control robots

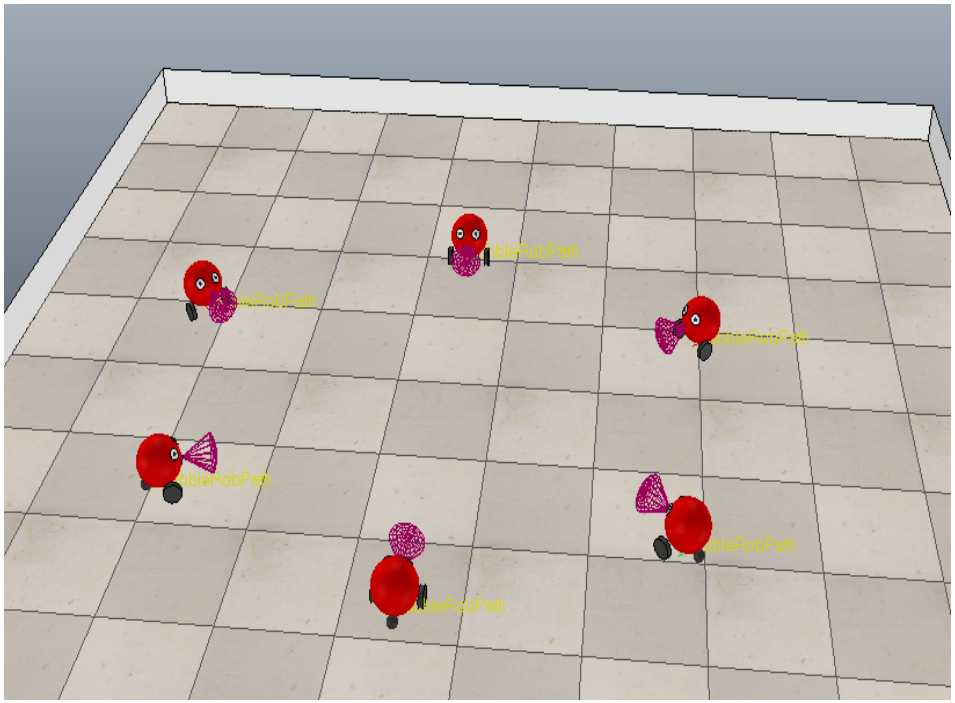


Figure: Example of a scene of six robots of Bubblerob's

Consensus of Multi-Robots

- Stopping condition with $d = \sum_{j \in N} |(x_1 - x_j) + (y_1 - y_j)|$

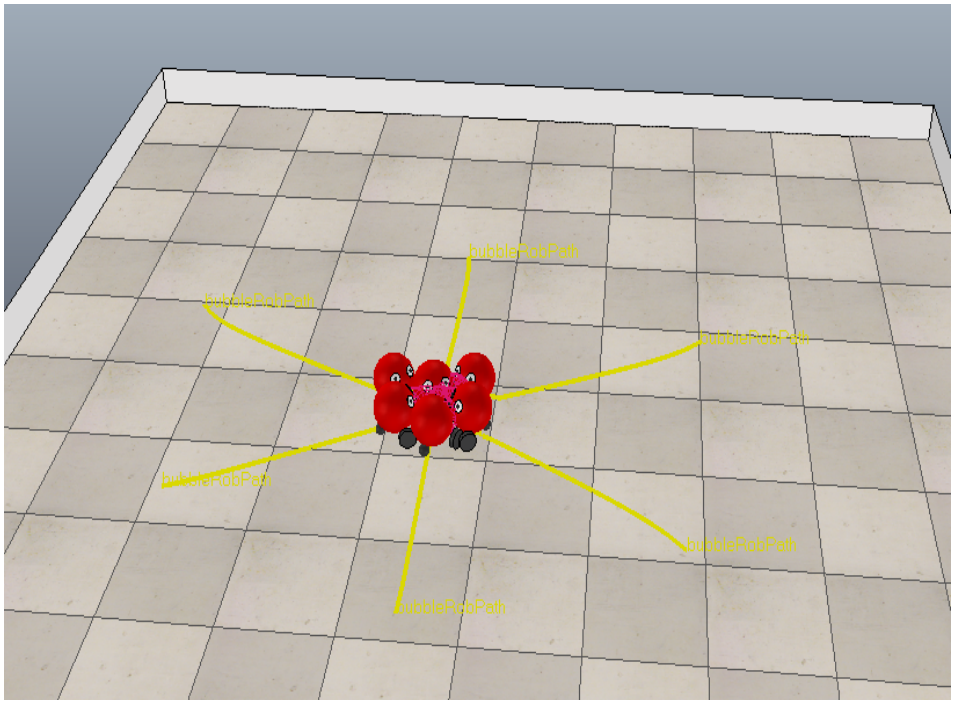


Figure: Example of a scene of six robots achieving consensus

Data Collection

Run an expert consensus algorithm:

$$\begin{cases} U_{x_i} = -\sum_{j=1}^{n} A_{ij} K_{ij} * [(x_i - x_j)] \\ U_{y_i} = -\sum_{j=1}^{n} A_{ij} K_{ij} * [(y_i - y_j)] \end{cases} \tag{5}$$

Centralized Controller

- Time to achieve consensus took around 1.5 seconds
- Implemented as a Fully Connected Graph

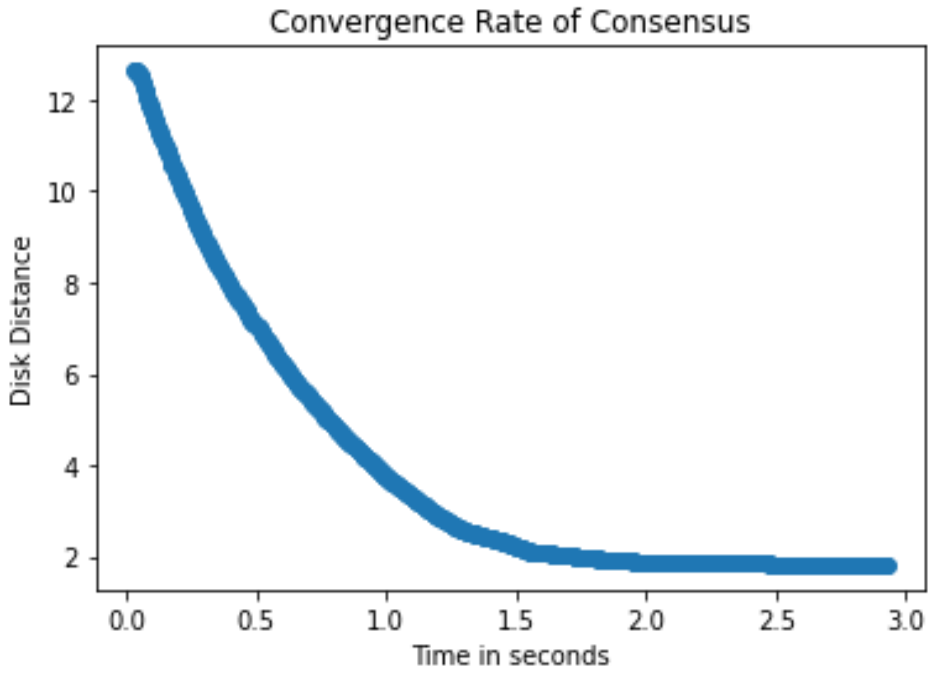


Figure: Convergence Rate of a Centralized Controller on Fully Connected Graph

GNN on Fully Connected Graph

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

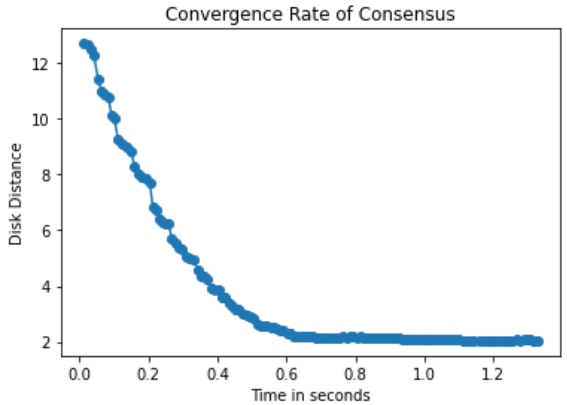


Figure: Convergence Rate of GNN controller on Fully Connected Graph

GNN on Cyclic Graph

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

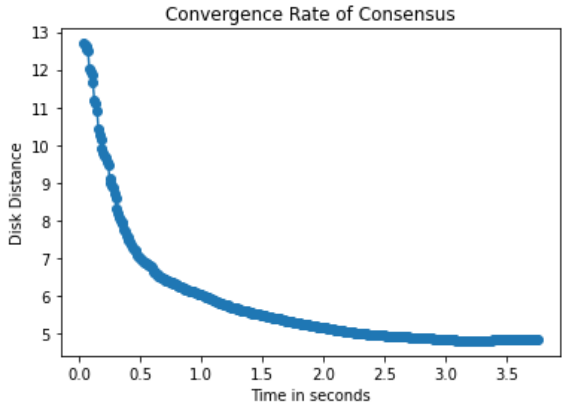


Figure: Convergence Rate of GNN controller on a Cyclic Graph

GNN on Line Graph

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

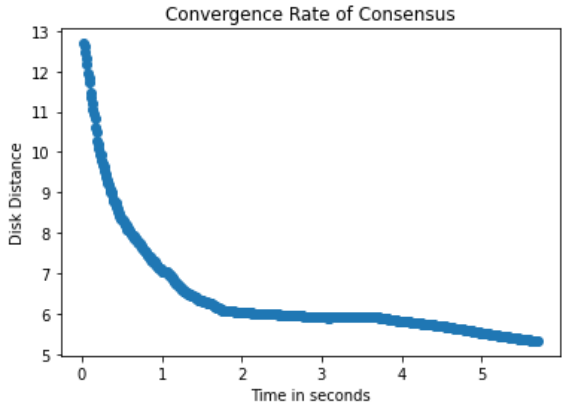


Figure: Convergence Rate of GNN controller on a Line Graph

GNN on Random Weak Graph

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

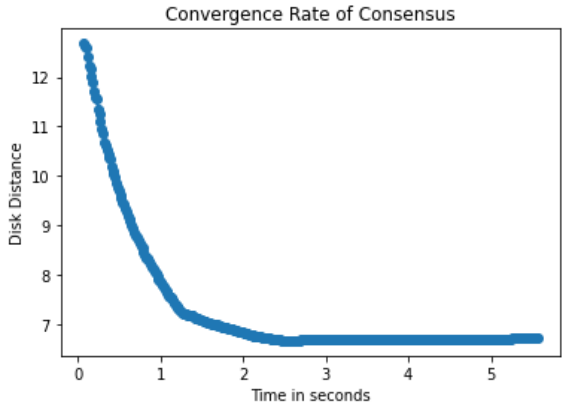


Figure: Convergence Rate of GNN controller on a Random Graph

Real-Time Demo

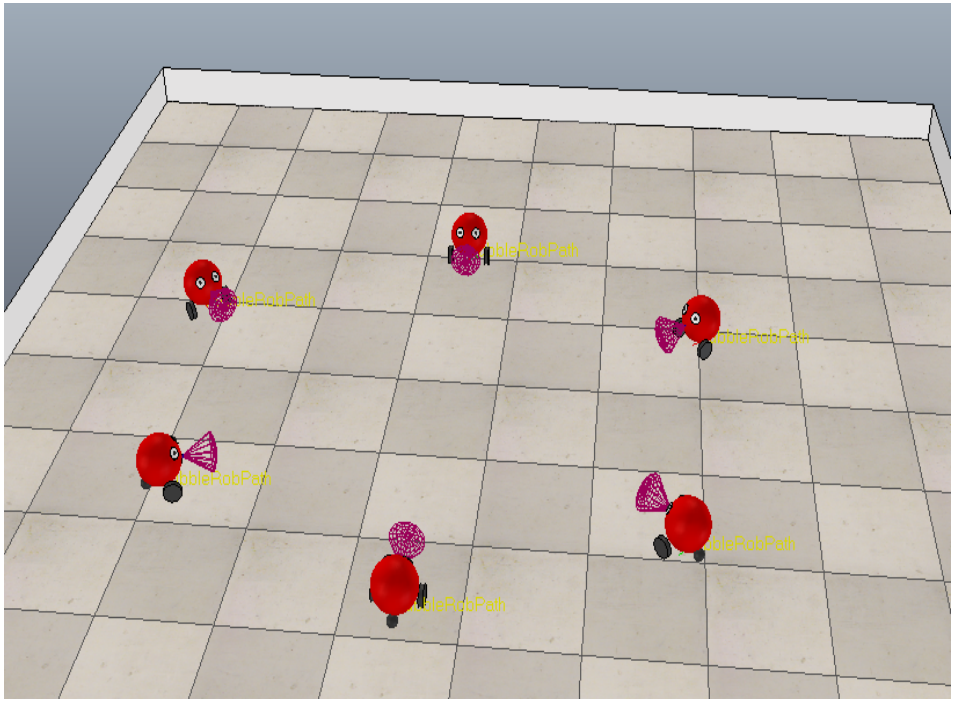


Figure: Demo of GNN model for Fully Connected Graph

Conclusion

- Developed decentralized framework powered by Graph Neural Nets and Reinforcement Learning
- GNN model is robust, scalable, and computationally fast
- Can be used for different multi-agent motion planning case studies

Future Work

- Training the GNN model with a Reinforcement Learning algorithm like Q-Learning, since we can manually write the reward function
- Implementing the GNN-DQN model on a different motion planning problem where decentralized algorithms are essential

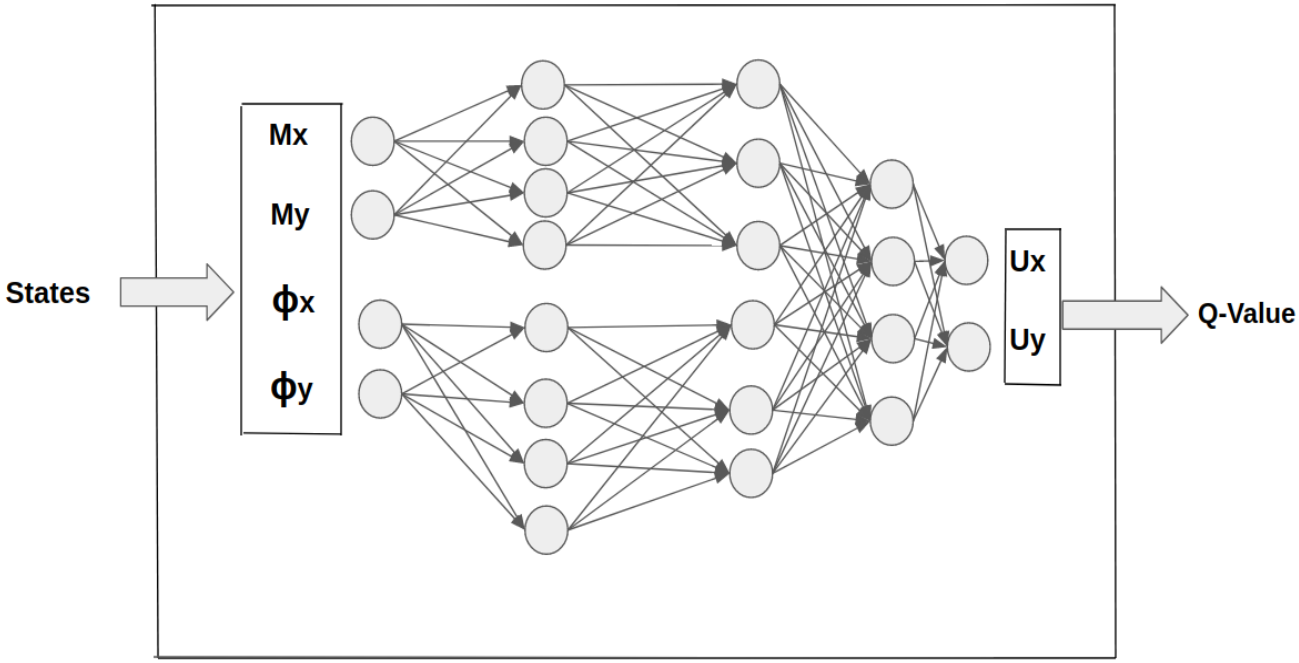


Figure: DQN with GNN architecture

Graph Neural Network v1

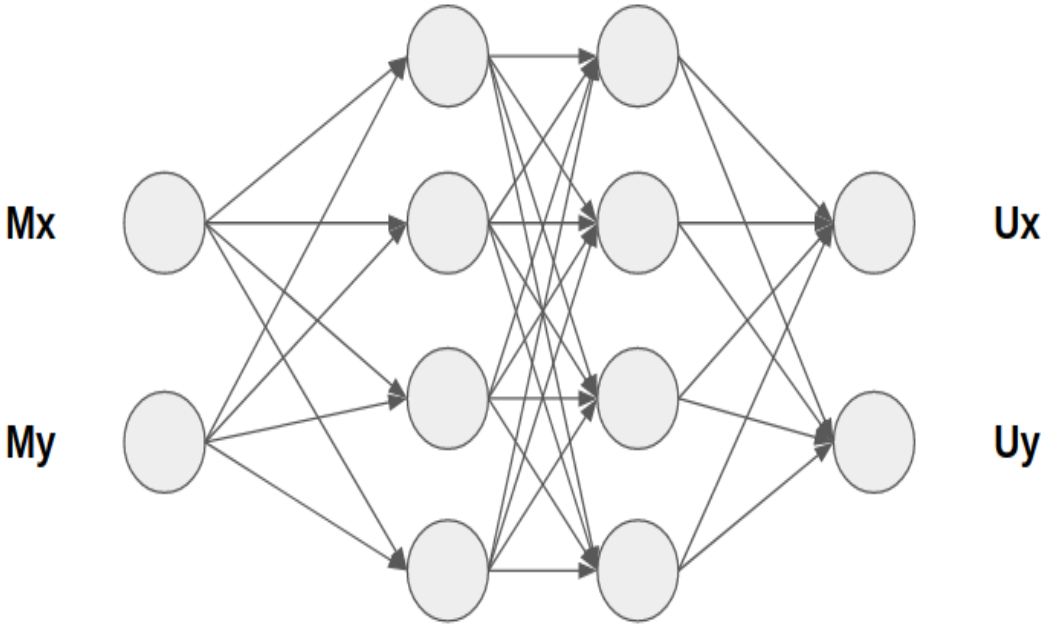


Figure: First version of the GNN model

- Convergence wasn't fast, and performance was like that of a line graph at test time
- Lacks generalization power to different communication topologies

Graph Neural Network v2

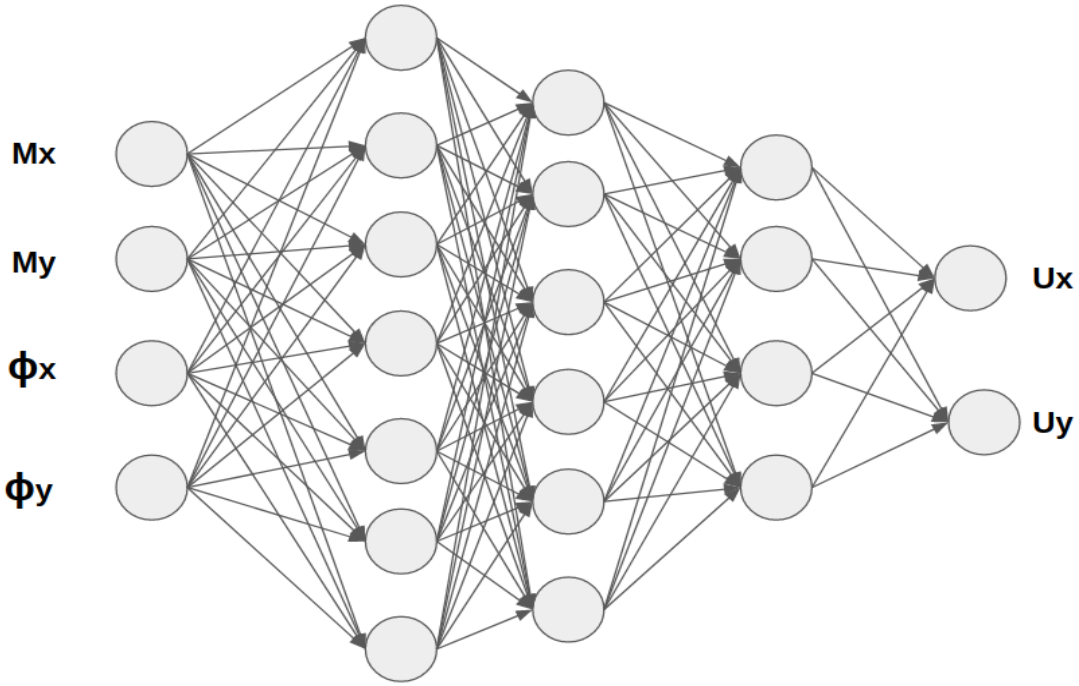


Figure: Second version of the GNN model

- The additional two terms enhanced the performance dramatically
- However, due to correlation between both input couples the performance was poor and we had to decouple them