Dream Inspired Deep Reinforcement Learning for Training Agent in Navigation Task Settings

Christopher Pham
Virginia Tech
glacegon@vt.edu

Vaidhyanathan Chandramouli
Virginia Tech
vaidhyanathan@vt.edu

Abstract

During sleep, the human brain engages in a process of memory consolidation and learning, often manifesting as the reenactment of familiar scenarios. Inspired by this phenomenon, our research endeavors to improve agent navigation skills through a novel approach. Integrating deep reinforcement learning with diffusion processes, we simulate a dream-like environment for our agent to navigate within. This environment is generated by amalgamating elements from the actual environment encountered during initial navigation. By training the agent within this simulated realm, we aim to augment its comprehension and proficiency in real-world navigation tasks. This interdisciplinary project not only applies concepts from reinforcement learning and computer vision but also offers insights into the remarkable adaptability of the human brain.

1. Introduction

With the significant advancements and ongoing research in computer vision, object detection, and control, autonomous vehicles and robots have become ubiquitous in various applications. Navigation task optimization, particularly obstacle avoidance, stands as a pivotal technology in local path planning for autonomous vehicles, ensuring both human and vehicular safety [8]. This technology finds diverse applications, including guiding autonomous underwater vehicles to explore regions of interest while avoiding collisions with reefs, underwater obstacles, and the seabed [9]. Additionally, it plays a crucial role in front collision warning systems deployed in highway environments [11], as well as in visual-based automated collision avoidance systems [12]. Moreover, obstacle avoidance algorithms are integral to unmanned aerial vehicles' vision-based trail following approaches [10]. Furthermore, these algorithms are instrumental in enhancing the navigation capabilities of car-robots, especially in scenarios involving exploration and traversal of unfamiliar environments [13].

1.2 Literature review

This paper [14] utilizes a policy gradient method within a Markov decision process setup, where the policy is defined by the reverse sampling process of the diffusion model. This method can be technically adapted to dream-inspired diffusion learning by employing reward functions specifically tailored to evaluate the dream-like qualities of generated images, such as surrealism and abstractness. By optimizing these reward functions, the model can iteratively learn to produce images that mimic the often illogical and fantastical nature of dreams. Another paper [15] introduces Denoising Diffusion Policy Optimization (DDPO), a policy gradient method for optimizing diffusion models directly on downstream tasks like image quality and prompt-image alignment. This approach frames the denoising process of diffusion models as a multi-step decision-making task within a Markov decision process, leveraging exact likelihoods at each denoising step for optimization. Specifically, the method can fine-tune text-to-image diffusion models to enhance alignment with complex, non-explicitly promptable objectives such as aesthetic quality or unusual compositional tasks, using feedback from a vision-language model instead of extensive human labeling**.**

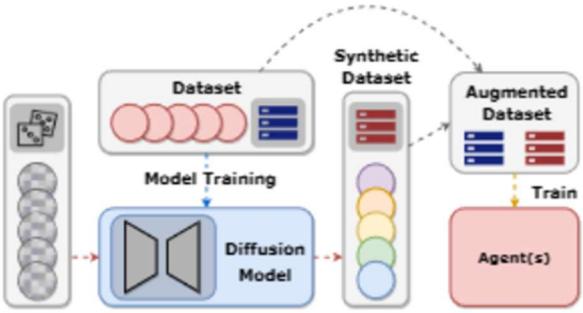


Figure 1: MaxDiff RL framework overview as proposed by paper[16]

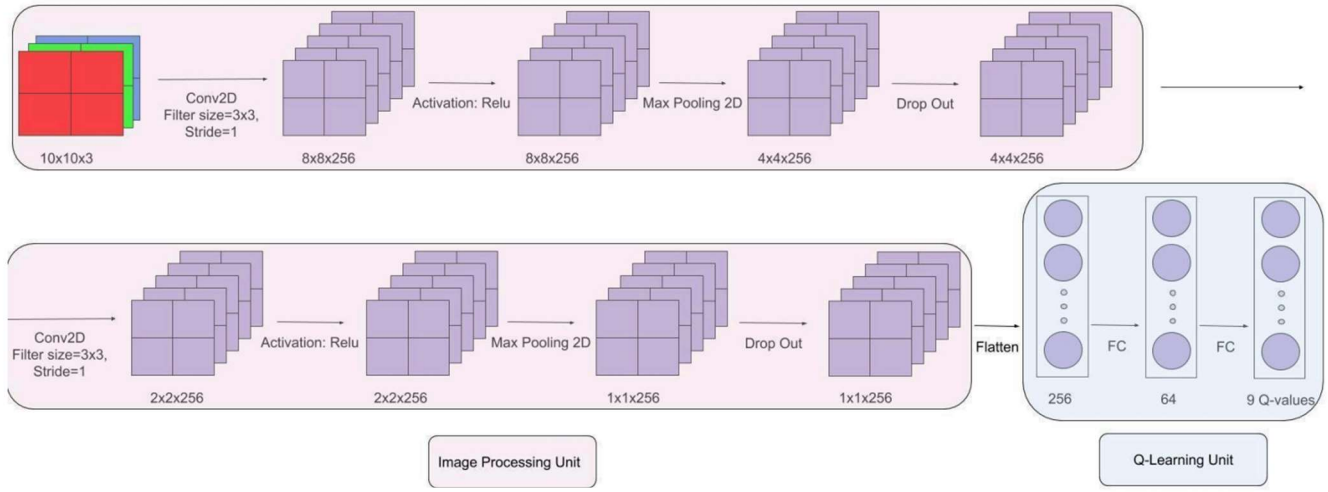


Figure 2: An illustration of the DQN Architecture

The paper [16] "Maximum Diffusion Reinforcement Learning" introduces MaxDiff RL, a framework that enhances traditional reinforcement learning by utilizing statistical mechanics to decorrelate agent experiences, akin to independent and identically distributed sampling. For dream-inspired diffusion learning, MaxDiff RL could effectively generate diverse and surreal imagery by optimizing reward functions that measure the 'dream-likeness' of outputs.

Diffusion models offer a more robust approach by learning the entire data distribution from the real dataset These models parameterize and learn the distribution,

2. Methodology

2.1. Mathematical Formulation

For this project, we utilized a DRL variance called Deep Q Network (DQN) approach. The main idea of this algorithm is to learn the Q-function or, in other words, to estimate the state-action pair value denoted as $Q(s, a)$, where s is the state and a is the corresponding available action chosen from that state. In order to learn the Q-function, we apply Bellman equation for DQN [7]:

$$Q^{\pi}_{DQN}(s,a) \approx r + \gamma max_{a'} Q^{\pi}(s', a') \quad (1)$$

Where

- $Q^{\pi}_{DQN}(s,a)$: The optimal Q-value that we want to learn for the current state given.
- r: The reward that the agent receives from the environment by performing an action.
- γ: Discounted rate, this makes the new information learn less valuable as time goes by and helps eliminate infinite horizon scenarios. A discounted rate of 1 will make the agent always consider long term reward while a discounted rate of 0 will value immediate reward only.
- $max_{a'} Q^{\pi}(s', a')$: The action that maximizes the Q-value of the next state.

2.2. Machine Learning Model & Architecture

Many authors misunderstand the concept of anonymizing for blind review. Blind review does not mean that one must remove citations to one's own work—in fact it is often impossible to review a paper unless the previous citations are known and available.

Blind review means that you do not use the words "my" or "our" when citing previous work. That is all. (But see below for tech reports.)

Saying "this builds on the work of Lucy Smith [1]" does not say that you are Lucy Smith, it says that you are building on her work. If you are Smith and Jones, do not say "as we show in [7]", say "as Smith and Jones show in [7]" and at the end of the paper, include reference 7 as you would any other cited work.

This network architecture is designed to be better suited with reinforcement learning tasks. Specifically, it consists of two main components as highlighted in the above image: Image Processing unit and Q-Learning unit.

2.2.1. Image Processing unit

The 'Image Processing unit' follows a similar structure to most convolutional neural networks (CNN) that are used for image recognition such as AlexNet [3]. Specifically, it contains: 2D convolutional layers which use filters to convolve with the image and create corresponding feature

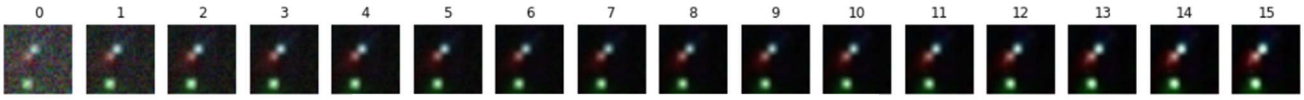
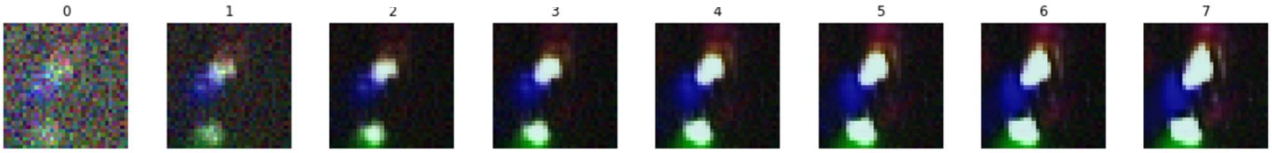


Figure 4: Image over 16 steps



initial iteration of diffusion model

maps, max pooling layer for reducing the size of feature maps, drop out layer for output normalization and finally, Relu Activation layer for adding nonlinearity to the function approximation of the neural net while preventing the vanishing gradient problem. When we stack them in the order Conv2d -> Relu -> MaxPool2D -> DropOut, we have a module that is widely used for image related tasks. Additionally, we can also stack multiple modules together for better performance in training, however, it will increase the training time due to additional parameters. For this reason, in the project, we decided to stack 2 modules since it gave us the best overall result without compromising too much of the training time.

.

2.2.2. Q-Learning Unit

The 'Q-Learning unit' is what differentiates DQN from other CNN used purely for image tasks. This unit consists of a fully connected layer that is used for estimating Q-value of each action a in state s, instead of classifying images to their corresponding class label at the output of the network. In other words, the output of DQN is a vector that contains the Q-value of each action a in state s. The dimension of DQN output depends on the number of actions that the agent can take at any given time. Since, we want our agent to utilize all the image information from the previous layer before estimating its Q-value, a fully connected layer is used so that all image weights can be taken into account.

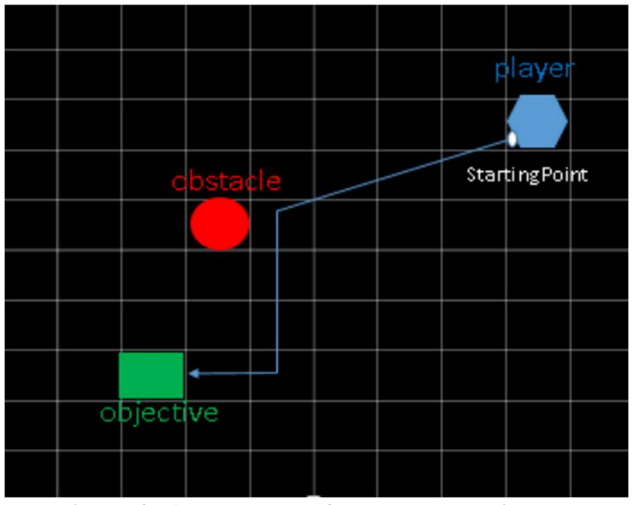


Figure 3: An example of the dream environment

3. Experiments

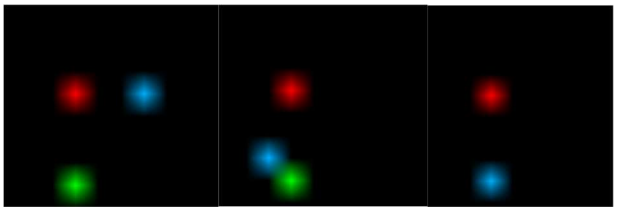


Figure 5: Image describes agent (blue) reaching the destination(green) avoiding the obstacle(red).

For the simulation, we generated a 2D environment as an image, in which each element has specified coordinates within the array of pixels and a specific colorization: red for the obstacle, blue for the player and green for the objective. Any pixels that are not one of the three aforementioned elements will be black (Figure 5).

Action	Reward
Each movement	-1
Colliding with the obstacle	-300
Reaching the destination	+25

Table 1: Rewards associated with each action

Standard functions from the OpenCV library are used to generate these images. We are using synthetic images we generated, rather than real images, in order to keep the project self-contained and allow the agent to control the player to get to the destination.

4. Results

We got the below results described in table 1 by running our simulation for 100 epochs.

Episode	Reward
99	-419
199	-424
299	-313
399	-320
499	-303
599	-295
699	-207
799	-95
899	5
999	19

Table 2: Episode vs Reward Observations

Desired output for this experiment is the successful navigation of the agent toward the goal. Specifically, the agent will be able to find a path and move along that path until reaching the destination without colliding with the obstacle on the way within the step limitation.

References

1] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).

[2] Hester, Todd, et al. "Deep q-learning from demonstrations." arXiv preprint arXiv:1704.03732 (2017).

[3] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Communications of the ACM 60.6 (2017): 84-90.

[4] Niroui, Farzad, et al. "Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments." IEEE Robotics and Automation Letters 4.2 (2019): 610-617.

[5] Sampedro, Carlos, et al. "A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques." Journal of Intelligent & Robotic Systems 95.2 (2019): 601-627.

[6] Zhang, Kaichena, et al. "Robot navigation of environments with unknown rough terrain using deep reinforcement learning." 2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR). IEEE, 2018.

[7] Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018.

[8] Cho, Jang-Ho, et al. "A Real-Time Obstacle Avoidance Method for Autonomous Vehicles Using an Obstacle-Dependent Gaussian Potential Field." Journal of Advanced Transportation 2018 (2018).

[9] Gaya, Joel O., et al. "Vision-based obstacle avoidance using deep learning." 2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR). IEEE, 2016.

[10] Back, Seungho, et al. "Autonomous UAV Trail Navigation with Obstacle Avoidance Using Deep Neural Networks." Journal of Intelligent & Robotic Systems (2020): 1-17.

[11] Pyo, Junghwan, Jiwon Bang, and Yongjin Jeong. "Front collision warning based on vehicle detection using CNN." 2016 International SoC Design Conference (ISOCC). IEEE, 2016.

[12] Xu, Qingyang, Chengjin Zhang, and Li Zhang. "Deep convolutional neural network based unmanned surface vehicle maneuvering." 2017 Chinese Automation Congress (CAC). IEEE, 2017.

[13] https://spectrum.ieee.org/automaton/robotics/industrial-robots/japan-earthquake-more robots-to-the-rescue

[14] Elegoo Smart Robot Car Kit." Github, 3 Mar. 2018, github.com/keopx/Arduino/blob/master/Elegoo Smart Robot Car Kit V2.0.2017.08.21/elegoo-smart-robot-car-2.0.jpg

[15] Zhang, Yinan, et al. "Large-scale Reinforcement Learning for Diffusion Models." arXiv preprint arXiv:2401.12244 (2024).

[16] Black, Kevin, et al. "Training diffusion models with reinforcement learning." arXiv preprint arXiv:2305.13301 (2023).

[17] Berrueta, Thomas A., Allison Pinosky, and Todd D. Murphey. "Maximum diffusion reinforcement learning." Nature Machine Intelligence (2024): 1-11.

[18] Sinha, Samarth, Ajay Mandlekar, and Animesh Garg. "S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics." Conference on Robot Learning. PMLR, 2022.