Improving Output in Generative Models

Tarun Amarnath
S. Shankar Sastry, Ed.

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2024-171
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2024/EECS-2024-171.html

August 9, 2024

Improving Output in Generative Models

by

Tarun Amarnath

A thesis submitted in partial satisfaction of the

requirements for the degree of

Masters of Science

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor S. Shankar Sastry, Chair
Professor Koushil Sreenath

Summer 2024

Improving Output in Generative Models

by Tarun Amarnath

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor S. Shankar Sastry
Research Advisor

8/7/2024

(Date)

* * * * * * *

Professor Koushil Sreenath
Second Reader

8/7/2024

(Date)

Improving Output in Generative Models

Abstract

Improving Output in Generative Models

by

Tarun Amarnath

Masters of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor S. Shankar Sastry, Chair

Generative models have proven widely successful in creating language and image responses to prompts. These techniques have been extended to video creation, robotics, and personal assistants, among other use cases. However, as their pervasiveness grows, a major issue arises: they have no guarantees of correctness. The networks themselves are encapsulated into black boxes, making them difficult to interpret. This work instead aims to improve their output in the pre- and post-processing stages, increasing overall accuracy. The problem is approached through 3 projects: Drone Diffuser (a diffusion-based path planner for drones), RAG Alignment (checking for factual generation of text data in Retrieval-Augmented Generation systems), and IMAGINE (improving output of RAG-based image retrieval).

Drone Diffuser Diffusion models have been successful in image and video generation, leveraging massive amounts of data to achieve remarkable results. Recently, these models have been adapted for the robotics domain, demonstrating advantages such as better performance in long-horizon contexts and more stable training processes. This research extends the application of diffusion models to aerial vehicles. The task is to generate high-level path plans to a goal position denoted by a gate, akin to racing scenarios, in a receding-horizon manner. Two policies are trained: the first, $\pi(\Delta)$, utilizes state information as conditioning to characterize the goal, while the second, $\pi(I)$, directly uses FPV images from a drone. These policies mimic a privileged expert that employs RRT* to generate near-optimal paths. The reverse diffusion process has no guarantees in its output. As a result, both the training data given for imitation and the output from the policy are fit to a polynomial trajectory using minimum snap optimization to ensure dynamic feasibility for quadrotors. The state-based policy performs exceptionally well, achieving a 100% accuracy on every plan in the testing set, whereas the image-based policy requires further refinement. Future work can focus on translating these findings to real-world systems.

RAG Alignment In order to better align Retrieval-Augmented Generation (RAG) systems with intended behaviors and factual consistency, we propose the notion of Fact-Bearing

Terms (FBTs) as the terms in a sentence upon which its factuality rests, such as proper nouns or direct objects. Applying this notion of FBTs, we demonstrate significant performance improvements over both encoding- and part-of-speech- based approaches to text retrieval in RAG systems using sources from a custom dataset created for this task outside of the training distribution of most large-scale models. In addition, we use this metric to demonstrate a visible boost in performance on HyDE (hypothetical document embeddings)-based retrieval after fine-tuning the HyDE model. We show several practical applications of Fact-Bearing Terms, such as warning users of higher risks of hallucination or citing sources.

IMAGINE Within the language domain, HyDE and other techniques prompting the model to retrieve with a guess or estimate of a desired output demonstrate incredible improvements over standard RAG. Selecting the most relevant images to an abstract user input can be a far more challenging task because there can be a large semantic gap between the user's query and desired output: image similarity may not strongly correlate to semantics for many user contexts. In order to better align image RAG systems with intended behaviors, we propose the IMAGINE system, which creates hypothetical outputs and performs retrieval based on those responses. Depending on the user context and resources available, we present IMAGINE-T (text-based) and IMAGINE-I (diffusion-based) multimodal retrieval mechanisms. We present the CameraRollQA benchmark based on real-world images to evaluate both IMAGINE solutions and demonstrate improved performance over CLIP RAG baselines.

To my family, whose unwavering support has been the foundation for everything I have done.

Contents

List of Figures

List of Tables

Acknowledgments

I owe a significant amount of my academic and professional progress to Professor Shankar Sastry, without whom this thesis would not have been possible. Thank you for being my teacher in EECS C106A right after the pandemic. Thank you for guiding me when I took over as head TA of this very class (and its subsequent offering, C106B). And thank you for agreeing to become my advisor for this 5th Year Masters, which has been monumental to discovering my interests and developing my mindset both inside and outside of academics. It was a pleasure to also be advised by Professor Koushil Sreenath, whose steady presence kept C106A on track. I will always be grateful for your reliable mentorship.

My collaborators on projects this year are exceptional individuals who have opened my eyes to new fields - Devan Shanker, Josh Barron, Shrey Aeron, Moaaz Akbar, Anish Gollakota, and Joel Jaison. Without all of you, I would not have discovered new areas of robotics, machine learning, or safety. Thank you for the effort you put in to make our vision a reality.

I would also like to acknowledge my fellow lab members of Cory 337 - C.K. Wolfe, Eric Berndt, Valmik Prabhu, Adith Sundram, Addison Kalanther, Emma Stephan, Nima Rahmanian, Chris Lai, and Daniel Bostwick. We have found joy in our struggle together. Your steady presence at meetings is always appreciated, and I am glad we were able to bring some vibrant life to our time in the lab. Thanks for your technical, logistical, and emotional support while we all juggled so many different projects.

Thank you to my fellow course staff members who made C106AB such a pleasure to teach - Shrey Aeron, Kirthi Kumar, Kaylene Stocking, Eric Berndt, Mingyang Wang, Michael Psenka, Nima Rahmanian, Karim El-Refai, Daniel Bostwick, Anuj Raichura, Charles Xu, Chris Lai, and Martin Zeng. Your dedication to educate students and open up the world of robotics inspired me to put so much of my life into these classes. They ran exceptionally well, and the legend of banana gibbons will live on in the halls of Cory 105. Also, thank you to my fellow JamCoders TAs for showing me the joy of the service we do. Education is one of the most valuable activities I have been involved in and will always hold great significance in my life.

The people around me throughout the years have been my support system, and there are far too many of you to list. You have been invaluable in both forming the wonderful experiences I have had and supporting me during the low points. To all of my friends - whenever and wherever I may have met you, your story has contributed to mine in an incredibly positive way.

Rohit, Anirban, Saajid, Bryan - thank you for being the hilarious, friendly, supportive people you are. Derek, Rob, Ava - odes to Bonita Base Bouse will be sung for generations. Devan and Ana - I cannot fully express how much both of you mean to me, but I think you understand. Finally, my family - thank you for being the bedrock of my growth. This thesis would not have happened without any of you.

Chapter 1

Drone Diffuser: Diffusion-Based Path Planning for Drones

1.1 Introduction

Background

Generative models have made significant advancements in the creation of text and visual data. Diffusion models in particular have found success in the image and video domains after training on the massive corpus of data available on the Internet. The technique particularly excels at replicating complex data distributions and creating diverse outputs, which are desirable in visual contexts but applicable to other areas as well [1].

The field of robotics has adapted the method for path planning. Approaches with diffusion vary. Some modify camera images to create subgoals that can be tracked using a classical reinforcement learning planner [2], while others generate trajectories based on conditioning data that can be followed either directly or in a receding-horizon manner [3][4].

In the latter case, several advantages can be seen when using diffusion over classical control or reinforcement learning. Because the technique performs sampling over the gradient field of the action score function, it inherently sees greater success representing the multimodal distributions often present when creating path plans [5][6]. Evaluating the loss of the gradient also achieves more stable training. Additionally, diffusion models for robot action generation perform better in long-horizon contexts, necessary for robotic planning, as the error of full trajectories are evaluated rather than calculating single-step rewards. Diffusion models can compose multiple tasks together, thereby using training data effectively to generalize to new situations.

Path planning with diffusion has not yet been extended to aerial vehicles, which this work aims to do. Quadrotors in rescue scenarios, for example, are left to human experts who can deftly fly through obstacles without compromising the safety of the payload or the drone. An effective autonomous controller would have the capability to generalize to different locations, seamlessly adapting to various environmental conditions and obstacle configurations.

This would involve not only precise maneuvering skills but also real-time decision-making, ensuring the feasibility, safety, and efficiency of accomplishing the mission in diverse and unpredictable scenarios. Diffusion models in conjunction with provable optimization techniques have the potential to make this happen.

One situation with UAVs where autonomous controllers work very effectively is the field of racing. Several methods have been proposed to fly a quadrotor through multiple gates as quickly as possible before reaching the finish line [7][8]. None use diffusion models, both because of the novelty of the technique and because they require millisecond-level computation times. Some papers separate perception and control, with one or both parts of the pipeline performed with classical approaches, while others leverage end-to-end networks that directly translate sensor inputs into action space commands [9] [10]. This chapter experiments with diffusion-based planning with both a separate perception preprocessing stack and a combined sensor-to-output network.

Diffusion models perform well with a large, diverse training set, as seen in the efficacy of image models like DALL-E [11]. Collecting data for Drone Diffuser has several limitations if performed in the real world. The intensive effort required, poor sample efficiency because of limited variance in training data generation, and resulting difficulty in dealing with variance of the goal make empirical data collection unsuitable to the task. As a result, Drone Diffuser makes use of domain randomization within a high-fidelity simulator [12] to generate a training dataset.

A major issue with diffusion models is an absence of true guarantees in their output; we have no certainty that, even if trained on many examples of drone trajectories, the output of a drone diffusion policy will be feasible by a quadrotor. This does not pose a problem for the robot paths performed in earlier work, as inverse kinematics solvers can calculate joint angles to achieve any position in the reachable workspace. Diffusion policies do have the advantage of temporal locality [3], meaning they can make decisions about the action at a given timestep while considering the actions at both the next and previous timesteps; however, this still does not guarantee feasibility.

A key insight in the physics of quadrotors is their differential flatness property [13]. This implies that all states and control inputs are expressible using a fully controllable output and its derivatives. In the context of drones, Mellinger and Kumar prove that the position and heading can be used as these flat outputs. Additionally, they show that a polynomial trajectory that minimizes snap, the fourth derivative of position, generates smooth and dynamically feasible paths. A minimum snap optimization can be layered on top of a higher-level path planner to guarantee viability on the physical system.

Approach

This chapter proposes a method to perform diffusion-based path planning for aerial vehicles while taking into consideration the feasibility of those trajectories. The objective is to generate a high-level path plan (composed of x, y, z, and yaw displacements) from a starting position to a goal denoted by a gate, akin to those in drone racing environments, for which control inputs can be calculated and tracked in a receding-horizon manner. While racing scenarios serve as a reasonable starting point for development, the overarching goal is not to achieve the fastest lap time but to determine whether diffusion is viable as a planner for quadrotors. The technique's strength is generalizing to novel situations, which would show up more dramatically in real-world settings like rescue operations.

Conditioning Data

Drone Diffuser generates output based on two modalities of conditioning data (incorporated into the policy using FiLM conditioning).

- The first type contains state information that directly characterizes the goal, which specifically is displacement of the drone body frame from the gate at the time of the query.
- The second kind is FPV images taken from the drone. The AirSim simulation environment [12] allows for collection of high-fidelity training data, moving a virtual quadrotor through space and collecting images from its onboard camera. An encoding generated by a different but similar policy, DroNet [14], transforms the image into vector space for conditioning Drone Diffuser.

Privileged Expert

The training data for paths comes from a privileged path planning expert with access to full state information. Specifically, an RRT* planner generates paths from the starting position to the center of the gate. To incorporate feasibility of drone trajectories into this model, the waypoints generated by RRT* are optimized using minimum snap trajectory generation.

Policy Outputs

Drone Diffuser outputs paths in a receding-horizon manner, effectively planning to replan; given a conditioning input, it generates waypoints for the next 8 timesteps, after which another observation is taken and further waypoints are returned. The results created by the diffusion planner are also optimized to fit a polynomial that minimizes snap, translating to feasibility on a physical quadrotor model.

Contributions

The contributions of this work are

- Translating diffusion-based trajectory generation to quadrotors by performing behavioral cloning of an RRT*-based privileged expert

- Developing and testing conditioning based on a) state information fed to the model and b) FPV images
- Guaranteeing dynamic feasibility through minimum snap optimization

1.2 Related Work

Diffusion

Diffusion [15], which began as a concept in thermodynamics, [16], allows for high-quality samples in high-dimensional spaces that have intractable distributions to sample, such as images. The algorithm iteratively adds noise to a known sample in the form of a Markov chain. Then, a network is taught to remove the noise to retrieve the source image in an iterative process. The model learns to estimate the original data distribution.

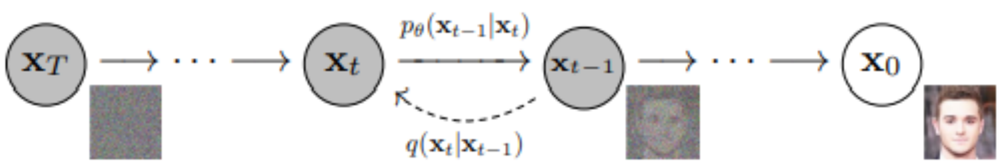


Figure 1.1: Illustrated diffusion process. Image taken from Ho et. al. [15]

The Forward Diffusion

The forward diffusion process is a fixed, known Markov chain. The forward diffusion kernel is defined by a transition function $q(x_t|x_{t-1})$. Gaussian noise is added at each timestep in the Markov chain to x_{t-1} to produce a new latent variable x_t. The Gaussian noise added at each step of the chain is based on a variance scheduler with (either learned or fixed) constants $\beta_1 ... \beta_T$. Specifically,

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

The PDF of our full forward diffusion process can be defined as

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1})$$

Intuitively, we can model this same process as Gaussian noise slowly being added through our Markov chain according to variance scheduler β:

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z_t$$

where z_t is sampled from $\mathcal{N}(0, I)$.

Naturally, because we know the parameters of each step in our diffusion throughout the Markov chain and we know that they are Gaussian, we come to the conclusion that we do not actually have to step through the chain and can instead sample directly using the reparameterization trick:

- $\alpha_t = 1 - \beta_t$
- $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$
- $\epsilon_i \sim \mathcal{N}(0, I)$

Redefining our forward process using our new notation, we have

$$\begin{aligned} x_t &= \sqrt{\alpha_t}x_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1} \\ &= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{(1-\alpha_t) + \alpha_t(1-\alpha_{t-1})}\bar{\epsilon}_{t-2}, \text{ summing up the variances} \\ &= ... \\ &= \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \end{aligned}$$

We have $\beta_{1...T}$ as hyperparameters, allowing us to compute our $\alpha_{1...T}, \bar{\alpha}_{1...T}$ and thereby the results of the diffusion at any arbitrary timestep. Finding the β vector tends to be done using linear interpolation, cosine interpolation, or in a learned manner [17].

Reverse Diffusion

Overview

The reverse diffusion process aims to move our noisy data back into the initial distribution. The reverse of a Gaussian diffusion process can also be modeled as a Gaussian for sufficiently small variance β_t [18]. As a result, we model each step of the reverse diffusion process as

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

μ_θ and Σ_θ are learned parameters. The distribution for the full reverse diffusion process then becomes

$$p_\theta(x_{0:T}) = p_\theta(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$$

Loss Function

The goal is to train a model to recover the parameters of this reverse diffusion process while conditioning on timestep t to ensure we get the correct outputs. Specifically, we want to maximize the likelihood (or log likelihood) of our training data. We can accomplish this alternatively by minimizing the negative log likelihood. Our loss therefore takes the form

$$L = -log(p_\theta(x_0)), \;\; p_\theta(x_0) = \int p_\theta(x_{0:T})dx_{1:T}$$

This integral is intractable, however. Instead, we use the following inequalities to reformulate our objective:

$$\mathbb{E}[-\log p_\theta(x_0)] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right] = \mathbb{E}_q\left[-\log p(x_T) - \sum_{t\geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right]$$

This equality holds because a Gaussian PDF must be ≤ 1. A more formal justification for this can be found using Jensen's inequality or by breaking the problem down using KL-divergence (a measure calculating the distance between two probability distributions). A full proof is available in the original papers or in [19]. We reformulate our expectation to the following:

$$\begin{aligned}
\mathbb{E}_q&\left[D_{KL}(q(x_T|x_0)||p(x_T)) + \sum_{t>1} D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) - \log p_\theta(x_0|x_1)\right] \\
&= \mathbb{E}_q\left[L_T + \sum_{t=1}^{T-1} L_t + L_0\right]
\end{aligned}$$

where we define each of the L terms as

$$\begin{aligned}
L_T &= D_{KL}(q(x_T|x_0)||p(x_T)) \\
L_{t-1} &= D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) \\
L_0 &= -\log p_\theta(x_0|x_1)
\end{aligned}$$

Reversing the Steps

Recall that we can approximate our reverse diffusion steps also as a normal distribution. Specifically, we can calculate $q(x_{t-1}|x_t, x_0)$ as a normal distribution. Intuitively, it should make sense that we are conditioning on x_0 because we need to know where we are trying to reach in the long term. Here are the mean and standard deviation terms for that reverse diffusion Gaussian:

$$\begin{aligned}
q(x_{t-1}|x_t, x_0) &= \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I) \\
\tilde{\mu}_t(x_t, x_0) &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \\
\tilde{\beta}_t &= \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t
\end{aligned}$$

We saw above that we can define x_t in terms of x_0, which means we can also define

$$x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon), \epsilon \sim \mathcal{N}(0, 1)$$

As a result, we can define $\tilde{\mu}$ as

$$\tilde{\mu}_t(x_t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$$

Learning the Reverse Diffusion Process

We could approximate $\tilde{\mu}_t(x_t)$ directly to get the expected value for a step of the reverse diffusion (i.e. the mean for x_{t-1}). However, we know x_t at both training and test time; the only thing we do not know is the noise term (we are trying to slowly remove noise to get to the original, and we want our network to predict this noise). As a result, we use our neural network to predict noise. We reformulate our mean to be the following:

$$\tilde{\mu}_t(x_t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right)$$

Our network predicts the noise term ϵ directly - the noise that we would have to remove in order to get to the next step of the algorithm. We sample x_{t-1} using this noise term as the mean value. Recall though that the reverse diffusion process is a normal distribution, so we must also calculate variance. The variance above $\tilde{\beta}_t$ works, but Ho et. al. find that simply using β_t as the variance works just as well.

To train the model, we have some x_0 from our training set. We add noise to it according to ϵ at some uniformly random step of the diffusion process. Then, we have the network predict the noise it has to remove, minimizing the loss function.

In the loss function, we can ignore the L_T term because it's constant due to our variance schedule. We will ignore L_0 for our reverse diffusion process because the Ho et. al. calculate the final step separately from the rest of the process. We are just left with L_{t-1}, which equals the following:

$$L_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}||\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)||^2\right] + C$$

Subtracting out the constant term and simplifying due to the fact that our network is now predicting ϵ, we get

$$\mathbb{E}_{x_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma^2\alpha_t(1-\bar{\alpha}_t)}||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)||^2\right]$$

The authors find that sample quality improves if we remove the constant term in the front and used the following loss:

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,x_0,\epsilon}\left[||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)||^2\right]$$

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Figure 1.2: The training and sampling algorithms from [15]

Conditional Diffusion

Generally, we want to perform reverse diffusion given a particular conditioning value. Mathematically, we are trying to maximize $p_\theta(x_0|y)$, where y is our conditioning variable.

Classifier-Guided Diffusion

We can train a second classifier to predict the probability that the given image is the class that we want it to be, $f_\phi(y|x_t)$.

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
$\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
$x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma\, \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$
end for
return x_0

Figure 1.3: Classifier guidance [20]

Increasing s improves sample quality at the cost of diversity. We could (in addition to classifier guidance, and perhaps more intuitively) also train a diffuser that predicts $\mu_\theta(x_t|y)$, which improves results further.

Classifier-Free Guidance

Classifier-free guidance does not require a secondary classification model trained, which has natural limitations in terms of creating text classification [21]. Instead, it uses a normal conditioned diffusion model along with 10-20% dropout during training, where the label is replaced with the null label (effectively the training for an unconditional model). Then, for sampling, the model is guided toward some caption c using the following expression:

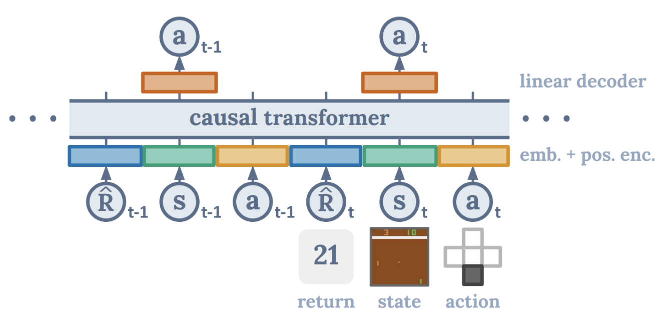
$$\hat{\epsilon}_\theta(x_t|c) = \epsilon_\theta(x_t|\emptyset) + s \cdot (\epsilon_\theta(x_t|c) - \epsilon_\theta(x_t|\emptyset))$$

Generative Models for Robotics

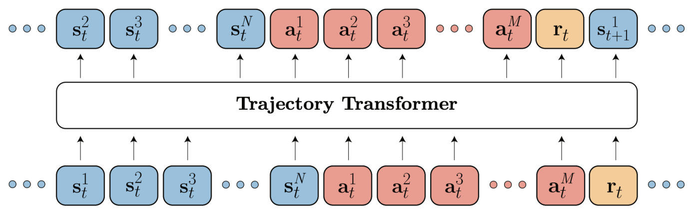
Generative models have been used in conjunction with robotics. Different authors approach this challenge in unique ways. Some diffusion policies generate sub-goals for the robot with image editing models, such as [2], but this project aims to predict the trajectory directly, which will be the focus of this section.

Transformers and RL

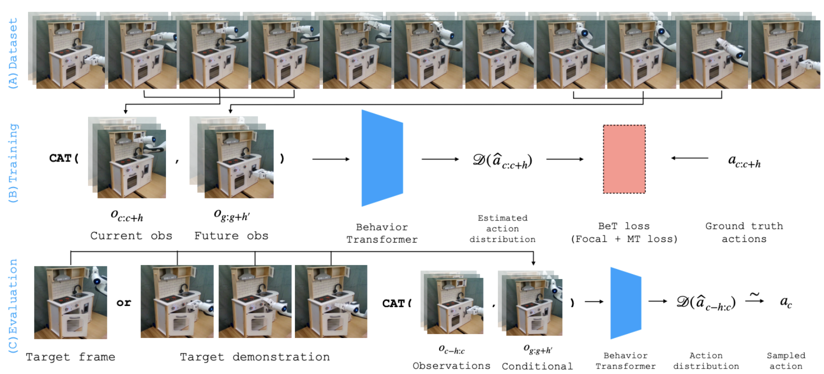
The first steps to apply a generative model lens to reinforcement learning use the transformer architecture [22]. Rewards-to-go, actions, and states are fed into a causal transformer, which then outputs the next action to take. The transformer is trained on offline data and compared to an offline reinforcement learning approach (TD learning using CQL). It is also compared to direct behavioral cloning. The results show success in matching, or in some environments exceeding, state-of-the-art methods in the field.



Simultaneously, Janner et. al. propose a different version of a transformer-based sequence planner called Trajectory Transformer [23]. The ideas from natural language processing are directly translated into sequence planning, and the architecture maintains a very similar structure. This paper performs a beam search, selecting the most likely state/action value, and then evaluates the highest-reward trajectories. The probabilities are calculated using a sequence transformer. The desired final state in the trajectory is actually placed at the start of the input sequence to guide the transformer into choosing to move in the correct direction without modifying the underlying architecture. These authors discretize the state, action, and reward space because of the nature of the transformer model. This effectively performs offline RL with behavior cloning with some goal conditioning. The authors show that it demonstrates success in long-horizon, sparse-reward tasks.

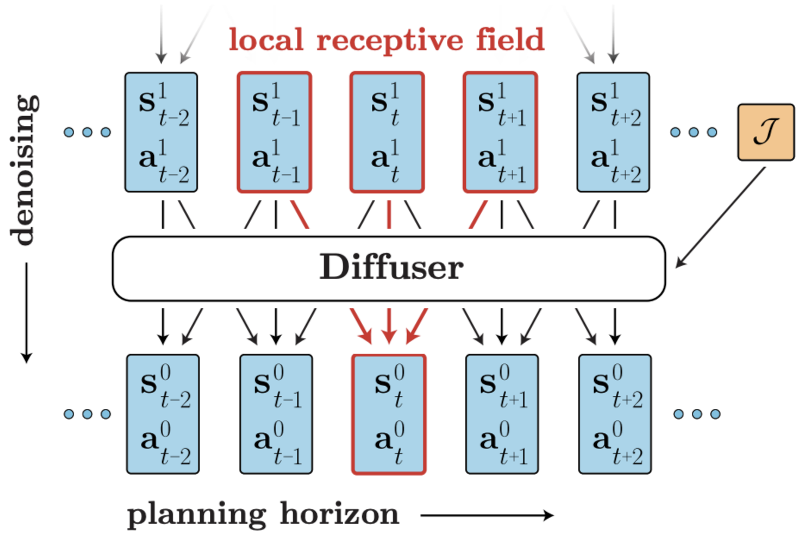


Another transformer-based generative model for trajectories called BeT (Behavioral Transformers) [24] applies k-means clustering to discretize actions. An extra head is then added to the transformer to make the actions continuous using an offset from the cluter centers. This is built upon in the following months with C-BeT, or Conditional Behavioral Transformers [25]. The new paper combines the ideas of the behavioral transformer with previous work on conditioning. The sequence model takes in the most recent observations as well as desired observations and, in a sequence-to-sequence manner intrinsic to the architecture, generates an action trajectory meant to move from the observations to the goals. A diffusion-based extension of this paper attempts to imitate human behavior with the idea of k-means and a residual [26].



Diffuser

Diffuser introduces diffusion to trajectory planning [3]. The general design of the planner is as follows:



The authors highlight four main properties of a diffusion-based model that makes it superior for trajectory planning: 1) long-horizon scalability (because the model trains for trajectories over single steps), 2) task compositionability (compose multiple rewards), 3) temporal compositionability (stepping ahead of classical behavioral cloning by combining multiple sequences), and 4) non-greedy planning (like above, generalizing to sparse-reward, long-term sequences).

The model used in the paper has a few advantages over existing works. First is the idea of temporal ordering, in that unlike model-based methods, the full trajectory must be generated at once (instead of working autoregressively, generating one state at a time). Temporal consistency is enforced; by expanding the receptive field of the generation of a single timestep to include the prior and subsequent steps, the model can enforce local consistency, which, when extrapolated, creates global consistency. The design of the inputs and the outputs of the model are a two-dimensional array (a trajectory equivalent to standard image diffusion models), with the timesteps across the first dimension and the concatenated state/action pair along the second.

A key insight of the paper comes from the method to maximize the reward function. A binary indicator random variable O_t is set to equal 1 if timestep t of a trajectory is optimal. We want to sample optimal trajectories, so we have

$$\tilde{P}_\theta(\tau) = P(\tau | O_{1:T} = 1)$$

Reformulating with Bayes Rule, we end up with

$$\tilde{P}_\theta(\tau) \propto P(\tau) P(O_{1:T} = 1 | \tau)$$

This follows the form of conditional diffusion.

For reasons described in [27], we set $P(O_t = 1) = exp(r(s_t, a_t))$. For classifier-guided conditioned diffusion, we add an additional term that depends on the log probability of the classifier 1.3. The gradient is taken:

$$\begin{aligned}\nabla_{x_t} logP_\phi(y|x_t) &= \nabla_\tau logP(O_{1:T}|\tau) \\ &= \sum_{t=0}^{T} \nabla_{s_t,a_t} r(s_t, a_t) \\ &= \nabla J\end{aligned}$$

As a result, the classifier guidance is the gradient of the reward function, for which a separate model is trained. New reward functions are easily incorporated. Actuation is done in a receding-horizon fashion. Constraints on the trajectory come from setting states to the desired values after every iteration in the denoising process.

The authors highlight a few key properties of diffusion planners. First, because planning and sampling are merged, predicting and planning in a long-horizon manner are coupled. Additionally, the generative process can create new trajectories by merging known trajectories. The length of the trajectories generated is variable. New reward functions can be easily incorporated into the formulation of the model.

A few different experiments are given: maze solving, block stacking, and offline RL. One conclusion drawn is that Diffuser's effectiveness comes from coupled modeling and planning, not prediction accuracy.

Diffusion Q-Learning

Wang et. al. [28] introduce Diffusion Q-learning as a way to perform policy regularization in offline reinforcement learning in an alternate manner to Janner et. al. The deviation of generated policies from the training set is regularized, preventing significant divergence from exhibited behavior (contrasting from the Janner paper, which does everything from a model-based planning perspective as opposed to a model-free policy-optimization perspective). The loss function for the diffusion model contains two parts: one to mimic the training set (effectively the training data, becoming the regularization term) and one to sample high-value actions from a learned critic.

Decision Diffuser

Ajay et. al. [29] use conditional diffusion models, where constraints and skills are the conditioning variables, to perform trajectory generation, stitching together sub-optimal trajectories to create an optimal output, potentially performing dynamic programming implicitly. The model is trained with a dataset of reward-labeled trajectories. They perform classifier-free guidance. Their work can combine skills and constraints in novel situations.

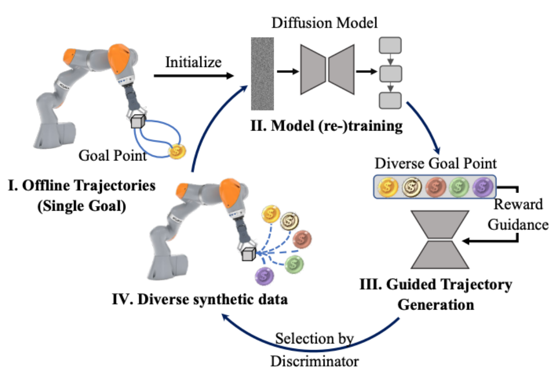
A classifier-free conditional diffusion is set up with the condition $y(\tau)$ being some information about the trajectory, such as return, constraint, or skill it satisfies. Because states are more continuous than actions, the authors decide to diffuse only states; actions between consecutive states are interpolated using some inverse dynamics model (learned with the same offline data as diffusion). To determine $y(\tau)$, the authors consider training a model to predict rewards from the trajectory, but they choose not to because of its complexity, instead conditioning on the returns in the offline dataset. Classifier-free guidance with low-temperature sampling extracts high-likelihood trajectories that correspond to the best behaviors. Receding-horizon control is used. The authors also mention that $y(\tau)$, instead of being the expected rewards, can be a one-hot encoding of constraints satisfied.

The paper's results are excellent. They outperform Diffuser on and surpass some offline RL algorithms as well.

AdaptDiffuser

AdaptDiffuser allows for better generalization to new tasks [30]. The methods are very similar to Diffuser [3]. The authors improve the original paper by using the diffusion model to create many trajectories using different reward functions, generating synthetic data. Not all of this data is necessarily a) physically possible, or b) high-reward. Therefore, these generated trajectories are plugged into a rule-based discriminator before being added to the training set.

To deal with the feasibility of state tracking, only states are generated through the diffusion model. Then, an inverse dynamics model derives executable actions $\tilde{a}_t$, and the next state is revised to $\tilde{s}_t$. Trajectories with too large of a difference between the original s_t and $\tilde{s}_t$ are discarded. Reward is calculated using the remaining ones, and only those with high reward are kept in the dataset.



Diffusion Policy

Similar to prior work, this paper uses diffusion to create actions for a robot. The reverse diffusion process is transformed using FiLM conditioning with visual data and does not utilize a reward function gradient like Diffuser or AdaptDiffuser. The policy does not predict future states, which speeds up diffusion and improves the accuracy of generated actions.

The authors experiment with two different architectures for predicting ϵ_θ:

1. CNN-based: This follows from [3], and observation conditioning is added using FiLM (a technique that involves modifying the output of each layer of the CNN using a predicted affine function [31]). However, it does not work very well for high-frequency action sequences, such as velocity commands.

2. Transformer: The minGPT transformer is used for action prediction. Input tokens (the current actions with noise) and the positional embeddings are passed in, and the observation encoding is used as the input feature.

Diffusion Policy achieves state-of-the-art results, and the architecture is the basis for this chapter's reverse diffusion codebase.

Drone Navigation

Minimum Snap Trajectory Generation

Mellinger and Kumar [13] propose minimum snap trajectory generation and control as a method to guarantee convergence and stability of quadrotor paths when a CLF cannot be formulated or a linearized model is not fully controllable. They explicitly move beyond the small angle approximations common to drone controllers up until that point for motion in three dimensions that require significant deviations from the hover state. Their algorithm satisfies constraints on system-specific input limits as well as obstacles present in the environment.

The basis for the work rests on the differential flatness of quadrotors. Specifically, states and control inputs can be formulated in terms of flat outputs and their derivatives; these outputs are $[x, y, z, \psi]$, where ψ represents the yaw angle of the drone. These outputs are solved for using a quadratic program whose cost function minimizes snap, or the fourth derivative of position. Once the values for the outputs are returned, the remaining states and control inputs can be calculated. While the optimization problem can take a significant number of cycles, future research [32] has worked on reducing computation time of the algorithm.

Racing Approaches

Autonomous drone racing has seen many different approaches to the problem [9]. The problem has many different aspects to it, including drone modeling, perception, planning, and control.

The classical stack breaks down perception, planning, and control into different submodules and tackles each one individually. On the planning front, a majority of approaches fall into the categories sampling-based and combinatorial methods, both of which take advantage of the differential flatness of quadrotors. Minimizing snap starts off supreme, but racing teams have experimented with other methods that are more optimal to fast lap times as opposed to dynamic feasibility of waypoints.

More recent learning-based approaches employ neural network architectures for perception, planning, and control. Different papers combine portions of the stack in different ways. Some leave all three separate [33][34][35], some combine perception and planning [36][37][38][39], and others join planning and control [40][41][42], including the state-of-the-art racing algorithm [8]. End-to-end learning, where networks take sensor data as input and output control inputs, are rare because reliable methods exist for parts of the pipeline, and relying on an ML black box does not necessarily promise success. Some however have made the algorithm work [43][44][10].

Other Path Planning Techniques

Other path planning papers that have inspired this work include forming methods for high speed flight in novel environments, avoiding obstacles while following a given vector as best as possible [37]. This paper uses depth images in conjunction with the current state of the drone to construct a series of waypoints for the quadrotor to follow. These are projected onto the space of polynomial trajectories based on minimum snap optimization, and a model preditive controller tracks the paths.

Other key papers map environment images to paths. Dai et. al. use a CNN architecture to predict a steering angle and collision probability from environment pictures [45]. They then transform the steering angle to a yaw angle to best avoid obstacles. Bhattacharya et. al. employ vision transformers, key components of generative networks today [46]. Depth images of the environment, along with the current state of the drone, are passed to a policy, which is learned using behavioral cloning, that outputs linear velocity commands for the drone. A minimum-snap trajectory is formed from these velocity waypoints that is then tracked with a geometric controller. The paper's overarching goal is to build a reactive policy that can fly through obstacles in a straight line, establishing a proof-of-concept for ViTs as quadrotor planners; they do not aim to race as fast as possible, making the work most comparable to Drone Diffuser.

1.3 Methods

Task Formulation

Inspired by the AirSim Drone Racing Lab [47], which is designed to provide a racing testbed in the AirSim simulator, the goal for this chapter is to navigate a drone to a virtual gate in the environment. The task formulation derives from [46], [8], and [4].

A privileged expert $\pi_{\text{expert}}(s_t, g) = s_\tau$ takes as input the starting state of the quadrotor s_t along with the location of the goal frame with respect to the quadrotor body frame g. It returns a near-optimal and feasible trajectory for the quadrotor to follow through the environment. A dataset $\mathcal{D}$ is collected following the expert policy, with each state $s_t = \pi_{\text{expert}}(s_t, g)_t$. The image from the quadrotor is recorded at each position in time, $\mathbf{im}_t$.

Two different student policies are trained. The first $\pi(\Delta)$ takes as input the displacement from the quadrotor body frame to the goal at a particular instance in time, with the assumption that a separate thread calculates this value based on either SLAM or ground-truth input from a controller network. The second $\pi(I)$ takes as input a single FPV image taken by the quadrotor at that moment in time. The policy, operating as a receding-horizon planner, outputs displacement actions for the next 8 timesteps, $a_{t:t+7}$, in the form of $[\delta x, \delta y, \delta z, \delta yaw]$. This is learned from the privileged expert in a behavioral cloning fashion using MSE loss:

$$L(\theta) = \mathbb{E}_{\tau \sim \mathcal{D}}[\frac{1}{T}\sum_{t=0}^{T-1} ||\delta_{\text{expert}}(t) - \delta_{\text{pred}}(t;\theta)||_2^2$$

Both the inputs by the privileged expert and the outputs from Drone Diffuser are passed through a minimum snap trajectory optimizer for feasibility guarantees.

Environment Creation

AirSim

Testing of Drone Diffuser takes place within Microsoft AirSim [12]. Environments can be made using Unreal Engine and loaded into the simulator. A vehicle, by default either a multirotor (used for this project) or a car, is placed at a defined starting position. The physics engine calculates interactions with the environment as the vehicle moves through space. AirSim provides APIs designed to monitor and control the vehicle. `Get` methods include measuring speed, evaluating position, and querying camera data located either on the multirotor or from a 3rd person point of view. `Set` methods allow for programmatically defining the vehicle's speed, position, or rotation values.

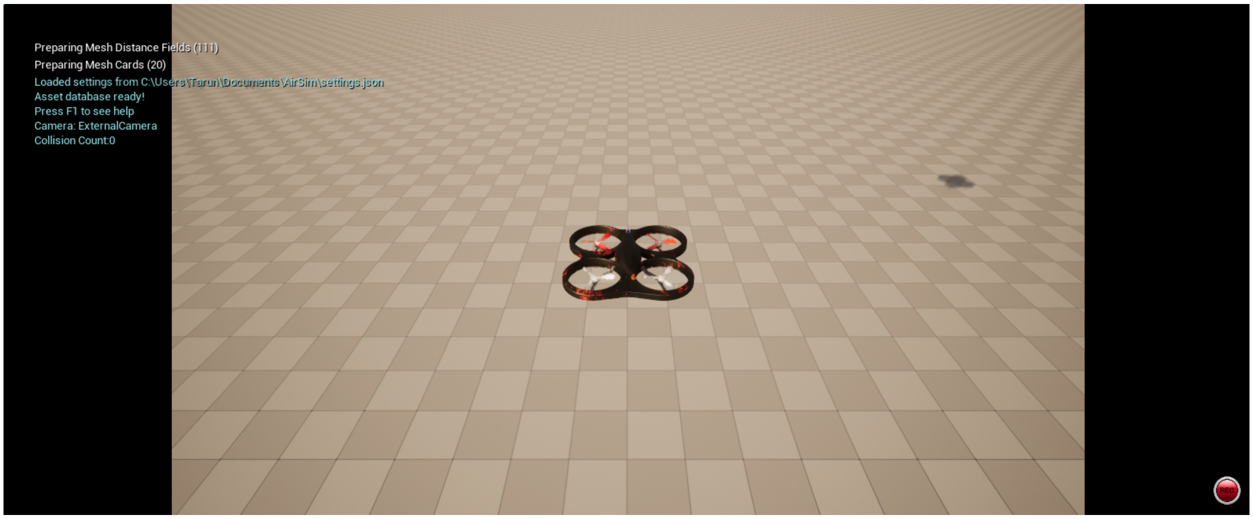


Figure 1.4: An Unreal Engine environment loaded with the multirotor vehicle from AirSim.

Blocks Environment

The environment being used for this project is a modification of the default Blocks Unreal Engine environment distributed by the AirSim library. It is chosen because of its uniform structure outside of the objects of interest for the task at hand. Confounding variables as a result of the environment, such as non-uniform lighting, can lead to positive results in a visual policy without an understanding of the actual goal. The existing blocks in the environment are removed, to be replaced by new obstacles and the goal.

Diffusion Policy

The backbone code for the dataset parser and diffusion process is adapted from Diffusion Policy[1][4]. The input to the policy consists of the current state observation, or the conditioning value. The output is the series of actions that the robot, or in our case, the drone, is to perform for the next T_a steps. The actions start as noise, similar to general image diffusion models; the policy uses the observation conditioning to diffuse this noise into a reasonable sequence of actions.

Prior to being passed through the diffusion process, the conditioning value is normalized. The network for predicting each diffusion step follows a U-Net architecture, with each block of the network downsampling and upsampling consisting of convolutional neural network-based predictors. FiLM conditioning [31] is applied on the conditioning value at each layer to influence the output depending on the current state. A separate model maintains an

[1] https://github.com/real-stanford/diffusion_policy

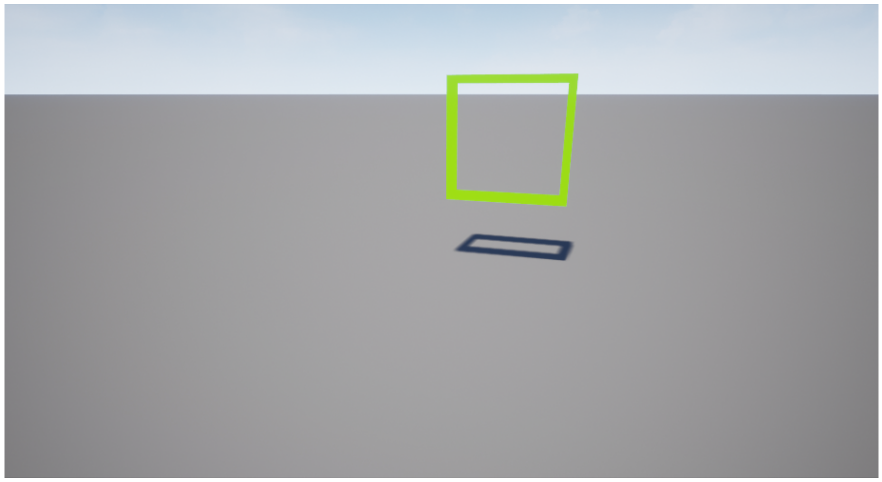


Figure 1.5: An example gate placed in the blocks environment.

exponential moving average of the model weights as training happens to increase stability of the results values. Hyperparameters used for the policy are included in A.2.

Data Generation Pipeline

Takeoff Dataset

The initial experiment serves as a sanity check for both the dataset generator and the diffusion policy code using a straightforward task. AirSim includes a built-in takeoff sequence for its multirotor aircraft model. The drone takes off from its starting location and hovers approximately 5 meters above the ground. The simulator offers functionality to record data from any sequence of actions at user-specified intervals. For this experiment, data from a takeoff-then-land sequence is recorded every 0.01 seconds, capturing velocity, positional data, and image data. This recorded data forms the initial dataset for training the diffusion model.

Gate Creation

In typical drone racing scenarios, gates take the form of virtual rectangles that quadrotors must fly through in succession in the fastest possible time. Rather than working with the complexity of a succession of gates, for exploratory purposes, a singular gate is placed at a random location 20-40 meters ahead of the starting position, up to 10 meters left or right, and 5-15 meters above ground level. The variability of the gate position allows for the creation of a diverse training set. The passable area of the gate is 4x4 meters, comparable to typical sizes in similar scenarios. The color is a bright green to distinguish the gate from the surrounding environment; typical gates have green, red, or blue checkered patterns.

The drone begins 10 meters above ground level and must use Drone Diffuser to fly through the gate. Checking whether the quadrotor has successfully reached the target is done using the following process:

Assume we use the following notation to represent points along the rectangular prism of the gate:

- P_1: bottom-left-front
- P_2: bottom-left-back
- P_3: bottom-right-front
- P_4: top-left-front

We define the following vectors between those points:

- $u = P_2 - P_1$
- $v = P_3 - P_1$
- $w = P_4 - P_1$

Additionally, say the position of the drone is d. If all of the following conditions hold true, then the drone has successfully reached the gate:

- $u \cdot P_2 > d \cdot u > u \cdot P_1$
- $v \cdot P_3 > d \cdot u > u \cdot P_1$
- $w \cdot P_4 > d \cdot u > u \cdot P_1$

Privileged Expert

In order to build a drone diffusion policy, which can generalize to new trajectories, we require sample paths to train on. Past work, including Diffusion Policy, has relied upon imitating the movements of a human. While transferable to flying a multirotor as well, a more consistent and generalizable method is desirable to allow for both more iterations and a larger dataset.

Higher-Level Planner

Initially, sample paths are generated using the RRT (rapidly-exploring random trees) algorithm [48]. The position of the quadrotor and the center of the gate are given as inputs for the start and goal. RRT is particularly well-suited for this task due to its ability to efficiently explore high-dimensional spaces and find feasible paths in complex environments. The algorithm works by incrementally building a tree that explores the space by randomly sampling points and extending the nearest existing tree node towards the sample point. This results in a tree structure that rapidly covers the search space, providing a path from the start to the goal.

The algorithm to sample paths is then changed to RRT*, an extension of the original RRT algorithm that guarantees asymptotic optimality [49]. Unlike RRT, which focuses on finding any feasible path quickly, RRT* aims to find the optimal path by continuously improving the quality of the path as more samples are added.

RRT* works by not only extending the tree towards the random samples but also by rewiring the tree to ensure that the path cost from the start to the new sample point is minimized. This rewiring process involves checking the potential new paths through each neighbor and updating the tree to include the most cost-effective routes. Over time, it ensures that the path taken by the drone is near-optimal in terms of length and smoothness. This is crucial for high-speed navigation, where minimizing travel time and avoiding sharp turns can greatly enhance performance and safety.

Parameters for the RRT* algorithm are included in A.1.

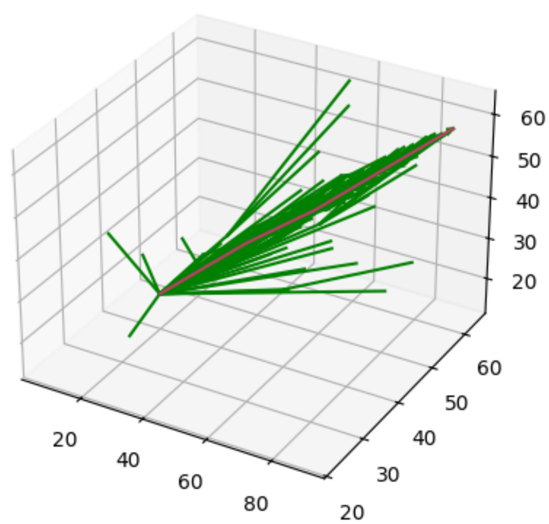


Figure 1.6: An example path generated using the RRT* algorithm.

Feasible Trajectories

While the RRT* algorithm generates a reasonable path plan, the trajectories need to be optimized for quadrotor motion. Because quadrotors are differentially flat systems, given the flat outputs of the system (specifically x, y, z, and yaw), the necessary control inputs (thrust and moments) can be computed. As a result, we can work directly with these outputs to determine our trajectories.

Mellinger and Kumar propose an algorithm [13] to create smooth and dynamically feasible trajectories by minimizing the integral of snap, the fourth derivative of the position,

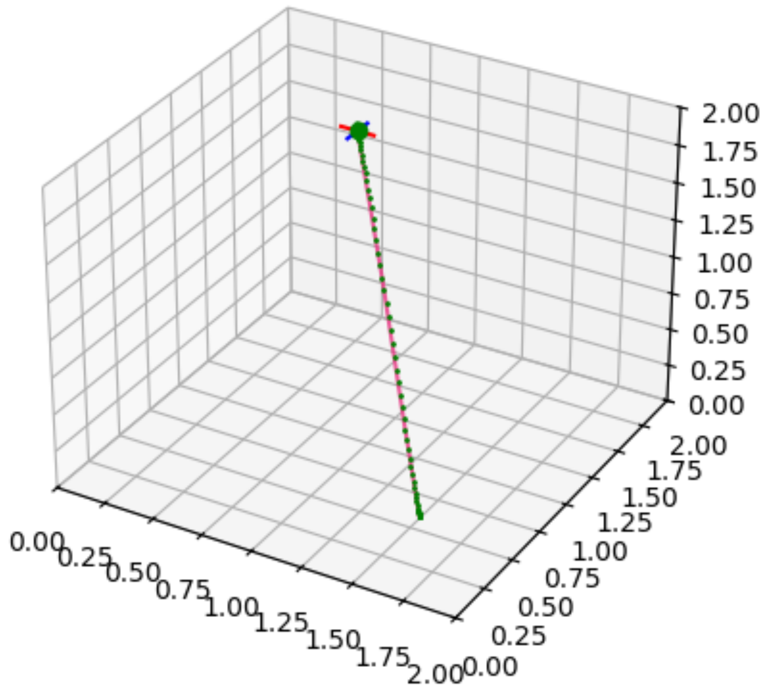


Figure 1.7: A scaled-down visual of the RRT-generated path (solid line) and the path that the drone follows after minimum-snap trajector optimization (dotted line).

along the path taken by the quadrotor. The RRT* algorithm generates waypoints from the starting position to the goal. A quadratic program computes the optimal trajectory given the waypoint constraints and the snap-based cost function. This step smooths the RRT* trajectories and ensures training inputs into the diffusion model match those feasible by a quadrotor. The quadratic program additionally outputs thrust and moments, but the adjusted position and yaw data are used as inputs for the diffusion model, as it is meant to be a higher-level planner.

The diffusion itself has no guarantees of generating feasible outputs, although by preprocessing training data with the quadratic program, the process is more likely to mimic dynamically smooth trajectories. In order to guarantee that the quadrotor can perform the path generated by the drone diffusion policy, however, paths generated are also passed through the minimum snap optimizer.

Implementations for the RRT* planner and the minimum snap trajectory generation module are based on the Quadrotor-Simulation repository published by Bharath Irigireddy[2].

[2] https://github.com/Bharath2/Quadrotor-Simulation

Image-Based Trajectory Planning

Retrieving Path Images

The data pipeline up until this point allows for generation of trajectories using ground truth information about the gate's position with respect to the quadrotor. In order to generalize to new scenarios, creating a foundation model for quadrotor movement, an image-based policy is necessary.

The steps above take place to generate a gate at a random position and calculate a smooth path to that position using the combination of RRT* and minimizing snap. This happens without using AirSim. The simulator then provides the functionality to capture images at specified intervals or upon an API call. Trajectories generated to random gate positions are performed in the simulator, and the associated image at each waypoint is added to the training dataset. This picture comes from the forward-facing camera of the quadrotor with a field of view of 120 degrees.

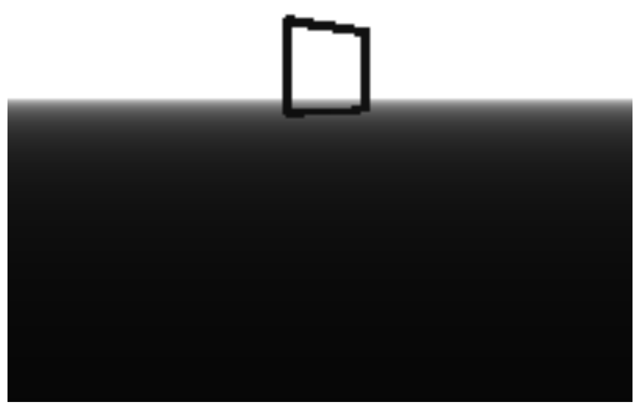


Figure 1.8: An image of the goal gate taken with the FPV camera on the drone.

Encoding Images

Because the drone diffusion network uses the FiLM conditioning method to modify its outputs depending on the inputs, image inputs cannot directly be used in the policy. Instead, the pictures must have some encoding applied.

Transfer learning in robotics has clearly shown superiority for learning new tasks, as opposed to training a model from scratch [50]. Leveraging prior knowledge with some retraining for a new scenario, similar to the way humans learn, should intuitively also perform better than beginning with random noise, or square one.

For this project, an encoding is transferred from a visual policy called DroNet [14]. The original version of this task is a transfer learning problem itself; the authors use the plethora of data available from ground vehicles to train a drone policy that predicts steering angle from visual information. The output of DroNet is a vector in $\mathbb{R}^2$ that includes the steering angle and a probability of collision. The encoding being transferred is the output of a previous layer once the result of the convolutional layers is flattened, which is in $\mathbb{R}^{6272}$.

DroNet is trained on an old version of TensorFlow and can no longer be inferenced using the version of Python necessary for the drone diffusion policy. Additionally, the computation requires versions of adjacent libraries, including numpy, from the initial training date of the network. As a result, in order to make use of this encoding, a separate conda environment is created. To interface with Drone Diffuser, a Flask application handles input containing images from AirSim, queries the model, and returns the encoding.

Other models were considered prior to choosing DroNet, including ones trained on more recent data and with newer techniques, including Vision Transformers and Autoencoders. These include [51], [52], [53], and [54]. However, because these models are trained on data from other robotics applications, not quadrotors, they likely have very little to no similarity with the task at hand, a distributional shift issue that the additional training by Drone Diffuser is unlikely to overcome. An examination of existing drone image-to-policy models such as [55], [56], or [10] either have no open-source code to reference or require significant compute for retraining.

Representing Rotations

The yaw angle of the quadrotor is predicted in radians. While this works for many tasks, Euler angle representations are by nature discontinuous (as shown by the directional limits $\theta = \pi$ and $\theta = -\pi$), increasing difficulty for neural networks to directly predict them [57]. Quaternions and other common representations, including the axis-angle method, have the same issue. Zhou et. al. propose using a continuous representation of the rotation space using $SO(n)$ for networks to output while dropping the last column, both to avoid excessive data and because orthonormalization is necessary in either case. They show prediction performance improvements on scenarios including inverse kinematics and pose estimation.

In this chapter, because the output only includes yaw, we can generate predictions in $SO(2)$. The rotation matrix about the z-axis follows the form

$$R_z(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The final row and column are dropped because they remain constant. Additionally, the second column is dropped because it can be derived through Gram-Schmidt orthonormalization from the first. Additional experiments are conducted to determine whether this change of basis improves results of the diffusion process.

1.4 Experiments

Initial Testing

The initial sanity test for the drone diffusion model involves executing the simple takeoff and landing sequence described above. The observation consists of the current location, and the action diffused is the next 8 positions for the drone, performed in receding-horizon fashion. Because the dataset consists of a single example of this sequence, the policy is fully overfit to the sample, which means a successful result implies a nearly exact imitation of this action executed. The model converges in 5 epochs, confirming the code works as expected.

Gate Displacement Policy

The gate-delta policy $\pi(\Delta_g)$ takes as conditioning input the relative pose of the gate with respect to the body frame of the quadrotor. It returns the positional and angular displacements for the quadrotor for the next 8 timesteps in a receding-horizon manner. Pose is defined as a vector in $\mathbb{R}^4$ comprised of (x, y, z, yaw), with the positional displacement values measured in meters and the angular displacement measured in radians.

400 uniformly random gate locations are sampled within the bounds described in 1.3. RRT* and minimum snap optimization are used to generate trajectories from the starting position to the gate. While recorded data consists of absolute values of position and orientation, it is converted to displacement in a postprocessing step.

The gate-delta policy converges in about 10 epochs. It is evaluated with 30 gate locations sampled at the uniform random distribution as the training set, and the policy successfully navigates the quadrotor to the gate with a 100% accuracy. The positions are located at different places than those in the training set, although they are within the same uniform distribution; the policy adjusts to account for the difference.

The quadrotor is also flown to gates located at out-of-distribution positions. This works successfully for some cases within a small range of the training distribution margins, but it struggles at further positions.

Visual Conditioning Policy

Yaw for Rotations

The visual conditioning policies $\pi(I)$ use image encodings from DroNet as conditioning to predict displacements for the quadrotor. Similar to the Gate Displacement Policies, the training set comprises of paths to 400 uniformly random gate locations generated with RRT* and smoothed with minimum snap optimization.

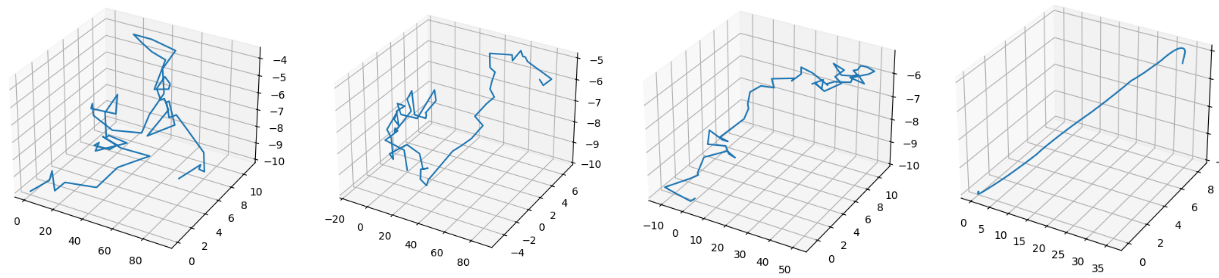


Figure 1.9: Visualization of the reverse diffusion process. The figure depicts the path generated after 25%, 50%, 75%, and 100% of the reverse diffusion steps. The path becomes more refined and imitates the training set as noise is removed. The drone begins at (0, 0, -10) and travels to (36, 7, -6). Note the scale on the axes.

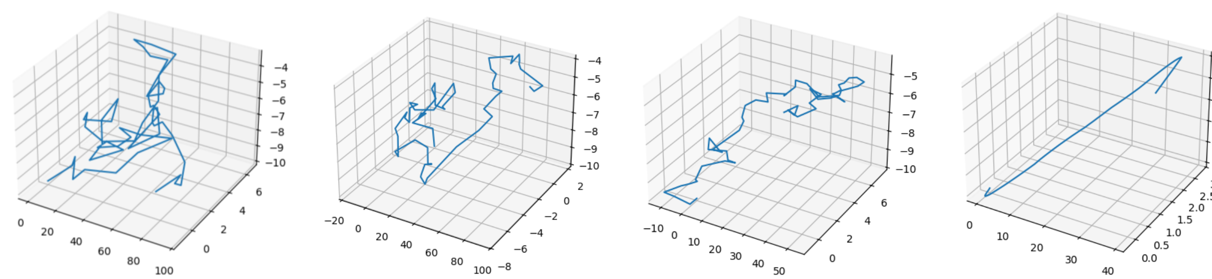


Figure 1.10: Path to (40, 1, -5)

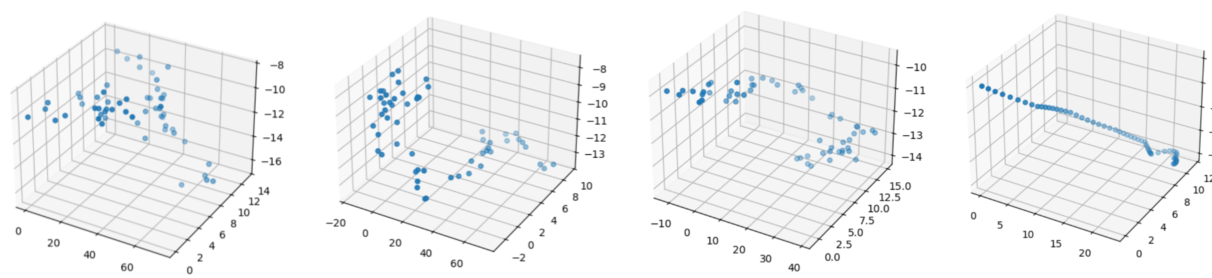


Figure 1.11: Path to (20, 10, -14). A scatterplot is used to depict the waypoints generated and illustrate the distribution of points reducing in variance from the final path.

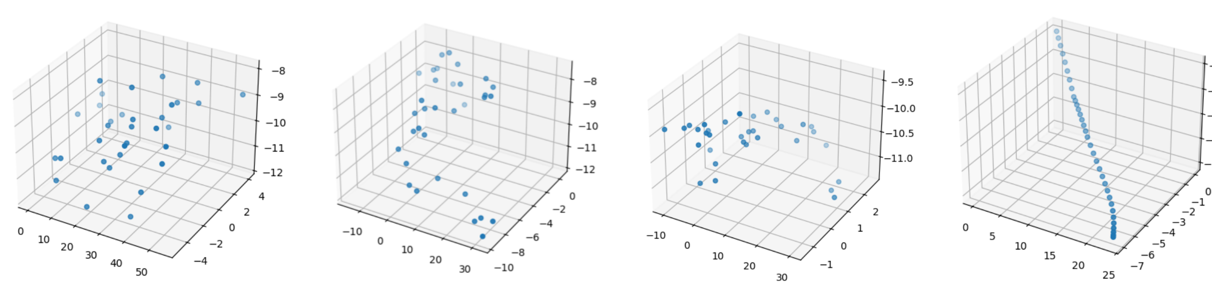


Figure 1.12: Path to (24, -7, -12)

While initial tests work with the full DroNet encodings, memory limitations on the hardware available prevent the vectors from being usable as conditioning. As a result, a downsampling layer is added after the final convolution layer, and a compressed version of the encodings is derived.

Drone Diffuser fails to work for this experiment. While training and validation loss go down over the course of training, the mean squared error increases significantly. The trajectories evaluated during the testing stage do not move the drone towards the gate; the quadrotor spins randomly and moves in seemingly arbitrary directions.

Rotation Transformation

The same image dataset is used, but instead of having Drone Diffuser output yaw values, the policy instead returns the first column of the z-axis rotation matrix to preserve continuity. The attempt does not yield any significant improvement, as issues in image encoding likely overshadow any other potential enhancements to the Drone Diffuser itself.

1.5 Conclusion and Further Work

This chapter shows that diffusion-based path planning has the potential to work for drones. Generative methods that make use of diffusion have clear advantages over optimization and RL planners, including better long-horizon planning, improved adaptability to new scenarios within the training distribution, and stability during the gradient descent process.

While the reverse diffusion process in and of itself has no feasibility guarantees, supplying training data with paths that a quadrotor can fly as well as taking a postprocessing step to ensure the outputs are doable allows for successful trajectory generation.

Providing goal state information has shown to result in high success values in simulation for scenarios in which the quadrotor must navigate to a designated rectangular gate, akin to drone racing competitions. The conditioning values are assumed to have come from a separate part of the onboard processing pipeline, perhaps from a central communication server, incorporation of SLAM, or a different network altogether.

For obstacle avoidance and more generic situations, such as rescue operations, however, an image-based method must still be proven to reliably work. The visual encoding chosen for this project likely falls short of the quality necessary to guide drone flight. Further experimentation can use an encoder-decoder model or involve vision transformers, which have shown success in end-to-end drone navigation [46].

To implement Drone Diffuser in a real quadrotor will likely take some specialized computing resources, advancements in existing hardware, or techniques that identify the most salient parts of the network to compress it. The reverse diffusion process by nature takes time to run, and Drone Diffuser specifically generates paths on the order of seconds. The speed issues make the technique impractical for path planning in real-time scenarios like racing or danger zones, although it has the current potential to operate successfully in less time-critical environments. Other further experimentation can attempt alternative diffusion-based path planning techniques, such as modifying the current image to a goal state and training an RL planner to reach that position.

In the long-term, an aerial vehicle foundation model that can take as input some specified task (ex. pass through the gate, find missing people, photograph a monument from an angle) and can predict a trajectory that performs this motion would significantly advance the field of autonomous systems. Processes that would ensure this motion is performed in a safe, reliable, and feasible manner are necessary prior to rollout into the public domain.

Chapter 2

RAG Alignment

2.1 Problem Statement

When training language models on large volumes of data, significant chunks of information and skills can be lost, and it can be difficult for models to effectively store and retrieve the contents of the entire dataset from the weights directly. As a result, Retrieval-Augmented Generation aims to improve the factual correctness and reliability of model outputs by providing the model with a vector database of ground truth. Retrieving from this database and providing data elements to the language model when prompting it can produce more accurate and consistent model outputs, allowing the model to even answer questions that may fall outside of its original training distribution.

But even with access to exact the training data, retrieval documents, or model weights, language model behavior can be extremely difficult to predict and interpret. For instance, language models can demonstrate unpredictable or unsafe behavior when user questions fall outside of the training distribution. If the data available to a language model contains all the information it needs to answer a user input, the model may still not include all of that information or convey it accurately depending on the extent to which it weighs different sources and its own training knowledge base.

Fine-tuning is one technique applied to better understand and control the behavior of a language model in specific contexts. The general fine-tuning process involves adjusting the parameters of a pretrained model for a specific set of tasks or data. Despite the effectiveness of fine-tuning in achieving performance improvements on certain tasks, it can result in several unintended outcomes that pose risks to users or systems implementing fine-tuned language models. The behavior of the models may be more controllable or better understood across a specific subset of inputs, but fine-tuning can result in unintentional forgetting, training data memorization, and degradation of model alignment and safety. Additionally if information

Disclaimer: This section of the thesis is work collaboratively done with co-authors Devan Shanker (devanshanker@berkeley.edu) and Josh Barron (josh-ee@berkeley.edu). To ensure proper credit is given to all contributors, please do not cite this thesis for this particular project. For accurate referencing and further information, please contact us directly.

needs to be added or removed from the models knowledge base, a new fine-tuning is required, the cost of which can quickly become unmanageable depending on the rate that information needs to be added. Taking into account the risk of hallucinations and the price of repeated fine-tuning, it is viable to have a hybrid approach that equips a fine-tuned model with Retrieval-Augmented Generation.

Issues with Retrieval-Augmented Generation for both pretrained and fine-tuned models can pose tremendous risks to the safety of users interacting with a model. Models may hallucinate and risk presenting inaccurate or misleading information to users without any warning or justification. In addition, language models rarely demonstrate warnings when asked to complete tasks far outside their original training distributions, often resulting in hallucinations for fine-tuned models. When working with fine-tuned models, it can be difficult to assess the relative effectiveness of a specific fine-tuning. Overall, the concerns with hallucination and factual inaccuracies of retrieval-based language models motivate a system for detecting hallucinations and other inconsistencies between retrieved data and model outputs.

2.2 Related Work

Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) [58] is an approach to augmenting the quality of external language models by retrieving the most relevant information from an additional datastore, before providing it as additional context to the language model. After creating vector embeddings for all text chunks stored in a vector database, the most similar stored chunks of data can be retrieved by embedding the model input with the same approach. Retrieval-Augmented Generation allows a language model to draw from knowledge and answer questions outside of its original training distribution, supporting the convenient addition, removal, and updating of data stored in the RAG database as required by the overarching system. Depending on the quality of the embeddings and the nature of the user input, retrieval-augmented generation systems may fail to find the most relevant and appropriate sources to generate a response to a user input. In addition, contradictory or outdated sources may confuse the model, increasing the probability of a hallucination or factually incorrect output.

Hypothetical Document Embeddings

Hypothetical Document Embeddings (HyDE) [59] intend to improve upon the performance of traditional RAG systems by attempting to shift the embeddings of the user input closer to the distribution of data contained in the vector database. For instance, if we asked the model to name the president of the United States (based on data in the vector database), formulating a hypothetical answer *("the president of the United States is Dan Hendrycks")* may more closely match relevant data in the retrieval data distribution (*"the president of the U.S.A. is Barack Obama"*) than the original question *("who is the president of the United States")*. Even if the hypothetical answer itself is incorrect, the changes in language, sentence structure, and vocabulary used to formulate an answer can all help shift the embeddings closer to the embeddings of potential answers. Hypothetical Document Embeddings for RAG have demonstrated significant performance improvements over standard RAG systems, and play a critical role in allowing language models to draw from additional data distributions and external knowledge bases.

Efficient Fine-Tuning

The fine-tuning process involves adapting the weights of a pretrained model to better fit a particular set of tasks and corresponding data distribution. Parameter-Efficient Fine-Tuning (PEFT) [60] is a line of work that attempts to fine-tune by modifying only a small proportion of the model parameters, resulting in a more efficient fine-tuning process. Recent work in Low-Rank Adaptation of Large Models (LoRA) [61] demonstrates how the pre-trained model matrices can be reduced into low-rank decomposition matrices for each layer of the Transformer architecture. LoRA is a type of PEFT that enable extremely computationally efficient model fine-tuning, allowing for the deployment of domain-specific language models with far less computational resources than previously required. Quantized LoRA (QLoRA) [62], a variation of LoRA, introduces quantization to further reducing the memory and computation overhead of fine-tuning. Several recent works find risks associated with fine-tuning that may go unnoticed. For instance, fine-tuning can result in data memorization that may compromise the factual accuracy of certain generations. In addition, fine-tuning can result in unintentional forgetting of specific data elements or pieces of factual information.

Language Model Alignment

Recent work in the space of language model alignment has focused on creating safer and more usable models for humans to interact with on an everyday basis. However, results find that language model alignment to provide clearer replies, discourage unsafe content generation, and prevent generation of misinformation often involve more brittle or weak safeguards. Few production models provide warnings about hallucinations, and models may provide incorrect or outdated information without any indication or warning to users. Though several papers have explored the extent to which models understand their own training and input distributions, models tend to treat all inputs the same and most often fail to warn users about domains or topics that fall outside of the typical training distribution.

Model Hallucination

Even within a retrieval-augmented generation system architecture, language model hallucinations are an often unpredictable phenomenon that can be difficult to prevent or detect.

Though certain attacks can be used to elicit a mode of hallucination from language models at a higher rate, language models can demonstrate more subtle hallucinations and factual inaccuracies stemming from unintentional forgetting or retrieval issues. If a model retrieves a text chunk with the exact information the user is looking for, it may still omit this information from the completion. Language models can hallucinate when the training data contains outdated or contradictory information about a certain topic. However, models also demonstrate no awareness or warnings about the situations and domains that may fall further from their training distribution, even for individuals with access to the full weights and training data. Several existing approaches to hallucination detection involve classification of factual accuracy, faithfulness to a source document, or separation into types of hallucination errors [63]. However, these solutions often fail to scale or fit easily into consumer-facing language models.

2.3 Proposed Solution

The factors previously mentioned all directly contribute to challenges related to the deployment of customer-facing LLM-based chatbots and conversational tools. In light of these factors, we propose a solution to address these challenges through:

- Determining source utilization by the chatbot
- Detecting out-of-distribution (OOD) user questions
- Quantifying fine-tuning effectiveness with regards to factual statements
- Providing meaningful hallucination warnings for OOD questions and completions with low source utilization

We use the term "RAG Alignment" to refer to the process of addressing these challenges to support the deployment of user and customer-facing conversational agents. Our proposed solution to the issue of RAG Alignment problem is comprised of a robust method for factual similarity comparisons between text.

To more closely align Retrieval Augmented Generation systems with user definitions of factual correctness, we propose the notion of a fact-bearing term (FBT). Based on this concept, we define a similarity formula supporting the comparison of a generated output against each retrieved data item used to generate it. Applying this comparison, we propose an algorithm designed to detect hallucinations and verify the truthfulness of a generated output against source data.

Fact-Bearing Terms

We define fact-bearing terms to be those that carry the factuality of the sentence. Changing or removing these words would completely modify the meaning of the statement. For instance, in the sentence, "The first president of the US is George Washington," the name *George Washington* is the fact-bearing term. Meanwhile, in the sentence "The printing press was invented in 1440", the year *1440* is fact-bearing. After local experiments involving a RAG-equipped LLM, we identify named entities (including, but not limited to, proper nouns), direct object nouns, and verbs as fact-bearing terms.

Similarity Scores Calculation

Step 1: Extract FBTs from Answer & Source

In order to perform the similarity score calculation that computes the relevance of a particular source and the accuracy of the answer with respect to that source, we must first extract the Fact-Bearing Terms from each source and the answer.

Step 2: Calculate Similarity Score

Part 1: Per Key Similarity Score Calculation For each key $k \in \{\text{ent}, \text{direct\_object}, \text{verb}\}$:

- Let A_k and B_k represent the sets of terms for key k in dict_a and dict_b, respectively.
- After removing exceptions:

$$A'_k = A_k \setminus \text{Exceptions}_k, \quad B'_k = B_k \setminus \text{Exceptions}_k$$

- The similarity score per key, Score_k, is computed as:

$$\text{Score}_k = \frac{|A'_k \cap B'_k|}{|A'_k|} + \left(\frac{|A'_k \setminus B'_k|}{|A'_k|}\right) \times \left(\frac{|\text{SubWords}(A'_k \setminus B'_k) \cap \text{SubWords}(B'_k \setminus A'_k)|}{|\text{SubWords}(A'_k \setminus B'_k)|}\right)$$

  where $\text{SubWords}(S)$ denotes the set of sub-words obtained by splitting the terms in set S.

Part 2: Total FBT Similarity Score Calculation

- Define the discount factor:

$$DF = 0.7$$

- The total number of terms in A is computed with a weighting scheme:

$$\text{NumAll}_A = \text{lenA}_{\text{ent}} + (\text{lenA}_{\text{noun}}^{0.9}) + (\text{lenA}_{\text{verb}}^{0.5})$$

- The final FBT similarity score, FBT_Score, is calculated using a recursive weighted reduction of scores:

$$\text{FBT\_Score} = \text{Score}_{\text{ent}} + (1 - \text{Score}_{\text{ent}}) \cdot DF \cdot (\text{Score}_{\text{noun}} + (1 - \text{Score}_{\text{noun}}) \cdot DF \cdot \text{Score}_{\text{verb}})$$

Step 3: Repeat for Each Source

Algorithm 1 Similarity Calculation Per Inference

```
function SIMILARITY(question_str, response_str, sources_list)
    question_fbts ← EXTRACTFBTS(question_str)
    exceptions = question_fbts
    response_fbts ← EXTRACTFBTS(response_str, exceptions)
    similarity ← empty list
    for each source_str in sources_list do
        source_fbt ← EXTRACTFBTS(source_str, exceptions)
        Append FBT_SCORE(response_fbts, source_fbt) to similarity
    end for
    return FBT_Score
end function
```

2.4 Methods

Data

Data Sourcing

To perform RAG experiments, we must utilize a database that has not been indexed by standard language models to verify the success of our algorithm. We settle on two sources that take the form of questions and answers, making them more conducive to testing. First, we use information from the 1921 Ford user manual that contains some specific instructions on how to operate the car. Second, we find an old Colorado school charter that has some specific answers without necessarily using archaic terms or proper nouns. In order to identify promising data sources for retrieval evaluation, we prioritize data sources with highly specific Q/A contents irrelevant to more general knowledge settings from at least one hundred years ago.

Data Cleaning

Though the data is likely unindexed and presents itself in a form for straightforward testing, significant preprocessing is required to clean the text such that it is more readable in a human context. For the Model T manual, we find the data in the form of an HTML page. We parse the information and extract the relevant details, dividing the data into topics. The Colorado school charter similarly contains extraneous textual artifacts, which are removed, and the data is split based into questions and their respective answers.

Database

The RAG pipeline has two steps, the vector database creation and then retrieval at inference time. For our database, we are using the open source project Chroma DB [1]. To create our Chroma Vector DB for the RAG system, we parse all of the scraped data from our chosen sources, split each source into chunks, and then embed the chunks.

The parameters that gave us the best results are as follows:

- Embedding Model: GIST-small-Embedding-v0
- Normalize Vectors: True
- Max Chunk Size: 512

The `Create_Vector_DB.py` code loops through all our scraped sources and fills the `RAG_ALL_Vector_DB` according to the above parameters.

FBT Experiments

Information Extraction

We use the small Spacy model to extract the most important pieces of information from the text. This allows us to parse through verbs, proper nouns, objects, emails, numbers, dates, and other important pieces of information that comprise fact-bearing terms.

RAG

Sources from the database are scored based on the similarity of their vector embeddings to the prompt. After sorting, the ones with the highest scores are added to the prompt, which gives the language model context for the output.

HyDE

The language model is requested to first return an answer based on its understanding of the query without any sources. Instead of the question, the answer is used for vector embedding comparison to find relevant sources.

Fine-Tuning

We also fine-tune our language model to improve the HyDE model as well as increase the accuracy of our generated results. This new feature OpenAI provides allows us to train GPT-3.5 for a few more iterations using our dataset specifically. We use `gpt-3.5-turbo-0125` fine-tuned for 3 epochs.

[1] https://www.trychroma.com

Additionally, we repeated the experiment with `Mistral-7B-Instruct-v0.2` running locally on a M2-Max equipped MacBook Pro with 32GB of RAM. Fine-tuning of Mistral-7B-Instruct-v0.2 was done utilizing the mlx-examples library [64]. Specifically, we created a 32-layer LoRA which was trained for 1000 iterations. The LoRA was created with the exact same train-test split as the GPT-3.5 fine-tuning. Figures in the result are generated from GPT-3.5 to demonstrate the generalizability of the approach. The Mistral-7B plots accompanying our own fine-tuning of a local model can be found in the appendix B.

2.5 Results

To determine the effectiveness of FBT similarity RAG Alignment, we construct four evaluations based on the question-answer dataset created from the 1921 Ford Model T user manual.

Determining Source Utilization

To determine a baseline for source utilization, we decided upon using the same L_2 similarity calculation performed by the ChromaDB upon retrieval. Specifically:

$$\begin{aligned} \text{l2\_distance} &= \sqrt{\sum_{i=1}^{512} (\text{input}[i] - \text{source}[i])^2} \\ \text{embedding\_similarity} &= e^{-\text{l2\_distance}} \end{aligned}$$

Where *input* and *source* are the respective 512-dimension embedding vectors.

For the baseline, we embed the final chat-bot answer and then compute the embedding_similarity for each source. These are the **triangles** in all the result figure plots. To compare FBT effectiveness, we compute the FBT similarity score as outlined in Algorithm 1 and plot that as the **plus symbol** in all the result plots.

Each symbol is colored according to its source, displaying blue for source 1, red for source 2, and green for source 3.

Detecting out of distribution (OOD) questions

To test for OOD questions we constructed a test set of 30 questions where the first 10 are directly in the 1921 Ford user manual, the next ten (10-20) are questions that were in the test set (not fine-tuned on), and 20-30 are questions that are car related but not in the 1921 Ford user manual and thus are out of distribution.

To visualise the drop in FBT score as the questions leave RAG distribution we added a Second-degree polynomial line of best fit for the highest FBT score in each question, which are distinguishable by a black square.

Analyzing Figure 2.1 we collect a couple of key takeaways.

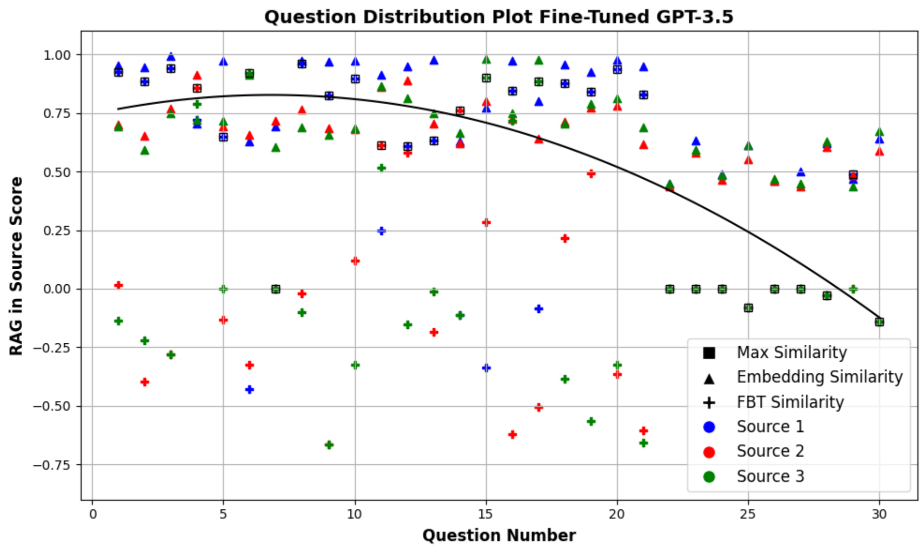


Figure 2.1: FBT vs Embedding Similarity as questions leave dataset distribution on Fine-Tuned GPT-3.5

1. Order of FBT Similarity matches with embedding
2. Once questions leave data distribution, both FBT Similarity and Embedding Similarity drop
3. FBT Similarity has much higher variance

Quantifying effectiveness of fine-tuning with regard to factual statements

To determine if FBT Similarity could be used to provide a numerical answer to the effectiveness of fine-tuning we run the same experiment on the base and fine-tuned versions of our models. For the experiment we enable HyDE and then ask the chat-bot every question in the 1921 Ford user manual.

In this experiment, we decided to add a additional point on the plot for a "Weighted Score" represented by a **star** which is calculated:

$$\text{weighted\_score} = 1 + \text{w1} \times \text{emb\_score} + \text{w2} \times \text{fbt\_score}$$

For our experiments, we set weight 1 (w1) and weight 2 (w2) equal to 0.5.

Since the questions are chosen from the Ford user manual, we know which source was the "correct" source when we ask the question. Knowing this, we are able to detect when the correct source was returned and in which spot it was returned (source 1, 2, or 3). To visualize this, we add **vertical lines** that correspond to when the correct source is returned with the color of which location it was returned in.

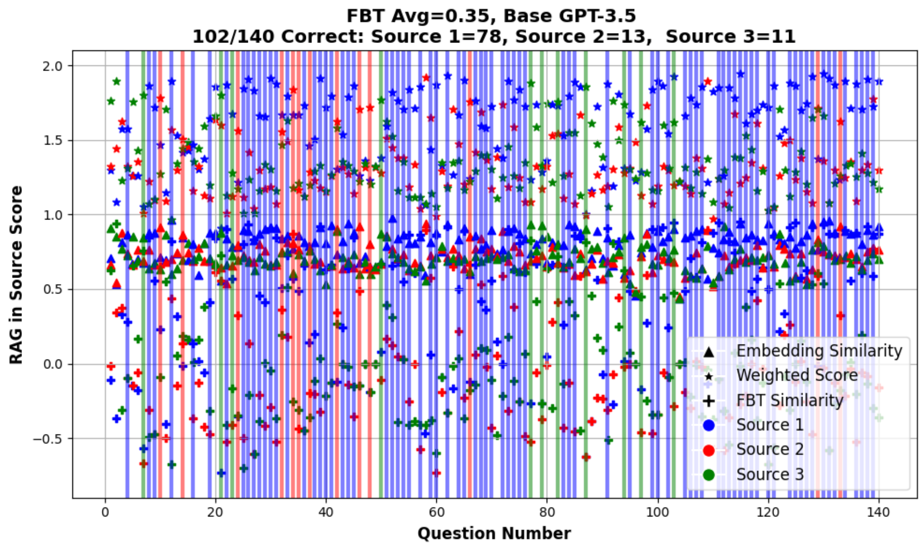


Figure 2.2: Chat-Bot RAG performance with HyDE and Base GPT-3.5

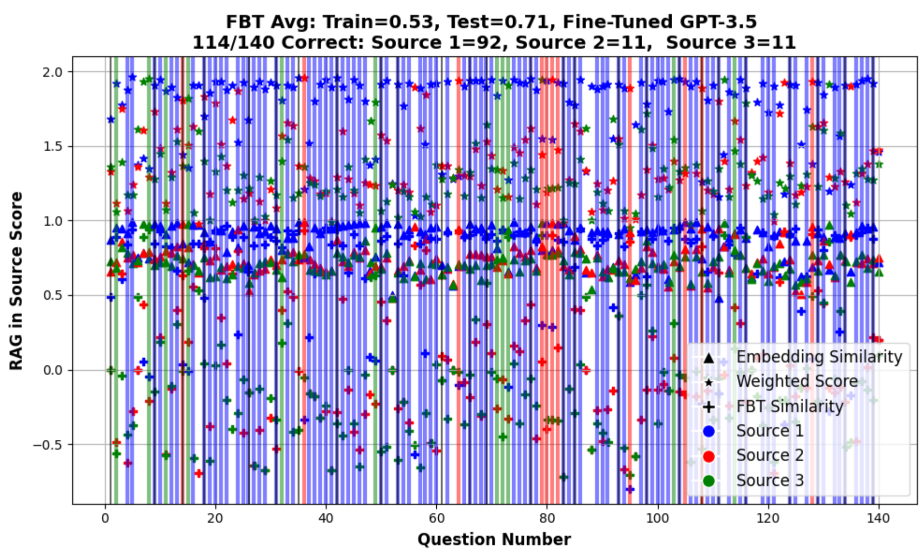


Figure 2.3: Chat-Bot RAG performance with HyDE and Fine-Tuned GPT-3.5

Analyzing Figure 2.2 we see that the chatbot returns the correct source 102 times out of the 140 questions; additionally we see that a majority of the time (78) the correct source is returned first (Source 1).

Figure 2.2 uses the same fine-tuned GPT-3.5 HyDE and Answering model. Additionally, since we perform a train-test split during fine-tuning, we mark those questions with an additional thin black vertical line. The high FBT Avg in the test set is unexpected and not observed in the Mistral Model (see B).

Comparing Figure 2.2 to 2.3, we see that fine-tuning our model has two major benefits of better RAG performance and improved factual alignment.

Better RAG performance is clear when comparing the number of correct sources returned between the figures. The fine-tuned version returns the correct answer 114 times compared to 102, and the correct source is returned as the Source 1 more often. Having the correct source being first is advantageous, as it may reduce the requirement for RAG Rerankers.

The improved factual alignment is clearly visible in the higher average FBT score (0.35 vs 0.53), which demonstrates that the word selection for the Answering LLM is closer to the sources.

Providing Meaningful Hallucination Warnings

Combining all the information gathered from the above experiments, we are able to create a chat-bot that warns the user of possible hallucinations for OOD questions and answers with low source utilization.

By observing our dataset we see that generally, when sum of FBT similarity score fails to go above 0.1, the model does not have the correct source. We can use this as a threshold to warn the user. This scenario is demonstrated in Figure 2.4, where how to paint the car is not in the dataset and thus the hallucination warning is triggered. Additionally, we think it would be helpful to cite the FBTs in the answer to help the user investigate the answer when desired.

This example UI is created with Chainlit and is just a sample. If desired, it is possible to provide multiple types of warnings depending on the individual FBT score. Overall, we are pleased with the progress towards making chatbots more reliable.

2.6 Conclusion and Future Work

In this work, we present the notion of a Fact-Bearing Term, a term in a sentence bearing the load of factual correctness of the sentence. We identify named entities, direct object nouns, and verbs as FBTs that hold the weight of truth of a sentence.

Based on the known limitations of retrieval-augmented generation systems, we first propose an algorithm that computes a measure of similarity between a generated output and one of the retrieved sources used to generate it. This algorithm allows for the detection of questions that fall outside of the training or fine-tuning distribution we have specified. In addition, these metrics can be used to benchmark RAG systems and different fine-tunings.

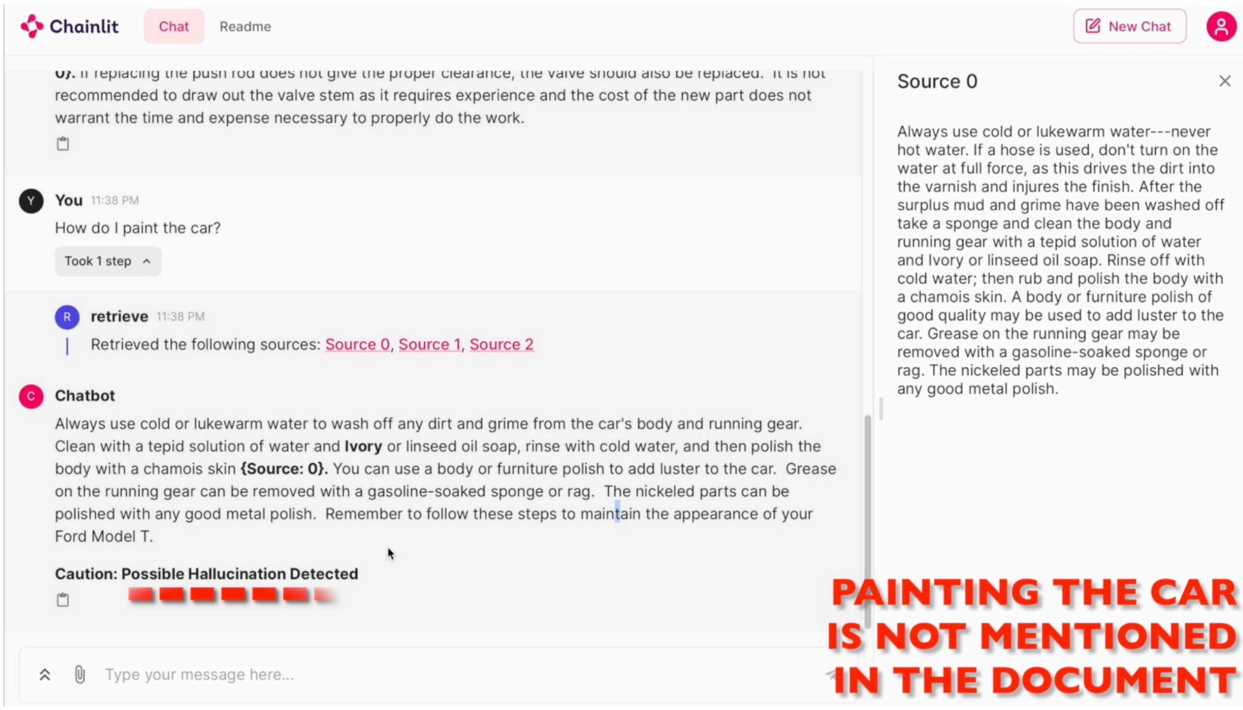


Figure 2.4: Chat-Bot Warning user due to Low FBT similarity

Most importantly, the measure of FBT-based similarity allows for hallucination detection and citation of sources at inference time. This allows for an efficient, scalable implementation of context-specific hallucination warnings within a larger scale language model pipeline.

We are excited about the opportunities that FBT-based and similar hallucination detection systems and hope Fact-Bearing Terms play a useful role in future evaluations of language model outputs and fine-tunings. More specifically, we hope to see a more precise and intuitive mathematical formalization of the per-key similarity score calculation. In addition, it would be interesting to see a deeper linguistics-inspired study of the most effective parts of speech, as well as the feasibility of training a model to detect and compare fact-bearing terms across sources automatically.

Chapter 3

IMAGINE: Improved Multimodal Augmented Generation through Imagined Neural Embeddings

3.1 Problem Statement

In addition to information retrieval, transformer-based conversational models are quickly rising in popularity for usage involving image and multimodal data retrieval tasks. In order to augment language models with the capacity to understand and operate on images according to user instructions, RAG techniques involve retrieving the data with embeddings most similar to the embedding of target user inputs. Due to their reliance on embedding distances, RAG systems can be severely performance constrained by the quality of embedding models and the semantic distances between user inputs and desired retrieved data outputs. This phenomenon can have an even harsher impact on retrieval-based image systems as a result of the context sensitivity of many image-related use cases and the known limitations of image and multimodal embedding models.

In the context of retrieval systems, text chunks may often contain more consistent contextual meanings and factual uses across different user queries and use cases. More specifically, we propose it can be much more difficult to predict what questions a user may ask about a specific collection of pixels than a specific collection of words. We highlight this through a real-world use case that will serve as a guiding example for the benchmarks proposed later on in the paper. Suppose we want to create a conversational agent that lets a user chat with their camera roll. For this example, we consider the retrieval data distribution to be a collection of images (i.e. your camera roll from the last month). From these distributions, we hope to retrieve the most relevant images that help us answer the question, *"what did*

Disclaimer: This section of the thesis is work collaboratively done with co-author Devan Shanker (devanshanker@berkeley.edu). To ensure proper credit is given to all contributors, please do not cite this thesis for this particular project. For accurate referencing and further information, please contact us directly.

[the user] do over spring break?"

We highlight three key issues with a naive embedding-based approach to image retrieval in the image space. (1) The CLIP embedded notion of spring break in the user input may fall extremely semantically far from embeddings of target images of the user dancing, skiing, or relaxing on the beach. We define the term *indirect retrieval* to highlight this distance between a more abstract input and concrete desired outputs. (2) Multiple correct images may exist and fall very far apart from each other in the embedding space. Unlike three pictures of different dogs, pictures of us reading a book on the beach and hiking through nature are fairly different. We refer to this challenge as *desired output sparsity*. (3) Depending on additional user and environment information, the notion of a term like *spring break* can contain dramatically different meanings. If a seven-year-old user gained access to our system, we may desire our expected distributions of embeddings of images to fall further from exotic destinations and closer to pictures of the park or crayons. We use the term *contextual sensitivity* to highlight the possibility of differing output distributions for the same input embedding when provided with additional information. The third issue of contextual sensitivity only applies in specific cases when additional information is available. However, the risks of indirect retrieval tasks and desired output sparsity can occur for all users across a variety of inputs for image retrieval models.

One promising line of research in improving retrieval systems in other domains focuses on bringing search embeddings for retrieval closer to the distribution of the data. In the context of text, the notion of Hypothetical Document Embeddings (HyDE) aims to bridge the distributional gaps between user inputs and desired retrieved data by forming a guess (hypothetical embedding) to the user query. HyDE demonstrates impressive performance improvements over RAG in the language domain. However, many production vision LM and multimodal retrieval systems tend to rely on naive RAG implementations that simply retrieve CLIP embeddings or human-written captions for stored images. This leads us to the question, *can we apply the notion of hypothetical embeddings to image and multimodal tasks for improved retrieval accuracy and model performance?*

In order to evaluate the feasibility of imagined image embeddings against naive and baseline RAG implementations, we propose a benchmark for indirect vision RAG tasks. Each use case operates on a collection of real-world images and only accepts a single user text input. This benchmark, the CameraRollQA task, involves question and answer tasks on 50 hand-picked images forming a camera roll that elicit unique replies to highly specific questions when the right information is retrieved. One example CameraRollQA task is the user input, "Do I have any dietary restrictions?".

Based on these evaluations, we present the following contributions:

- The IMAGINE-I and IMAGINE-T caption-based and diffusion-based imagined image embedding systems for improved indirect sparse image retrieval

- A simple formulation of multi-top-k image embedding similarity to ensure more robust outputs in indirect retrieval contexts with larger desired output sparsities

- An end-to-end conversational camera roll chat implementation freely available that completes CameraRollQA tasks for any language model and collection of images
- An open-source augmentation of pretrained multimodal models with abilities to visualize & verbalize as they answer user questions and the capacity to interact with and search over visual datasets

3.2 Related Work

RAG and HyDE

For a detailed overview of Retrieval-Augmented Generation and Hypothetical Document Embeddings, see 2.2. Through observing textual visualization outputs of IMAGINE-T, we hypothesize the set of adjusted stylistic characteristics in text has an image analog including references to environment, shape, size, movement, color, and lighting. Several examples suggest that hypothetical embeddings implicitly weight emphasis on different aforementioned visual style features depending on the task.

Vision Language and Multimodal Model Alignment

Recent work in the space of transformer model alignment has focused on creating safer and more usable models for humans to interact with on an everyday basis. Vision language models demonstrate notable performance on a growing set of tasks including image/video captioning, visual question answering, and image-text matching. However, models operating over less strictly defined multimodal training distributions can struggle to deliver safe and consistent outputs to users. In addition, model alignment and fine-tuning on a task such as spatial image reasoning can have an unpredictable impacts on other model capabilities. As a result, we aim to demonstrate an improvement on image retrieval tasks that can be applied to pretrained models with unknown weights. For this purpose, we use the large multimodal model (LMM) GPT-4 Turbo with Vision, capable of answering basic questions about the contents of input images. GPT-4V is trained with and involves reinforcement learning in the training process to demonstrate improved performance on VQA and image captioning tasks.

CLIP Embeddings

The CLIP system, developed by OpenAI, addresses the classical computer vision challenge of describing images via natural language, rather than restricting them to a strict set of categories [65]. CLIP embeddings place text and images into the same vector space, creating better correlations between an image and its contents. The authors demonstrate clear transferability of their results, achieving state-of-the-art success in ImageNet classification without utilizing any of the original training examples. In this work, we leverage CLIP embeddings to streamline the operations between the image and text space and allow for consistent benchmarking between our proposed IMAGINE methodology and comparable baselines.

Image Diffusion and Generation

A background on diffusion networks is presented in detail in 1.2. In this work, we leverage image diffusion and generation to create imagined images, embedded and compared against camera roll embeddings (stored image data) in the IMAGINE-I approach.

3.3 Proposed Solution

The factors mentioned above discuss ways in which generation can be improved in the language domain. We seek to apply similar ideas in the image domain in order to augment generation and improve retrieval of photos that a user supplies from their camera roll. Specifically, we compare a few different approaches that are image analogues to text HyDE to determine the best way to find relevant images from a user's photo library for the prompt that they supply. See C.1 for prompts.

Naive Baseline: Direct CLIP Embedding RAG

As a baseline, we used a standard RAG technique to determine which images end up being retrieved corresponding to a user prompt. The user query is embedded using a CLIP model, and the CLIP embeddings of the images are compared with that of the prompt using cosine similarity to establish the top-k matches. These are returned for further postprocessing. We observe that the naive CLIP baseline involves no analysis or filtering of the original CLIP input based on the desired user task.

Augmented Baseline: LM Prompt Modification

We propose the LM Prompt Modification (augmented[1] baseline) to provide a more holistic evaluation of the effectiveness of imagined embeddings, especially for the CameraRollTransform task. A user query may have extra elements that push the CLIP embedding away from the space that encompasses the desired images to be retrieved. For example, in the prompt, "turn the house into a cartoon", only "the house" describes the images that must be searched for in the database for modification. Similarly, in the sentence, "Find pictures for my time on the beach", "the beach" is our indicator term. We use a language model query to modify the prompt, narrowing down the fact-bearing portions of the statement. The CLIP

[1] We originally referred to this as the *improved baseline* due to its markedly higher performance on the CameraRollTransform task. The naive baseline was far from usable for this task in comparison. We change the name to augmented baseline in light of evaluation metrics in comparison to the naive baseline under this experimental configuration.

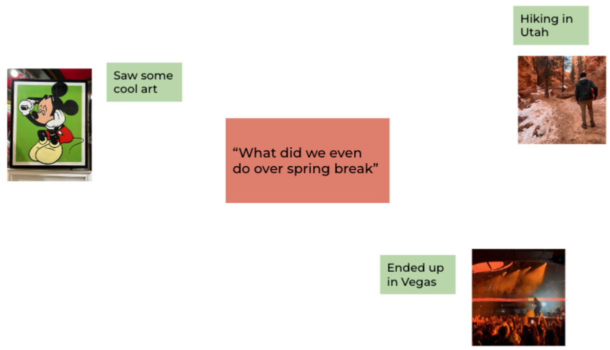


Figure 3.1: The naive baseline attempts to retrieve CLIP embeddings based on the embedding of the user input. Due to the notion of indirect retrieval and desired output sparsity mentioned earlier, we observe that the desired output images can fall very from the user input in the CLIP embedding vector space.

embeddings of these portions are calculated, and the top-k images are found, similar to the baseline.

IMAGINE-I

IMAGINE-I extends the notion of imagined embeddings to the image domain. In the text domain, HyDE generates a guess solution before RAG retrieves the relevant solution, which tends to significantly improve results. In this preliminary application to imagined embeddings of pixels, we use diffusion to generate hypothetical (imagined) images that may corresponding to the user request as an analog. Specifically, the pipeline follows the following steps:

1. Generate text prompts for photorealistic images corresponding to the user's request with no prior knowledge of the camera roll.
2. Use a diffusion model to generate images corresponding to these prompts.
3. Embed the generated images using CLIP.
4. Search for the database using these new embeddings and retrieve the multi-top-k[2] matches.

[2]Please refer to the Future Work for a more thorough discussion of the implications of multi-top-k embedding retrieval and opportunities to further enhance IMAGINE retrieval results.

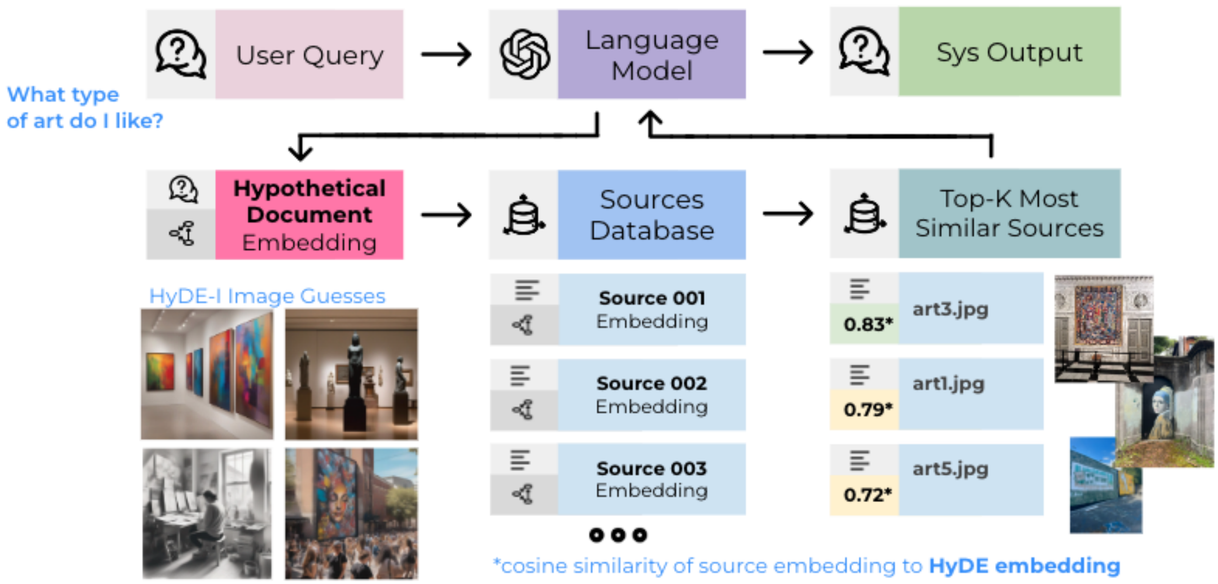


Figure 3.2: The IMAGINE-I architecture embeds multiple imagined guesses of what the user's desired imagines created with previously described image diffusion techniques. These embeddings are used to retrieve the most similar images to the stored image data, in the context of the user query.

IMAGINE-T

IMAGINE-T extends HyDE into the image domain using text generation instead of image generation. Specifically, a language model is tasked to generate captions for the image the user wants retrieved or modified. For improved performance, these captions are expressed in the form of hypothetical DALL·E 3 prompts that would generate hypothetical images of the desired nature. These captions are embedded using CLIP, and the multi-top-k matches from the database corresponding to any of these embeddings are returned. See C.1 for the corresponding text prompts.

3.4 Methods

Data

Data Sourcing

To perform RAG experiments, we wanted to utilize a dataset that has not been indexed by any language or diffusion model. We also wanted to apply our method in a practical use case with real users. As a result, we test on ourselves, uploading about 50 pictures from from

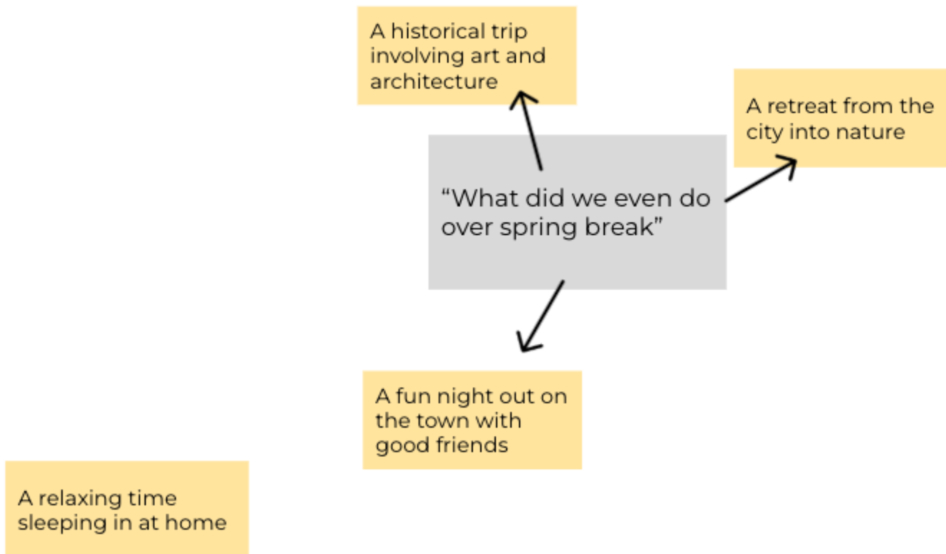


Figure 3.3: The IMAGINE-T architecture embeds multiple imagined guesses of what the user’s desired images may look like, with no knowledge of the target distribution.

our camera roll into the database. While the images take many different forms, including pictures of art, food, and lifestyle, we do curate our collection to target specific questions whose answers we know are available in the database so that we can evaluate our method over baseline retrieval.

Database

We create our own simple image database class rather than using off-the-shelf libraries developed for RAG applications. Our database is optimized for image and embedding access as well as retrieving the top-k matches for a particular prompt.

Models

OpenAI’s CLIP embedding model available on HuggingFace is used for all embeddings. OpenAI’s GPT-4 API is used for text generation, and StabilityAI’s Stable Diffusion is used for image generation.

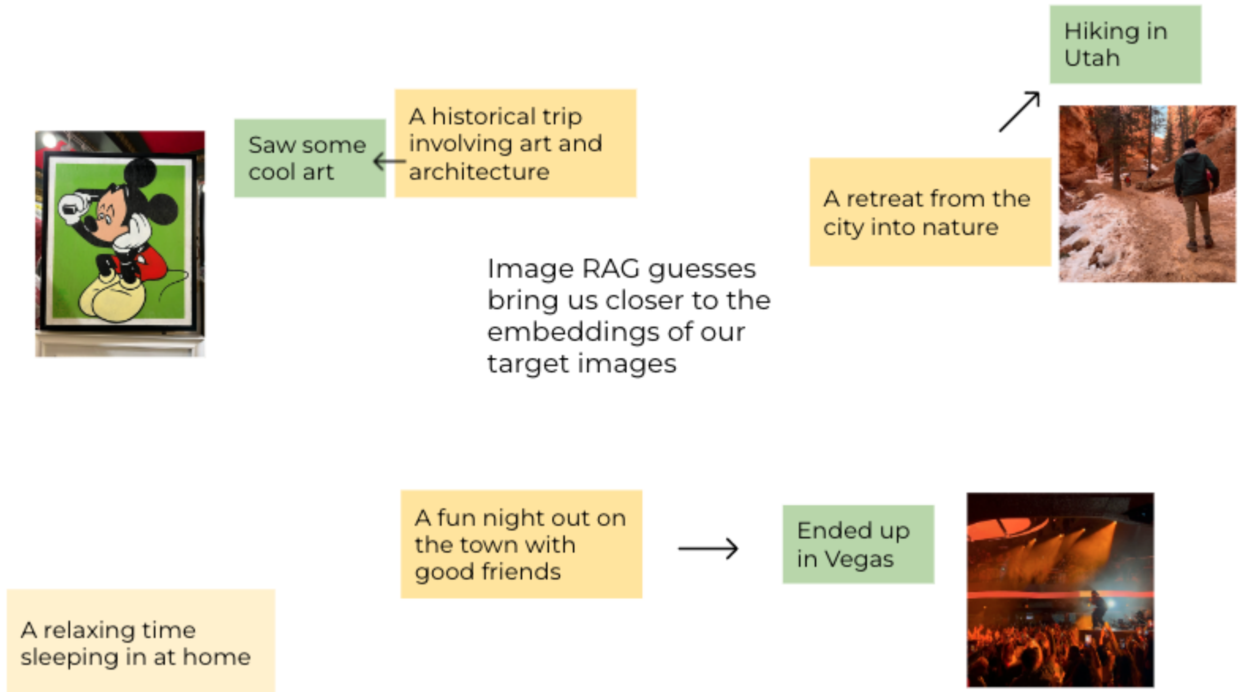


Figure 3.4: The IMAGINE-T architecture retrieves based on the *imagined embeddings* as opposed to the direct question. This can help address issues of *desired output sparsity* for this more *indirect retrieval* problem.

3.5 Results

Results on the CameraRollQA task show that the IMAGINE-T approach does significantly improve retrieval for user queries for many examples in both the access and modification contexts. However, while IMAGINE-I had comparable results to baseline RAG, we do not find that it significantly contributes to better outputs. We observe a large amount of variation and more artistic styling (as opposed to photorealism) of images created through the diffusion process used in IMAGINE-I. We hypothesize these factors likely contribute to the gap in performance between the diffusion-based IMAGINE-I and the DALL·E 3 prompt-based IMAGINE-T.

One exciting result of the IMAGINE-I system in comparison to the LM Prompt Modification baseline is the much higher consistency of retrieval despite relatively similar average performances, indicating that IMAGINE-I more consistently captures a portion of the desired retrieved image distribution even with a more loosely defined image diffusion process. Please refer to Table 3.1 and the appendix for a more detailed overview of results. Overall, we are extremely excited to see the IMAGINE-T system demonstrated the highest (or tied for highest) performance across each image and highest overall performance in our evaluations.

	Art	Pets	Food	Clothing	Interests	**Average**
Baseline	60%	100%	100%	100%	80%	**88%**
LM Prompt Modification	40%	100%	0%	0%	80%	**44%**
IMAGINE-I (ours)	80%	40%	40%	40%	40%	**48%**
IMAGINE-T (ours)	100%	100%	100%	100%	80%	**96%**

Table 3.1: Retrieval accuracy for various retrieval tests within the CameraRollQA context.

3.6 Future Work

We are excited about the opportunities that IMAGINE and hypothetical image embeddings present for image and multimodal retrieval tasks, and hope to see a broader trend towards enhanced reasoning and imagination powering visual retrieval systems. In the context of general image retrieval, we feel large-scale image and multimodal retrieval benchmarks would help promote and direct future research into this growing field. There is currently an abundance of retrieval benchmarks in other domains that we hope to see transfer over to the image domain to accelerate the development of embodied visual agents and safer vision-based systems. We predict verifiably safe image retrieval systems will play a vital role in the future of healthcare, mobility, and assistive technology.

Another line of work we hope to see more deeply explored is the notion of *indirect retrieval tasks*. Within the image domain, there exists a subset of use cases and problem contexts we hypothesize even perfect CLIP embeddings can fail to capture in indirect contexts. This is a fundamental consequence of the fact that due to the inherent complexity of images, the same image can mean very different things to different people and very different things in different contexts. As a result, we hope to see a larger-scale benchmark pushing the limits of indirect retrieval and formalizing the challenges and advantages of taking different approaches to indirect retrieval problems. Perhaps a better solution adapts based on the profile of the user, generating context-aware outputs and providing results based on specific backgrounds and interests.

In addition, we hope to see more precise definitions of the *multi-top-k* solution that is used for top-k retrieval from a database with multiple embeddings. For our provided implementations of IMAGINE-I and IMAGINE-T, we prioritize maximum CLIP embedding similarities across all top-k matches as the most important indication of retrieved relevance. However, we hope to see future research explore mathematical modifications to this simpler approach that may more effectively trade off CLIP embedding match and sparsity of retrieved outputs.[3]. This may extend to using other embedding models, as CLIP has known limitations [66].

[3]More concretely, suppose we ask for the user's favorite activities and receive four outputs involving basketball and one involving squash. Depending on the number of requested images and the desire use case, we may want to adjust the level of output sparsity to include more similar or more unique images.

More broadly, we hope to see a push towards imagination and visualization of thoughts and reasoning. We observe that IMAGINE-T and IMAGINE-I embeddings place different levels of emphasis depending on stylistic image elements like environment and color, and prompts seem to optimize for probable relevance and coverage (maximal sparsity) when generated together. For instance, when asked about a favorite sport activity for the CameraRollQA task, IMAGINE-I opts to generate a mix of indoor and outdoor, team and individual, ball and no-ball, day and nighttime sports to maximize coverage the expected distribution of possible camera roll contents. On the flip side, missing concepts from its diffusion prompts can highlight image compositional elements the model considers less important or user preferences that the model does not know of.

It would be fascinating to run larger-scale experiments to better understand how agents with different reasoning capacities exploit the notion of imagined embeddings to optimize performance. In addition, several techniques including arithmetic operations or averages over multiple imagined embeddings and/or user inputs may demonstrate interesting results.

3.7 Conclusion

In this work, we present the novel formulation of imagined embeddings for improved imagine retrieval. Through the IMAGINE-I diffusion-based hypothetical embedding system and IMAGINE-T DALL·E input-based hypothetical embedding system, we demonstrate our image retrieval system can efficiently and effectively be integrated into user-facing compositional systems of pre-trained models. We present CameraRollQA, an evaluation benchmark for image-retrieval tasks involving user datasets. We make a full implementation of our system freely available[4] and allow users to interact with and manipulate their own camera rolls by simply uploading a set of images. We hope this work becomes part of a growing movement to increase levels of imagination and reasoning in image retrieval contexts in order to create safer and more reliable user-facing systems.

[4] For the most up-to-date implementation, please refer to the link `github.com/tarunamarnath/Image-RAG`, and feel free to contact the authors with any questions.

Chapter 4

Conclusion

Neural networks reside in black boxes. Once the structure of the model is set, the inner workings of why a particular parameter is trained or reasoning behind an output is not intuitively comprehensible. We know that feeding more data improves output and can understand the mechanisms of broader ideas like attention [67], but beyond a superficial level, generative models, composed of billions of parameters, remain a mystery. Some papers have attempted to peel the layers back by studying where attention is applied or guiding LLMs through reasoning tasks to determine the level to which they succeed with deriving new conclusions beyond the training data provided [68][69]. While this stops short of true explicability, the black box has been cracked, ever so slightly.

The work in this thesis discusses pre-processing and post-processing results, suggesting guardrails beyond the parameters of the network themselves. While the core generation module might stay shrouded in darkness for a while, we can ensure that if the outputs are used for any purpose, they are taken at more than direct face value. Ensuring feasibility guarantees, providing sources, and improving retrieval can all improve results when implemented in real-world systems.

When AI is placed into the public domain, engineers have a responsibility to understand the dangers of their implementations and curb negative side-effects of their work. End-to-end design must take these ramifications into account. One important consideration is the target audience of any systems; scientists have a duty to collaborate with those whom their developments can effect. For example, while education stands to benefit from deployment of personalized study guides and resources become ubiquitous through AI tools, over 90% of online higher education materials used as training data are sourced from North America or Europe, which may leave most of the world's knowledge behind [70]. Social manipulation and deception, including deepfakes, can target more vulnerable groups in society, spread false information, and disrupt logical discourse, which is best combated through enforcing transparency and non-agentic guardrails [71]. Jobs will shift between sectors, and shuffling employment will naturally ferment dissatisfaction [72]. Approaching shifting societal dynamics with empathy over confrontation and determining a suitable middle ground can mitigate the effects of the displacement while considering the human factor in this change.

The widespread use of GPUs has significant environmental consequences as well [73]; climate change effects have disproportional effects that are more likely to target groups not actively involved in creating AI systems. Well-meaning engineers have long wielded the vision of safe AGI as a utopian future on the horizon. However, this promise of a brighter reality has pushed aside legal and social barriers in favor of building vast all-encompassing systems. They may work extremely well but do not give second thought to the ideas that power them and have little representation from groups less involved in building out the models [74].

This technology absolutely has potential as well. Education, transportation, medicine, and many other fields have space to grow with the productivity brought by AI and even generative models specifically. Perhaps we will soon understand how the black box of neural networks work or transition to a different kind of architecture that uses as little energy as our own brains. Until then, thoroughly studying the consequences involved, clearly defining task specifications, and establishing necessary guardrails can lead to a safe, sustainable future in conjunction with the scientific advancements.

Bibliography

[1] M. Chen, S. Mei, J. Fan, and M. Wang, "An overview of diffusion models: Applications, guided generation, statistical rates and optimization," *arXiv preprint arXiv:2404.07771*, 2024.

[2] K. Black, M. Nakamoto, P. Atreya, *et al.*, *Zero-shot robotic manipulation with pre-trained image-editing diffusion models*, 2023. arXiv: `2310.10639 [cs.RO]`.

[3] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine, *Planning with diffusion for flexible behavior synthesis*, 2022. arXiv: `2205.09991 [cs.LG]`.

[4] C. Chi, S. Feng, Y. Du, *et al.*, "Diffusion policy: Visuomotor policy learning via action diffusion," *arXiv preprint arXiv:2303.04137*, 2023.

[5] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," *Advances in neural information processing systems*, vol. 32, 2019.

[6] R. M. Neal, "Mcmc using hamiltonian dynamics," *arXiv preprint arXiv:1206.1901*, 2012.

[7] P. Foehn, D. Brescianini, E. Kaufmann, *et al.*, "Alphapilot: Autonomous drone racing," *Autonomous Robots*, vol. 46, no. 1, pp. 307–320, 2022.

[8] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, 2023.

[9] D. Hanover, A. Loquercio, L. Bauersfeld, *et al.*, "Autonomous drone racing: A survey," *IEEE Transactions on Robotics*, 2024.

[10] M. Muller, V. Casser, N. Smith, D. L. Michels, and B. Ghanem, "Teaching uavs to race: End-to-end regression of agile controls in simulation," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[11] A. Ramesh, M. Pavlov, G. Goh, *et al.*, *Zero-shot text-to-image generation*, 2021. arXiv: `2102.12092 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2102.12092`.

[12] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics: Results of the 11th International Conference*, Springer, 2018, pp. 621–635.

[13] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE international conference on robotics and automation*, IEEE, 2011, pp. 2520–2525.

[14] A. Loquercio, A. I. Maqueda, C. R. Del-Blanco, and D. Scaramuzza, "Dronet: Learning to fly by driving," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1088–1095, 2018.

[15] J. Ho, A. Jain, and P. Abbeel, *Denoising diffusion probabilistic models*, 2020. arXiv: 2006.11239 [cs.LG].

[16] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International conference on machine learning*, PMLR, 2015, pp. 2256–2265.

[17] A. Nichol and P. Dhariwal, *Improved denoising diffusion probabilistic models*, 2021. arXiv: 2102.09672 [cs.LG].

[18] W. Feller, *On the theory of stochastic processes, with particular reference to applications, p 403–432*, 1949.

[19] L. Weng, "What are diffusion models?" *lilianweng.github.io*, Jul. 2021. [Online]. Available: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/.

[20] P. Dhariwal and A. Nichol, *Diffusion models beat gans on image synthesis*, 2021. arXiv: 2105.05233 [cs.LG].

[21] J. Ho and T. Salimans, *Classifier-free diffusion guidance*, 2022. arXiv: 2207.12598 [cs.LG].

[22] L. Chen, K. Lu, A. Rajeswaran, *et al.*, *Decision transformer: Reinforcement learning via sequence modeling*, 2021. arXiv: 2106.01345 [cs.LG].

[23] M. Janner, Q. Li, and S. Levine, *Offline reinforcement learning as one big sequence modeling problem*, 2021. arXiv: 2106.02039 [cs.LG].

[24] N. M. M. Shafiullah, Z. J. Cui, A. Altanzaya, and L. Pinto, *Behavior transformers: Cloning k modes with one stone*, 2022. arXiv: 2206.11251 [cs.LG].

[25] Z. J. Cui, Y. Wang, N. M. M. Shafiullah, and L. Pinto, *From play to policy: Conditional behavior generation from uncurated robot data*, 2022. arXiv: 2210.10047 [cs.RO].

[26] T. Pearce, T. Rashid, A. Kanervisto, *et al.*, *Imitating human behaviour with diffusion models*, 2023. arXiv: 2301.10677 [cs.AI].

[27] S. Levine, *Reinforcement learning and control as probabilistic inference: Tutorial and review*, 2018. arXiv: 1805.00909 [cs.LG].

[28] Z. Wang, J. J. Hunt, and M. Zhou, *Diffusion policies as an expressive policy class for offline reinforcement learning*, 2023. arXiv: 2208.06193 [cs.LG].

[29] A. Ajay, Y. Du, A. Gupta, J. Tenenbaum, T. Jaakkola, and P. Agrawal, "Is conditional generative modeling all you need for decision-making?" *arXiv preprint arXiv:2211.15657*, 2022.

[30] Z. Liang, Y. Mu, M. Ding, F. Ni, M. Tomizuka, and P. Luo, *Adaptdiffuser: Diffusion models as adaptive self-evolving planners*, 2023. arXiv: 2302.01877 [cs.LG].

[31] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer. arxiv," *arXiv preprint arXiv:1709.07871*, 2017.

[32] D. Burke, A. Chapman, and I. Shames, "Generating minimum-snap quadrotor trajectories really fast," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 1487–1492.

[33] H. X. Pham, H. I. Ugurlu, J. Le Fevre, D. Bardakci, and E. Kayacan, "Deep learning for vision-based navigation in autonomous drone racing," in *Deep learning for robot perception and cognition*, Elsevier, 2022, pp. 371–406.

[34] E. Kaufmann, M. Gehrig, P. Foehn, *et al.*, "Beauty and the beast: Optimal methods meet learning for drone racing," in *2019 international conference on robotics and automation (ICRA)*, IEEE, 2019, pp. 690–696.

[35] H. X. Pham, I. Bozcan, A. Sarabakha, S. Haddadin, and E. Kayacan, "Gatenet: An efficient deep neural network architecture for gate perception using fish-eye camera in autonomous drone racing," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2021, pp. 4176–4183.

[36] E. Kaufmann, A. Loquercio, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: Learning agile flight in dynamic environments," in *Conference on Robot Learning*, PMLR, 2018, pp. 133–145.

[37] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza, "Learning high-speed flight in the wild," *Science Robotics*, vol. 6, no. 59, eabg5810, 2021.

[38] T. Lee, S. Mckeever, and J. Courtney, "Flying free: A research overview of deep learning in drone navigation autonomy," *Drones*, vol. 5, no. 2, p. 52, 2021.

[39] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement learning for uav attitude control," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 2, pp. 1–21, 2019.

[40] Y. Song, A. Romero, M. Müller, V. Koltun, and D. Scaramuzza, "Reaching the limit in autonomous racing: Optimal control versus reinforcement learning," *Science Robotics*, vol. 8, no. 82, eadg1462, 2023.

[41] R. Penicka, Y. Song, E. Kaufmann, and D. Scaramuzza, "Learning minimum-time flight in cluttered environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7209–7216, 2022.

[42] B. Amos, I. Jimenez, J. Sacks, B. Boots, and J. Z. Kolter, "Differentiable mpc for end-to-end planning and control," *Advances in neural information processing systems*, vol. 31, 2018.

[43] G. Li, M. Mueller, V. Casser, N. Smith, D. L. Michels, and B. Ghanem, "Oil: Observational imitation learning," *arXiv preprint arXiv:1803.01129*, 2018.

[44] M. Müller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem, "Sim4cv: A photo-realistic simulator for computer vision applications," *International Journal of Computer Vision*, vol. 126, pp. 902–919, 2018.

[45] X. Dai, Y. Mao, T. Huang, N. Qin, D. Huang, and Y. Li, "Automatic obstacle avoidance of quadrotor uav via cnn-based learning," *Neurocomputing*, vol. 402, pp. 346–358, 2020.

[46] A. Bhattacharya, N. Rao, D. Parikh, P. Kunapuli, N. Matni, and V. Kumar, "Vision transformers for end-to-end vision-based quadrotor obstacle avoidance," *arXiv preprint arXiv:2405.10391*, 2024.

[47] R. Madaan, N. Gyde, S. Vemprala, *et al.*, *Airsim drone racing lab*, 2020. arXiv: `2003.05654 [cs.RO]`. [Online]. Available: `https://arxiv.org/abs/2003.05654`.

[48] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[49] S. Karaman and E. Frazzoli, *Sampling-based algorithms for optimal motion planning*, 2011. arXiv: `1105.1186 [cs.RO]`. [Online]. Available: `https://arxiv.org/abs/1105.1186`.

[50] N. Jaquier, M. C. Welle, A. Gams, *et al.*, "Transfer learning in robotics: An upcoming breakthrough? a review of promises and challenges," *arXiv preprint arXiv:2311.18044*, 2023.

[51] J. Whitman and H. Choset, "Learning modular robot visual-motor locomotion policies," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023, pp. 11 908–11 914.

[52] E. S. Hu, K. Huang, O. Rybkin, and D. Jayaraman, "Know thyself: Transferable visual control policies through robot-awareness," *arXiv preprint arXiv:2107.09047*, 2021.

[53] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, *R3m: A universal visual representation for robot manipulation*, 2022. arXiv: `2203.12601 [cs.RO]`. [Online]. Available: `https://arxiv.org/abs/2203.12601`.

[54] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.

[55] R. Bonatti, R. Madaan, V. Vineet, S. Scherer, and A. Kapoor, "Learning visuomotor policies for aerial navigation using cross-modal representations," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 1637–1644.

[56] Z. Xue and T. Gonsalves, "Vision based drone obstacle avoidance by deep reinforcement learning," *Ai*, vol. 2, no. 3, pp. 366–380, 2021.

[57] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5745–5753.

[58] P. Lewis, E. Perez, A. Piktus, *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[59] L. Gao, X. Ma, J. Lin, and J. Callan, "Precise zero-shot dense retrieval without relevance labels," *arXiv preprint arXiv:2212.10496*, 2022.

[60] H. Liu, D. Tam, M. Muqeeth, *et al.*, "Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1950–1965, 2022.

[61] E. J. Hu, Y. Shen, P. Wallis, *et al.*, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[62] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[63] A. Mishra, A. Asai, V. Balachandran, *et al.*, "Fine-grained hallucination detection and editing for language models," *arXiv preprint arXiv:2401.06855*, 2024.

[64] A. Hannun, J. Digani, A. Katharopoulos, and R. Collobert, *MLX: Efficient and flexible machine learning on apple silicon*, version 0.0, 2023. [Online]. Available: https://github.com/ml-explore.

[65] A. Radford, J. W. Kim, C. Hallacy, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, PMLR, 2021, pp. 8748–8763.

[66] S. Tong, E. Jones, and J. Steinhardt, "Mass-producing failures of multimodal systems with language models," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[67] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[68] Y. Tian, Y. Wang, Z. Zhang, B. Chen, and S. Du, *Joma: Demystifying multilayer transformers via joint dynamics of mlp and attention*, 2024. arXiv: 2310.00535 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2310.00535.

[69] K. Li, A. K. Hopkins, D. Bau, F. Viégas, H. Pfister, and M. Wattenberg, *Emergent world representations: Exploring a sequence model trained on a synthetic task*, 2024. arXiv: 2210.13382 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2210.13382.

[70] UNESCO. "Generation ai: Navigating the opportunities and risks of artificial intelligence in education." (2024), [Online]. Available: https://www.unesco.org/en/articles/generation-ai-navigating-opportunities-and-risks-artificial-intelligence-education.

[71] C. Tarsney, *Deception and manipulation in generative ai*, 2024. arXiv: 2401.11335 [cs.CY]. [Online]. Available: https://arxiv.org/abs/2401.11335.

[72] A. S. George, "Artificial intelligence and the future of work: Job shifting not job loss," *Partners Universal Innovative Research Publication*, vol. 2, no. 2, pp. 17–37, 2024.

[73] J. B. Keller, M. Donoghoe, and A. M. Perry. "The tescreal bundle: Eugenics and the promise of utopia through artificial general intelligence." (2024), [Online]. Available: https://www.brookings.edu/articles/the-us-must-balance-climate-justice-challenges-in-the-era-of-artificial-intelligence.

[74] T. Gebru and É. P. Torres, "The tescreal bundle: Eugenics and the promise of utopia through artificial general intelligence," *First Monday*, 2024.

[75] S. Aeron, E. LLontop, A. Adler, W. C. Agboh, M. R. Dogar, and K. Goldberg, *Pushmog: Efficient pushing to consolidate polygonal objects for multi-object grasping*, 2023. arXiv: 2306.14021 [cs.RO].

[76] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 5026–5033. DOI: 10.1109/IROS.2012.6386109.

[77] W. C. Agboh, J. Ichnowski, K. Goldberg, and M. R. Dogar, *Multi-object grasping in the plane*, 2022. DOI: 10.48550/ARXIV.2206.00229. [Online]. Available: https://arxiv.org/abs/2206.00229.

[78] T. Sakamoto, W. Wan, T. Nishi, and K. Harada, "Efficient picking by considering simultaneous two-object grasping," 2021.

[79] A. Shenoy, T. Chen, and Y. Sun, *Multi-object grasping – generating efficient robotic picking and transferring policy*, 2021. arXiv: 2112.09829 [cs.RO].

[80] W. Yuan, J. A. Stork, D. Kragic, M. Y. Wang, and K. Hang, "Rearrangement with nonprehensile manipulation using deep reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 270–277. DOI: 10.1109/ICRA.2018.8462863.

[81] E. Arruda, M. J. Mathew, M. Kopicki, M. N. Mistry, M. Azad, and J. L. Wyatt, "Uncertainty averse pushing with model predictive path integral control," *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 497–502, 2017.

[82] D. Müllner, *Modern hierarchical, agglomerative clustering algorithms*, 2011. DOI: 10.48550/ARXIV.1109.2378. [Online]. Available: https://arxiv.org/abs/1109.2378.

[83] Z. Bar-Joseph, D. K. Gifford, and T. S. Jaakkola, *Fast optimal leaf ordering for hierarchical clustering*, Jun. 2001. DOI: 10.1093/bioinformatics/17.suppl_1.S22. [Online]. Available: https://doi.org/10.1093/bioinformatics/17.suppl%5C_1.S22.

[84] A. Lobbezoo, Y. Qian, and H.-J. Kwon, "Reinforcement learning for pick and place operations in robotics: A survey," *Robotics*, vol. 10, no. 3, 2021, ISSN: 2218-6581. DOI: 10.3390/robotics10030105. [Online]. Available: https://www.mdpi.com/2218-6581/10/3/105.

[85] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Icml*, Citeseer, vol. 99, 1999, pp. 278–287.

[86] H. Ha, P. Florence, and S. Song, *Scaling up and distilling down: Language-guided robot skill acquisition*, 2023. arXiv: 2307.14535 [cs.RO].

[87] W. C. Agboh, S. Sharma, K. Srinivas, *et al.*, *Learning to efficiently plan robust frictional multi-object grasps*, 2022. DOI: 10.48550/ARXIV.2210.07420. [Online]. Available: https://arxiv.org/abs/2210.07420.

[88] K. Zakka, Y. Tassa, and MuJoCo Menagerie Contributors, *MuJoCo Menagerie: A collection of high-quality simulation models for MuJoCo*, 2022. [Online]. Available: http://github.com/google-deepmind/mujoco_menagerie.

[89] P. Virtanen, R. Gommers, T. E. Oliphant, *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[90] C. R. Harris, K. J. Millman, S. J. van der Walt, *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2.

[91] V. Makoviychuk, L. Wawrzyniak, Y. Guo, *et al.*, *Isaac gym: High performance gpu-based physics simulation for robot learning*, 2021. arXiv: 2108.10470 [cs.RO].

Appendix A

Drone Diffuser Appendix

A.1 Hyperparameters for RRT*

Max Branch Length	10
Path Resolution	0.5
Goal Sample Rate	0.05
Maximum Iterations	200

Table A.1: Hyperparameters for RRT*

A.2 Hyperparameters for the Drone Diffusion Policy

Horizon	16
Observation Steps	1
Action Steps	8
Diffusion Timesteps	100
Beta Start	0.0001
Beta End	0.02
Beta Schedule	Squared Cosine
Diffusion Step Embed Dimension	128
Convolution Dimensions	[512, 1024, 2048]
Kernel Size	5
Convolution Groups	8
EMA Power	0.75
Learning Rate Scheduler	cosine
Optimizer	AdamW
Learning Rate	1e-4
Weight Decay	1e-6
Random Seed	42
Validation Ratio	0.05

Table A.2: Drone Diffusion Policy Hyperparameters

Appendix B

RAG Alignment Appendix

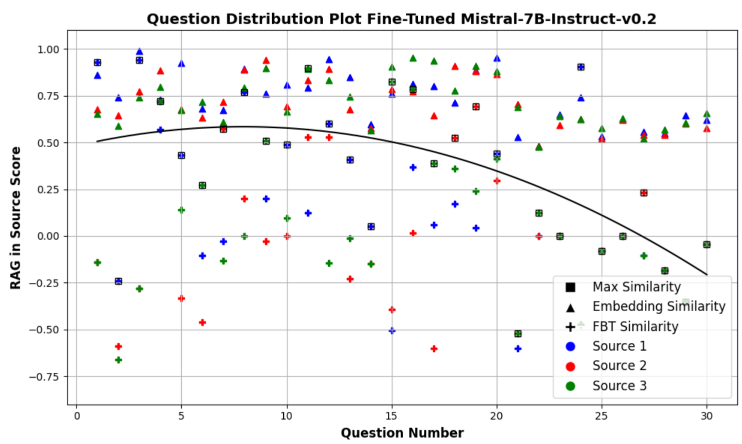


Figure B.1: FBT vs Embedding Similarity as questions leave dataset distribution on Fine-Tuned Mistral-7B

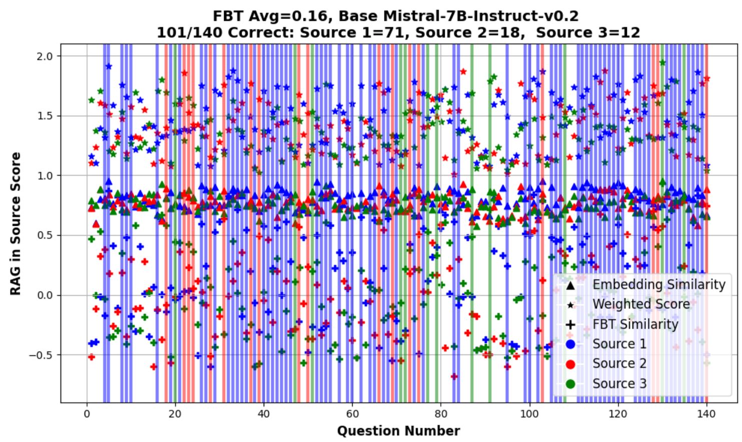


Figure B.2: Chat-Bot RAG performance with HyDE and Base M7B

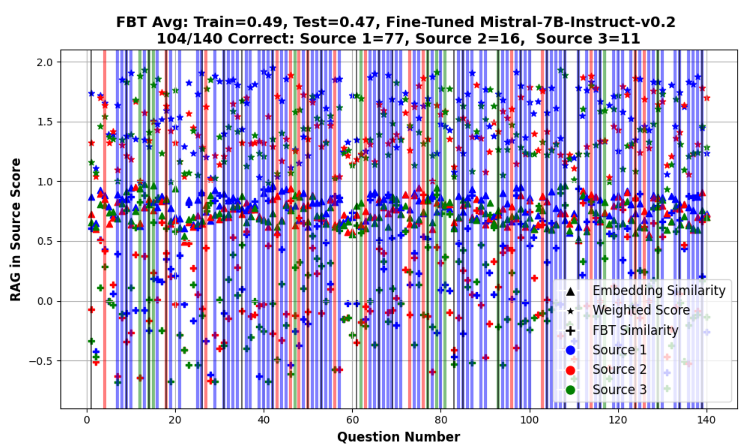


Figure B.3: Chat-Bot RAG performance with HyDE and Fine-Tuned M7B

Appendix C

IMAGINE Appendix

C.1 Language Model Prompts

We used the following language model prompts to generate the necessary text that would feed into later parts of the IMAGINE pipeline. Note that the naive CLIP baseline involves no analysis or filtering of the original CLIP input, motivating the creation of the LM Prompt Modification C.1 to provide a more fair and direct evaluation of the effectiveness of imagined embeddings.

LM Prompt Modification (Augmented Baseline)

- For image retrieval:

> Consider the examples: input - 'remember a time when you got hurt', output - 'hurt'; input - 'think about a time you played a sport', output - 'played a sport; input - 'remember a time on the beach', output - 'beach'. In one sentence, describe the parts of the user input that describe relevant source images for the requested memory: {user prompt}

- For image modification:

> Consider the examples: input - 'turn the house into a cartoon', output - 'house'; input - 'put a hat on my professor', output - 'my professor'; input - 'make the boy and the girl switch sides', output - 'the boy and the girl'. In one sentence, describe the parts of the user input that describe a relevant source image for the requested image edit: {user prompt}

IMAGINE-I Prompts

- For image retrieval:

> Please generate {k} photorealistic Dall-E prompts for relevant source images for the requested memory, (ex: input - ‘recall a time when you got sunburned’, output - ‘1. A beach scene with a clear sky and blazing sun. The exposed skin of the subject shows the effects of sunburn. The reddened areas are visible, conveying discomfort. 2. A summer picnic in a lush green park. The subject is seated on a blanket, enjoying the warmth of the sun. However, the aftermath of a day spent outdoors is evident on their skin, with patches of redness indicating sunburn. 3. A day at the pool with friends, laughter echoing in the background. The subject’s shoulders and face are visibly sunburned from hours of swimming and soaking up the sun’s rays. 4. A hiking trip in the mountains on a sunny day. The subject’s face and arms show the telltale signs of sunburn, contrasting with the breathtaking scenery of towering peaks and clear blue skies.’). Arrange them numerically. The memory is: {user prompt}

- For image modification:

> Please generate k photorealistic Dall-E prompt{’s’ if k $\geq$ 2 else ”} for relevant source images for the requested image edit, IGNORING information about the transformation (ex: if the input transforms the dog into a mouse, the generated output for the source image description should include no mention of the word mouse). Example: (input - ‘change the glasses into sunglasses’, output - ‘1. An image of a person wearing glasses sitting outdoors on a sunny day. 2. A portrait of a woman wearing glasses standing against a city skyline backdrop. 3. An image of a man wearing glasses smiling while holding a cup of coffee. 4. A close-up photo of a person wearing glasses reading a book in a cozy library setting.’). Arrange them numerically: {user prompt}

IMAGINE-T PROMPTS

- For image retrieval:

> Please generate {k} captions for relevant source images for the requested memory. Arrange them numerically: **user prompt**

- For image modification:

> Please generate {k} caption{‘s’ if k $\geq$ 2 else ”} for relevant source images for the requested image edit, IGNORING information about the transformation (ex: if the input transforms the dog into a mouse, the generated output for the source image description should include no mention of the word mouse). Arrange them numerically: {user prompt}

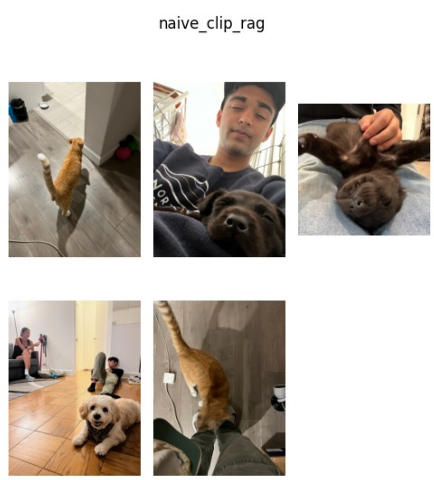
C.2 Evaluations

What pets do I have?

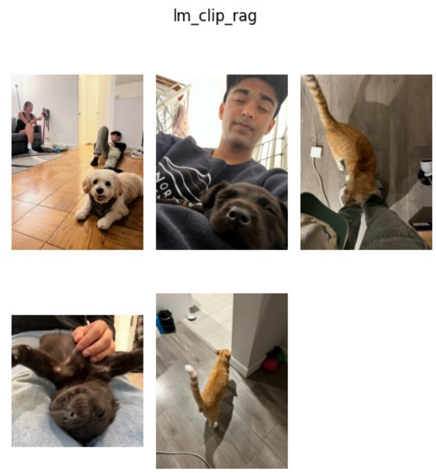
Naive CLIP RAG Based on the images provided, it appears that you have five pets: two cats, two dogs, and a puppy. The first cat is an orange tabby, recognizable by its distinctive mackerel pattern, seen walking ahead in the corridor. There's also a charming black cat held playfully in your lap, showing off a relaxed and affectionate moment between the two of you.

Additionally, you have two dogs. The white dog with a lovely fluffy coat and an adorable expression seems to be a well-loved member of your home, spotted lounging comfortably on the hardwood floor. A distinctive feature is its stylish, patterned bandana. Moreover, there's a heartwarming photo of you holding a very young black puppy, which looks to be a new addition to your family, comfortable and sleepy in your arms.

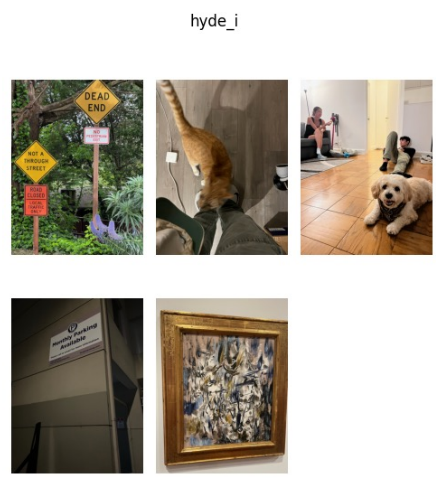
The collection of images collectively tells a story of a diverse and loving pet family, each animal unique in its way, sharing moments around your home.



LM CLIP RAG Based on the images provided, you have a diverse group of pets including at least two dogs and three cats. The dogs appear to be a fluffy white breed, possibly a Bichon Frise or a similar small breed, and a small, black puppy of an unspecified breed. The cats include two that are orange tabby and one that is a dark grey or black. Each pet seems well-loved and integrated into various aspects of your home life.



IMAGINE-I (Error: LMM failed.)

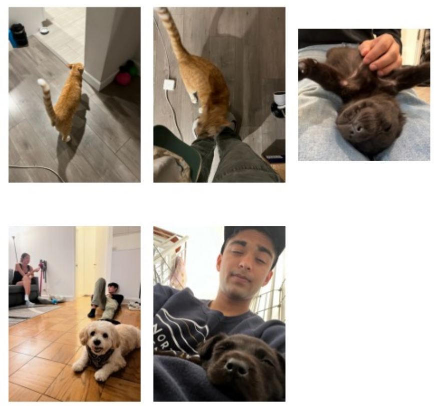


IMAGINE-T Based on the images you've provided, it looks like you have several pets. You have three cats and two dogs:

1. A ginger tabby cat with a white tip on its tail. This cat is seen in a few images, indicating that it is likely a significant part of your home life.
2. A black cat, shown being cradled while lying upside down, which suggests it enjoys being close and comfortable with humans.
3. Another cat visible in one image interacting around your living space.
4. A small white dog with a curly coat, likely a breed such as a Bichon Frise or Maltese, which is lying on the floor in one of the images.
5. A small dark brown or black puppy, seen being cuddled by someone, indicative of likely being a recent addition to your household.

These pets seem to be well-integrated into your home, with scenes of interaction and comfort with humans, which suggests they are well-loved members of your household.

hyde_t



What style of art do I enjoy?

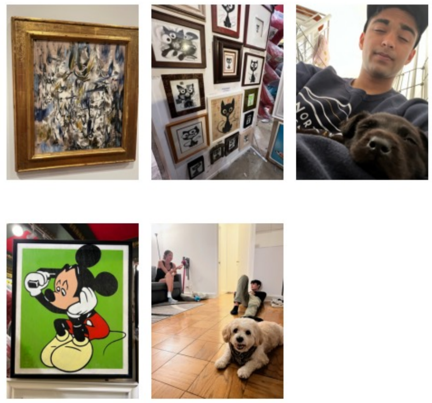
Naive CLIP RAG From the snapshots you've provided, it's evident that you enjoy a mix of art styles ranging from whimsical and cartoon-inspired pieces to more abstract and expressionistic works.

1. The first image showcases a painting with abstract and expressionistic strokes, suggesting your appreciation for more spontaneous and gestural art forms. This appreciation is typical of individuals who enjoy the emotional depth and personal interpretation that expressionistic art offers.

2. The second image, which features a collection of playful, cartoon-style cat artworks, indicates a fondness for whimsical and charming imagery that sparks joy and amusement.

3. The other images, including animated character art like the painting of Mickey Mouse, add to the evidence that you have a penchant for iconic, cartoon-based art. This style is characterized by bold, simple graphics and is closely associated with childhood and nostalgia.

Overall, your art preferences seem to lean towards engaging, expressive pieces that either evoke a sense of fun and familiarity with cartoon icons or allow for a deeper emotional and aesthetic interpretation naive_clip_rag



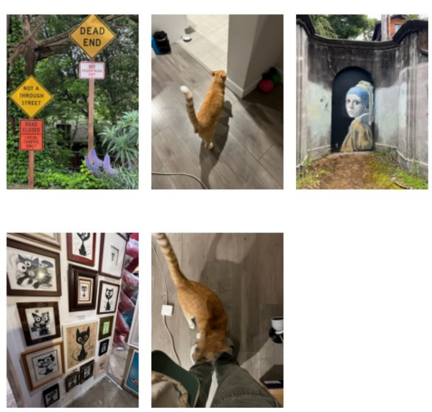
LM CLIP RAG Based on the images you've shared, it seems you enjoy a variety of art styles, reflecting both a sense of humor and a taste for classical influences. The images suggest you appreciate artistic expressions in daily life settings—even in the form of quirky road signs or casual moments with a pet, alongside more traditional gallery settings:

1. Street Art: The image featuring a mural of the "Girl with a Pearl Earring" suggests an appreciation for street art adaptations of classical masterpieces. This piece demonstrates a blend of historical art in a non-traditional, more accessible context, which might indicate your enjoyment of classical art with a modern twist.

2. Cartoony and Animated Illustrations: The multiple cartoony cat illustrations in various frames suggest a penchant for whimsical, playful art that employs simplistic but expressive characters. This style is often inviting and humor-filled, which might resonate with your lighter, more joyful side.

In essence, you likely enjoy art that is varied, accessible, and playful, with [cut off by token limit]

lm_clip_rag



IMAGINE-I Based on the images you've shared, it seems you have a diverse appreciation for different styles of art. Considering the variety of art pieces in the images:

1. The first image shows an abstract or expressionist painting, which suggests you appreciate art that emphasizes visual emotion and unique perspectives through dramatic, expressive use of color and strokes.

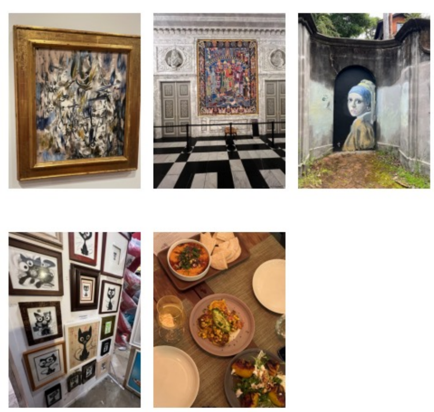
2. The second image of a tapestry with a very detailed and vibrant portrayal hints at an enjoyment of art with historical narratives or traditional motifs, often rich in color and complexity.

3. The third image indicates a fondness for urban street art, showcasing a realistic and slightly stylized portrait placed in an unconventional setting. This suggests a liking for art that integrates into everyday environments, perhaps with a touch of social commentary.

4. In the fourth image, you've shown an interest in whimsical or cartoon-style art, characterized by simplistic yet expressive characters. This style is often fun and accessible, suggesting a light-hearted approach to art appreciation.

5. While the final image [cut off by token limit]

hyde_i



IMAGINE-T Based on the images provided, it appears that you enjoy a diverse range of art styles, including contemporary and abstract art, pop art, cartoon illustrations, urban street art, and decorative or symbolic tapestries. Here is a breakdown of the styles reflected in the images:

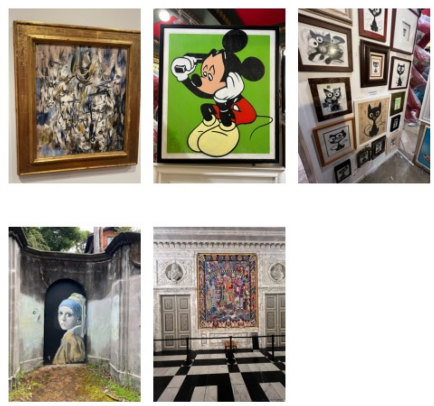
1. **Abstract and Contemporary Art**: The first image shows a painting with abstract forms and brushstrokes, indicating a taste for modern, expressionist paintings that prioritize color and form over representation.

2. **Pop Art and Iconic Cartoons**: The second image, featuring a stylized depiction of Mickey Mouse, and the various frames with cartoon characters indicate an appreciation for pop art and culturally iconic characters represented in a simple, impactful style.

3. **Illustrative and Whimsical Styles**: The third image of multiple framed illustrations of cartoon-like cats suggests a liking for whimsical and cute characters, using simple lines and colors often found in modern illustrations.

4. **Street Art**: The depiction of a classical-style painting in an [cut off by token limit]

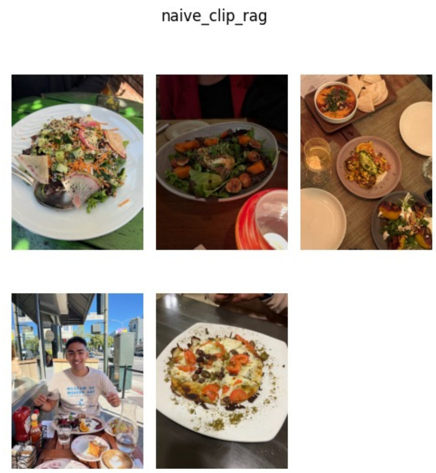
hyde_t



Do I have any dietary preferences?

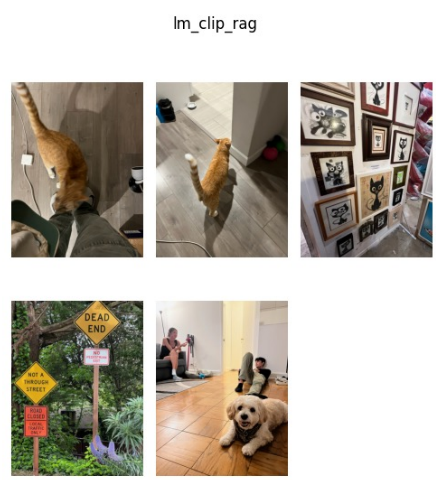
Naive CLIP RAG Based on the images you've shared, it appears that you enjoy a variety of dishes including salads, dips, grilled fruits and vegetables, pancakes, and what looks like a man'oushe or pizza topped with tomatoes, olives, and cheese. The diverse selection of foods, including fresh vegetables, fruits, grains, and dairy products, does not specifically indicate any clear dietary restrictions like strict vegetarianism, veganism, or gluten avoidance.

Your meals are rich in vegetables, suggesting a preference for healthy, possibly plant-centric cuisine, but also include cheese and possibly other ingredients like bread. Therefore, there are no overt indications of dairy or gluten restrictions based on these images alone. If you have any specific health-related dietary restrictions, such as allergies, intolerance, or other dietary conditions, these would not necessarily be evident from the photos. Always consider your personal experiences or any medical advice you've received when considering your dietary needs.



LM CLIP RAG Based on the images you've provided, it's difficult to determine if you have any specific dietary restrictions. None of the images directly depict food or meals, food packaging, or any indications of dietary preferences like vegan or vegetarian symbols, gluten-free labels, or indications of allergy warnings. The images display various scenes, including a cat at home, art featuring cats, road traffic signs, and individuals in a domestic setting without any visible food.

To accurately answer questions about dietary restrictions, details like visible food labels, meals, or a description of dietary habits from the images or additional information from you would be necessary. If you have specific dietary concerns or conditions, you might want to consider a direct consultation with a healthcare provider or nutritionist for personalized advice.



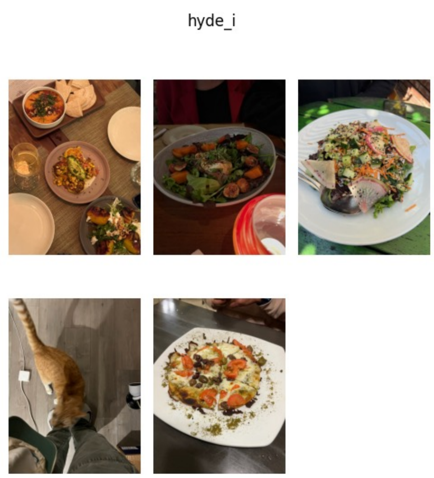
IMAGINE-I Based on the images you've shared, your dietary preferences appear to lean towards health-conscious and potentially vegetarian choices. Most of the dishes feature an array of vegetables, grains, and legumes, showcasing a focus on fresh and nutritious meals. Here's a breakdown:

1. The first image displays dishes including what looks like a creamy tomato soup with herbs and a chopped nut topping, served with pita bread; a plate with corn, avocado, and grain combos; and a salad with grilled fruits and feta cheese, suggesting a preference for substantial, veggie-rich dishes.

2. The second image features a colorful salad with mixed greens, figs, and roasted vegetables, signifying an inclination towards fresh, unprocessed ingredients.

3. The third image again highlights a fresh, colorful salad with an assortment of vegetables and seeds, reinforcing the preference for healthy and fresh ingredients.

4. The fourth image, despite being somewhat unrelated to your dietary preferences, shows a cat, which doesn't provide information about your [cut off by token limit]

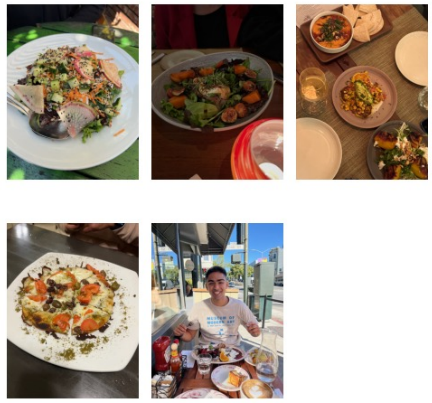


IMAGINE-T From the images you've shared, it seems that you have a preference for varied, possibly health-conscious dishes that prominently feature fresh vegetables and fruits. The meals include a mix of salads, vegetable-rich main dishes, and some vegetarian options such as a flatbread topped with vegetables and what appears to be a vegan or vegetarian pâté with pita bread.

The dishes suggest a lean towards vegetarian or possibly vegan choices, as they are primarily plant-based and include a variety of nuts, seeds, and legumes. However, there's a presence of what might be dairy or cheese in some dishes, like the flatbread, indicating you might be flexible with dairy products if you adhere to a vegetarian diet.

Additionally, the image with the breakfast setting includes a diverse plate showcasing berries, citrus fruits, and pancakes, suggesting an enjoyment of both sweet and savory breakfast items. The presence of coffee and a casual dining setting also hints at an enjoyment of relaxed, café-style meals.

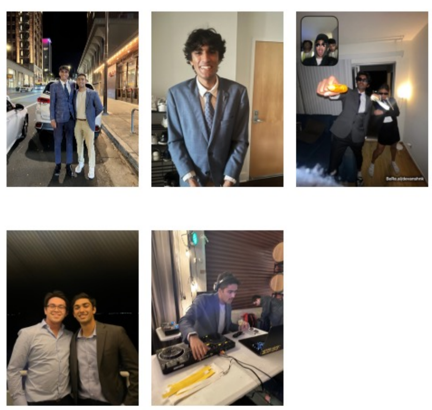
Overall, it seems your dietary preferences [cut off by token limit] hyde_t



What color is my suit jacket?

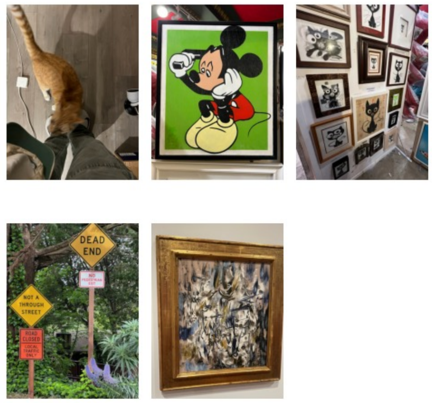
Naive CLIP RAG Your suit jacket appears to be blue in color. In the various lighting conditions shown in the images, the suit maintains a consistent blue hue, ranging from a deeper royal blue to a lighter, shinier blue depending on the light. It's styled with a classic cut, which is evident across the different events captured in the snapshots.

naive_clip_rag



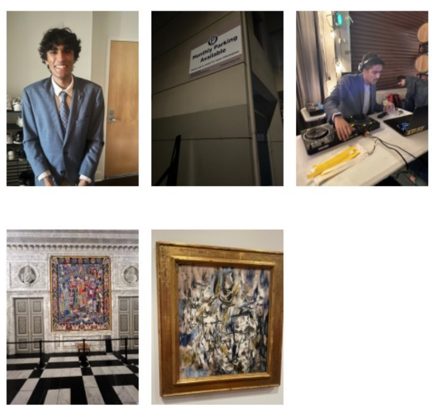
LM CLIP RAG None of the images provided show a clear view of you or your suit jacket. These images feature an orange cat, various artworks depicting cartoon characters, road signs, and a painting. Hence, I'm unable to determine the color of your suit jacket from the visuals you've shared. If you can provide an image of the suit or describe it, I'd be happy to help with your question regarding its color.

lm_clip_rag



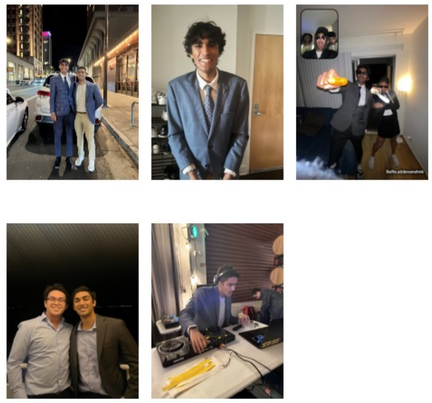
IMAGINE-I Your suit jacket is blue. In the images provided, the lighting conditions vary, so the exact shade of blue might look slightly different in each setting. In the brighter setting of the first picture, the jacket appears to be a lighter, vibrant shade of blue. In the dimmer lighting where you are DJing, the color looks a bit darker but it retains its blue hue.

hyde_i



IMAGINE-T Your suit jacket in the images provided appears to be blue. The specific shade of blue seems to vary slightly under different lighting conditions; it looks like a lighter blue under bright or flash lighting and a darker blue under dim or night lighting. The jacket also features a pattern or texture that gives it a stylish and distinct look.

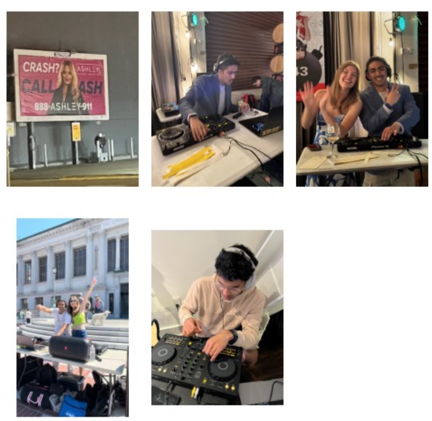
hyde_t



Have I performed live music?

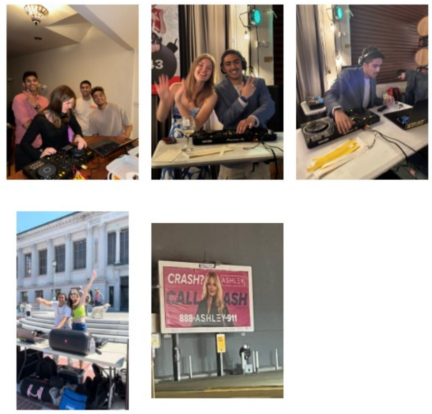
Naive CLIP RAG Yes, you have indeed performed live music. The images you provided show you engaged in DJing activities, from spinning at indoor events with other people around and equipment like laptops and DJ mixers in use, to performing outdoors in more casual settings. These snapshots clearly depict you actively involved in operating DJ equipment and engaging with the audience, which indicates live performance experiences.

naive_clip_rag



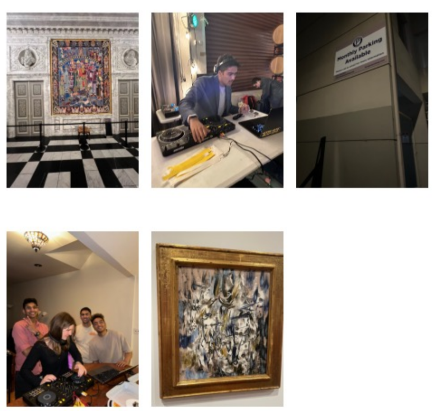
LM CLIP RAG Yes, you have performed live music. The images clearly show you engaged in activities involving mixing or playing music on electronic DJ equipment in various settings. The images display you actively using DJ controllers and managing music setups in front of audiences, both small gatherings and public venues. These snapshots express your involvement in live music performances in diverse environments, indicating a passion for and experience in playing music for live audiences.

lm_clip_rag



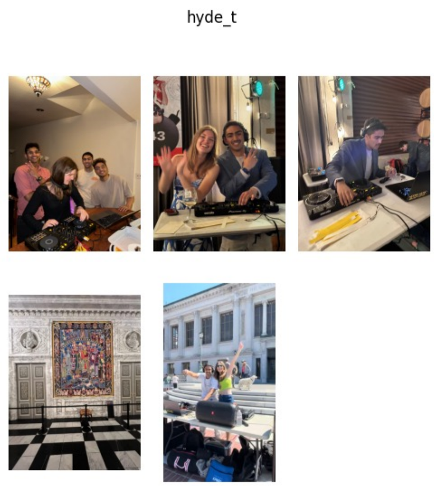
IMAGINE-I Yes, based on the images you've provided, it looks like you have indeed performed live music. Two of the photos show scenes with DJ equipment and people engaging actively in music playback and mixing, which is a form of live musical performance. In one of the images, there's a person working with DJ equipment in what appears to be a lively setting, typically associated with live DJing. In another, there's a group of people gathered around a DJ setup at what looks like a more intimate, perhaps home-based setting, also indicating a live performance scenario. These suggest that you've had experiences playing music live, likely as a DJ.

hyde_i



IMAGINE-T Yes, you have performed live music. The images you provided show you engaged in DJing activities in various settings that include both indoor and outdoor environments. These snapshots depict you operating DJ equipment, actively interacting with guests or attendees, and exuding a vibrant energy, indicative of live music performance scenarios.

Whether it's in more intimate indoor settings or larger, open, outdoor venues, you've clearly taken part in the act of playing and mixing music live for an audience.



Appendix D

Supplemental Work: RL-MOG

D.1 Introduction

In recent years, robots have witnessed a surge in applications within residential and warehouse settings, particularly in the task of decluttering spaces by efficiently collecting and relocating objects from planar surfaces into designated containers. While prevailing methodologies predominantly focus on the individual grasping of objects, Multi-Object Grasping (MOG) emerges as a promising strategy to amplify operational efficiency by maximizing the average number of objects grasped per retrieval trip (OpT). However, the conventional MOG paradigm faces challenges stemming from the requirement for precise object alignment and proximity, hindering its broader applicability.

Addressing these challenges, Push-MOG was introduced. Utilizing an algorithm designed to compute "fork pushing" actions, this approach strategically clusters objects into arrangements conducive to efficient grasping [75]. Through physical decluttering experiments, Push-MOG demonstrated a 34% increase in OpT, showcasing its effectiveness in facilitating multi-object grasps.

This paper details the development and evaluation of RL-MOG, highlighting its application in real-world de-cluttering scenarios and showcasing its potential to surpass the performance achieved by Push-MOG. By exploring the integration of deep RL, RL-MOG aims to advance the state-of-the-art in robotic manipulation strategies, paving the way for more sophisticated and adaptable solutions for multi-object grasping in diverse environments.

We evaluate RL-MOG in a simulation MuJoCo environment consisting of randomly distributed objects on a flat surface, in which the robot executes both pushes and grasps using a parallel-jaw gripper [76]. We consider two baselines: (i) Single-Object Grasping (SOG), which removes objects at random, one at a time; (ii) RL Single-Object Grasping (RL-SOG) which is a trained-from-scratch version of SOG using Proximal Policy Optimization (PPO)

Disclaimer: This section of the thesis is work collaboratively done with co-authors Shrey Aeron (aeron@berkeley.edu) and Moaaz Akbar (moaazakbar@berkeley.edu). To ensure proper credit is given to all contributors, please do not cite this thesis for this particular project. For accurate referencing and further information, please contact us directly.

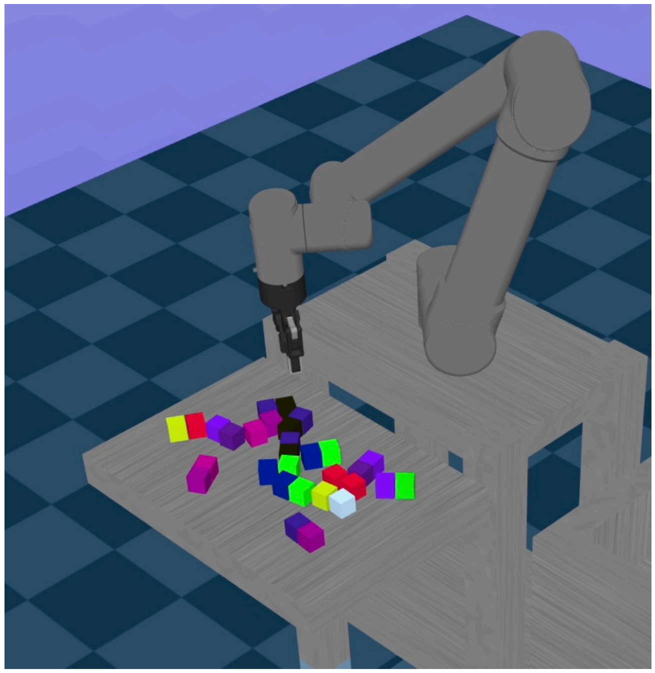


Figure D.1: The RL-MOG system executing a grasp on a cluster to grasp. (1) A cluster is identified containing 2 or more objects; (2) the gripper grasps the two or more objects that will be transported to the bin; (3) the objects are transported to the bin.

and a custom reward function. For the sake of this paper we slightly simplify some assumptions from Agboh et al. [77]. The objects are now extruded squares with roughly uniform mass and are assumed to stay flat when pushed. Due to the symmetric nature of the cuboids, the environment interacts the same with the pieces regardless of orientation.

In this work, we make the following contributions:

- We propose a new methodology of optimizing the push and grasp action, in which we reward the robot for executing actions where it pushes and grasps objects in groups to clear the bins.
- We also develop a testbed environment for training and evaluating UR5 Pick and Place optimization tasks.
- We evaluate the performance of variations of RL-MOG and two baselines

D.2 Related Work

Multi-object grasping

Sakamoto et al. proposes a method of pick and place with 2 objects at the same time [78]. The algorithm chooses between three candidate policies of single object picking, double object picking, and pushing followed by double object picking using a cost function. It utilizes a mask R-CNN for policy selection and dynamic programming for optimal pick and place planning.

Work also exists to create efficient robotic pick and place policies for a robotic hand [79]. The work uses analytical procedures to find optimal hand configurations for grasping as well as MDP modeling to generate efficient routines to clear spaces by determining when and where to multi-object grasp.

Pushing

Previous work exists specifically utilizing deep reinforcement learning to manipulate objects on flat surfaces using robot arm push motions in Yuan et al. [80]. The scope of the work establishes a system for pushing objects to a goal region without collisions, based on vision input and using deep Q-learning to learn an optimal policy.

There is also work minimizing uncertainty for robot pushing movements using different methods including Gaussian Process Regression and Mixture Density Networks in combination with a model predictive path controller in other to develop and obtain robust solutions to pushing tasks in both simulation and real robot settings in Arruda et al. [81]

Push-MOG

The previous work, Push-MOG develops an algorithm that combines the processes of pushing and multi-object grasping to improve pick and place performance. This was evaluated through a novel metric called objects per trip (OpT), a measure to determine the efficiency of multi-object grasping over single-object grasps. Fork pushing was utilized to execute stable pushing maneuvers and previous work on point clustering [82] [83] was adapted to determine how to group objects via linear push maneuvers before executing multi-object grasping.

Reinforcement Learning for Robotics

Various previous works explore reinforcement learning for pick and place procedures utilizing robotic arms. A survey that examines common methodologies for the formulation, policy optimization, reward shaping, etc. is done by Lobbezoo et al. [84]. Reward shaping [85] in particular introduces methods to effectively train policies for multi-step tasks or in complex environments applicable to robotics.

Recent work from Ha et al. [86] integrates deep learning for robotics tasks with large language models to establish an easy-to-use pipeline for learning skills. It also utilizes LLM technology to improve the training and data collection process by automatically labeling whether trajectories were successful.

D.3 Problem Statement

Agboh et al [77] studied the MOG problem to transport O objects from a flat surface to a box, and Aeron et al [75] explored pushing objects together to facilitate this action. Push-MOG, while effective, has inherent limitations in the way it computes optimal pushes. Namely, only one push can happen at a time before a grasp is attempted, and that push must maintain a constant gripper width and orientation.

Our objective with incorporating reinforcement learning is to train a more flexible policy that can perform a wider variety of actions, thereby being a more adaptable solution to the MOG problem. Namely, the policy can learn to perform multiple push maneuvers to optimally cluster objects for pickup rather than being restricted to just one like previous work was. Additionally, a wider action space opens more possibilities for the movement of the end effector. For example, the gripper can rotate as objects are pushed, or the jaws may open more widely in different scenarios to navigate around clutter while executing a push or pick action.

Using reinforcement learning, we experiment with a few different methods to learn and improve robot performance:

1. RL-based SOG
2. RL-based pushing
3. RL-based combination of MOG and pushing

Similar to the previous papers by Aeron et al. [75] and Agboh et al. [87], we consider friction in the MuJoCo model, which is applied to the objects, as well as the gripper, and the table. Frictional grasps are considered because friction matches real-world scenarios more closely.

State

Let the set of objects on the work surface be $O = \{o_0, o_1, \ldots, o_{N_o-1}\}$. We use the original formulation in the Agboh paper but slightly constrain it such that each o_i is a cube-shaped object. N_o is the number of objects on the work surface. The system is initialized with the objects arranged randomly in a pick bin, and the desired goal location is a place bin on a different side of the robot.

Observation

In our simulator, we used a camera positioned immediately above the center of the table to capture information about the workspace. This includes both RGB and depth data. One data augmentation option we experimented with was creating one-hot encodings for the locations of each cube, which is described further in Section D.4.

Action

The robot executes a sequence of *pushes* and transports (which we call *trips*); each push translates and/or rotates the gripper in the workspace to rearrange the objects, while each trip grasps a set of closely clustered objects and transfers them to the bin.

Because the robot has a choice between a pick and a push action at each step, the action space is comprised of multiple discrete parts:

1. Action type $\in \mathbb{Z}^2$: a choice between pick and push.
2. Action location 1 $\in \mathbb{Z}^{n \times m}$: either the starting location for the push or the position for the grasp, given in (x, y) coordinates of the pick table.
3. Rotation 1 $\in \mathbb{Z}$: the angle the gripper will start at for the push or go down in for the grasp in the range of $[-90, 90)$.
4. Action location 2 $\in \mathbb{Z}^{n \times m}$: the ending location for the push. This is ignored for grasp actions.
5. Rotation 2 $\in \mathbb{Z}$: the angle the gripper will end up in for the push. This is ignored for grasp actions.
6. Gripper width $\in \mathbb{Z}$: a number ranged from 0 to 25, scaled to the gripper width of 0 to 255 using a linear function. For grasps, the minimum width is bounded to 80 to ensure a block can be picked up, but for pushes, the full range is used.

We evaluate actions by the numbers of successes, s, and fails f they generate. We define one success as completing a grasp that moves blocks from the pick box to the place box and fails as pushing a block over the edge of the pick box, thereby making it irretrievable.

D.4 Experimental Setup

To evaluate the performance of the policies developed in the paper, we run experiments in a simulated workspace.

MuJoCo Setup

As seen in Fig. D.1, we use a simulated Universal Robotics (UR) 5 Robot with a Robotiq 2F-85 gripper mounted to its wrist. Simulation is performed using the MuJoCo simulator [76]. The UR5 Model is obtained from Bielefeld University[1] and the 2F-85 model is from Google Deepmind, Zakka et al. [88]. The textures for the table are obtained from Paul

[1] UR5 Model: https://github.com/corlab/cogimon-gazebo-models

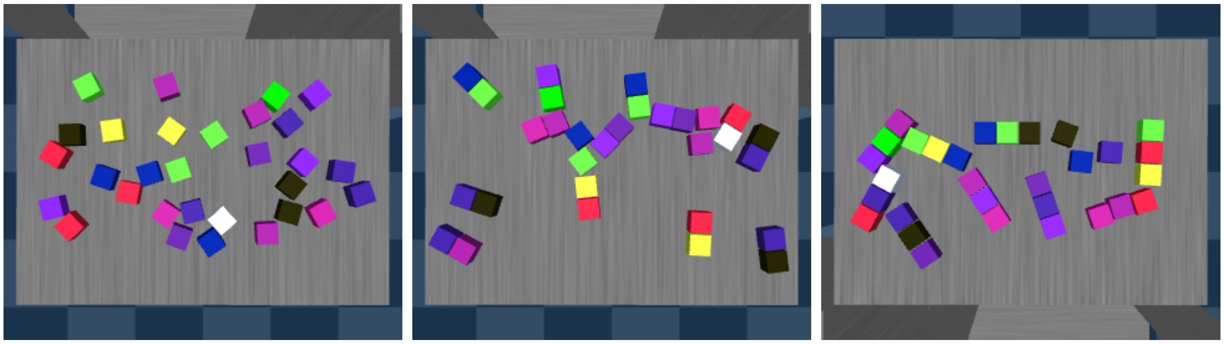


Figure D.2: Cluttered workspaces with blocks in grouped in configuration of groups of 1, 2 and 3, (left to right, respectively)

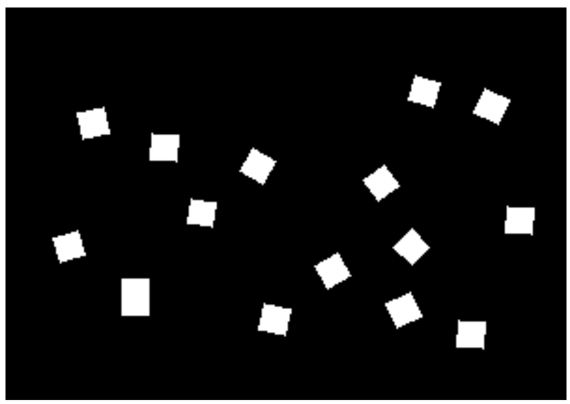


Figure D.3: This is an example output of the combined one hot encoded masks. Realistically, each of the separately polygons has its own layer and mask. However we combine them for stylistic and visual purposes.

Daniel on GitHub[2]. The coefficient of frictions on the gripper model are edited to 1.5 to better deal with the physics engine and prevent slippage.

We generate 30 MuJoCo geometries with a size of $0.015 \times 0.015 \times 0.015$ and a coefficient of friction μ_O of 1.5. Their relative mass is defined as 0.03, and the diagonal inertia matrix of $I = \begin{bmatrix} 0.01 & 0.01 & 0.01 \end{bmatrix}$. Using a random seed, we scatter the objects in the workspace. To facilitate all experiment types, we generate a few types of random placements.

1. Random single objects: Each of the 30 objects are placed in the workspace with a spacing of 0.057 to give the gripper space to grasp each object

2. Random multiple objects: The objects are considered as groups of n, such that $\frac{30}{n}$ random points are generated, with the same constraints at the single objects. However,

[2]Table Mode: https://github.com/PaulDanielML/MuJoCo_RL_UR5

here the spacing is $0.057 \cdot 1.25 \cdot 2 = 0.1425$ to give space for the larger cluster sizes. Then, the n objects are laid out linearly and rotated at random $\theta \in [-90, 90) \in \mathbb{Z}$ to generate MOG groups. For this paper, we consider $n \in [1, 2, 3]$ to generate all these points.

Each rollout generates a random scene - a few examples of randomly generated scenes can be seen in Figure D.2.

Camera Setup

To perceive the environment, we use a generic camera with depth, mounted above the pick box if the environment. This gives us RGB-D images which we can use to target points on the Pick Table's plane. The output size of the camera image is $W \times H$ pixels In addition to this, we also generate a one-hot-encoded mask of each polygon, as seen from the top down view. This is generated through the following steps.

1. We iterate through each object O in the MuJoCo model and extract its `xpos` and `xmat` parameters, which contain the object's global position and rotation, respectively.

2. The real-world coordinates are converted to image coordinates using the following formulas, based on the location of the table in space, where $I_?, W_?$ are the respective x or y coordinate in the image or world space, respectively:

$$I_y = 564.71 \cdot (W_y + 0.67) + 39$$
$$I_x = -564 \cdot (W_x - 0.25) + 19)$$

3. Based on the type of object, we calculate its vertices in 3D space and apply the rotation and transformation on it to get the real world coordinates.

4. To generate a binary mask for an object, the SciPy library [89] is utilized to calculate the projection and convex hull of the points.

5. The Pillow Python library is then used to generate a one hot encoded layer for that polygon. In this way we generate an output of $|O| \times W \times H$, which may be used to train in the modality of choice.

An example combined output of the one-hot-encoded objects is shown in Figure D.3.

Motion Setup

As this is a simulation-based experiment for a robot in real, it is important to outline the constants, constraints, and primitives defined to conduct experiments.

1. **Move in Z primitive**: the following primitive is used to define a vertical movement to or from the pick table. For movements down, as series of 2 NumPy linear spaces are [90] are utilized. The first space covers 80% of the downwards move at a relatively quick pace of 1500 milliseconds (ms) of simulator time, while the last 20% are traveled at pace that is 37.5% of the initial downwards movement, taking 4000 ms. This is done to ensure that the robot does not fling objects out of the pick table. If the z-movement is upwards, we simply complete the whole trajectory in 1500 ms as there is a near zero risk of bumping into anything!

2. **Gripper movement primitive**: Similar to the z movement primitive, we use a linear space to ensure that the movement between two gripper states is not sudden. In our initial experiments, we noticed that the forces and jerky movement of the simulator negatively affected performance. By spreading movement out into a linear space, we slow down gripper actions to take 1250 ms, which prevents large forces and accelerations on surrounding objects.

3. **Inverse Kinematics (IK)**: While IK is a well-known and relatively "solved" problem in robotics, we will outline the techniques where used to calculate positions for robot locomotion. This is due to our slightly different use of a custom IK chain and solver. We modified the IK chain, and apply a small offset of $\begin{bmatrix} 0 & 0 & 0.16 \end{bmatrix}$ to fix the extra height added from the Robotiq gripper attached. Additionally, the IK solver is *constrained* such that the gripper always faces downwards.

4. **Pushing Primitive**: One of the outputs of the network can be the push points and associated angles. To accommodate this, and more simple straight line pushes, we create a pushing primitive which takes in the start point and angle of the push, as well as the ending push and angle. A pushing action is executed the fashion seen in Figure D.4.

D.5 Experiments and Results

See Section D.3 for a refresher of the information about different metrics used in our evaluations, as they are important to the following section.

Baselines

We compare RL-MOG against Analytic Frictional SOG, which uses single-object grasps, and RL-SOG, which is a reinforcement learning implementation of SOG. This is then also compared to the analytic SOG algorithm.

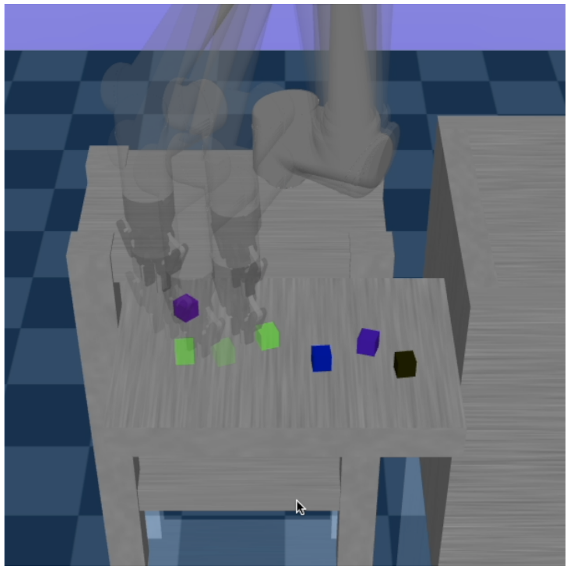


Figure D.4: This is an example of the full pushing action from end to end, where a block is pushed to another block.

Analytic SOG

This SOG assumes full knowledge of the state of the environment, meaning that it can access the low-level data stored in MuJoCo, similar to a perfect perception stack. The algorithm finds the location and rotation of a random block in the pick table and then executes a pick and place clearing action primitive aligned to it. It proceeds to repeat this for every remaining block on the pick table, until the table is cleared, or 5 consecutive failures are reached.

RL-SOG

This baseline was developed as a first attempt at having the robot learn a fairly standard pick-and-place action. We reduced the action space to only contain a single action location, orientation, and rotation, in contrast to the action space described in Section D.3. This allowed for a simple learning operation. The input was the same One Hot Encoded tensor and depth data.

Reward Functions

We experimented with many different reward functions to help the robot train a policy that pushes and grasps multiple objects.

Expected Reward

The first function we experimented with is designed to provide an explicit improvement over the Push-MOG results.

The probability of a successful grasp in earlier work was 65%, so we aim to succeed at least 80% of the time with RL-MOG. Additionally, when the original Push-MOG succeeded, it managed to pick up an average of 2.5 blocks at once. We keep this value the same in our calculation given the limitations of the size of our gripper. The following expectation is formulated:

$$\mathbb{E}[reward] = p\mathbb{E}[success] + (1-p)\mathbb{E}[fail]$$

We also want to reasonably balance push-and-pick actions. To do so, we estimate a reasonable linear amortization scheme. Specifically, based on the speed of the robot and the arrangement of the blocks, we decide that a collection of two blocks with five intermediate push steps approximately equates to a single pick. If a single pick receives a reward of 1, this gives us the success reward function $S(x) = 5x - 4$.

With a successful average of 2.5 blocks, we have an always-success reward of $5(2.5) - 4 = 8.5$. Setting the expected reward to 0 without loss of generality and using $p = 0.8$, we can calculate our fail reward: $(0.8)(8.5) + (0.3)(x) = 0$, $x = -22.67$.

Thereby, we have an initial reward function of

$$R(s, f) = 8.5 \times s - 22.67 \times f$$

Sigmoid Function

To punish failures less heavily during the initial parts of the training, we incorporated a scheduled increase in the weight of the fail reward in the form of a sigmoid function. Specifically, a weighted version of the training step was used as the input. The reward function now is

$$R(s, f) = 8.5 \times s - 22.67 \times f \times sigmoid\left(\frac{global\ step}{2500}\right)$$

As training progressed with this formulation, we saw that the punishment for failures still discouraged the robot significantly from attempting both pushes and grasps. The physics simulation software in MuJoCo calculated a push directly down above a cube as a spring force that ended up flinging the block into space, and push motions forced the objects off the table. As a result, the policy trained the robot to simply do nothing; it would attempt to grasp and push in areas of the table where no objects were positioned. The reward ended up converging to zero.

Distance Incorporation

As this null policy was trained, we noticed that the push actions the robot took did not arrange the blocks in a way that allows multi-object grasps. We incorporate a distance component into the reward function to incentivize larger clusters of blocks, thereby giving the grasping primitive the potential to pick up more than one at a time:

$$R_{push} = \max(10, 1500 \times (\bar{D}_{inital} - \bar{D}_{final})) - 50f$$

where $\bar{D}$ is the mean distance between all centers of blocks in the pick box, with a ceiling of 10 for the difference to disincentivize erratic swipes. The penalty for objects knocked out of the pick box remains. Additionally, if the robot moved no block, we factored in an additional punishment -1, helping prevent null actions.

MOG pure-pick reward

Because we were not seeing high rewards with a combination of both push and pick, we decided to train our pick and push actions separately. For pure-pick MOG, we utilize an environment with objects initialized in groups of 2 and limit the action space to only picking. To incentivize multi-object grasping over single-object grasping, we use an exponential reward function to heavily promote the successful pick-and-place of multiple blocks in a single action. The reward function takes the form:

$$R(s, f) = ((2 \times s)^3 - 7) - 20 \times f$$

This gives a significant positive reward for 2 or more grasps (r = 57 for 2 successes) and a relatively small reward for single object grasping (r = 1 for 1 success).

Pure-push reward

Here we use an environment with initially randomly spaced objects and train only with the push action space on a reward function incentivizing the grouping of blocks by positively rewarding for reducing the mean distance between blocks. Because no pick-and-place occurs, the reward does not incorporate success counts but does still penalize for fails when blocks are knocked out of the pick box:

$$R(f) = \max(10, 1500 \times (\bar{D}_{inital} - \bar{D}_{final})) - 50f$$

with the same -1 penalty for failed pushes.

Results and Analysis

Push and Pick - Sigmoid with Distance Incorporation

As seen in Figure D.5, in the short term we see successful training of the push and pick towards increased reward as the policy achieves increased successful grasps. However, in the long term, this reward structure causes the policy to converge to 0 as it learns to avoid interacting with the blocks to minimize the negative penalty associated with knocking blocks out of the pick box. This result leads us to rethink the reward function to also penalize inaction going forward.

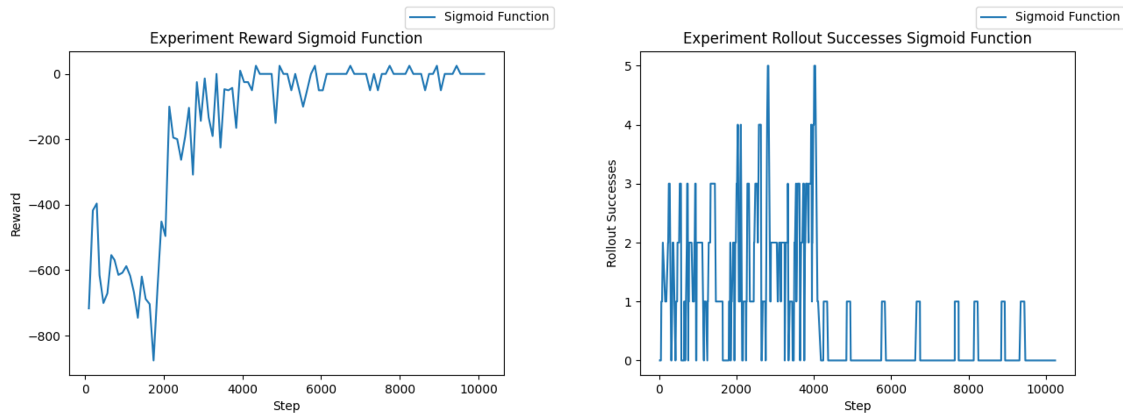


Figure D.5: The reward (left) and success (right) for each Sigmoid rollout and iteration

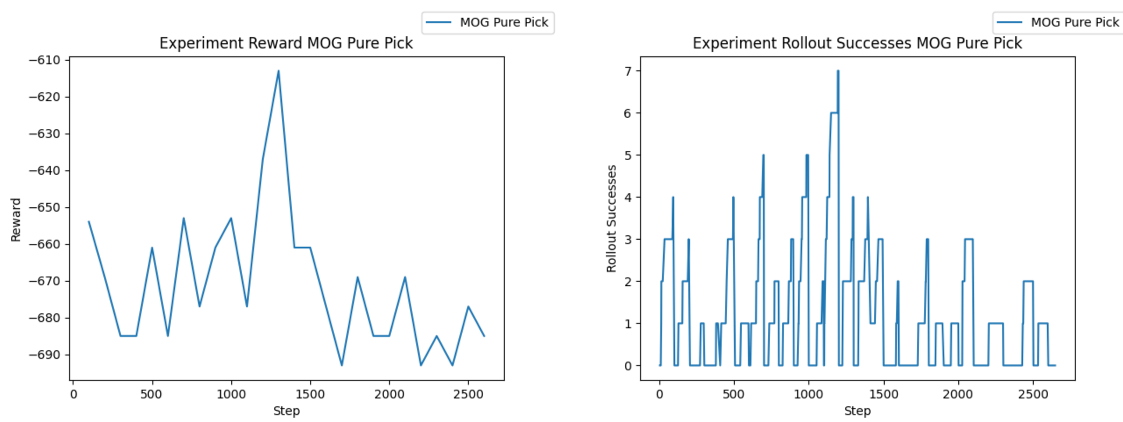


Figure D.6: The reward (top) and success (bottom) for each Pure Pick rollout and iteration

MOG pure-pick

Our experimental results for MOG pure-pick highlighted the success of our exponential success reward for learning multi-object grasps while also demonstrating overfitting due to a necessarily high learning rate in limited simulation trajectories. The policy successively improves the number of successful single pick and place actions per trajectory to a rate of 7 out of 15 blocks early in training. It then receives a massive reward after a series of successful multi-object grasps, as seen in Figure D.6. However, the policy then begins to perform significantly worse in terms of total grasp due to poor single-object grasping. This decline can be seen in Figure D.6.

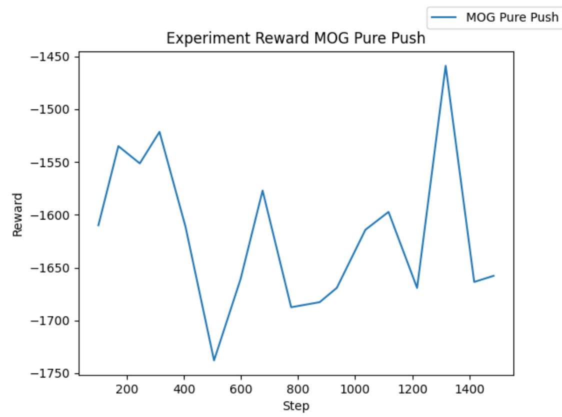


Figure D.7: The reward for each Pure Push rollout and iteration. Successes were not measured as pushing does not clear the workspace.

Pure Push

While pure push aims to bring together objects together, and is thus rewarded for its actions, we were not able to detect any significant improvements. This attests to the in-effectiveness of using mean distance as the sole metric for rating the agent's performance. We noticed that the physics engine would tend to throw off objects into ungraspable space, reaping a highly negative reward for its actions. The stagnation can be seen in Figure D.7. However, this brings up a great point for future work, as outlined in Section D.6. We could potentially use the clustering metrics introduced in Push-MOG to better quantify the notion of clusters and distances [75] to define a reward function based not only on mean distance, but also on a sense of optimal clusters, and learning from there.

D.6 Limitations and Future Work

This work proposes RL-MOG, a technique that aims to use RL to consolidate and clear objects from a flat workspace. This work has the following limitations: 1) It can only be applied to extruded squares 2) Pushing and Picking are not reliably robust, due to pushing speed and computational limits 3) The simulation environment is limited to an accuracy of ± 0.05. This is due to joint accuracy and physics limits in the MuJoCo software with our current hardware and setup.

There are multiple facets of the research which is currently and can still be expanded upon. We would like to add support for polygonal objects, as in the previous MOG papers, to introduce more complexity and generalization to the clustering problem. The variety of polygons has already been integrated into the MuJoCo simulation, but interacting with various shapes and edges has yet to be integrated into the action space of the model. An

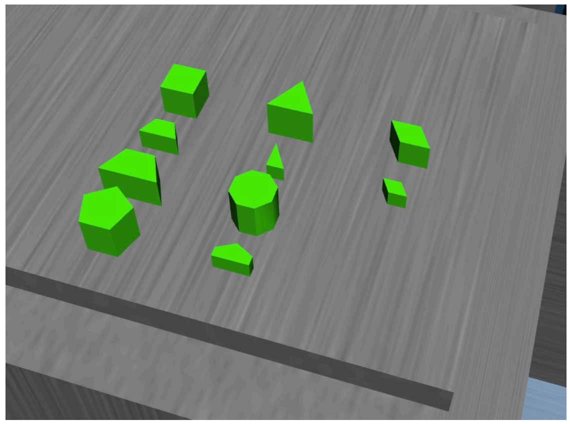


Figure D.8: This is an example of the polygons to be used in future work for RL-MOG. This will aim to make the agent more robust, as it can act regardless of the polygon shapes and edges.

example of such objects in the simulation environment is in Figure D.8.

In addition, it would be useful to explore other simulation parameters to improve simulator speed and performance in MuJoCo, with `friction`, `inertia`, `condim`, and `tendon`, to name a few. These would allow us to 1) employ better and more realistic speed and physics in the environment and 2) unlock the use of MJX, the new MuJoCo simulator, to run in parallel and directly on the GPU. An alternative would be to look into Isaac Gym [91], which allows for parallel simulation in a single environment with high performance on a GPU. This talk on simulators also leaves room for us to develop an easily reusable, robust, and fast simulation environment for not only the UR5 and Robotiq Gripper combination but also other commonly used robots.

In addition, the environment and experimental setup provide a testbed for implementing more complex architectures to better capture the combination of pushing and grasping actions required. This includes experimenting with Mutli-Task RL, hierarchical RL, and Meta-RL.

Finally, it would be beneficial to incorporate analytical clustering algorithm data as input into our policy. Experiments with pure pushing using block distance-based reward failed to give stable improvement in clustering, so combining this training with more precise measurements of success based on analytical object clustering can provide an avenue to greatly improve results.