Optimizing Active Vision-Based Policy for Robotic Grasping through 3D Point Cloud Feature Extraction- A Comparison Study

Uthiralakshmi Sivaraman
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, MA, USA
usivaraman@wpi.edu

***Abstract*—The majority of robotic grasping algorithms rely on a single image of the object, which makes the algorithm's performance highly dependent on the camera's viewpoint and significantly limits its effectiveness. Active vision enables robots to intelligently adjust their sensors, facilitating quick and reliable interactions with unknown objects in uncertain conditions. Our approach focuses on identifying the most efficient subsequent viewpoint for data collection in vision-based grasping. To optimize a vision-based policy for robotic grasping, choosing the most effective 3D point cloud feature extraction technique is crucial. These techniques convert raw 3D spatial data into structured forms that learning algorithms can use to make well-informed decisions about object manipulation. Our research compares several global feature representations to find the most suitable candidate for developing an active vision policy using a Dagger-like imitation learning algorithms.**

***Index Terms*—Robot grasping, vision-based grasping, Active Vision, Imitation Learning, Dataset Aggregation - Dagger, 3D point cloud feature extraction, object manipulation**

I. INTRODUCTION

Robotic grasping is a fundamental capability in robotic arm applications, particularly in service-oriented scenarios. Traditional grasping algorithms typically rely on data derived from a single viewpoint to generate grasps. However, these algorithms often encounter difficulties when the camera's viewpoint differs from those used during training. This discrepancy can significantly hinder grasp synthesis, especially when certain graspable features of an object are obscured or self-occluded from the camera's current viewpoint.

Historically, vision-based grasping has been constrained by these limitations, as noted in studies such as those by Fischinger et al [2], who discuss the implications of relying on single viewpoints. To address these challenges, recent advancements have incorporated active vision frameworks. These systems enhance data acquisition by allowing the camera to move dynamically around the object, thereby broadening the observational perspectives and improving grasp synthesis [3]. One innovative approach involves heuristic-based methods for optimizing camera viewpoints to ensure comprehensive coverage and focused observation of the target object. For example, Aldoma et al. [4] explored how oriented and repeatable clustered viewpoint feature histograms can significantly aid in recognizing and estimating the pose of objects, thereby facilitating more effective grasping strategies.

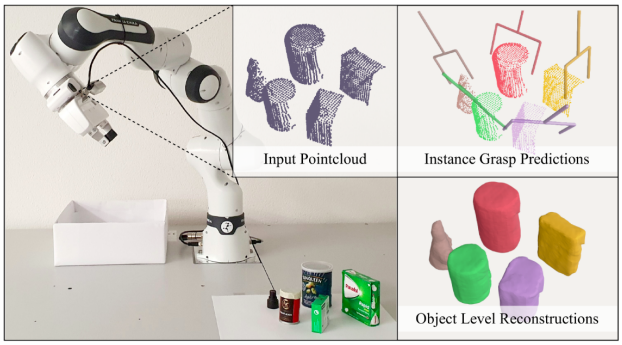


Fig. 1: Single Viewpoint Grasping [1]

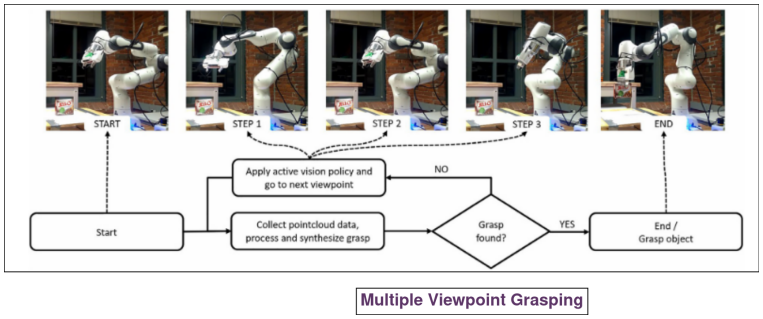


Fig. 2: Multiple Viewpoint Grasping [5]

Active vision necessitates complex optimizations across various dimensions including camera motion, robot trajectory, and data analysis. This project prioritizes minimizing camera movement to quickly and efficiently identify viable grasps for unknown objects. By reducing the travel distance of the camera, we aim to decrease the time required to establish a functional grasp, optimizing the process for practical robotic applications [6], [7]. This approach aligns with the practical needs of service robotics, where speed and reliability are

paramount. By integrating these considerations, our methodology provides a robust framework for enhancing robotic grasping capabilities in real-world applications.

II. Background

A. Advanced Strategies for Robot Grasping

Robotic grasping in unpredictable environments benefits immensely from active vision techniques. These approaches enable robots to adapt to changing conditions and optimize their grasp. Several methods focus on synthesizing grasps from a single image, while others tackle grasping in cluttered spaces without updating their strategies based on new information. Some methods use cost-effective imaging solutions to perform precise grasps but fail with new or deformed objects due to their dependency on predefined templates. Machine learning is also employed to determine optimal grasp points directly from image data, though such methods often miss out on utilizing historical image data which could further refine their processes. Active vision elevates these techniques by dynamically collecting data through scene exploration, thus enhancing the grasp quality and algorithm efficiency.

B. Expanding the Scope of Active Vision

Active vision is applicable well beyond grasping, influencing the design of non-grasping algorithms as well. [2]employs active vision for 3D scanning, not for grasping, and aims to scan new objects as quickly as possible, unlike our goal of rapid grasping. Similarly, [8] utilizes active vision for 3D reconstruction, employing an information gain metric. In our project, we want to minimize search time as well as maximize grasp quality and use expert solutions within optimal space to enhance a more realistic grasping scenario.

C. Reinforcement Learning

Reinforcement Learning (RL) [9]is utilized when traditional supervised or unsupervised machine learning methods do not fit a problem's requirements. It is especially beneficial for continuous, non-episodic challenges, which is why it has become widely adopted in robotics. RL tackles a range of problems that make it well-suited for these applications. Advances in deep RL have expanded its capability into continuous action spaces by applying function approximation not just across state spaces but also action spaces. This allows RL to evaluate the potential rewards of continuous actions within these spaces, diverging from the traditional method which involves selecting from a set of discrete actions. Recent studies have applied RL to enhance active vision for robotic grasping without expert demonstrations. By paralleling training sessions and significantly enhancing the data collected, these systems compensate for the absence of expert input. However, we now have the capability to integrate expert demonstrations into the training process.

D. Expert Training in Reinforcement Learning

In reinforcement learning, [9] [10]expert demonstrations are leveraged primarily for initial training, to inform model design with human-understandable reasoning, to establish optimization criteria, and to enhance exploration. Our research utilizes these aspects, except for legibility. illustrate that even suboptimal expert demonstrations can significantly enhance training speed and accuracy. We address common RL challenges, such as exploding gradients and vanishing gradients where predictions become overly confident and resistant to contradictory evidence—as observed in various neural networks but particularly problematic in RL due to the dependency of the reward function on network predictions.

E. Epsilon Optimal Demonstrations

Expert demonstrations are a foundational element in the field of robotics and artificial intelligence, facilitating the transfer of complex task execution capabilities to robotic systems. Human experts perform a variety of tasks, ranging from simple object manipulation to intricate navigation maneuvers, in the presence of robots. The robots, through observation, compile a detailed blueprint of task execution based on these expert demonstrations. This method circumvents the need for extensive programming for each specific task, which is both impractical and labor-intensive.

The efficacy of expert demonstrations lies in their ability to condense human expertise into a format easily digestible by robotic systems. This accelerates the robot's learning curve, enabling performance that closely mirrors human dexterity. The concept of epsilon-optimal demonstrations introduces an advanced approach where the demonstrations provided are nearly optimal, falling within a small epsilon margin of error. This ensures that not only are the tasks performed correctly, but they are also executed in the most efficient and effective manner possible.

In practical applications, epsilon-optimal demonstrations have proven to significantly speed up the learning process for robots, allowing them to quickly adapt to a wide range of tasks. This method has been particularly impactful in service robotics, where tasks require a high degree of adaptability and human-like decision-making. As robots continue to integrate into diverse sectors, the role of epsilon-optimal demonstrations in pushing the boundaries of what is achievable in robotics is increasingly recognized and valued in contemporary research.

Applying expert demonstrations in real-world scenarios has enabled robots to tackle a diverse array of challenges, from assembly line tasks to domestic service roles, where the quality and nuance of task performance are paramount. As robots continue to permeate various sectors, the role of epsilon-optimal demonstrations in advancing the state-of-the-art in robotics is increasingly becoming a focal point of contemporary research.

1) Optimal path finding as Expert demonstration: As illustrated in Figure 5 the problem involves finding the shortest path along a view sphere to view two graspable points on an object point cloud. A dense network of points is created on the view sphere using a Fibonacci lattice to facilitate this. The

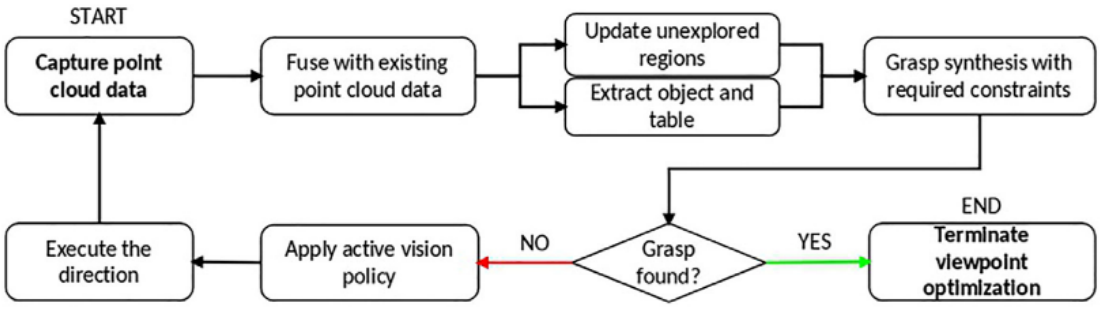


Fig. 3: Active Vision Workflow [5]

process involves three main steps: calculating visibility regions for each point on the object, identifying pairs of points that can form a grasp based on specific criteria, and computing the shortest distances between viewpoints that can view these graspable pairs. Once these distances are recorded, an optimal path is determined through brute force search, starting from the camera's initial position and calculating the shortest route connecting the viewpoints necessary to view at least one valid grasp along the path, thereby determining the most efficient viewing sequence.

F. Behavioral Cloning and Dataset Aggregation

[11] Behavioral Cloning (BC) is a form of supervised learning where the model is trained to mimic expert behavior. In this context, the "expert" refers to the optimal path finder algorithm that determines the best camera viewpoints for object manipulation tasks. By using the decisions made by this expert system as a dataset, the RL model learns to replicate these choices. BC effectively jump-starts the model's training by providing it with high-quality, expert-level decision patterns from the get-go. However, BC's main limitation is that it doesn't account for the model's interactions with the environment, which can lead to discrepancies between the training scenarios and real-world applications.

Dataset Aggregation (DAgger) [12] addresses the limitations of BC by iteratively refining the training dataset. DAgger involves the model making decisions in the simulated environment, after which the expert reviews these decisions. If the model's decisions deviate from what the expert deems optimal, these instances are added to the training dataset with the correct actions specified by the expert. This process allows the model to learn from its mistakes in a controlled manner, gradually aligning its decision-making process with that of the expert's. DAgger ensures that the model is not only trained on a static set of expert decisions but also learns to correct its course based on dynamic feedback from the expert regarding its actions.

III. Optimization of 3D Point Cloud Feature Extraction Techniques

In the pursuit of optimizing a vision-based policy for robotic grasping, selecting the most effective 3D point cloud feature extraction technique is crucial. These techniques transform raw 3D spatial data into a structured form that enables learning algorithms to interpret and make informed decisions regarding object manipulation. Some earlier works [4]have shown that the introduction of 3D feature descriptors has significantly improved the efficiency of the grasping algorithm.

A. Feature Extraction Techniques

This section details several key feature extraction techniques, each uniquely contributing to the enhancement of robotic grasping systems by providing crucial data representations.

1) Height Accumulated Features (HAF): [13] Developed by Fischinger and Vincze and later adapted by Calli et al., Height Accumulated Features (HAF) generate a grid-based representation of the scene. This representation accumulates the maximum height within each grid cell, thereby creating a feature map that effectively captures the vertical structure of objects and the spatial relationship of their surrounding areas. This technique is particularly beneficial for scenarios where the vertical profile of objects dictates their graspability.

2) Viewpoint Feature Histogram (VFH): [3] The Viewpoint Feature Histogram (VFH) focuses on capturing the viewpoint orientation of the object relative to the sensor. This histogram-based descriptor not only encodes the shape of the object but also its orientation, making it highly useful for both object recognition and precise pose estimation tasks in cluttered or densely populated environments.

3) Clustered Viewpoint Feature Histogram (CVFH): [4] Building on the principles of VFH, the Clustered Viewpoint Feature Histogram (CVFH) extends its capabilities by incorporating the local geometry of object points and their spatial layout relative to a specific viewpoint. This method offers a descriptor that is sensitive to the object's orientation and can effectively be clustered to recognize and differentiate similar shapes in varying conditions.

4) Global Aligned Spatial Distribution (GASD): [14] The Global Aligned Spatial Distribution (GASD) technique provides a spatially-aligned, comprehensive description of the entire point cloud. By capturing the global distribution of points, GASD enhances robustness against varying object

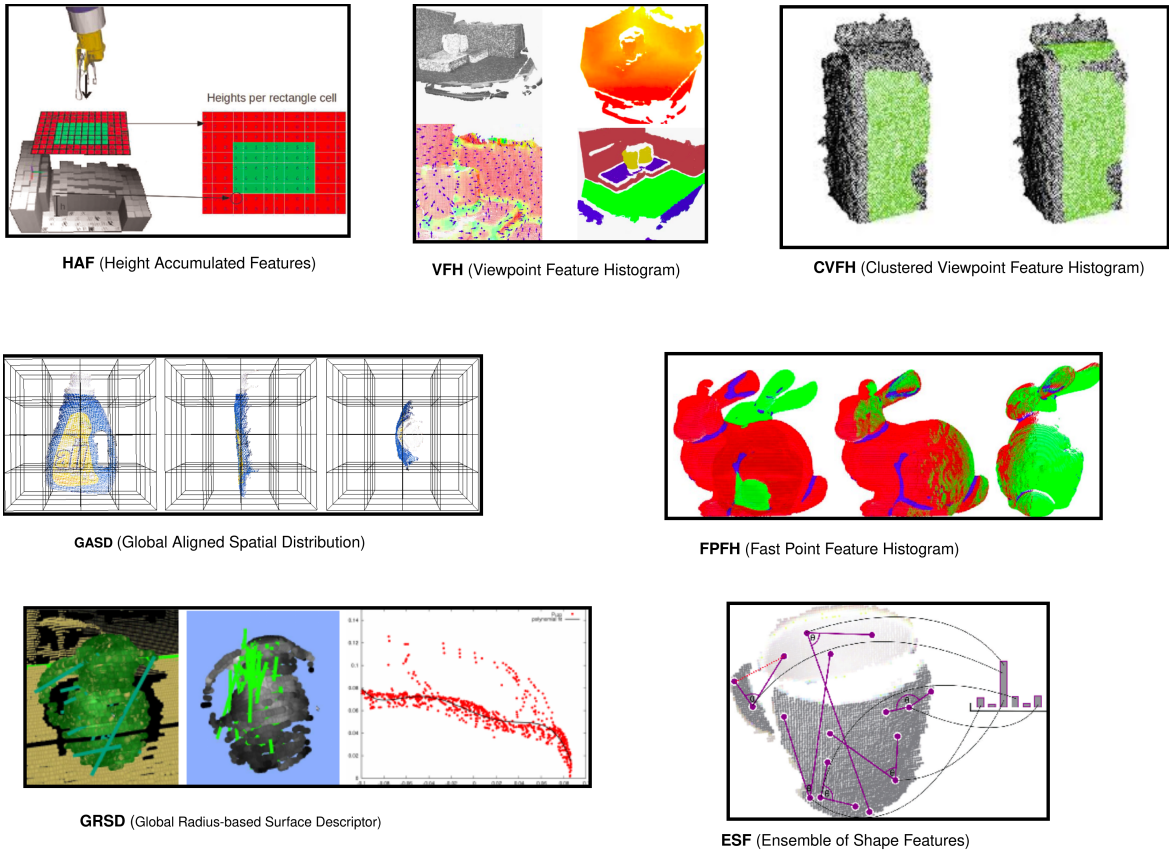


Fig. 4: Overview of Feature Descriptors

orientations and can be particularly effective in environments where objects of interest present diverse rotational forms.

5) Global Radius-based Surface Descriptor (GRSD): [15] The Global Radius-based Surface Descriptor (GRSD) quantifies the distribution of surface normals within a global context. This descriptor is adept at discerning between different object textures and geometries, proving invaluable for distinguishing between objects that have subtle physical differences but significant textural or geometric variations.

6) Ensemble of Shape Functions (ESF): [16] The Ensemble of Shape Functions (ESF) aggregates multiple shape functions to globally describe the point cloud. This approach offers a comprehensive understanding of an object's overall shape, independent of its size and orientation, making it suitable for applications requiring a broad overview of object forms without detailed geometric fidelity.

7) Fast Point Feature Histograms (FPFH): [7] Fast Point Feature Histograms (FPFH) focus on computing a local histogram of features based on the relative orientations of surface normals. Despite its rapid computation, FPFH provides a detailed and descriptive representation of local geometry, suitable for tasks where speed and efficiency are critical, such as in real-time grasping applications.

Each of these techniques plays a pivotal role in enhancing the capability of robotic arms to perform precise and reliable grasps, tailored to the specific demands of varied application environments.

IV. METHODOLOGY

Figure 7 illustrates the overall workflow of this project. GsubsectionData Acquisition and Preprocessing The initial stage in our pipeline is the acquisition of a 3D RGB point cloud. This rich dataset captures the full spectrum of the scene, containing both the target objects and the surrounding space yet to be explored. The point cloud is processed to segregate the two key components: the object point cloud and the unexplored point cloud. This segmentation is critical, as it distinguishes between the already scanned areas and those that require further examination.

A. Point Cloud Processing Overview

The initial handling of point cloud data involves downsampling the data captured by the camera, which serves to reduce sensor noise and accelerate subsequent processing phases. This preparatory step is depicted in Figure 6, showcasing how the environment appears to the camera after the data has been simplified. Identifying unexplored areas around the object is crucial for effective grasp synthesis and to guide active vision strategies. To achieve this, the vicinity of the object is populated with an evenly spaced point cloud. Each point within this cloud is then sequentially assessed to determine

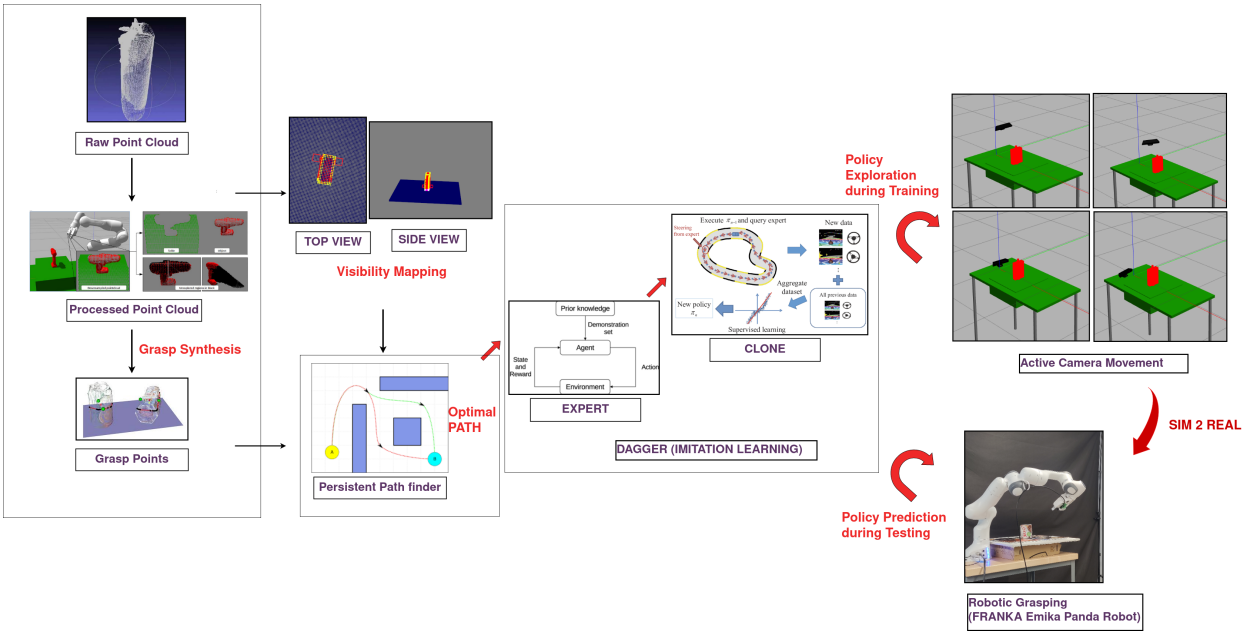


Fig. 5: Expert Demonstration

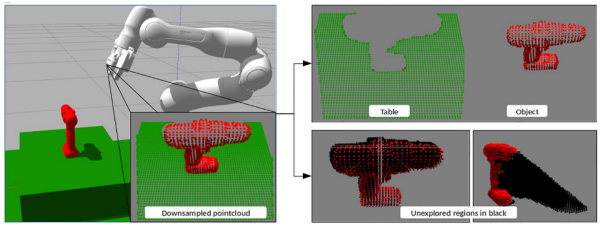


Fig. 6: Point Cloud Processing [5]

if it is occluded, avoiding the computationally intensive ray-tracing method by leveraging the structured nature of the point cloud.

Occlusion detection utilizes a projection technique where 3D points are mapped onto the image plane using the transformation:

$$X_p = \frac{KX}{z_0}$$

where X_p represents the projected pixel coordinates, $X = (x_0, y_0, z_0)^T$ is a 3D point, and K is the intrinsic camera matrix expressed as:

$$K = \begin{bmatrix} fx & 0 & ppx \\ 0 & fy & ppy \\ 0 & 0 & 1 \end{bmatrix}$$

This process marks points as occluded if their depth, z_0, exceeds the depth at X_p. The visual representation of these unexplored and occluded areas is dynamically updated with every new camera viewpoint, continually integrating new point cloud data with existing environmental information to refine the mapping of object data and unexplored regions.

B. Feature Extraction and State Vector Representation

Once the object and unexplored regions are defined, we proceed to the feature extraction phase. Utilizing a suite of seven feature extraction techniques, we decode the point clouds into a structured state vector representation. These feature vectors articulate the spatial and geometrical properties of the environment, translating complex 3D information into a format that is conducive to algorithmic interpretation.

Additionally, the state vector encapsulates the current camera position, which provides context regarding the sensor's perspective relative to the objects and unexplored areas. By integrating the camera position with the feature vector data, we form a comprehensive state vector that embodies all the critical information required for subsequent decision-making processes.

In our framework for facilitating data-driven policies in robotic grasping, we employ a uniform state vector as the foundational input. This state vector encapsulates the essence of the object's three-dimensional spatial information. To construct such a vector, a critical step is the selection and application of an appropriate feature extraction technique that can effectively condense the raw point cloud data.

One exemplary method is the Height Accumulated Features (HAF), which exemplifies the process of converting point clouds into a structured and informative representation. Originally introduced by Fischinger and Vincze [2]and later applied in the work of Calli et al. [17], the HAF approach involves segmenting the point cloud into a grid and recording the maximum height within each cell. This results in a feature map that distinctively portrays the object's verticality, an attribute often crucial in determining the feasibility of a grasp.

Our experimental evaluations explored the implications of

varying the grid resolution, with both 5x5 and 7x7 configurations tested for their impact on the learning algorithm's performance. Despite both configurations exhibiting comparable results, the 5x5 grid was chosen, favoring computational efficiency without significant loss of performance. The state vector thus consists of the flattened HAF-derived height maps combined with the polar and azimuthal coordinates of the camera, yielding a vector dimension of $2n^2 + 2$, where n is the grid size.

While HAF serves as a prime example in our study, the methodology applies broadly to other feature extraction techniques such as VFH, CVFH, GASD, GRSD, ESF, and FPFH. Each possesses unique attributes that render them more or less suitable for specific robotic grasping tasks. The process of feature selection and vectorization is pivotal, as it translates the complexity of the object's physical characteristics into a form that is amenable to algorithmic analysis and learning, thus influencing the system's grasp synthesis capabilities.

C. Imitation Learning and Training

The core of our training mechanism is based on imitation learning, specifically utilizing the DAgger (Dataset Aggregation) algorithm. The system is fed the state vector representation, and through the imitation learning framework, it learns to mimic expert behavior. The training is driven by a reward model that seeks to optimize cumulative rewards over time. This model favors actions that align closely with the epsilon-optimal demonstrations provided by human experts, which have been pre-established as nearly optimal paths or solutions.

D. Testing and Grasp Synthesis

In the testing phase, a pre-trained clone predicts the next optimal camera movement. This prediction aims to enhance the robot's understanding of the scene, leading to more informed decision-making in subsequent actions.

Simultaneously, a grasp synthesis algorithm is employed. This algorithm processes the current state vector to generate viable grasp candidates. The algorithm's effectiveness hinges on its ability to translate the learned imitation model and the intricacies of the feature representations into physical actions that the robotic system can perform. Grasp synthesis

V. EXPERIMENTAL SETUP AND VALIDATION

A. Dataset and Object Selection

The experimental validation of our robotic grasping algorithm was conducted using a carefully selected subset from the Yale-CMU-Berkeley (YCB) Object and Model Set. This dataset is a standardized collection widely recognized for its diversity and relevance in robotic manipulation benchmarks. It includes objects of daily life that vary in shape, size, texture, weight, and rigidity, providing a robust foundation for testing under realistic conditions.

For this study, we chose 11 items from the YCB dataset to cover a broad spectrum of shapes and symmetry of the objects by doing a comprehensive analysis of these objects in a Fold wise we test and train on different subsets of objects for the same feature to understand the implications.

B. Training Procedure

The training of our neural network employed the Imitation Learning approach using the Stable Baselines' DAgger (Dataset Aggregation) function [12]. This method involves training the network to imitate epsilon-optimal paths, which are near-optimal solutions derived from our predefined criteria. The training continued until the network achieved convergence, indicating that it had effectively learned the desired grasping behaviors.

C. Simulation Testing and Cross-Validation

Following training, the network's performance was evaluated in a simulated environment that mimics real-world conditions. This testing phase is crucial for assessing the robustness and reliability of the learned grasping strategies before any real-world deployment.

To comprehensively evaluate the effectiveness of the trained network, we conducted a complete cross-validation on the 11 selected objects. This involved testing each object multiple times under varying conditions to ensure that the network's performance was not only effective but also consistent across different instances. The cross-validation process focused on different feature types of the objects, allowing us to assess the impact of shape, size, texture, and other characteristics on the performance of the grasping algorithm.

This methodical approach to training and validation ensures that our findings are both robust and replaceable, providing valuable insights into the capabilities and limitations of the current state-of-the-art robotic grasping technologies.

VI. RESULTS AND DISCUSSION

A. Training and Testing Configurations

The experimental setup involved four-folds, each with a distinct combination of training and testing objects, as delineated in Table 1. The diversity of objects aims to assess the adaptability and robustness of different feature descriptors in a machine-learning context.

B. Computational Performance

An analysis of the computational performance revealed significant disparities in training and evaluation times across different features. In Folds 1 and 3, where a total of 8000 training steps were employed, certain features demonstrated efficiency in both learning and computation. Conversely, Fold 2, with 1500 steps, saw increased times, highlighting a critical balance between the number of steps and computational demand. In fold 4, we use a in-between number of training steps of 5500.

C. Feature Analysis

Across all folds, the plots demonstrate varying efficiency and efficacy levels for the seven features in question. Features like HAF and FPFH showed a steep learning curve in the plots, indicating a rapid attainment of a high success rate, which corresponds to their lower computational times as observed in the tabulated results. In contrast, features such as CVFH and

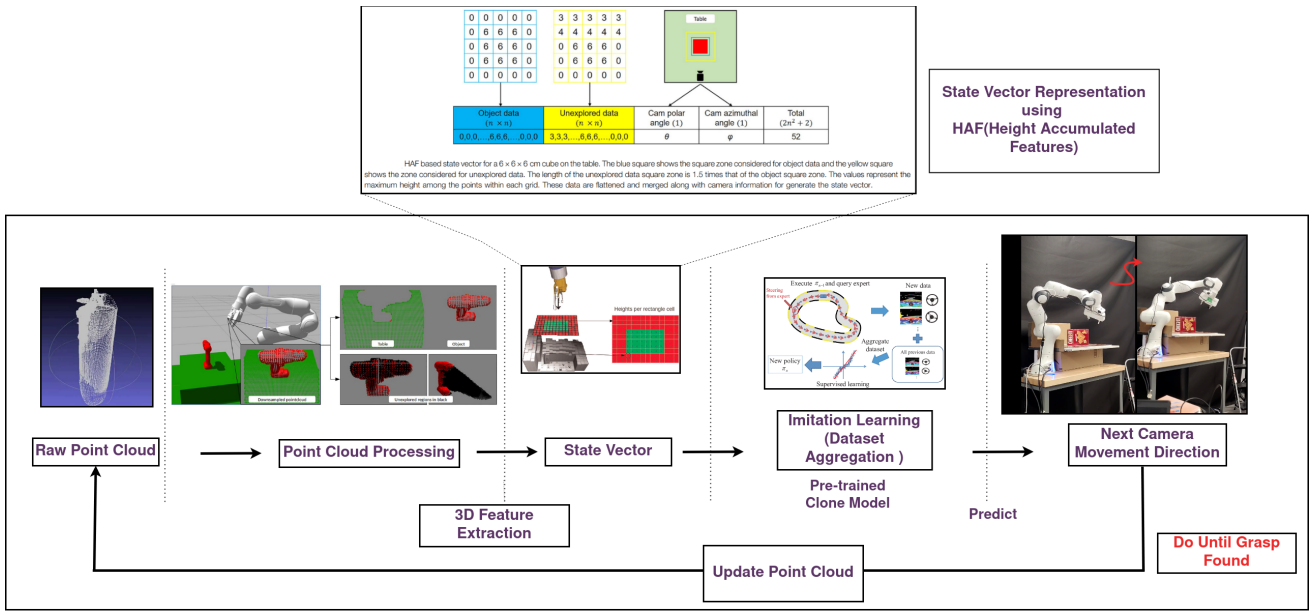


Fig. 7: Next Camera Movement Prediction workflow

Fig. 8: YCB(Yale-CMU-Berkeley Dataset) [12]

GASD required more iterations to reach similar success rates, as reflected in their longer computational times.

D. Consistency Across Folds

When considering the consistency across folds, features that consistently performed well in both the plots and the tabulated data were indicative of a stable and reliable performance. For instance, FPFH not only showed a quick convergence in the plots but also had the fastest evaluation pre-train times in Fold 1, suggesting a high degree of computational efficiency. Also, we observe that HAF was consistently performing better on training and in FOLD4 it is seen that HAF is performing well in testing unknown objects too.

E. Comparative Evaluation

The comparative evaluation of features indicated that the feature descriptors' performance is highly context-dependent. However, taking into account both the learning curves from the plots and the computational times from the tables, FPFH stands out as the feature with a superior balance of high success rate and computational efficiency across all three folds.

Fold	Training Objects and IDs	Testing Objects and IDs
1	009 gelatin box (8) 055 baseball (41) 072-a toy airplane (51) 010 potted meat can (9) 003 cracker box (2) 035 power drill (28) 006 mustard bottle (5) 021 bleach cleanser (19) 013 apple (12) Weisshai Great White Shark (65)	004 sugar box (3) 005 tomato soup can (4)
2	055 baseball (41) 072-a toy airplane (51) 010 potted meat can (9) 003 cracker box (2) 005 tomato soup can (4) 006 mustard bottle (5) 021 bleach cleanser (19)	009 gelatin box (8) 035 power drill (28) 013 apple (12)
3	009 gelatin box (8) 055 baseball (41) 010 potted meat can (9) 003 cracker box (2) 035 power drill (28) 005 tomato soup can (4) 006 mustard bottle (5) 021 bleach cleanser (19) 013 apple (12)	072-a toy airplane (51) Weisshai Great White Shark (65)
4	009 gelatin box (8) 072-a toy airplane (51) 010 potted meat can (9) 003 cracker box (2) 035 power drill (28) 005 tomato soup can (4) 006 mustard bottle (5) 013 apple (12) Weisshai Great White Shark (65)	055 baseball (41) 021 bleach cleanser (19)

TABLE I: Training and testing objects across different folds.

F. Failure Cases Analysis

Despite the overall successful outcomes highlighted in the previous sections, it is crucial to address and analyze the instances of failure as depicted in the plots. These cases offer invaluable insights into the limitations of the current feature set and the challenges faced during the project's execution.

In Fold 1, while the success rate for most features rapidly increased, certain features like GRSD plateaued early, indicating a potential mismatch between feature descriptors and the training objects. Similarly, in Fold 2, the slower increase in success rates for features such as CVFH could be attributed to the reduced number of training steps, which may not have been sufficient for the feature to capture the complexity of the objects effectively.

Fold 3's failure cases, while fewer in comparison to Fold 2, still present critical learning opportunities. For instance, the consistent underperformance of GASD, even with an increased number of training steps, prompts further investigation into the adequacy of the feature's design for the types of objects in the dataset.

Moreover, the failure cases across all folds suggest that while some features have a high success rate for specific objects, they may not generalize well across different object classes. This particular insight underscores the necessity for a diverse and comprehensive training set to build a robust feature set for real-world applications.

Our project's exploration into object recognition and manipulation revealed critical limitations in the features tested. Despite varying training steps across Folds 1 to 4, certain test objects consistently evaded a successful grasp by all features. This consistent lack of success suggests a fundamental mismatch between the current feature set and specific object properties, underscoring the need for more sophisticated feature engineering. The persistent failures across different training conditions highlight the importance of developing a more robust and versatile feature set capable of handling a diverse range of objects. This study's failure cases prompt further research into advanced feature descriptors to address these gaps and improve grasp success in robotic applications.

In the evaluation of computational efficiency for imitation learning in robotic grasping simulations, different GPUs exhibit significant variance in training times for a fixed number of steps. Specifically, the NVIDIA RTX 3070 Ti processes 8000 training steps in approximately 8 hours, resulting in a calculated time of about 0.5 hours for 500 steps. Conversely, the NVIDIA GTX 1650, a less powerful GPU, takes a full 5 hours to complete just 500 steps, indicating its lower efficiency for such computationally intensive tasks. Meanwhile, the NVIDIA RTX 4070, despite its advanced architecture, requires 16 hours to complete 8000 steps, leading to a training time of 1 hour for 500 steps. These differences underscore the importance of selecting appropriate hardware based on the specific demands and scale of training in robotic simulations to optimize both time and resource utilization.

GPU Model	Maximum Training Steps	Training Time for 500 Steps
RTX 3070 Ti	9000	0.5 hours
GTX 1650	2500	5 hours
RTX 4070	9000	1 hour

Fig. 9: Comparison of Imitation Learning training time on GPU Hardware

VII. Conclusion

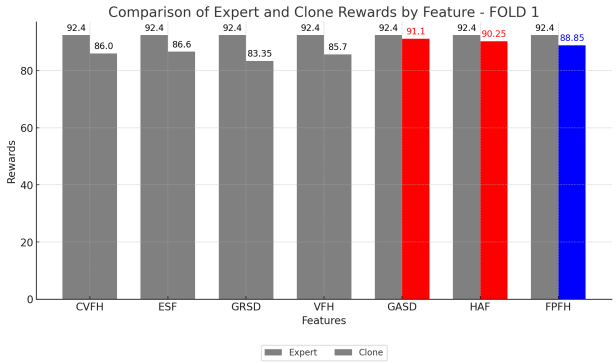
Considering the overall performance in terms of learning efficiency, computational speed, and consistency across different training/testing conditions, FPFH is concluded to be the most efficient feature among the seven evaluated with HAF being the second best. FPFH's rapid learning curve and quick evaluation times underscore its potential for applications where both performance and computational resources are of paramount concern. HAF was suited for applications when the camera is located on top of the object, in applications like Robotics grasping.

A. Future Work

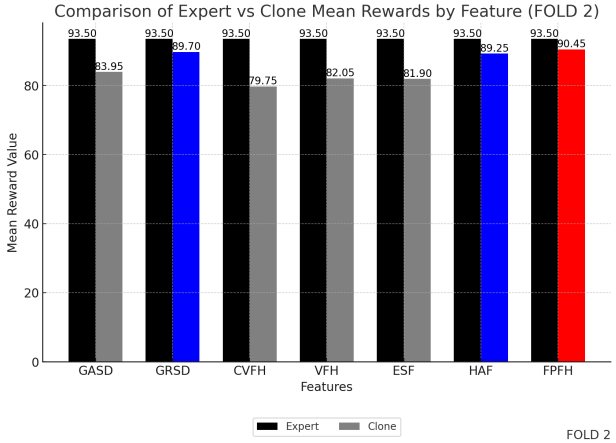
Future investigations should further explore the scalability of FPFH and HAF features across a broader range of objects and conditions. Additionally, studies could focus on optimizing training strategies to further reduce computational times without compromising the performance quality.

Acknowledgment

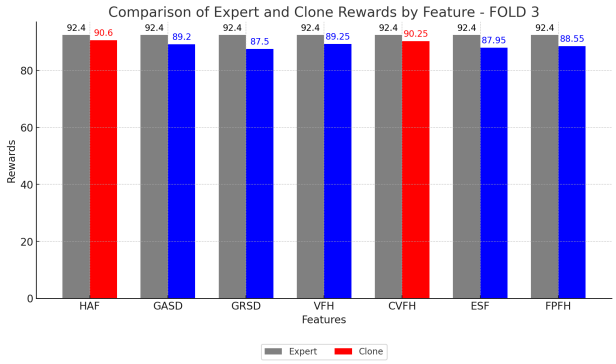
I extend my heartfelt thanks to Prof. Berk Calli for his invaluable contribution to our discussion on robotics and artificial intelligence. My appreciation also goes to my project mate, Galen Brown, whose dedication and assistance were instrumental in my navigation through the various stages of the project. I am thankful for the chance to use the Franka Emika Panda Arm equipped with the Intel RealSense D430I camera, which was crucial for my work. Additionally, I am indebted to the Robotics Engineering department, as well as my lab peers and friends, who provided indispensable support from the initial setup to the execution of my project.



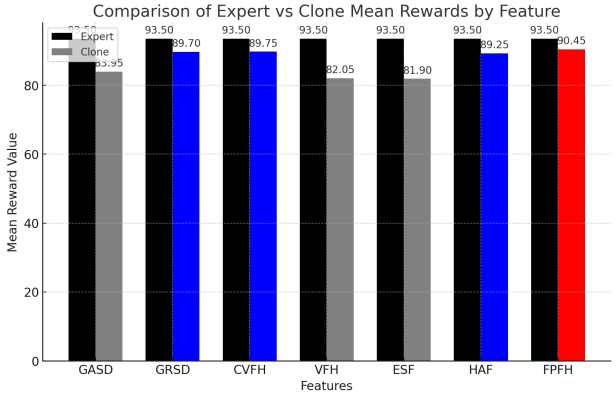
(a) FOLD 1



(b) FOLD 2



(c) FOLD 3



(d) FOLD 4

Fig. 10: Expert Vs Clone Cumulative Reward Comparison in Training

REFERENCES

[1] R. Zurbrügg, Y. Liu, F. Engelmann, S. Kumar, M. Hutter, V. Patil, and F. Yu, "Icgnet: A unified approach for instance-centric grasping," *Journal Name Here*, vol. xx, no. xx, pp. xx–xx, 2024.

[2] D. Fischinger and M. Vincze, "Empty the basket - A shape based learning approach for grasping piles of unknown objects," in *IEEE International Conference on Intelligent Robots and Systems*, pp. 2051–2057, 2012.

[3] R. B. Rusu *et al.*, "Fast 3d recognition and pose using the viewpoint feature histogram," in *Proceedings of the Conference on 3D Vision*, IEEE, 2010.

[4] A. Aldoma, F. Tombari, R. Rusu, and M. Vincze, "Our-cvfh - oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6dof pose estimation," in *Proceedings of the International Conference on 3D Vision*, IEEE, 2012.

[5] S. Natarajan, G. Brown, and B. Calli, "Grasp synthesis for novel objects using heuristic-based and data-driven active vision methods," in *Proceedings of the Conference on Robotics and Automation*, IEEE, 2021.

[6] Z. Marton, D. Pangercic, N. Blodow, J. Kleinehellefort, and M. Beetz, "General 3d modelling of novel objects from a single view," 2010.

[7] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Proceedings of the International Conference on Robotics and Automation*, IEEE, 2009.

[8] J. Daudelin and M. Campbell, "An Adaptable, Probabilistic, Next-Best View Algorithm for Reconstruction of Unknown 3-D Objects," *IEEE Robotics and Automation Letters*, vol. 2, pp. 1540–1547, jul 2017.

[9] S. Ross, G. J. Gordon, and J. A. Bagnell, "No-regret reductions for imitation learning and structured prediction," *CoRR*, vol. abs/1011.0686, 2010.

[10] I. Sebag, S. Cohen, and M. P. Deisenroth, "On combining expert demonstrations in imitation learning via optimal transport," 2023.

[11] P. Florence, C. Lynch, A. Zeng, O. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," 2021.

[12] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," 2011.

[13] D. Fischinger, A. Weiss, and M. Vincze, "Learning grasps with topographic features," in *Proceedings of the International Conference on Robotics and Automation*, IEEE, 2015.

[14] "Gasd estimation." Accessed: 2024-04-27.

[15] Z. Marton, D. Pangercic, N. Blodow, J. Kleinehellefort, and M. Beetz, "General 3d modelling of novel objects from a single view," in *Proceedings of the Conference on 3D Vision*, IEEE, 2010.

[16] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3d object classification," in *Proceedings of the International Conference on 3D Vision*, IEEE, 2012.

[17] B. Calli, W. Caarls, M. Wisse, and P. Jonker, "Viewpoint optimization for aiding grasp synthesis algorithms using reinforcement learning," *Advanced Robotics*, vol. 32, no. 20, pp. 1077–1089, 2018.

FOLD1

Feature	Total Train Steps	Base (Mean, Std Dev.)	Expert (Mean, Std Dev.)	Clone (Mean, Std Dev.)	Time to train last 500 steps (n-1 th iteration) in secs	Time to Evaluate Pretrain in secs
CVFH	8000	69.70, 12.83	92.40, 4.58	86.00, 10.18	764.55	273.04
ESF	8000	69.70, 12.83	92.40, 4.58	86.60, 8.88	529.08	229.48
GRSD	8000	69.70, 12.83	92.40, 4.58	83.35, 9.73	547.67	224.93
VFH	8000	69.70, 12.83	92.40, 4.58	85.70, 10.79	546.06	202.01
GASD	8000	69.70, 12.83	92.40, 4.58	91.10, 5.17	503.95	146.90
HAF	8000	69.70, 12.83	92.40, 4.58	90.25, 6.34	505.34	153.35
FPFH	8000	69.70, 12.83	92.40, 4.58	88.85, 9.73	741.84	771.33

(a) FOLD 1

FOLD2

Feature	Total Train Steps	Base (Mean, Std Dev)	Expert (Mean, Std Dev)	Clone (Mean, Std Dev)	Time to train last 500 steps (n-1 th iteration) in secs	Time to Evaluate Pretrain in secs
GASD	1500	(68.15, 12.09)	(93.50, 2.20)	(83.95, 7.76)	6097.43	3824.68
GRSD	1500	(68.15, 12.09)	(93.50, 2.20)	(89.70, 11.09)	19566.95	21918.48
CVFH	1500	(68.15, 12.09)	(93.50, 2.20)	(79.75, 10.17)	8230.38	6693.06
VFH	1500	(68.15, 12.09)	(93.50, 2.20)	(82.05, 7.67)	7079.97	6337.00
ESF	1500	(68.15, 12.09)	(93.50, 2.20)	(81.90, 9.67)	6634.51	5949.13
HAF	1500	(68.15, 12.09)	(93.50, 2.20)	(89.25, 9.20)	3485.65	3942.40
FPFH	1500	(68.15, 12.09)	(93.50, 2.20)	(90.45, 4.72)	4166.38	3618.02

(b) FOLD 2

FOLD3

Feature	Total Train Steps	Base (Mean, Std Dev)	Expert (Mean, Std Dev)	Clone (Mean, Std Dev)	Time to Train Last 500 Steps (n-1 th iteration) in secs	Time to Evaluate Pretrain in secs
HAF	8000	(69.70, 12.83)	(92.40, 4.58)	(90.60, 8.31)	1483.48	644.27
GASD	8000	(69.70, 12.83)	(92.40, 4.58)	(89.20, 5.37)	1391.02	770.79
GRSD	8000	(69.70, 12.83)	(92.40, 4.58)	(87.50, 4.83)	1724.03	1102.40
VFH	8000	(69.70, 12.83)	(92.40, 4.58)	(89.25, 7.67)	1060.89	549.97
CVFH	8000	(69.70, 12.83)	(92.40, 4.58)	(90.25, 4.75)	1745.03	971.92
ESF	8000	(69.70, 12.83)	(92.40, 4.58)	(87.95, 7.70)	1491.67	885.52
FPFH	8000	(69.70, 12.83)	(92.40, 4.58)	(88.55, 4.88)	1718.91	906.31

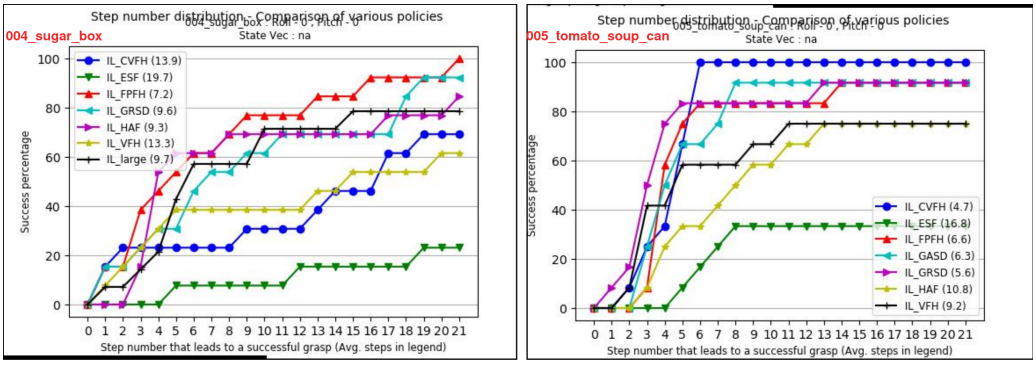
(c) FOLD 3

FOLD 4

Iteration	Feature	Base (Mean, Std Dev)	Expert (Mean, Std Dev)	Clone (Mean, Std Dev)	Time to train last 500 steps (s)	Time to Evaluate Pretrain (s)
5500	CVFH	71.50, 14.01	93.20, 3.04	89.30, 8.72	4164.77	1476.56
5500	ESF	71.50, 14.01	93.20, 3.04	90.45, 4.44	4079.29	1290.87
5500	GRSD	71.50, 14.01	93.20, 3.04	89.55, 4.70	3978.12	1314.63
5500	VFH	71.50, 14.01	93.20, 3.04	87.50, 9.63	3773.80	1695.45
5500	GASD	71.50, 14.01	93.20, 3.04	90.00, 7.78	5277.05	2427.22
5500	HAF	71.50, 14.01	93.20, 3.04	92.90, 4.48	2572.35	688.87
5500	FPFH	71.50, 14.01	93.20, 3.04	86.65, 10.80	6157.30	2104.35

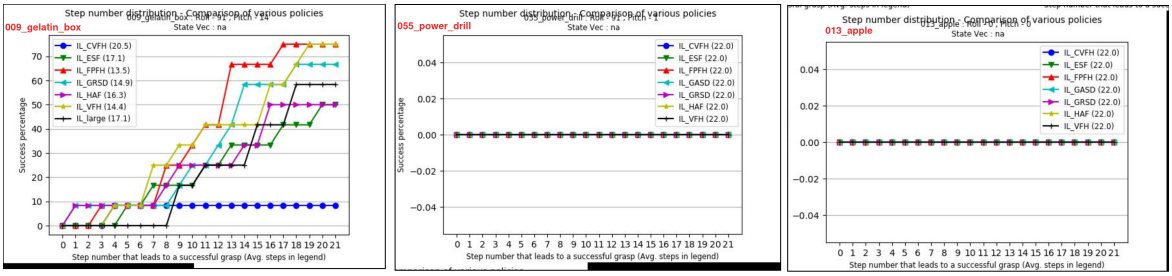
(d) FOLD 4

Fig. 11: Random, Expert, Clone Cumulative Reward and standard deviations and pre-train evaluation



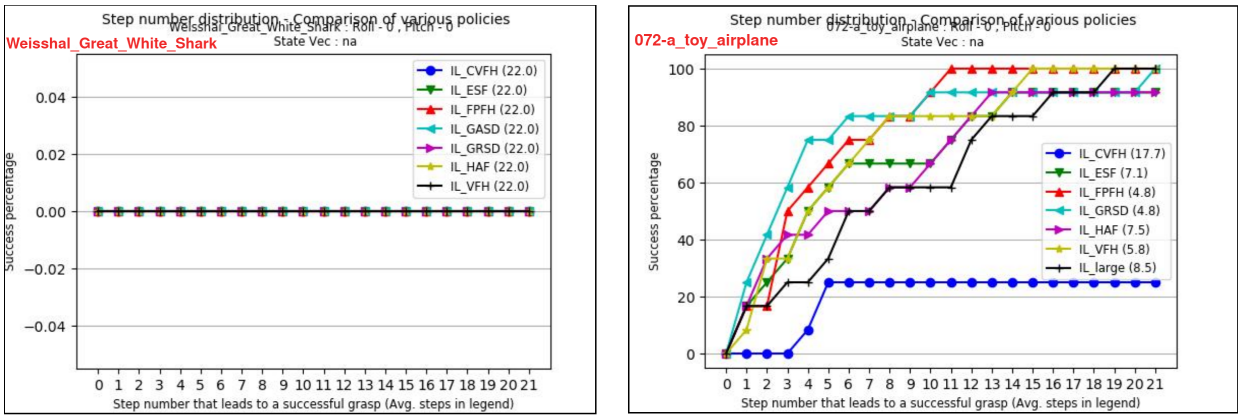
FOLD1

Fig. 12: Fold 1 Test results



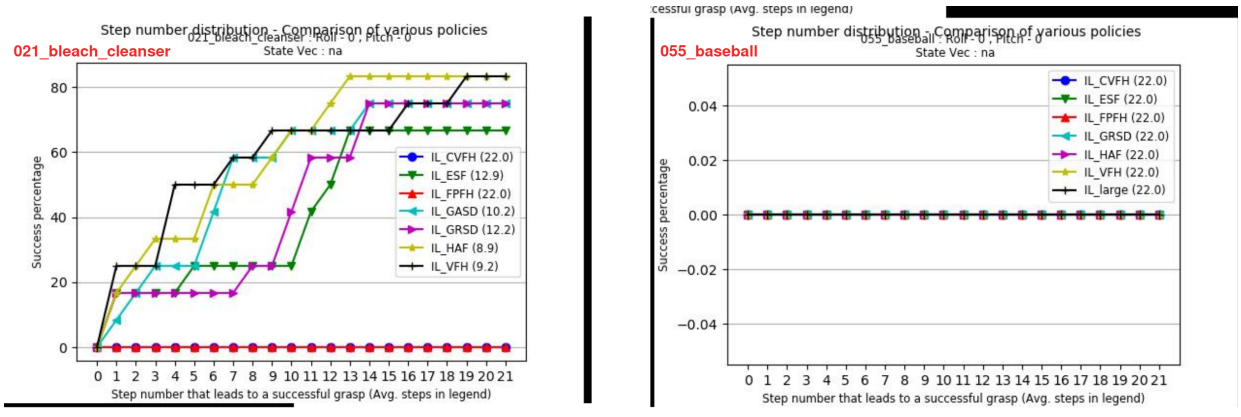
FOLD2

Fig. 13: Fold 2 Test results



FOLD3

Fig. 14: Fold 3 Test Results



FOLD 4

Fig. 15: Fold 4 Test Results