Towards Generalized Context-Aware Navigation for Mobile Robots in Human-Centered Environments

by

Siddarth Narasimhan

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science

Mechanical and Industrial Engineering
University of Toronto

Towards Generalized Context-Aware Navigation for Mobile Robots in Human-Centered Environments

Siddarth Narasimhan

Master of Applied Science

Mechanical and Industrial Engineering
University of Toronto

2025

Abstract

Mobile robots are increasingly deployed in human spaces such as healthcare settings, office spaces, and households, with the objective of navigating safely to their destinations to perform tasks such as delivery or search. Effective navigation requires robots to adapt to diverse forms of context, such as interactions between people in the scene or semantic relationships between objects. This thesis presents two main contributions in context-aware robot navigation: 1) a social navigation algorithm that leverages foundation models to interpret social context and navigate among people while adhering to social norms, and 2) an instance image-goal navigation architecture that uses a Gaussian Splatting representation and semantic context of the environment to guide robots towards image goals. These works demonstrate how context-aware reasoning can enable robots to navigate both among people and to people. Extensive experiments validate the effectiveness of our approaches, advancing the literature in context-aware robot navigation.

Acknowledgments

It is quite unfortunate that I am forced to write only one name as the author of this thesis. In reality, this work was a collaborative effort from working alongside advisors, colleagues, friends and family. This thesis would not have been possible without the support, guidance and comradery from many people, whom are the reason I'm writing this paper today.

I would firstly like to express my sincere gratitude to Professor Goldie Nejat for giving me the opportunity to work at ASBLab as a bachelor's thesis student and now a graduating master's student. The most valuable skill that I've learned from you is how to write. While reading research papers from even the most prestigious universities, I cannot help but wanting to dissect their sentences and commenting "TOO VAGUE", which is a habit I owe entirely to you. You have made me into a better researcher by challenging me to work harder and aim higher. I cannot thank you enough for the countless hours and late nights you spent working towards helping me reach publication deadlines. You really pushed the boundaries of what was possible during my thesis. I extend my thanks to my committee member, Prof. Eric Diller, who's feedback has been very helpful.

I am very fortunate to have met Aaron Tan at the very beginning of my bachelor's thesis. I would not have decided to go into research if it wasn't for your early mentorship and encouragement. You have been by my side, supporting me throughout my research during its ups and downs. Our Zoom calls were some of the most productive meetings I've ever had with anyone. I am also thankful for the immeasurable help you've given to improve my profile and support my applications. What I carry with me the most is the lasting imprint of your life's guiding philosophy; mamba mentality.

Haitong Wang, you have grown from being an academic senior to a great friend over the years. I never felt concerned when faced with what seemed like an impossible technical problem, because I knew I could turn to you for guidance. You are also one of the few people who I've been humbled by in table tennis. The thing I admire most about you is your patience and calmness (like the sea), which always made you an enjoyable person to be around.

Angus Fung and Daniel Choi, thank you for being a constant form of support throughout my thesis. You guys have always had a strong drive and ambition to have a lasting impact on the world,

which I've always admired. I've also been fortunate to get a glimpse of your alter egos, Nψ Angus and Hamniel, whose relentless work ethics have motivated me to work harder.

Jerry Cheng and Souren Pashangpour, I will never forget the many 3-hour podcasts we've had debating deep tech, culture, politics, and religion. You guys have given me the confidence to unforgivingly express my sincerely-held beliefs, while also offering perspectives on life that have greatly shaped my worldview.

Matthew Lisondra, I can't think of a better person to talk to during stressful times. Your unique sense of humor would always bring levity to any situation. Thank you for the encouragement and support in the last year. I hope that you will someday 1) become the director of NASA and Google Deepmind to live up to your honor, and 2) meet Elisabeth to shed some light on the past.

Thank you also to Dr. Ficocelli and Prof. Bensiyon Benhabib for being the most inspiring teachers, and a source of deep guidance, enlightenment and joy for my thesis.

I would also like to extend my thanks to many of my other colleagues, including Cristina Getson, Nan Liang and Yizhu Li for the many enjoyable conversations we've had throughout the years.

My family has always been a great support for me for my entire life. Thank you to my mom, dad, sister, uncle and brother-in-law for everything you've done for me.

Finally, this thesis would not have been possible without God. His grace has been the cornerstone of my success in ways I cannot fully express in words.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Motivation

Mobile robots operating in human-centered environments must have the ability to navigate safely and efficiently to their destinations in order to complete useful tasks [1]-[9]. In healthcare settings, robots are used to search for and deliver medical equipment through crowded hallways [1], or locate specific patients for activity monitoring [2], [3]. In office spaces and households, robots must search for necessary objects and tools around the space in order to complete tasks like floor cleaning [4], meal preparation [5], or laundry [6]. In public or commercial spaces, robots can be used for search an assistance tasks, such as locating missing people in shopping malls and airports [7], [8], or guiding visitors to specific locations [9]. Overall, these robots must be able to *generalize* to any situation or scenario to complete their tasks – namely, the robots must be able to complete the tasks in any environment (indoor vs outdoor), among many demographics of people (ethnicity, age), and with varying types of objects (household vs outdoor objects).

In all of the above scenarios, robots must interpret the surrounding context provided to them in order to make effective navigation decisions [10]. For example, when navigating towards a goal location in the presence of dynamic humans, a robot must reason about the social context of the scene to move safely and in accordance to social norms [11]. This capability, known as robot social navigation, refers to the ability of an autonomous robot to move towards a goal while adhering to socially acceptable behaviors [12]. This involves understanding interactions between people, such as avoiding interruption of conversational groups or yielding to groups moving together [13]. It must also adapt its behavior according to social contexts in the scene such as the type of environment, such as whether it is a narrow hallway or an open space [14], and also the characteristics of the people involved, such as whether they are elderly or young, in order to balance safety and speed of the maneuver [3].

Similarly, mobile robots must also reason about visual and semantic context of the scene when locating specific objects or people [15]. A key navigation task in this domain is Instance Image Goal Navigation (IIN), where the robot is provided a reference image of the goal object or person from any arbitrary viewpoint and must use contextual information from current and prior

observations in order to recognize potential matches [16]. For instance, if a robot is provided a goal image of a bed, it can use visual context of the bed's background to distinguish the correct instance from other similar beds, while also leveraging the semantic context of the environment to guide navigation (e.g. recognizing that a hallway is more likely to lead to a bedroom, and therefore prioritizing exploration in that direction). Without context-aware navigation capabilities, robots would risk unsafe interactions and inefficient behaviors when completing their given tasks [17]-[20].

1.2 Research Challenges

This thesis will address the primary challenges of context-aware robot navigation models in two key domains, robot social navigation and instance image goal navigation. These challenges are described in each of the subsections below.

1.2.1 Robot Social Navigation Challenges

Robot social navigation is a challenging problem as robots must react to human behavior and contexts in real time [11], while dealing with varying social conditions [21]. Existing robot social navigation approaches have used either human-model-based (HMB) [13], [22]-[28] or human-model-free (HMF) [14], [29]-[33] methods. In HMB methods, human trajectories are explicitly predicted and then incorporated into a navigation policy primarily using deep reinforcement learning (DRL) [13], [22]-[28]. HMF methods implicitly account for human trajectories by: 1) learning social navigation policies using imitation learning (IL) [29]-[32], or 2) leveraging social reasoning capabilities of large foundation models such as large language models (LLMs) [33], or VLMs [14]. However, HMB and HMF methods do not account for social or environment context, which are important for robot path planning [14]. Furthermore, they are unable to adapt to new social scenarios (unexpected human behaviors, changes in the environment), resulting in degraded performance in real-world deployment [11].

1.2.2 Instance Image Goal Navigation Challenges

In IIN, the following challenges need to be addressed: 1) a robot is not provided with a map of the environment prior to deployment, therefore, it must explore and map the environment using visual observations, while avoiding redundant coverage [34], 2) the viewpoint of the provided goal image may differ from the robot's viewpoint, increasing the difficulty of recognizing the goal object

during navigation [35], and 3) a robot must leverage visual and semantic context from earlier exploration to improve the efficiency of future navigation actions [36].

To-date, existing approaches to IIN have primarily used deep learning [16], [34], [37]-[48], and large foundation models [36], [49]-[52]. Deep learning methods have used convolutional encoders to extract features from visual observations and the goal image, capturing information such as scene geometry and visual appearance (e.g. color, texture, and object shape). These features are used to predict the robot's next action and are encoded into latent vectors processed by actor-critic models [34], [38]-[40], transformers [37], [41], recurrent neural networks [44], or inverse dynamics models [46]. Large foundation model approaches use pre-trained encoders from large vision transformers to convert the image observations and the goal image into semantic embeddings [49]-[52]. This enables the robot to query the environment for the goal image using language prompts. The semantic embeddings are then used by scene graphs [36], [51] or reinforcement learning architectures [49], [50], to direct the robot towards the goal. These existing methods assume that the goal image is captured from a viewpoint that is similar to the robot's viewpoint. For example, if the goal image is captured from a bird's-eye-view, the robot may fail to recognize it from its frontal view due to feature differences across the two viewpoints. As a result, their performance can degrade when provided with a goal image from an arbitrary viewpoint at the goal location [35].

To address this limitation, where goal images may be provided from perspectives unseen during exploration, recent research has explored the use of alternative scene representations that can render the environment from novel viewpoints. In particular, 3D Gaussian Splatting (3DGS) [53] enables photorealistic 3D scene reconstruction of an environment using collections of 3D Gaussian ellipsoids. 3DGS has been applied to robot navigation in dynamic human environments [54], [55], where they must navigate safely to a provided goal location [56], [57], as well as for path-planning in indoor environments such as supermarkets [58] and classrooms [59]. With respect to the IIN task, 3DGS has been able to assign semantic labels to the Gaussians, allowing a robot to localize a goal image within the map of an environment and then navigate to the goal using waypoint-based planning [35], [60], [61], or model predictive control [62]. However, these approaches all assume that a detailed 3DGS map is available prior to robot deployment, which can limit their usability in unknown environments where there is sparse-view or incomplete scene reconstruction [63].

1.3 Thesis Objective

The challenge of context-aware robot navigation lies in developing algorithms that can generalize across diverse environments, objects and interaction types. Specifically, for robot social navigation, this encompasses the ability to navigate in various indoor and outdoor spaces, and among various social contexts and interactions such as conversational groups, large moving crowds and people with diverse demographics (elderly vs young, ethnicity, etc.). For IIN, the broader challenge is to enable robots to identify and navigate to arbitrary goals objects in any indoor and outdoor environment. Addressing these challenges within these two domains of robot navigation remains an open challenge.

The objective of this thesis is to build two novel architectures centered around problems in context-aware navigation for indoor dynamic and human-centered environments. While the broader challenge of social navigation involves generalizing to diverse spaces, interactions and objects, the scope of this thesis is on indoor environments. Namely, this thesis will develop: 1) an Online Lifelong Vision Language architecture, OLiVia-Nav [64], for mobile robot social navigation which considers both social and environment context during robot trajectory planning and adapts to new social scenarios, and 2) a novel 3DGS-based architecture, SplatSearch, where a mobile robot searches for a static object or person using visual and semantic context, provided from a single reference image from an arbitrary viewpoint. These methods are incorporated on a mobile robot for social navigation and IIN in real-world, human-centered environments.

1.4 Organization

The work of this thesis is organized as follows:

Chapter 2 contains a detailed literature review on 1) robot social navigation approaches and 2) instance image goal navigation approaches, covering existing state-of-the-art methods in their respective navigation domains.

Chapter 3 presents a novel social navigation architecture, OLiVia-Nav, which considers both social and environment context during robot trajectory planning and adapts to new social scenarios. The main contributions are: 1) the development of a novel distillation process, Social Context Contrastive Language Image Pre-training (SC-CLIP), that transfers the social reasoning capabilities of large VLMs into two lightweight encoders. These encoders extract social context

embeddings from both visual and token semantic features from image and text captions, respectively, for predicting robot trajectories, and 2) the development of a trajectory planning network that uniquely utilizes multi-head attention to account for these social context embeddings during the generation of socially compliant robot trajectories. The encoders support online lifelong learning to adapt to new unseen social navigation scenarios during robot navigation.

Chapter 4 presents a novel 3DGS-based architecture for IIN, SplatSearch, where a mobile robot searches for a static object or person, provided a single reference image from an arbitrary viewpoint in unknown environment using sparse reconstructions. SplatSearch is the first architecture to incrementally build a sparse 3DGS representation of the environment to achieve novel viewpoints around objects, and use a multi-view diffusion to inpaint missing regions of these viewpoints for robust goal recognition. The key contributions of this work are: 1) The development of an IIN method using 3DGS that is the first to utilize a multi-view diffusion model to address viewpoint-invariant goal recognition using sparse reconstructions of an unknown environment, and 2) A novel frontier exploration strategy that uniquely combines visual context from the novel viewpoints and semantic context from prior observations to guide the robot towards the goal image.

Chapter 5 summarizes the key contributions in this thesis, highlighting the advancements in robot social navigation and IIN for generalized context-aware navigation, and offers recommendations for future improvements.

Chapter 2
Literature Review

This chapter provides a detailed summary of existing literature related to the two context-aware navigation domains. The literature review can be categorized as follows: 1) robot social navigation, and 2) Instance Image Goal Navigation for mobile robots.

2.1 Robot Social Navigation

Existing robot social navigation approaches can be categorized into: 1) human-model-based (HMB) methods [13], [22]-[28], 2) human-model-free (HMF) methods [14], [29]-[33], and 3) lifelong learning methods [65], [66].

2.1.1 Human-Model-Based (HMB) Methods

In general, HMB methods explicitly predict the trajectories of people in the robot's surrounding and use them to inform robot social navigation.

Human trajectories have been predicted using constant velocity (CV) models [26], transformer models [22], [24], [28], or human tracking (HT) models [13], [23], [25], using RGB images and LiDAR point clouds. In CV models, human trajectories are predicted by assuming a constant speed and direction [26]. In transformer models, a spatial-temporal graph transformer is used to encode distance relationships between people overtime to predict their positions [22], [24], [28]. Lastly, HT models utilize tracker methods such as YOLO [67] to detect and track human positions using either bounding boxes [23], [13], or LiDAR point clouds [25]. These models then predict trajectories by estimating velocity and direction from the tracked human positions.

The predicted trajectories are then used to generate social navigation policies by using DRL methods [22], [24]-[28], or heuristic control methods [23], [13]. Namely, DRL techniques include actor-critic [22], [25], proximal policy optimization (PPO) [24], [26], [27], and double deep Q networks [28], which optimize reward functions to maximize personal space and minimize collisions. Heuristic control policy methods consist of predefined rules used to generate policies to avoid human trajectories [23], [13].

The HMB methods are trained using 2D simulated environments, where humans are represented as point masses [22], [23], [25], [26], [28], or 3D simulated environments with procedurally generated human trajectories [24], [27].

2.1.2 Human-Model-Free (HMF) Methods

HMF methods consist of: 1) imitation learning (IL) methods [29]-[32] which learn social navigation policies from expert knowledge in datasets, or 2) large foundation model methods (e.g. LLMs/VLMs [33], [14]), that generate robot actions based on their social reasoning capabilities.

IL methods use behavior cloning [29], transformer architectures [30], generative adversarial IL [31], or inverse reinforcement learning [32], to predict robot trajectories. They have leveraged social navigation datasets like SCAND [68], MuSoHu [69], and THOR-Magni [70], which contain expert demonstrations of robot trajectories, RGB images, and LiDAR point clouds from real-world environments.

LLMs and VLMs exhibit social reasoning capabilities as they are pre-trained on internet scale data [71]. These methods generate social navigation policies in two stages. Firstly, LLMs and VLMs are prompted to generate social and environment context captions using audio [33] or image inputs [14]. Secondly, a navigation planner (DRL [33] or the dynamic window approach (DWA) [14]) uses these context captions to generate socially compliant navigation actions.

2.1.3 Lifelong Learning Methods

Lifelong learning methods incrementally update navigation model parameters to adapt to new social scenarios overtime, and have only been applied to HMB methods [65], [66]. For example, in [65], human trajectories were first tracked using a Kalman filter (KF) with visual and LiDAR data to estimate their positions and velocities. Lifelong learning was achieved by updating probability distribution maps with this tracking data, predicting human positions relative to the robot at discrete future time steps. In [66], a DRL architecture using PPO was trained with the THOR-Magni dataset [70] to predict human trajectories using gated recurrent units (GRUs). The predicted trajectories were used to quantify deviations from socially acceptable behaviors through a social cost function, which was incorporated into the DRL reward function. The lifelong learning process continuously updated the weights for the navigation policy by retraining the model with new human trajectory data collected during navigation.

2.1.4 Summary of Limitations

Existing HMB methods do not account for social and environment context found in real-world scenarios [13], [22]-[28]. Furthermore, lifelong learning methods that use HMB methods rely on predefined human motion models within a KF. These motion models do not incorporate visual cues from the environment or the behaviors of nearby people [65]. They also cannot be directly applied in real-world environments due to their training on simulation-based datasets, creating a sim-to-real gap [66].

HMF methods that use IL [29]-[32] can only handle social scenarios and the environment directly observed in their training datasets. This is a limitation as existing navigation datasets cannot comprehensively encompass every possible social scenario that may occur [68]. LLM and VLM methods [33], [14] are limited by their slow response times resulting in poor social navigation performance, as they cannot adapt to fast human movements leading to collisions [72].

To address the limitations, we propose OLiVia-Nav to: 1) account for social and environment context using SC-CLIP to generate social context embeddings. These embeddings encode the robot's surrounding environment, human behavior, and high-level navigation actions for socially compliant robot trajectory planning and selection; and 2) adapt to new social scenarios by incorporating online lifelong learning through SC-CLIP's lightweight encoders. These encoders are updated during navigation to continuously account for new unseen social scenarios.

2.2 Instance Image Goal Navigation

We categorized IIN methods into: 1) deep learning-based methods [16], [34], [37]-[48], 2) large foundation model methods [36], [49]-[52], and 3) gaussian splatting methods [35], [60]-[62].

2.2.1 Deep Learning-Based Methods

Deep learning-based methods for the IIN task have mainly used deep reinforcement learning (DRL) [16], [34], [37]-[41], [44], [46]-[48] or imitation learning (IL) [42], [43], [45].

DRL-based methods used convolutional encoders such as ResNet to generate joint embeddigns from RGB image observations and the goal image to inform the navigation policy. Architectures including actor-critic networks [38]-[40], [47], proximal policy optimization (PPO) [16], [34], [46], [48], or transformers [37], [41] were used to generate future navigation actions given the

current and history of image observations. In [48], an additional switch policy was introduced to decide between exploration and verification actions, enabling hybrid DRL and classical feature matching. To enable long-horizon IIN tasks where the robot must navigate to multiple objects and retain information about previously seen locations, policies were memory-augmented using attention networks [34], [41], topological memory graphs [47], or feature-extraction encoders [38]. Experiments were conducted in the Habitat [16], [34], [38], [41], [44], [46]-[48], or Gazebo [37] simulators with 3D photorealistic environments, and real-world indoor environments [37], [39] [40].

IL-based methods generated navigation actions by learning from expert demonstrations from other robot platforms in pre-existing IIN datasets [42], [43], [45]. The datasets used to train the IL architectures include the SACSoN dataset [42], RECON [43] or Active Vision Dataset [45], to learn how to convert current observations into actions. In [42], a vision transformer was used to encode past observation sand the goal image into a latent representation, which was then used by a diffusion model to predict future actions. In [43], a topological memory module was used to keep a compact representation of the environment and generate subgoals towards the goal image. In [45], a variational generative encoder was used to predict next actions towards the goal while adhering to safety and collision constraints with obstacles using a deep convolutional neural network.

2.2.2 Large Foundation Model Methods

Large foundation model methods have used pre-trained VLMs to generate semantic priors with the robot's current RGB observations for locating the goal image in the environment, or query an LLM directly to inform the navigation policy.

In [36], [49], [50], the RGB observations were converted to a semantic embedding space using contrastive language image pretraining (CLIP), while the goal was processed through a ResNet encoder to align semantic features and locate the goal. Namely, in [36], a semantic map of the environment is incrementally constructed using CLIP to localize and keep a memory of instances of object categories, after which the robot navigation policy is executed using the Fast-Marching Method. In [49], an additional semantic training strategy was introduced to recognize the goal object under viewpoint changes, and enable the robot to reach the target location provided varying camera goal images. In [50], the embeddings from observations and the goal were concatenated

into a joint embedding and trained with an actor-critic policy network to determine the robot actions.

Other methods have directly queried and LLM to generate navigation actions for the robot by prompting the LLM with an online scene graph of the environment, describing spatial and semantic relationships between objects [51], [52]. The observations with the scene graph are use to generate plans towards objects of the same instance, and a local feature matching method using LightGlue [51] or SBERT [52], is used to compare the current observation against the goal image.

2.2.3 Gaussian Splatting Methods

3DGS methods have leveraged photorealistic scene modeling and novel view-synthesis to preserve detailed visual information across diverse viewpoints of the goal image [35], [60]-[62]. Prior to navigation, these methods build a high-fidelity 3DGS map and embed semantic features into the map using CLIP [35], [61], or create topological graphs [60] to enable querying of the goal image at test-time. GauScoreMap, [61], improved on GaussNav, [35], for goal search efficiency by estimating optimal viewpoints for querying using hierarchical scoring, instead of performing an exhaustive search across all possible viewpoints. Navigation policies used to predict actions to reach the goal image included diffusion [60], model predictive control with dynamic Bayesian updates [62], or the Fast-Marching Method [35], [62]. Experiments were conducted in the Habitat simulator [35], [60], [61], or real-world environments [62].

IGL-Nav, [73], on the other hand builds an incremental 3DGS of the environment instead of using a pre-built 3DGS representation. The 3DGS map is updated online with new RGB observations and is used for navigation by localizing the 6DoF pose of the goal image, and iteratively navigating towards this location. Localization of the goal pose is performed by creating a voxel grid on the scene and using 3D convolution to generate an activation map, which identifies the next exploration frontier, and the pose is further refined using gradient descent.

2.2.4 Summary of Limitations

DRL-based methods used fixed encoders to extract and compare features between the current observation and goal image from the robot perspective [16], [34], [37]-[41], [44], [46]-[48]. However, these encoders are unable to recognize the goal object from different viewpoints and under various lighting conditions, resulting in degraded performance [37]. IL-based methods rely

on trained policies from existing datasets, limiting their recognition ability to the specific set of objects included in these datasets [42], [43], [45]. This is a limitation as existing datasets cannot comprehensively encompass every goal object, and thus, cannot generalize to novel IIN scenarios [42].

Large foundation model approaches use semantic priors to enable searching for the goal image using text instead of visual matching [36], [49]-[52]. However, these methods are limited to the quality and granularity of the language to image alignments, as information is lost when converting the goal image to semantic labels. Hence, the robot is unable to differentiate between semantically similar, but visually distinct objects [74].

3DGS methods enable viewpoint invariance at the goal location but all assume that a detailed 3DGS map is provided to the robot prior to deployment [35], [60]-[62]. This assumption is unrealistic in unknown or novel environments where scene reconstructions are either sparse of unavailable [37]. The closest to our work is [73], however, this method relies on a dense 3DGS representation of the environment to localize the goal pose effectively. Additionally, it does not consider semantic context during exploration, leading to sub-optimal and redundant exploration.

To address these limitations, we propose SplatSearch, the *first* online 3DGS-based architecture to address the problem of goal recognition under arbitrary viewpoints in sparsely reconstructed environments. SplatSearch employs a novel viewpoint synthesis module that renders multiple views around candidate objects using the 3DGS map, and integrates a multi-view diffusion model to inpaint incomplete renders for robust feature matching against the goal image. SplatSearch also introduces a frontier exploration strategy that leverages similarity scores from these rendered viewpoints to enable the robot to progressively refine its exploration towards the goal object.

Chapter 3
Robot Social Navigation Architecture using Vision Language Models

This chapter[1] introduces a social navigation architecture using vision-language models to enable mobile robots to adapt to social context and abide by social norms when navigating among people. Section 3.1 provides an introduction to the problem, Section 3.2 provide a detailed overview of the architecture, OLiVia-Nav, Section 3.3 details the datasets used for the training and evaluating the architecture, Section 3.4 discusses how the modules were trained, and finally, Section 3.5 presents the experimental results, demonstrating the effectiveness of our approach in real-world human-environments.

3.1 Problem Definition

Robot social navigation addresses the problem of a mobile robot that needs to navigate from its initial pose (x_0, y_0, ϕ_0) to a goal pose (x_G, y_G, ϕ_G) in an unknown environment consisting of dynamic people. The robot uses RGB images from its onboard camera, I_{RGB}, to detect visual features such as people and objects, and 3D LiDAR point clouds, $L_{(x,y,z)}$, to provide the 3D structural layout of the robot's surrounding. The task is to predict K socially compliant future trajectories, $\boldsymbol{\tau}^P$, from (x_0, y_0, ϕ_0) to (x_G, y_G, ϕ_G) given an expert demonstration trajectory $\boldsymbol{\tau}^E = \{(x_j, y_j, \phi_j)\}_{j=0}^{G}$:

$$\boldsymbol{\tau}^P = \left\{\tau_k^P : \tau_k^P = \{(x_i^k, y_i^k, \phi_i^k)\}_{i=0}^{G}, k \in [1, K]\right\}. \quad (1)$$

The trajectories, $\boldsymbol{\tau}^P$, are generated by a deep neural network, $f_\theta(I, L)$, with learnable parameters, θ. The overall objective is to learn θ in order to minimize the winner-takes-all (WTA) loss [75] between the predicted and expert trajectories:

---

[1] This chapter is based on and includes sections reprinted, with permission from IEEE: S. Narasimhan, A. H. Tan, D. Choi, G. Nejat, "OLiVia-Nav: An Online Lifelong Vision Language Approach for Mobile Robot Social Navigation", *May 2025, International Conference on Robotics and Automation.*

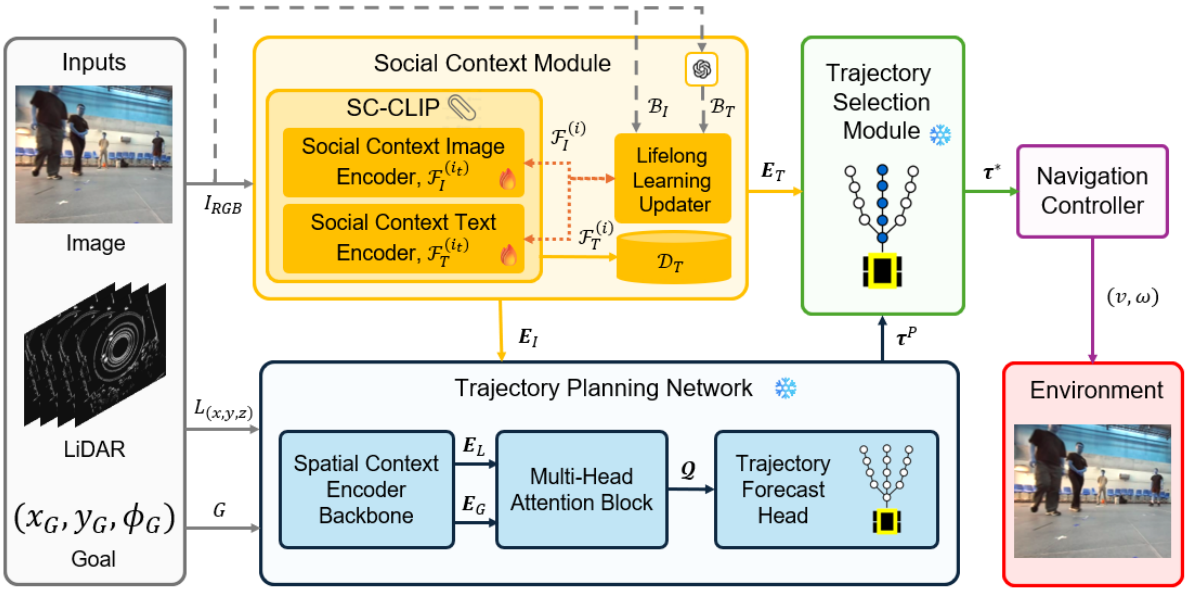


Fig. 1. OLiVia-Nav consists of four modules, 1) *Social Context Module (SCM)* extracts social context embeddings for trajectory planning and selection, and to update the social context image and text encoders for lifelong learning, 2) *Trajectory Planning Network (TPN)* generates socially compliant navigation trajectories using LiDAR data, goal and the social context image embedding, 3) *Trajectory Selection Module (TSM)* selects the trajectory that follows the high-level navigation action encoded in the social context text embedding, and 4) *Navigation Controller (NC)* uses a Proportional Derivative Integral (PID) controller to follow the selected trajectory. 🔥 and ❄ denote modules that are updated and frozen during navigation. denotes a VLM.

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathrm{WTA}(f_\theta(I, L), \boldsymbol{\tau}^E)\,. \tag{2}$$

3.2 Architecture

The OLiVia-Nav architecture consists of four main modules, Fig. 1: 1) *Social Context Module* (*SCM*), 2) *Trajectory Planning Network* (*TPN*), 3) *Trajectory Selection Module* (*TSM*), and 4) *Navigation Controller* (*NC*). The *SCM* generates social context embeddings from the robot's surroundings, which are used by the *TPN* and *TSM* to predict and select a socially compliant trajectory for execution by the *NC*. Each module is discussed in detail below.

3.2.1 Social Context Module (SCM)

The proposed *SCM* consists of two submodules: 1) *Social Context Contrastive Language Image Pretraining (SC-CLIP),* and 2) a *Lifelong Learning Updater (LLU),* Fig. 1.

Long Text Caption (T_l): In the provided image, the scenario involves navigating among dynamic people in a shopping area. There are several people present, including a person in an orange jacket walking ahead and slightly to the left, and another person in a red jacket standing near a flower planter on the right in the immediate vicinity. The pathway is clear, but it curves slightly to the right. To navigate safely, you should continue moving straight while gradually turning to the right to follow the pathway. Ensure to maintain a safe distance from the person in the orange jacket and avoid any sudden movements that could cause collisions.

Short Text Caption (T_s): People in orange and red jackets walking and standing in a busy shopping area. Curving pathway with flower planter on the right.

Fig. 2. Example of a long and short text caption generated by the large VLM for training *SC-CLIP*. The long text caption describes the social scenario, objects and people in the environment and the high-level navigation action for the robot. Image is from the MuSoHu dataset [69].

3.2.1.1 Social Context Contrastive Language Image Pretraining (SC-CLIP)

We introduce *SC-CLIP*, a distillation approach to transfer the social reasoning of a large VLM to two lightweight encoders: a social context image encoder (*SCIE*), $\mathcal{F}_I$, and a social context text encoder (*SCTE*), $\mathcal{F}_T$. The novelty of *SC-CLIP* is its ability to retain the social understanding of a large VLM, enabling OLiVia-Nav to generalize to diverse social scenarios, without the slow response speed of the large VLM.

The distillation approach consists of two stages. First, a large VLM is used to generate a long and short text caption, (T_l, T_s), to describe the social and environment context within I_{RGB}, Fig. 2. This context includes descriptions of: 1) the social scenario, 2) objects and people in the environment, and 3) the high-level navigation action that the robot should follow to remain socially compliant. Note that both T_l and T_s are required in the training procedure of *SC-CLIP* to enable long text caption understanding, as detailed in [76]. Then, in the second stage, *SC-CLIP* trains the $\mathcal{F}_I$ and $\mathcal{F}_T$ in parallel to align the image embedding, $\boldsymbol{E}_I$, from I_{RGB}, with the corresponding text embedding, $\boldsymbol{E}_T$, from (T_l, T_s), in their respective embedding spaces. Herein, $\mathcal{F}_I$ utilizes a ViT-L/14 transformer backbone [77] to extract visual semantic features from I_{RGB} in order to generate $\boldsymbol{E}_I$. $\mathcal{F}_T$ utilizes a self-attention transformer backbone [78] to extract token semantic features from (T_l, T_s) to generate $\boldsymbol{E}_T$. *SC-CLIP* is trained using the following loss function [76]:

$$\begin{aligned}\mathcal{L}_{SC-CLIP} = \mathrm{CE}(\mathcal{F}_I(I_{RGB}) \cdot \mathcal{F}_T(T_l)^T, labels) \\ + \mathrm{CE}(\mathrm{PCE}(\mathcal{F}_I(I_{RGB})) \cdot \mathcal{F}_T(T_s)^T, labels),\end{aligned} \tag{3}$$

where PCE is the Principal Component Extraction function, which extracts high-level context features from $\mathcal{F}_I(I_{RGB})$ [79], CE is the cross-entropy loss, and $labels$ are the ground truth indices that align $\boldsymbol{E}_I$ and $\boldsymbol{E}_T$.

The trained $(\mathcal{F}_I, \mathcal{F}_T)$ are used to incorporate social and environment context to be used for trajectory planning and selection. Specifically, $\mathcal{F}_I$ is used to generate $\boldsymbol{E}_I$, from RGB images as the robot navigates an environment. $\mathcal{F}_T$ is used to create $\mathcal{D}_T$, of size $|\mathcal{D}_T|$, from an offline dataset (discussed in Section V). During navigation, $\boldsymbol{E}_I$ is used for: 1) trajectory planning by *TPN*, and 2) retrieving $\boldsymbol{E}_T$ from $\mathcal{D}_T$ based on a cosine similarity score, for trajectory selection by *TSM*.

3.2.1.2 Lifelong Learning Updater (LLU)

The objective of the *LLU* is to update $\mathcal{F}_I$ and $\mathcal{F}_T$ during robot navigation to account for new social scenarios that were not present during training. *LLU* collects a batch, $\mathcal{B}_I$, of I_{RGB} during navigation and stores the batch in a buffer of size $|\mathcal{B}|$, which is a tunable parameter that determines the frequency of model updates. These images are passed to a large VLM (GPT4) to obtain the batch, $\mathcal{B}_T$, which is also used to update $\mathcal{F}_I$ and $\mathcal{F}_T$. A Symmetric Image-Text fine-tuning strategy is used to update $\mathcal{F}_I$ and $\mathcal{F}_T$ using the loss function [80]:

$$\mathcal{L}_{LLU} = -\sum_{\boldsymbol{E}_I \in V_I} \log \frac{\exp\left(\frac{\hbar(\boldsymbol{E}_I, \boldsymbol{E}_T)}{\mu}\right)}{\sum_{\boldsymbol{E}'_T \in \mathrm{V}_T} \exp\left(\frac{\hbar(\boldsymbol{E}_I, \boldsymbol{E}'_T)}{\mu}\right)}, \tag{4}$$

where V_I are image features from batch $\mathcal{B}_I$, and V_T are the text features from the batch $\mathcal{B}_T$, $\hbar(\cdot,\cdot)$ measures the cosine similarity score, and μ is the temperature to control the sharpness of the distribution. We denote $\mathcal{F}_I$ at update iteration i, as $\mathcal{F}_I^{(i)}$, and $\mathcal{F}_T$ at iteration i, as $\mathcal{F}_T^{(i)}$. The last iteration is denoted as $\mathcal{F}_I^{(i_t)}$ and $\mathcal{F}_T^{(i_t)}$, where i_t represents the latest *LLU* iteration update. The updated encoders, $\mathcal{F}_I^{(i_t)}$ and $\mathcal{F}_T^{(i_t)}$, are used to generate $\boldsymbol{E}_I$ and $\mathcal{D}_T$, respectively.

3.2.2 Trajectory Planning Network (TPN)

We propose a novel *TPN* to generate socially compliant trajectory candidates, $\boldsymbol{\tau}^P$, by uniquely incorporating LiDAR data $L_{(x,y,z)}$, navigation goal $G = (x_G, y_G, \phi_G)$, and the social context embedding, $\boldsymbol{E}_I$. The *TPN* consists of a spatial context encoder backbone (*SCEB*), a multi-head attention block (*MHAB*), and a trajectory forecast head (*TFH*), Fig. 1.

The *SCEB* consists of two encoders. A LiDAR encoder to extract voxel features for the 3D points residing in each voxel [81] using five residual blocks [82] in order to obtain the LiDAR embedding vector, $\boldsymbol{E}_L$. A goal encoder uses a single layer feed-forward network (FFN) and rectified linear unit (ReLU) activation to obtain the goal embedding vector, $\boldsymbol{E}_G$.

$\boldsymbol{E}_I$, $\boldsymbol{E}_L$ and $\boldsymbol{E}_G$ are fused together using the *MHAB* to exchange context-relevant information across each embedding representation, as detailed below. The attention process starts with a randomly initialized query, $\boldsymbol{Q}^{(0)}$, which passes through three cross-attention layers: 1) *SC-CLIP* cross-attention, which attends to $\boldsymbol{E}_I$ to incorporate social and environment context from I_{RGB}, 2) LiDAR cross-attention, which attends to $\boldsymbol{E}_L$ to incorporate geometric and motion features from $L_{(x,y,z)}$, and 3) goal cross-attention, which attends to $\boldsymbol{E}_G$ to condition the trajectory generation on G. Each attention block updates the query sequentially using [83]:

$$\boldsymbol{Q}_{att}^{(z)} = \mathrm{LN}\left(\boldsymbol{Q}^{(z)} + \mathcal{A}\left(\boldsymbol{Q}^{(z)}, \cdot\right)\right), \tag{5}$$

$$\boldsymbol{Q}^{(z+1)} = \mathrm{LN}\left(\boldsymbol{Q}_{att}^{(z)} + \mathrm{FFN}\left(\boldsymbol{Q}_{att}^{(z)}\right)\right), \tag{6}$$

where $\boldsymbol{Q}_{att}^{(z)}$ represents the intermediate query after performing cross-attention on the z^{th} attention layer, $\boldsymbol{Q}^{(z)}$ represents the query for the z^{th} attention layer, $\mathcal{A}$ represents one of the three aforementioned cross-attention operations, LN denotes layer normalization. The final output, $\boldsymbol{Q}^{(3)} = \boldsymbol{Q}$, is passed into the *TFH* for trajectory planning.

In the *TFH*, the output query is fed into N_{GRU} GRUs with each predicting one trajectory, τ_k^P. The outputs are concatenated into a single vector, $\boldsymbol{\tau}^P$, Eq. 1. The *TPN* is trained using the WTA loss function, Eq. 2, to predict multiple trajectories. The *TSM* then uses $\boldsymbol{\tau}^P$ for trajectory selection.

3.2.3 Trajectory Selection Module (TSM)

The objective of the *TSM* is to select a socially compliant trajectory from $\boldsymbol{\tau}^P$ using $\boldsymbol{E}_T$. Namely, $\boldsymbol{E}_T$ is passed through an FFN to produce $\boldsymbol{E}_\tau$, while $\boldsymbol{\tau}^P$ is processed by a separate FFN and GRU to produce embedding vector $\boldsymbol{E}_C$. The *TSM* is trained using the following loss function:

$$\mathcal{L}_{TSM} = \text{CE}(\text{FFN}(\boldsymbol{E}_C \oplus \boldsymbol{E}_\tau), k^*), \tag{7}$$

where $\oplus$ represents the concatenation operation and k^* is an index that identifies the trajectory, $\boldsymbol{\tau}^*$, described by the navigation action in T_l. $\boldsymbol{\tau}^*$ is used by the *NC* for execution.

3.2.4 Navigation Controller (NC)

The selected $\boldsymbol{\tau}^*$, is used by *NC* to generate robot linear and angular velocities, (v, ω) using a PID controller [84].

3.3 Datasets

We collected three datasets to train the modules of OLiVia-Nav: 1) Social Context Dataset, $\mathcal{D}_{SC}$, using images from MuSoHu [69] to train *SC-CLIP*, 2) Trajectory Planning Dataset, $\mathcal{D}_{TPN}$, using expert trajectories, LiDAR point clouds and RGB images from MuSoHu to train the *TPN*, and 3) Trajectory Selection Dataset, $\mathcal{D}_{TSM}$, using expert trajectories from MuSoHu to train the *TSM*.

1. **Social Context Dataset, $\mathcal{D}_{SC}$:** This dataset comprises of 20,000 RGB images, and corresponding (T_l, T_s), which are generated using GPT4o [72]. We use GPT4o for its ability to generate socially related and contextually relevant captions [85]. The text captions include descriptions of the social scenario, objects and people in the scene, and

the high-level navigation plan. The database, $\mathcal{D}_T$, used within the *LLU*, is curated by taking a randomized subset of T_l from $\mathcal{D}_{SC}$, and generating the corresponding $\boldsymbol{E}_T$ using $\mathcal{F}_T^{(i_t)}$.

2. **Trajectory Planning Dataset, $\mathcal{D}_{TPN}$:** This dataset comprises of 5,000 expert trajectories, LiDAR point clouds, and RGB images. The expert trajectories, $\boldsymbol{\tau}^E$, are represented by a sequence of 10 future robot poses $\{(x_i^E, y_i^E, \theta_i^E)\}_{i=0}^{10}$ from the current robot pose. For each sequence, all waypoints are transformed into the reference frame of the initial robot pose, $(x_0^E, y_0^E, \theta_0^E)$, for normalization and consistency in outputs. I_{RGB}, is taken at the robot's initial pose.

3. **Trajectory Selection Dataset, $\mathcal{D}_{TSM}$:** This dataset consists of 20,000 predicted trajectories, $\boldsymbol{\tau}^P$, along with the corresponding RGB images, I_{RGB}, T_l, $\boldsymbol{E}_T$, and the trajectory index, k^*, that corresponds to the high-level action in T_l.

3.4 Training

OLiVia-Nav was trained in three stages: 1) *SCIE* and *SCTE* using the *SC-CLIP* framework, Eq. 3, 2) *TPN* in an end-to-end manner, Eq. 2, and 3) *TSM* based on the trajectory plans of the *TPN*, Eq. 5 and 6. Training was conducted with NVIDIA H100 GPU with 80GB of VRAM and 32GB of RAM.

3.4.1 SC-CLIP Training

SC-CLIP was trained with a batch size of 256 and a cosine annealing learning rate scheduler with a learning rate (LR) of 0.0001 [86], to gradually reduce LR over time for convergence. An AdamW optimizer [87] was used with weight decay (WD) of 0.01 to prevent overfitting. *SC-CLIP* was trained for 100 epochs. We implemented early stopping and obtained the lowest validation loss epoch.

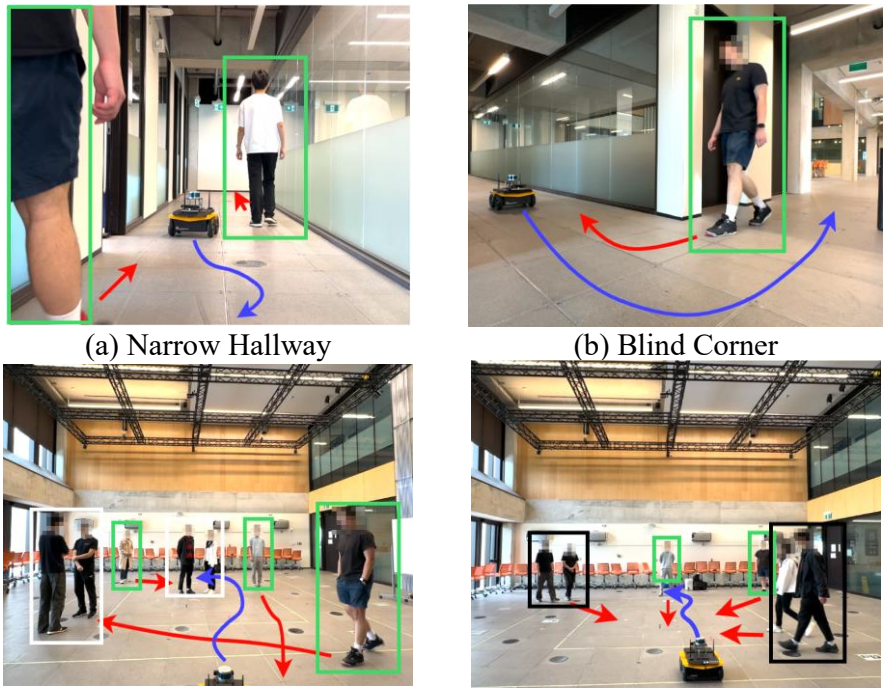
3.4.2 TPN Training

The *TPN* was trained with a batch size of 10, LR of 0.0008, and the AdamW optimizer with WD 0.0001. $\boldsymbol{E}_I$, $\boldsymbol{E}_T$ and $\boldsymbol{E}_G$ were projected into a common dimension of $C = 128$ using an FFN. For *MHAB*, 32 heads were used. The number of predicted trajectories was set to $K = 5$, and correspondingly $N_{GRU} = 5$. *TPN* was trained for 500 epochs until the lowest validation loss was achieved.

3.4.3 TSM Training

The *TSM* was trained with a batch size of 128 and an LR of 0.00001. A WD of 0.00001 was used with an AdamW optimizer. *TSM* was trained for 500 epochs similar to *TPN*.

3.5 Experiments



(a) Narrow Hallway (b) Blind Corner

(c) Static Groups + Dynamic People (d) Dynamic Groups + Dynamic People

Fig. 3. The four experimental scenarios: (a) narrow hallway with two people, (b) approaching a blind corner, (c) static groups + dynamic people, and (d) dynamic groups + dynamic people. Red arrows represent human trajectories, blue arrows represent the robot's predicted trajectory, green boxes represent dynamic people, white boxes represent static groups, and black boxes represent dynamic groups.

We conducted two sets of experiments with the Clearpath Jackal robot to evaluate the performance of OLiVia-Nav in: 1) a real-world comparison study with state-of-the-art (SOTA) social navigation methods, and 2) an ablation study to investigate the design choices of OLiVia-Nav. We used GPT4o [72] as the large VLM within the *SCM*.

3.5.1 Comparison Study

We conducted extensive experiments in a University of Toronto building to evaluate the performance of OLiVia-Nav. Three metrics were utilized: 1) mean squared error (MSE) to determine the positional error between the robot trajectory and a ground truth trajectory, 2) Hausdorff loss [29] [88] to evaluate the shape similarity between actual and ground truth trajectories, and 3) personal space violation duration (PSV) to determine the length of time the robot travels in the personal space of a person (< 0.25 m) [89]. The ground truth trajectory was collected through teleoperation by one of our researchers using North American social navigation rules.

Four distinct social scenarios were used in the experiments, similar to [68], with 3 trials per scenario; 1) narrow hallway, where two dynamic people approach the robot front-on, Fig. 3(a), 2) blind corner, where the robot needs to approach and turn a corner while avoiding a dynamic person, Fig. 3(b), 3) navigating past two static groups and three dynamic people, Fig. 3(c), and 4) navigating through an environment with two dynamic groups and two dynamic people, Fig. 3(d).

3.5.1.1 Comparison Methods

We benchmarked OLiVia-Nav against the following SOTA methods:

- **VLM-Social-Nav** [14]**:** The **VLM-Social-Nav** method is a HMF approach which utilizes a cost-based planner with GPT4o for social navigation. The costs include: 1) obstacle collision cost obtained from LiDAR point clouds, 2) goal cost from a goal position, and 3) a social cost obtained through GPT4o from RGB images. The planner generates a set of possible robot velocity pairs, (v, ω), where the pair with the lowest combined cost is selected as the action. We choose VLM-Social-Nav to compare how the direct usage of a large VLM (GPT4o) affects navigation performance, considering its slower response time

- **MultiSoc** [26]: The **MultiSoc** method is a HMB approach which predicts human trajectories by using a GNN with attention mechanisms to incorporate spatial relationships and the positions of people over time. The GNN takes as input RGB images and LiDAR point clouds to encode person positions, then uses the attention layers to predict future trajectories. The predicted trajectories are integrated into a DRL navigation policy trained

with proximal policy optimization [90]. The output of the policy are robot velocity commands (v, ω). We choose MultiSoc as it is trained in a 2D simulator with procedurally generated human trajectories. This allows us to evaluate the impact of using real-world collected trajectory data for training of OLiVia-Nav.

3.5.1.2 Results

TABLE I

COMPARISON RESULTS FOR THE FOUR SOCIAL SCENARIOS

Scenario	Method	MSE↓	Haus ↓	PSV (s) ↓
Narrow Hallway	OLiVia-Nav	0.1075	0.7348	1.2
	VLM-Social-Nav	0.1915	0.8785	1.9
	MultiSoc	0.2968	0.9095	2.1
Blind Corner	OLiVia-Nav	0.0236	0.2572	0.4
	VLM-Social-Nav	0.0755	0.5384	2.6
	MultiSoc	0.1021	0.4596	2.8
Static Groups + Dynamic People	OLiVia-Nav	0.2195	0.7563	2.1
	VLM-Social-Nav	0.4361	1.5579	3.5
	MultiSoc	0.2747	1.1081	3.2
Dynamic Groups + Dynamic People	OLiVia-Nav	0.0733	0.4813	3.3
	VLM-Social-Nav	0.1459	0.6832	4.7
	MultiSoc	0.1154	0.7929	4.5

The results of the comparison study are presented in Table I. OLiVia-Nav had the lowest MSE, Hausdorff loss, and PSV across all scenarios, generating social navigation trajectories that closely match ground truth trajectories in both proximity and shape. OLiVia-Nav utilizes the *TPN*, which was trained using IL on a dataset containing real-world human trajectories and diverse environments. This enabled OLiVia-Nav to predict robot trajectories that resemble human navigation behaviors in realistic social scenarios by directly learning from real-world trajectories [11]. In contrast, VLM-Social-Nav utilized hand-tuned cost functions to generate robot trajectories for obstacle avoidance and goal reaching instead of real-world data. This approach limited the VLM-Social-Nav's ability to adapt to nuanced real-world behaviors, such as a robot dynamically adapting its speed and orientation when navigating around a corner. As a result, VLM-Social-Nav had lower performance than OLiVia-Nav. Lastly, MultiSoc resulted in robot navigations that had frequent heading direction changes, since social and environment context were not explicitly incorporated during training [26].

As the number of dynamic people in the environment increased from 1 to 6, the PSV increased for all methods due to the larger number of people encountered. In general, OLiVia-Nav demonstrated lower PSV than both VLM-Social-Nav and MultiSoc. VLM-Social-Nav had a higher PSV due to its slower response speed (average 0.6Hz), compared to OLiVia-Nav (average 5Hz), as it relied on directly querying GPT4o (large VLM) to compute social costs. This caused delays in generating navigation actions which resulted in longer durations of personal space violations. Even though MultiSoc was able to plan in real time (average 5Hz), it exhibited a higher PSV compared to OLiVia-Nav due to inaccurate human trajectory predictions. Namely, MultiSoc frequently changed the robot's heading direction in response to pose errors in the predicted human trajectories, which resulted in the robot violating personal space. Friedman tests were performed on the MSE, Hausdorff and PSV metrics for all three methods across all social scenarios. The results show statistically significant differences ($p < 0.001$). A Post-hoc Wilcoxon Signed-rank test with Bonferroni correction further confirmed statistically significant difference in all three metrics when comparing OLiVia-Nav to each method across all social scenarios ($p < 0.025$). A video of our OLiVia-Nav approach and the SOTA methods navigating a hallway (Social Scenarios 1 and 2) and open space environment (Social Scenarios 3 and 4) is on our YouTube channel: https://youtu.be/eyFJiOIITO0

3.5.2 Ablation Study

TABLE II

ABLATION STUDY

Methods	**Before *LLU* Update**		**After *LLU* Update**	
	MSE ↓	**Haus ↓**	**MSE ↓**	**Haus ↓**
OLiVia-Nav (ours)	0.2981	1.8087	0.2521	1.4893
OLiVia-Nav w/o $\boldsymbol{E}_I$	0.4412	2.3772	0.3701	2.0839
OLiVia-Nav w/o $\boldsymbol{E}_T$	0.3838	2.1063	0.3791	2.0349
OLiVia-Nav w/o $\boldsymbol{E}_L$	0.4382	2.3681	0.4180	2.1894

We conducted an ablation study to investigate the design choices of OLiVia-Nav. Namely, we considered: **1. OLiVia-Nav without (w/o) $\boldsymbol{E}_I$** to determine the effect of visual semantic features from I_{RGB} on trajectory planning in the *TPN*, **2. OLiVia-Nav w/o $\boldsymbol{E}_T$** to evaluate the impact of token semantic features from (T_l, T_s) on the selected trajectory in *TSM*, **3. OLiVia-Nav w/o $\boldsymbol{E}_L$** to explore the influence of the LiDAR point cloud embedding on trajectory planning. Note that an

ablation on $\boldsymbol{E}_G$ does not provide useful insights since navigation requires a goal to be defined. The ablation study was conducted on a hold-out test dataset in MuSoHu.

For each of these variants, the performance was evaluated twice; Before *LLU* Update, using only $\mathcal{F}_I^{(0)}$ and $\mathcal{F}_T^{(0)}$, and After *LLU* Update, using $\mathcal{F}_I^{(i_t)}$ and $\mathcal{F}_T^{(i_t)}$, where $i_t = 1$ and $|\mathcal{B}| = 50$. The goal is to investigate the contribution of the *LLU* on all variants in terms of adapting to new social scenarios that were not present during the training of OLiVia-Nav. The performance metrics include the MSE and Hausdorff loss.

The ablation study results are presented in Table II. The Before LLU Update results show that the OLiVia-Nav had the lowest MSE and Hausdorff loss of 0.2981 and 1.8087 compared to all variants. The After LLU Update results also showed OLiVia-Nav having the lowest MSE and Hausdorff loss of 0.2521 and 1.4893, respectively. All ablation variants also improved their performance with the updates. For example, OLiVia-Nav w/o $\boldsymbol{E}_I$ achieved a higher MSE and Hausdorff loss compared to OLiVia-Nav as the predicted robot trajectories did not consider social and environment context. OLiVia-Nav w/o $\boldsymbol{E}_T$ also achieved degraded navigation performance as the social context embedding from the token semantic features of the text captions was not used. This embedding encodes the high-level navigation action and is necessary for trajectory selection. The variant was unable to utilize high-level action information without this embedding, resulting in the random selection of trajectories. Lastly, as OLiVia-Nav w/o $\boldsymbol{E}_L$ did not the use LiDAR embedding for obstacle detection, trajectory plans resulted in collisions with objects and people.

3.6 Summary

This chapter presents a novel lifelong vision language architecture OLiVia-Nav which uses VLMs to address the robot social navigation problem in dynamic human environments. It introduces a novel distillation process, SC-CLIP, to leverage the social reasoning capabilities of large VLMs for trajectory planning while being able to adapt online to new scenarios using the lifelong learning ability of the lightweight VLM. Extensive real-world experiments demonstrate that OLiVia-Nav follows expert trajectories more accurately compared to SOTA methods. Furthermore, ablation studies show the importance of each of the embeddings on robot social navigation performance.

Chapter 4
Instance Image Goal Navigation Architecture using Gaussian Splatting

This chapter introduces an instance image goal navigation (IIN) architecture, SplatSearch, for searching an unknown environment for an object or person, specified visually by an image, using semantic and visual context in the environment, Fig. 4. Section 4.1 provides an introduction to the problem, Section 4.2 explains the architecture of SplatSearch, Section 4.3 discusses the training of the multi-view diffusion model for inpainting and completing sparse-view images, and finally, Section 4.4 shows the results for a comparison study and ablation study, demonstrating the effectiveness of our approach over SOTA methods.

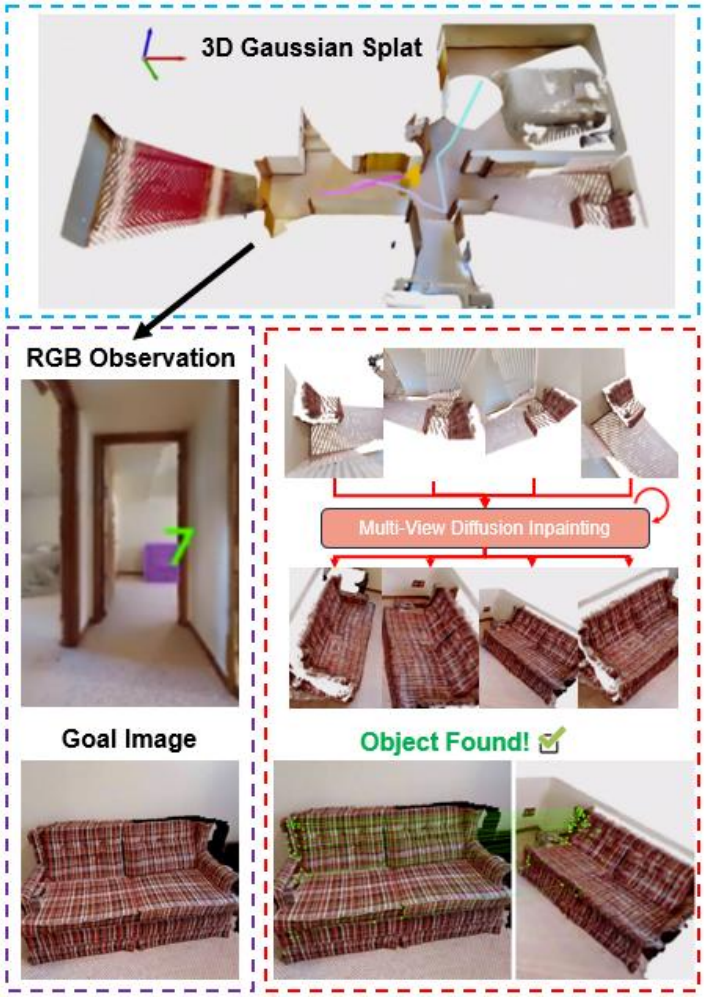


Fig. 4. An overview of SplatSearch. SplatSearch builds a sparse-view 3DGS map and inpaints incomplete regions of the novel viewpoints using a multi-view diffusion model

4.1 Problem Definition

Robot instance image goal navigation addresses the problem of a mobile robot that is placed at an initial randomized pose in an unknown, static, 3D environment and must navigate to a goal object or person represented by an image, $I^g \in \mathbb{R}^{H\times W\times 3}$, located at an unknown goal position x^g. At each timestep, t, the robot receives an RGB image and depth observation, $O_t = (I_t, D_t)$ from an onboard camera, and executes an action $a_t = \pi(O_t)$ according to a navigation policy, π. The total path length of the robot is given by:

$$d(\pi) = \sum_{t=0}^{T-1} \|x_{t+1} - x_t\|_2 \,, \tag{1}$$

where x_t is the robot position after action a_t, and T is the termination timestep. Navigation is considered successful if the robot reaches within a distance radius, τ_{succ}, of the true goal position. The objective is to execute a navigation policy that minimizes the expected path length subject to this success condition:

$$\pi^* = \underset{\pi}{\operatorname{argmin}}\, \mathbb{E}[d(\pi) \mid \|x_T - x^g\|_2 \leq \tau_{succ}] \,. \tag{2}$$

4.2 Architecture

The proposed architecture, shown in Fig. 5, consists of six stages: 1) Map Generation Module (MGM), 2) Semantic Object Identification Module (SOIM), 3) Novel Viewpoint Synthesis Module (NVSM), 4) View-Consistent Image Completion Network (VCICN), 5) Feature Extraction Module (FEM) and 6) Exploration Planner (EP). Using RGB-D inputs and odometry, the MGM incrementally builds a 3DGS map and a value map, in order to evaluate frontiers using semantic relevance to the goal image. If the SOIM detects an object of the goal class, the NVSM generates candidate viewpoints around the object, which are inpainted by the VCICN to handle missing regions. The FEM compares these inpainted views with the goal image to produce object scores. The EP uses the value map and object scores to select the next frontier, or navigates to the goal identified by the SOIM. Each module is discussed below.

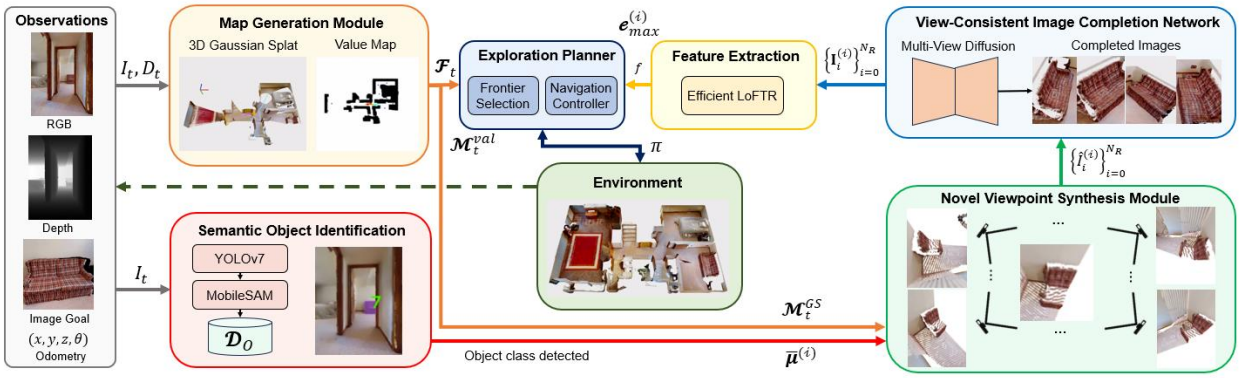


Fig. 5. SplatSearch consists of six modules, 1) Map Generation Module (MGM) generates a 3DGS map for photorealistic mapping and occupancy map for frontier exploration, 2) Semantic Object Identification Module (SOIM) detects if the current RGB image contains an object with the same class as the goal image, 3) Novel Viewpoint Synthesis Module (NVSM) generates multiple viewpoints around the object, 4) View-Consistent Image Completion Network (VCICN) inpaints the sparse viewpoint images from NVSM, 5) Feature Extraction Module (FEM) uses the inpainted images to feature match against the goal image, 6) Exploration Planner (EP) which selects and generates feasible paths towards new frontiers.

4.2.1 Map Generation Module

The MGM utilizes RGB images, I_t, depth images, D_t, and robot position, x_t, to incrementally construct 1) a sparse-view 3DGS map of the environment for novel viewpoint rendering around objects, and 2) a value map, which uses semantic context from the images to guide frontier selection.

4.2.1.1 3DGS Map

A 3DGS map of the environment at timestep t, $\boldsymbol{\mathcal{M}}_t^{GS}$, is represented by anisotropic 3D Gaussians, G_i, each parameterized by color $\boldsymbol{c}_i$, position $\boldsymbol{\mu}_i \in \mathbb{R}^3$, covariance $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times3}$, and opacity $\boldsymbol{o}_i$. Let $\mathcal{P}: \mathbb{R}^2 \to \mathbb{R}^3$ denote the mapping from a 2D pixel coordinate in the rendered image, $\boldsymbol{u}_k \in \mathbb{R}^2$, to its corresponding 3D point in the scene. The weight of 3D Gaussian, G_i, at a pixel $\boldsymbol{u}_k$ in the rendered image is given by the following equation [91]:

$$w_i(\boldsymbol{u}_k) = \boldsymbol{o}_i \exp\left(-\frac{1}{2}(\mathcal{P}(\boldsymbol{u}_k) - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\mathcal{P}(\boldsymbol{u}_k) - \boldsymbol{\mu}_i)\right), \quad (3)$$

where $i \in \{1, \dots, N_G\}$, indexes the set of 3D Gaussians and $k \in \{1, \dots, N_P\}$ indexes the pixels in the rendered image. Given the camera pose, the rendered color, depth and opacity images, can be created by alpha-compositing the 2D projection of each Gaussian [92]:

$$\hat{I}_t(\boldsymbol{u}_k) = \sum_{i=1}^{N_G} \boldsymbol{c}_i w_i(\boldsymbol{u}_k) \prod_{j=1}^{i-1} \left(1 - f_j(\boldsymbol{u}_k)\right). \tag{4}$$

$$\widehat{D}_t(\boldsymbol{u}_k) = \sum_{i=1}^{N_G} z_i w_i(\boldsymbol{u}_k) \prod_{j=1}^{i-1} \left(1 - f_j(\boldsymbol{u}_k)\right), \tag{5}$$

$$\hat{O}_t(\boldsymbol{u}_k) = \sum_{i=1}^{N_G} w_i(\boldsymbol{u}_k) \prod_{j=1}^{i-1} \left(1 - f_j(\boldsymbol{u}_k)\right), \tag{6}$$

where z_i is the depth of G_i in the camera frame. The 3DGS map is optimized online by minimizing the photometric loss, $\mathcal{L}_p$, and geometric loss, $\mathcal{L}_g$, between the rendered and the observed RGB-D frames at timestep t [91]:

$$\mathcal{L}_p = \left|I_t - \hat{I}_t\right| + \left(1 - \text{SSIM}\left(I_t, \hat{I}_t\right)\right), \tag{7}$$

$$\mathcal{L}_g = \left|D_t - \widehat{D}_k\right|, \tag{8}$$

$$\mathcal{L}_t = \lambda_p \mathcal{L}_p + \lambda_g \mathcal{L}_g, \tag{9}$$

where λ_p and λ_g are the weights assigned to the photometric and geometric losses, and SSIM is the Structural Similarity Index [93]. During online mapping, the 3DGS map is updated in regions which are not previously mapped or regions with low accumulated opacity:

$$M_t(u_k) = \left(\hat{O}_t < \tau_o\right) \cup \left(\left(D_t < \widehat{D}_t\right) \cap \left(\left|D_t - \widehat{D}_t\right| > \tau_M\right)\right), \tag{10}$$

where τ_o is the opacity threshold, and τ_M is the median depth error. Pixels satisfying $M_t(u_k) = 1$ are used to update the corresponding Gaussians in the 3DGS map.

4.2.1.2 Value Map

The depth images, D_t, are used to build an online occupancy map, $\boldsymbol{\mathcal{M}}_t^{occ}$, consisting of occupied, free and unknown cells. Frontier points, $\boldsymbol{\mathcal{F}}_t$, are then extracted as the boundary locations between free and unknown cells.

The value map, $\boldsymbol{\mathcal{M}}_t^{val}$, is constructed by assigning each free-space cell in $\boldsymbol{\mathcal{M}}_t^{occ}$, with a score quantifying its visual and semantic relevance to I^g. To obtain these values, we use the image encoder of a pre-trained BLIP-2 VLM [94] to generate semantic embeddings for the goal image, e_G, and the current RGB observation, e_t. The semantic score is computed through the cosine similarity between the embeddings:

$$v_t = \frac{e_t \cdot e_g}{\|e_t\|\|e_g\|}, \tag{11}$$

where a higher value indicates a higher semantic relevance to the goal image. The value v_t is stored in all of the free space pixels within the current field-of-view (FOV) of the robot.

We also introduce a confidence map, $\boldsymbol{\mathcal{M}}_t^{conf}$, to determine how a pixel's semantic value should be updated if it is already assigned a value and is within the current FOV of the robot. The confidence of a pixel i within the FOV of the robot is determined as $c_{t,i} = \cos^2(\pi\theta_i/\theta_{fov})$, where θ_{fov} is the field of view of the robot, and θ_i is the angle of the pixel relative to the optical axis of the robot. At each timestep, the new value, $v_{t,i}^{new}$, and confidence score, $c_{t,i}^{new}$, for all pixels in the robot FOV are recomputed [15]:

$$v_{t,i}^{new} = \frac{c_{t,i}v_t + c_{t,i}^{prev}v_{t,i}^{prev}}{c_{t,i} + c_{t,i}^{prev}}, c_{t,i}^{new} = \frac{c_t^2 + \left(c_{t,i}^{prev}\right)^2}{c_{t,i} + c_{t,i}^{prev}}, \tag{12}$$

where $v_{t,i}^{prev}$ is the previous value $c_{t,i}^{prev}$ is the previous confidence score of the pixel. For each frontier, $f \in \boldsymbol{\mathcal{F}}_t$, we compute a normalized average semantic score for that frontier, $\bar{V}_t(f)$ as the mean of $\boldsymbol{\mathcal{M}}_t^{val}$ within a circle of radius r_{val} around the frontier point, s. These scores are used by the EP to guide frontier selection towards regions that are semantically similar to the goal image.

4.2.2 Semantic Object Identification Module

The SOIM determines whether the current RGB image, I_t, contains an instance belonging to the image goal class and assigns a unique identifier to each detected object.

We use YOLOv7 [95] to generate bounding boxes around candidate objects, which are processed by MobileSAM [96] to generate instance masks $\{m_{t,n}\}_{n=0}^{n_t}$, where n_t denotes the number of masks detected at timestep t. For each mask $m_{t,n}$, the pixels are mapped to their corresponding 3D Gaussians in the 3DGS map, $\mathcal{M}_t^{GS}$, using $\mathcal{P}$. The corresponding set of Gaussians is denoted $\mathcal{G}^{(i)} = \{\boldsymbol{\mu}_i^{(i)}\}$, are aggregated and averaged to estimate the 3D centroid of the object, $\bar{\boldsymbol{\mu}}^{(i)}$, with identification i. Each object is stored in an object database $\mathcal{D}_O$ as:

$$\mathcal{O}^{(i)} = \{\mathcal{G}^{(i)}, \bar{\boldsymbol{\mu}}^{(i)}, \mathcal{V}^{(i)}\}, \tag{13}$$

where $\mathcal{V}^{(z)}$ represents set of viewpoints from which the object has been observed. When a new mask is generated by MobileSAM, its centroid $\bar{\boldsymbol{\mu}}^{new}$, is compared to existing entries in $\mathcal{D}_O$. If the Euclidean distance to an existing centroid is below a spatial threshold, τ_d, the object is merged into the existing entry and the centroid is updated via incremental averaging. Otherwise, a new object entry is created.

The centroid, $\bar{\boldsymbol{\mu}}^{(i)}$, is passed to the NVSM every time any object in $\mathcal{D}_O$ has been observed in f_O new distinct viewpoints, namely when $|\mathcal{V}^{(i)}| \bmod f_O = 0$. This ensures that the NVSM is only executed when increasingly refined object information is obtained.

4.2.3 Novel Viewpoint Synthesis Module

The NVSM generates camera viewpoints using the 3DGS map to render novel images around the object. The input to the NVSM is the object centroid, $\bar{\boldsymbol{\mu}}^{(i)}$, provided by the SOIM.

Around the centroid, $\bar{\boldsymbol{\mu}}^{(i)}$, viewpoints are sampled along a spherical surface of radius r_v, restricted to the upper hemisphere. A total of N_R viewpoints are generated by rotating at fixed angular steps, with each camera-to-world transformation matrix, $\mathbf{T}_{c\to w}$, positioned tangentially to the sphere and oriented towards the centroid. Using these transformation matrices, the set of novel images, $\{\hat{I}_i^{(i)}\}_{i=0}^{N_R}$, are rendered using the 3DGS map. They are subsequently processed by the VCICN to

inpaint incomplete pixels in the rendered images, and produce completed novel views suitable for feature matching.

4.2.4 View-Consistent Image Completion Network

The VCICN is used to complete rendered images from the NVSM by inpainting pixels that are unobserved or partially observed in the 3DGS map. The goal of this network is to improve the feature matching of the FEM by generating inpainted versions of the image, to address the issue of sparse-view 3DGS maps. The input to the VCICN consists of N_R rendered viewpoints, $\left\{\hat{I}_i^{(i)}\right\}_{i=0}^{N_R}$. Each image is paired with a binary mask, $\left\{\widehat{M}_i^{(i)}\right\}_{i=0}^{N_R}$, representing the observed and missing regions of the image. We set $\widehat{M}_0^{(i)} = \mathbf{0}$, as the first viewpoint is unmasked for training stability.

Each rendered image, is encoded into a latent feature map using a variational autoencoder (VAE):

$$z_0^i = \mathrm{VAE}(\hat{I}_i^{(i)}) \in \mathbb{R}^{H\times W\times 4}. \tag{14}$$

The binary mask $\widehat{M}_i^{(i)}$ is down-sampled using the same network to match the latent resolution. For each latent z_0^i, we simulate the forward diffusion process at a randomly chosen timestep t to produce a noisy latent vector:

$$z_t^i = \sqrt{\alpha_t} z_0^i + \sqrt{1-\alpha_t}\epsilon, \tag{15}$$

where α_t follows a consistent noise schedule and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. The sequence of inputs at timestep t is:

$$x_t = \left[z_t^{0:N_R}, z^{0:N_R}\left(1-\widehat{M}^{0:N_R}\right), \widehat{M}^{0:N_R}\right], \tag{16}$$

where $\widehat{M}$ is the down sampled mask. The VCICN predicts the noise $\epsilon_\theta(x_t, \tau_\phi(c), t)$, where c is the CLIP-encoded text prompt corresponding to the goal object [77]. The network is trained with an MSE loss, to encourage accurate noise prediction and enable reconstruction of complete, view-consistent images [97]:

$$\mathcal{L}_D = \mathbb{E}_{x_0,\epsilon,t,c}\left[\left\|\epsilon - \epsilon_\theta\left(x_t, \tau_\phi(c), t\right)\right\|_2^2\right]. \tag{17}$$

After reverse diffusion denoising, the VCICN outputs the completed viewpoint set of images, $\left\{\mathbf{I}_i^{(i)}\right\}_{i=0}^{N_R}$, which are then used the FEM for feature extraction against the goal image.

4.2.5 Feature Extraction Module

The FEM determines whether the current object instance corresponds to the goal image, I^g. The input is the set of completed novel viewpoints $\left\{\mathbf{I}_i^{(i)}\right\}_{i=0}^{N_R}$, generated from the VCICN.

The object score is computed using EfficientLoFTR [98] denoted by $\boldsymbol{E}(\cdot,\cdot)$, which returns the number of feature correspondences between two images. For each viewpoint, $\mathbf{I}_i^{(i)}$, the viewpoint with the highest match is computed as:

$$i^* = \underset{i}{\operatorname{argmax}}\, \boldsymbol{E}\left(I^g, \mathbf{I}_i^{(i)}\right). \tag{18}$$

If the object score satisfies $\boldsymbol{e}_{max}^{(i)} = \boldsymbol{E}\left(I^g, \mathbf{I}_{i^*}^{(i)}\right) \geq \tau_m$, where τ_m is the feature-match threshold, the object is declared to match the goal instance and is passed into the EP to select as the final frontier. If $\boldsymbol{e}_{max}^{(i)} < \tau_m$, however, $\boldsymbol{e}_{max}^{(i)}$ is passed into the EP to guide exploration.

4.2.6 Exploration Planner

The EP determines the next frontier point to navigate to using $\overline{V}_t(f)$ and $\boldsymbol{\mathcal{F}}_t$ from the MGM, and $\left\{\boldsymbol{e}_{max}^{(i)}\right\}_{i=0}^{|\boldsymbol{\mathcal{D}}_O|}$ from the FEM, enabling exploration to be guided by semantic and visual context.

During exploration, if the FEM has not detected the goal object, the EP selects from the set of frontiers, $f \in \boldsymbol{\mathcal{F}}_t$, that maximizes the utility function:

$$f_t^* = \underset{f \in \boldsymbol{\mathcal{F}}_t}{\operatorname{argmax}}[\alpha \cdot d_t(f) + \beta \cdot \mathcal{E}_t(f) + \delta \cdot \overline{V}_t(f)\,], \tag{19}$$

where α, β, δ are the weighting coefficients, $d_t(f)$ is the normalized inverse distance between x_t and f, and $\mathcal{E}_t(f)$ is the normalized object score computed as:

$$\mathcal{E}_t(f) = \frac{1}{C} \max_i \frac{\boldsymbol{e}_{max}^{(i)}}{\tau_m \|f - \overline{\boldsymbol{\mu}}^{(i)}\|}, \tag{20}$$

where C is the normalization constant. $\boldsymbol{\mathcal{M}}_t^{occ}$ is then used by the Navigation Controller, which employs the Fast Marching Method (FMM) [99] to generate the shortest feasible path to move from the current location to f_t^*. If the FEM confirms a match with goal image, FMM is used to plan a path to $\bar{\boldsymbol{\mu}}^{(i)*}$. Upon reaching the goal, the EP executes the stop action and the episode is completed.

4.3 Multi-View Diffusion Policy Training

The details of data collection and training for the VCICN are explained in the sections below.

4.3.1 Dataset Collection

We used the Habitat simulator with the HM3DSem-v0.2 dataset [100], which provides 145 training, 36 validation and 35 testing environments for the IIN task. To construct the training data for VCICN, we executed SplatSearch (without VCICN) on the 145 environments in the training set, spanning six object classes (TV, sofa, chair, table, plant, monitor) [16]. When the SOIM detects an object instance, we applied the NVSM to capture $N_R = 16$ viewpoints around the object. This process produced a dataset of approximately $20k$ objects, each with 16 sparse-views. The same viewpoints were captured from the scene file to retrieve the corresponding ground-truth renderings.

4.3.2 Training

Training was completed on an RTX 4090 GPU with 24GB of VRAM. The model was trained for 100k steps with a batch size of 64, learning rate of 0.0001 and with $N_R = 12$ frames. Training was completed over 2 days and obtained the checkpoints with the lowest validation loss.

4.4 Experiments

We evaluated the performance of SplatSearch by conducting: 1) a comparison study with state-of-the-art (SOTA) learning methods in a photorealistic simulator, 2) an ablation study to investigate the design choices of SplatSearch, and 3) real-world experiments to evaluate the generalizability of SplatSearch in real-world environments.

4.4.1 Comparison Study

We evaluated SplatSearch in the Habitat simulator using two validation sets: 1) the standard HM3DSem-v0.2 validation dataset (HM3D-val), and 2) a modified split that we constructed,

denoted HM3D-val-hard, designed to test robustness to goal images captures from more extreme viewpoints. We construct HM3D-val-hard by re-sampling goal images from HM3D-val in spherical coordinates around the object centroid with radial distance $r \in [0.5,2.5]m$, azimuth $\theta \in (0,\pi]$, and elevations, $r \in [1.0,2.0]m$, and camera angle further perturbed between $[-10^o,+10^o]$. These modifications entail more birds-eye-view perspectives, that are more extreme than HM3D-val [16], making feature matching from the robot viewpoint more challenging. The performance metrics to evaluate each benchmark include 1) Success Rate (SR), and 2) the success path length (SPL).

4.4.1.1 Comparison Methods

- **IEVE** [48]**:** A DRL approach that uses a switch policy to decide at every timestep to explore a new frontier, verify current observation against goal image or navigate towards the goal. We choose IEVE as it is the SOTA DRL approach in IIN to date.

- **UniGoal** [51]**:** This method uses an LLM to build an online scene graph in semantic space, and matches detected objects with the goal using language-grounded embeddings. This was selected as it is an approach that attempts to achieve IIN using semantics in the scene.

- **GaussNav** [35]**:** This method uses a pre-built 3DGS representation of the environment to generate candidate viewpoints around goal-class instances, using a local feature matcher to locate the goal, and using a classical planner to navigate to the goal. This was selected as a representative 3DGS approach.

4.4.1.2 Results

TABLE III
COMPARISON STUDY BETWEEN IIN BENCHMARKS IN HABITAT

Method	HM3D-val		HM3D-val-hard	
	SR ↑	SPL ↑	SR ↑	SPL ↑
IEVE	0.5320	0.2036	0.3910	0.1183
UniGoal	0.5520	0.2373	0.5037	0.1696
GaussNav	0.6492	0.0589	0.5828	0.0398
SplatSearch	**0.6983**	**0.3038**	**0.6639**	**0.2703**

The SR, SPL of SplatSearch with the other comparison methods are shown in Table III. SplatSearch achieved the highest SR (0.6983 & 0.6639) and SPL (0.3038 & 0.2703) across all four

SOTA methods and datasets. The performance degraded across all methods for HM3D-val-hard due to the increased difficulty in recognizing the goal image from BEV perspectives. IEVE achieved the lowest SR across all benchmarks (0.532 & 0.3910). This occurred due to its reliance on DRL for navigation and goal verification, which degrades when provided goal images that are out-of-distribution from the training dataset. UniGoal uses LLM-based semantic reasoning and feature matching, however, it has a lower SR (0.550 & 0.5037) as it is dependent on how well the feature matcher can identify the image across varying viewpoints. SplatSearch is able to render viewpoints from multiple viewpoints and also complete the images using diffusion, hence improves feature matches across arbitrary viewpoints. GaussNav has a much higher SR (0.6492 & 0.5828) because it varies the viewpoint using the 3DGS map, but has a much lower SPL (0.0589 & 0.0398) because it must generate the map first before searching, resulting in long and inefficient trajectories. SplatSearch on the other hand incrementally builds the map while also performing feature matching, hence achieving the highest performance.

4.4.2 Ablation Study

TABLE IV
ABLATION STUDY

Variants	HM3D-val		HM3D-val-hard	
	SR ↑	**SPL ↑**	**SR ↑**	**SPL ↑**
SplatSearch w/o NVSM	0.3409	0.1893	0.2383	0.1284
SplatSearch w/o VCICN	0.5380	0.2382	0.5291	0.2183
SplatSearch	**0.6983**	**0.3038**	**0.6639**	**0.2703**

We conducted an ablation study with different variants of SplatSearch, on the same HM3D-val and HM3D-val-hard datasets to evaluate the impact of different modules in the architecture. These included:

- **SplatSearch without NVSM:** This method used the RGB image, I_t, directly for feature matching against the goal image.
- **SplatSearch without VCICN:** This method used the NVSM to generate novel viewpoints but did not use multi-view diffusion to inpaint the missing regions of the images.

The results for the ablation study are shown in Table IV. Overall, SplatSearch achieved the highest SR and SPL across all of the variants. SplatSearch w/o NVSM achieved the lowest SR/SPL as it does not use the 3DGS map to render candidate objects from arbitrary viewpoints, hence, it is

unable to recognize the arbitrary goal images. SplatSearch w/o VCICN also achieved a lower SR and SPL. Although it can render objects from multiple viewpoints, the rendered images contained large missing regions, which degraded the feature matcher, which depends on complete image inputs.

4.5 Summary

In this chapter, we presented SplatSearch, a novel architecture for IIN in unknown environments. SplatSearch leverages sparse-view 3DGS reconstructions to synthesize candidate viewpoints around objects, and integrates a multi-view diffusion model to inpaint missing regions of the images for robust goal recognition across arbitrary viewpoints. A frontier policy future combines visual and semantic context from RGB observations to efficiently guide the robot towards the goal image. Through extensive experiments in both photorealistic simulation and real-world environments, we demonstrated that SplatSearch achieves state-of-the-art performance, outperforming existing baselines in the HM3D dataset. An ablation study validated our design choices in our architecture. Real-world experiments showed the generalizability of SplatSearch in indoor environments.

Chapter 5
Conclusion and Future Recommendations

This chapter provides a comprehensive summary of the work presented in this thesis, highlighting its contributions to addressing context-aware navigation domains in dynamic and human-centered environments, regarding robot social navigation and instance-image goal navigation. We then discuss the limitations of our work and recommendations for future research to improve on the limitations.

5.1 Summary of Contributions

5.1.1 Robot Social Navigation Contributions

This thesis has introduced a novel social navigation architecture, OLiVia-Nav, that accounts for both social and environmental context during robot trajectory planning while adapting to new social scenarios. The key achievements include: 1) the development of Social Context Contrastive Language Image Pre-training (SC-CLIP)**,** a novel distillation process that successfully transfers the social reasoning capabilities of large VLMs into two lightweight encoders. These encoders extract social context embeddings from both visual features and semantic text captions to inform trajectory prediction, and 2) the design of a trajectory planning network that uniquely leverages multi-head attention to incorporate these social context embeddings in generating socially compliant trajectories. Importantly, the encoders demonstrated the ability to support online lifelong learning, enabling adaptation to previously unseen social navigation scenarios in real-world settings.

5.1.2 Instance Image Goal Navigation Contributions

This thesis has also developed SplatSearch, a novel 3DGS-based architecture for Instance Image Navigation (IIN), where a robot locates a static object or person from a single reference image in an unknown environment. SplatSearch is the first architecture to incrementally construct a sparse 3DGS representation of the environment to synthesize novel viewpoints of candidate objects, and to employ a multi-view diffusion model to inpaint missing regions for robust goal recognition. The key outcomes of this work include: 1) the introduction of a 3DGS-based IIN method that leverages diffusion-based multi-view synthesis to achieve viewpoint-invariant goal recognition in sparse and

unknown environments, and 2) the design of a frontier exploration strategy that integrates both visual context from synthesized viewpoints and semantic context from prior observations to effectively guide the robot toward the goal image.

5.2 Limitations and Future Work

The limitations of our existing work are discussed below.

Specifically, regarding our robot social navigation architecture, OLiVia-Nav:

- **Limitations**
  - **1:** Each module in our architecture is trained independently.
  - **2:** Our system accounts for positions and trajectories but does not consider higher-level social contexts like gestures and speech that can help guide human navigation.
- **Future Work:**
  - **1:** We can use end-to-end architectures like vision-language action models (VLA) which provide more cohesive training and avoid the issues with module independence.
  - **2:** We can add additional modalities to our training dataset (audio, social cues) to better capture real-world navigation dynamics.

Regarding our limitations for our IIN architecture, SplatSearch:

- **Limitations**
  - **1:** The 3DGS representation increases VRAM and storage size as the scene size increases, which limits the deployment on robots with constrained GPU resources.
  - **2:** The pipeline may fail if the SOIM cannot detect the target class.
- **Future Work:**
  - **1:** We can mitigate the issue of GPU constraints by only mapping regions in the scene that contain the class of the goal object.
  - **2:** We can integrate multiple detection models or hierarchical recognition methods to mitigate SOIM failure cases.

An overall future extension is to combine the two systems to build a cohesive system for image goal navigation that can search and navigate among dynamic humans. This would simply require

adapting the instance image goal navigation architecture to map to account for dynamic obstacles. Another valuable extension would be extending the methodologies to multi-robot systems, where multiple robots aim to navigate among humans while searching for their own independent objects. This would allow for cross-collaboration between the robots and improved efficiency in navigation and finding the desired targets.

References

[1] A. A. Morgan, J. Abdi, M. A. Q. Syed, G. E. Kohen, P. Barlow, and M. P. Vizcaychipi, "Robots in Healthcare: a Scoping Review," *Curr Robot Rep*, vol. 3, no. 4, pp. 271–280, Oct. 2022, doi: 10.1007/s43154-022-00095-4.

[2] A. Fung, A. H. Tan, H. Wang, B. Benhabib, and G. Nejat, "MLLM-Search: A Zero-Shot Approach to Finding People Using Multimodal Large Language Models," *Robotics*, vol. 14, no. 8, p. 102, Jul. 2025, doi: 10.3390/robotics14080102.

[3] C. Getson and G. Nejat, "The adoption of socially assistive robots for long-term care: During COVID-19 and in a post-pandemic society," *Healthc Manage Forum*, vol. 35, no. 5, pp. 301–309, Sep. 2022, doi: 10.1177/08404704221106406.

[4] R. Memmesheimer *et al.*, "Cleaning Robots in Public Spaces: A Survey and Proposal for Benchmarking Based on Stakeholders Interviews," Jul. 23, 2024, *arXiv*: arXiv:2407.16393. Accessed: Sep. 15, 2024. [Online]. Available: http://arxiv.org/abs/2407.16393

[5] P. Bovbel and G. Nejat, "Casper: An Assistive Kitchen Robot to Promote Aging in Place1," *Journal of Medical Devices*, vol. 8, no. 3, p. 030945, Sep. 2014, doi: 10.1115/1.4027113.

[6] K. Black *et al.*, "$π_0$: A Vision-Language-Action Flow Model for General Robot Control," Nov. 13, 2024, *arXiv*: arXiv:2410.24164. doi: 10.48550/arXiv.2410.24164.

[7] S. C. Mohamed, S. Rajaratnam, S. T. Hong, and G. Nejat, "Person Finding: An Autonomous Robot Search Method for Finding Multiple Dynamic Users in Human-Centered Environments," *IEEE Trans. Automat. Sci. Eng.*, vol. 17, no. 1, pp. 433–449, Jan. 2020, doi: 10.1109/TASE.2019.2928774.

[8] S. C. Mohamed, A. Fung, and G. Nejat, "A Multirobot Person Search System for Finding Multiple Dynamic Users in Human-Centered Environments," *IEEE Trans. Cybern.*, vol. 53, no. 1, pp. 628–640, Jan. 2023, doi: 10.1109/TCYB.2022.3166481.

[9] L. Pérez, Í. Rodríguez, N. Rodríguez, R. Usamentiaga, and D. García, "Robot Guidance Using Machine Vision Techniques in Industrial Environments: A Comparative Review," *Sensors*, vol. 16, no. 3, p. 335, Mar. 2016, doi: 10.3390/s16030335.

[10] A. J. Sathyamoorthy *et al.*, "CoNVOI: Context-aware Navigation using Vision Language Models in Outdoor and Indoor Environments," Mar. 22, 2024, *arXiv*: arXiv:2403.15637. Accessed: Apr. 30, 2024. [Online]. Available: http://arxiv.org/abs/2403.15637

[11] C. Mavrogiannis *et al.*, "Core Challenges of Social Robot Navigation: A Survey," Mar. 16, 2021, *arXiv*: arXiv:2103.05668. Accessed: Apr. 04, 2024. [Online]. Available: http://arxiv.org/abs/2103.05668

[12] A. Francis *et al.*, "Principles and Guidelines for Evaluating Social Robot Navigation Algorithms," Sep. 19, 2023, *arXiv*: arXiv:2306.16740. Accessed: Jun. 18, 2024. [Online]. Available: http://arxiv.org/abs/2306.16740

[13] N. Hirose, D. Shah, A. Sridhar, and S. Levine, "SACSoN: Scalable Autonomous Control for Social Navigation," *IEEE Robot. Autom. Lett.*, vol. 9, no. 1, pp. 49–56, Jan. 2024, doi: 10.1109/LRA.2023.3329626.

[14] D. Song, J. Liang, A. Payandeh, X. Xiao, and D. Manocha, "Socially Aware Robot Navigation through Scoring Using Vision-Language Models," Mar. 29, 2024, *arXiv*: arXiv:2404.00210. Accessed: Apr. 05, 2024. [Online]. Available: http://arxiv.org/abs/2404.00210

[15] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher, "VLFM: Vision-Language Frontier Maps for Zero-Shot Semantic Navigation," Dec. 05, 2023, *arXiv*: arXiv:2312.03275. Accessed: Apr. 18, 2024. [Online]. Available: http://arxiv.org/abs/2312.03275

[16] J. Krantz, S. Lee, J. Malik, D. Batra, and D. S. Chaplot, "Instance-Specific Image Goal Navigation: Training Embodied Agents to Find Object Instances," Nov. 29, 2022, *arXiv*: arXiv:2211.15876. doi: 10.48550/arXiv.2211.15876.

[17] E. Stefanini, L. Palmieri, A. Rudenko, T. Hielscher, T. Linder, and L. Pallottino, "Efficient Context-Aware Model Predictive Control for Human-Aware Navigation," *IEEE Robot. Autom. Lett.*, vol. 9, no. 11, pp. 9494–9501, Nov. 2024, doi: 10.1109/LRA.2024.3461552.

[18] S. Pashangpour and G. Nejat, "The Future of Intelligent Healthcare: A Systematic Analysis and Discussion on the Integration and Impact of Robots Using Large Language Models for Healthcare," *Robotics*, vol. 13, no. 8, p. 112, Jul. 2024, doi: 10.3390/robotics13080112.

[19] M. Lisondra, B. Benhabib, and G. Nejat, "Embodied AI with Foundation Models for Mobile Service Robots: A Systematic Review," May 26, 2025, *arXiv*: arXiv:2505.20503. doi: 10.48550/arXiv.2505.20503.

[20] A. H. Tan, S. Narasimhan, and G. Nejat, "4CNet: A Diffusion Approach to Map Prediction for Decentralized Multi-Robot Exploration," Apr. 08, 2025, *arXiv*: arXiv:2402.17904. doi: 10.48550/arXiv.2402.17904.

[21] R. Mirsky, X. Xiao, J. Hart, and P. Stone, "Conflict Avoidance in Social Navigation -- a Survey," Dec. 28, 2022, *arXiv*: arXiv:2106.12113. Accessed: Apr. 05, 2024. [Online]. Available: http://arxiv.org/abs/2106.12113

[22] W. Wang, R. Wang, L. Mao, and B.-C. Min, "NaviSTAR: Socially Aware Robot Navigation with Hybrid Spatio-Temporal Graph Transformer and Preference Learning," Sep. 26, 2023, *arXiv*: arXiv:2304.05979. Accessed: Mar. 31, 2024. [Online]. Available: http://arxiv.org/abs/2304.05979

[23] A. J. Sathyamoorthy, U. Patel, M. Paul, N. K. S. Kumar, Y. Savle, and D. Manocha, "CoMet: Modeling Group Cohesion for Socially Compliant Robot Navigation in Crowded Scenes," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 1008–1015, Apr. 2022, doi: 10.1109/LRA.2021.3135560.

[24] V. Narayanan, B. M. Manoghar, R. P. RV, and A. Bera, "EWareNet: Emotion Aware Human Intent Prediction and Adaptive Spatial Profile Fusion for Social Robot Navigation,"

Mar. 07, 2023, *arXiv*: arXiv:2011.09438. Accessed: Apr. 08, 2024. [Online]. Available: http://arxiv.org/abs/2011.09438

[25] L. Kästner, J. Li, Z. Shen, and J. Lambrecht, "Enhancing Navigational Safety in Crowded Environments using Semantic-Deep-Reinforcement-Learning-based Navigation," 2021, doi: 10.48550/ARXIV.2109.11288.

[26] E. Escudie, L. Matignon, and J. Saraydaryan, "Attention Graph for Multi-Robot Social Navigation with Deep Reinforcement Learning," Jan. 31, 2024, *arXiv*: arXiv:2401.17914. Accessed: Jul. 29, 2024. [Online]. Available: http://arxiv.org/abs/2401.17914

[27] J. Liang, U. Patel, A. J. Sathyamoorthy, and D. Manocha, "Crowd-Steer: Realtime Smooth and Collision-Free Robot Navigation in Densely Crowded Scenarios Trained using High-Fidelity Simulation," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, Yokohama, Japan: International Joint Conferences on Artificial Intelligence Organization, Jul. 2020, pp. 4221–4228. doi: 10.24963/ijcai.2020/583.

[28] J. Xu, W. Zhang, J. Cai, and H. Liu, "SafeCrowdNav: safety evaluation of robot crowd navigation in complex scenes," *Front. Neurorobot.*, vol. 17, p. 1276519, Oct. 2023, doi: 10.3389/fnbot.2023.1276519.

[29] A. H. Raj *et al.*, "Rethinking Social Robot Navigation: Leveraging the Best of Two Worlds," Mar. 09, 2024, *arXiv*: arXiv:2309.13466. Accessed: Apr. 05, 2024. [Online]. Available: http://arxiv.org/abs/2309.13466

[30] B. Panigrahi, A. H. Raj, M. Nazeri, and X. Xiao, "A Study on Learning Social Robot Navigation with Multimodal Perception," Sep. 21, 2023, *arXiv*: arXiv:2309.12568. Accessed: Apr. 05, 2024. [Online]. Available: http://arxiv.org/abs/2309.12568

[31] L. Tai, J. Zhang, M. Liu, and W. Burgard, "Socially Compliant Navigation through Raw Depth Inputs with Generative Adversarial Imitation Learning," Feb. 26, 2018, *arXiv*: arXiv:1710.02543. doi: 10.48550/arXiv.1710.02543.

[32] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *The International Journal of Robotics Research*, vol. 35, no. 11, pp. 1289–1307, Sep. 2016, doi: 10.1177/0278364915619772.

[33] W. Wang, L. Mao, R. Wang, and B.-C. Min, "SRLM: Human-in-Loop Interactive Social Robot Navigation with Large Language Model and Deep Reinforcement Learning," Mar. 22, 2024, *arXiv*: arXiv:2403.15648. Accessed: Mar. 31, 2024. [Online]. Available: http://arxiv.org/abs/2403.15648

[34] L. Mezghani *et al.*, "Memory-Augmented Reinforcement Learning for Image-Goal Navigation," 2021, doi: 10.48550/ARXIV.2101.05181.

[35] X. Lei, M. Wang, W. Zhou, and H. Li, "GaussNav: Gaussian Splatting for Visual Navigation," Mar. 20, 2024, *arXiv*: arXiv:2403.11625. doi: 10.48550/arXiv.2403.11625.

[36] M. Chang *et al.*, “GOAT: GO to Any Thing,” Nov. 10, 2023, *arXiv*: arXiv:2311.06430. Accessed: Nov. 28, 2023. [Online]. Available: http://arxiv.org/abs/2311.06430

[37] H. Wang, A. H. Tan, and G. Nejat, “NavFormer: A Transformer Architecture for Robot Target-Driven Navigation in Unknown and Dynamic Environments,” *IEEE Robot. Autom. Lett.*, vol. 9, no. 8, pp. 6808–6815, Aug. 2024, doi: 10.1109/LRA.2024.3412638.

[38] P. Li, K. Wu, J. Fu, and S. Zhou, “REGNav: Room Expert Guided Image-Goal Navigation,” Feb. 15, 2025, *arXiv*: arXiv:2502.10785. doi: 10.48550/arXiv.2502.10785.

[39] A. Devo, G. Mezzetti, G. Costante, M. L. Fravolini, and P. Valigi, “Towards Generalization in Target-Driven Visual Navigation by Using Deep Reinforcement Learning,” *IEEE Trans. Robot.*, vol. 36, no. 5, pp. 1546–1561, Oct. 2020, doi: 10.1109/TRO.2020.2994002.

[40] Y. Zhu *et al.*, “Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning,” 2016, *arXiv*. doi: 10.48550/ARXIV.1609.05143.

[41] H. Li, Z. Wang, X. Yang, Y. Yang, S. Mei, and Z. Zhang, “MemoNav: Working Memory Model for Visual Navigation,” Mar. 28, 2024, *arXiv*: arXiv:2402.19161. doi: 10.48550/arXiv.2402.19161.

[42] A. Sridhar, D. Shah, C. Glossop, and S. Levine, “NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration,” Oct. 11, 2023, *arXiv*: arXiv:2310.07896. doi: 10.48550/arXiv.2310.07896.

[43] D. Shah, B. Eysenbach, G. Kahn, N. Rhinehart, and S. Levine, “Rapid Exploration for Open-World Navigation with Latent Goal Models,” Oct. 11, 2023, *arXiv*: arXiv:2104.05859. doi: 10.48550/arXiv.2104.05859.

[44] K. Yadav *et al.*, “OVRL-V2: A simple state-of-art baseline for ImageNav and ObjectNav,” Mar. 14, 2023, *arXiv*: arXiv:2303.07798. doi: 10.48550/arXiv.2303.07798.

[45] Q. Wu, X. Gong, K. Xu, D. Manocha, J. Dong, and J. Wang, “Towards Target-Driven Visual Navigation in Indoor Scenes via Generative Imitation Learning,” May 09, 2022, *arXiv*: arXiv:2009.14509. doi: 10.48550/arXiv.2009.14509.

[46] K. Yadav *et al.*, “Offline Visual Representation Learning for Embodied Navigation,” Apr. 27, 2022, *arXiv*: arXiv:2204.13226. doi: 10.48550/arXiv.2204.13226.

[47] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, “Neural Topological SLAM for Visual Navigation,” May 28, 2020, *arXiv*: arXiv:2005.12256. doi: 10.48550/arXiv.2005.12256.

[48] X. Lei, M. Wang, W. Zhou, L. Li, and H. Li, “Instance-aware Exploration-Verification-Exploitation for Instance ImageGoal Navigation,” Mar. 22, 2024, *arXiv*: arXiv:2402.17587. doi: 10.48550/arXiv.2402.17587.

[49] X. Sun, L. Liu, H. Zhi, R. Qiu, and J. Liang, “Prioritized Semantic Learning for Zero-shot Instance Navigation,” Jul. 16, 2024, *arXiv*: arXiv:2403.11650. doi: 10.48550/arXiv.2403.11650.

[50] A. Majumdar, G. Aggarwal, B. Devnani, J. Hoffman, and D. Batra, “ZSON: Zero-Shot Object-Goal Navigation using Multimodal Goal Embeddings,” Oct. 13, 2023, *arXiv*: arXiv:2206.12403. doi: 10.48550/arXiv.2206.12403.

[51] H. Yin, X. Xu, L. Zhao, Z. Wang, J. Zhou, and J. Lu, “UniGoal: Towards Universal Zero-shot Goal-oriented Navigation,” Mar. 18, 2025, *arXiv*: arXiv:2503.10630. doi: 10.48550/arXiv.2503.10630.

[52] Y. Tang, M. Wang, Y. Deng, Z. Zheng, J. Deng, and Y. Yue, “OpenIN: Open-Vocabulary Instance-Oriented Navigation in Dynamic Domestic Environments,” Jan. 08, 2025, *arXiv*: arXiv:2501.04279. doi: 10.48550/arXiv.2501.04279.

[53] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3D Gaussian Splatting for Real-Time Radiance Field Rendering,” Aug. 08, 2023, *arXiv*: arXiv:2308.04079. doi: 10.48550/arXiv.2308.04079.

[54] J. Zheng, Z. Zhu, V. Bieri, M. Pollefeys, S. Peng, and I. Armeni, “WildGS-SLAM: Monocular Gaussian Splatting SLAM in Dynamic Environments,” Apr. 04, 2025, *arXiv*: arXiv:2504.03886. doi: 10.48550/arXiv.2504.03886.

[55] L. Wen *et al.*, “Gassidy: Gaussian Splatting SLAM in Dynamic Environments,” Nov. 23, 2024, *arXiv*: arXiv:2411.15476. doi: 10.48550/arXiv.2411.15476.

[56] T. Chen *et al.*, “Splat-Nav: Safe Real-Time Robot Navigation in Gaussian Splatting Maps,” Apr. 26, 2024, *arXiv*: arXiv:2403.02751. doi: 10.48550/arXiv.2403.02751.

[57] G. Liu, W. Jiang, B. Lei, V. Pandey, K. Daniilidis, and N. Motee, “Beyond Uncertainty: Risk-Aware Active View Acquisition for Safe Robot Navigation and 3D Scene Understanding with FisherRF,” Mar. 18, 2024, *arXiv*: arXiv:2403.11396. doi: 10.48550/arXiv.2403.11396.

[58] R. Jin, Y. Gao, Y. Wang, H. Lu, and F. Gao, “GS-Planner: A Gaussian-Splatting-based Planning Framework for Active High-Fidelity Reconstruction,” May 24, 2024, *arXiv*: arXiv:2405.10142. doi: 10.48550/arXiv.2405.10142.

[59] T. Chen *et al.*, “SAFER-Splat: A Control Barrier Function for Safe Navigation with Online Gaussian Splatting Maps,” Sep. 15, 2024, *arXiv*: arXiv:2409.09868. doi: 10.48550/arXiv.2409.09868.

[60] K. Honda, T. Ishita, Y. Yoshimura, and R. Yonetani, “GSplatVNM: Point-of-View Synthesis for Visual Navigation Models Using Gaussian Splatting,” Mar. 10, 2025, *arXiv*: arXiv:2503.05152. doi: 10.48550/arXiv.2503.05152.

[61] Y. Deng, S. Yuan, G. C. R. Bethala, A. Tzes, Y.-S. Liu, and Y. Fang, “Hierarchical Scoring with 3D Gaussian Splatting for Instance Image-Goal Navigation,” Jun. 10, 2025, *arXiv*: arXiv:2506.07338. doi: 10.48550/arXiv.2506.07338.

[62] W. Meng, T. Wu, H. Yin, and F. Zhang, “BEINGS: Bayesian Embodied Image-goal Navigation with Gaussian Splatting,” Sep. 16, 2024, *arXiv*: arXiv:2409.10216. doi: 10.48550/arXiv.2409.10216.

[63] K. Chen *et al.*, “SLGaussian: Fast Language Gaussian Splatting in Sparse Views,” Dec. 11, 2024, *arXiv*: arXiv:2412.08331. doi: 10.48550/arXiv.2412.08331.

[64] S. Narasimhan, A. H. Tan, D. Choi, and G. Nejat, “OLiVia-Nav: An Online Lifelong Vision Language Approach for Mobile Robot Social Navigation,” in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, Atlanta, GA, USA: IEEE, May 2025, pp. 9130–9137. doi: 10.1109/ICRA55743.2025.11128004.

[65] C. Weinrich, M. Volkhardt, E. Einhorn, and H.-M. Gross, “Prediction of human collision avoidance behavior by lifelong learning for socially compliant robot navigation,” in *2013 IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany: IEEE, May 2013, pp. 376–381. doi: 10.1109/ICRA.2013.6630603.

[66] I. Okunevich, A. Lombard, T. Krajnik, Y. Ruichek, and Z. Yan, “Online Context Learning for Socially-compliant Navigation,” Jun. 17, 2024, *arXiv*: arXiv:2406.11495. Accessed: Jul. 16, 2024. [Online]. Available: http://arxiv.org/abs/2406.11495

[67] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” Apr. 08, 2018, *arXiv*: arXiv:1804.02767. Accessed: Sep. 14, 2024. [Online]. Available: http://arxiv.org/abs/1804.02767

[68] H. Karnan *et al.*, “Socially Compliant Navigation Dataset (SCAND): A Large-Scale Dataset of Demonstrations for Social Navigation,” Jun. 08, 2022, *arXiv*: arXiv:2203.15041. Accessed: Apr. 05, 2024. [Online]. Available: http://arxiv.org/abs/2203.15041

[69] D. M. Nguyen, M. Nazeri, A. Payandeh, A. Datar, and X. Xiao, “Toward Human-Like Social Robot Navigation: A Large-Scale, Multi-Modal, Social Human Navigation Dataset,” Aug. 09, 2023, *arXiv*: arXiv:2303.14880. Accessed: Apr. 05, 2024. [Online]. Available: http://arxiv.org/abs/2303.14880

[70] T. Schreiter *et al.*, “TH\"OR-MAGNI: A Large-scale Indoor Motion Capture Recording of Human Movement and Robot Interaction,” Mar. 14, 2024, *arXiv*: arXiv:2403.09285. Accessed: Sep. 13, 2024. [Online]. Available: http://arxiv.org/abs/2403.09285

[71] W. Chen, O. Mees, A. Kumar, and S. Levine, “Vision-Language Models Provide Promptable Representations for Reinforcement Learning,” Feb. 13, 2024, *arXiv*: arXiv:2402.02651. Accessed: Apr. 12, 2024. [Online]. Available: http://arxiv.org/abs/2402.02651

[72] OpenAI *et al.*, “GPT-4 Technical Report,” Mar. 04, 2024, *arXiv*: arXiv:2303.08774. Accessed: Jul. 29, 2024. [Online]. Available: http://arxiv.org/abs/2303.08774

[73] W. Guo *et al.*, “IGL-Nav: Incremental 3D Gaussian Localization for Image-goal Navigation,” Aug. 01, 2025, *arXiv*: arXiv:2508.00823. doi: 10.48550/arXiv.2508.00823.

[74] S. Zhu, G. Wang, X. Kong, D. Kong, and H. Wang, “3D Gaussian Splatting in Robotics: A Survey,” Dec. 19, 2024, *arXiv*: arXiv:2410.12262. doi: 10.48550/arXiv.2410.12262.

[75] S. Kim, H. Jeon, J. Choi, and D. Kum, “Diverse Multiple Trajectory Prediction Using a Two-stage Prediction Network Trained with Lane Loss,” *IEEE Robot. Autom. Lett.*, vol. 8, no. 4, pp. 2038–2045, Apr. 2023, doi: 10.1109/LRA.2022.3231525.

[76] B. Zhang, P. Zhang, X. Dong, Y. Zang, and J. Wang, “Long-CLIP: Unlocking the Long-Text Capability of CLIP,” May 23, 2024, *arXiv*: arXiv:2403.15378. Accessed: May 29, 2024. [Online]. Available: http://arxiv.org/abs/2403.15378

[77] A. Radford *et al.*, “Learning Transferable Visual Models From Natural Language Supervision,” Feb. 26, 2021, *arXiv*: arXiv:2103.00020. Accessed: Jun. 23, 2024. [Online]. Available: http://arxiv.org/abs/2103.00020

[78] A. Vaswani *et al.*, “Attention Is All You Need,” Aug. 01, 2023, *arXiv*: arXiv:1706.03762. Accessed: Jun. 26, 2024. [Online]. Available: http://arxiv.org/abs/1706.03762

[79] J. Shlens, “A Tutorial on Principal Component Analysis,” Apr. 03, 2014, *arXiv*: arXiv:1404.1100. Accessed: Jul. 29, 2024. [Online]. Available: http://arxiv.org/abs/1404.1100

[80] L. Wang, L. Xiang, Y. Wei, Y. Wang, and Z. He, “CLIP model is an Efficient Online Lifelong Learner,” May 23, 2024, *arXiv*: arXiv:2405.15155. Accessed: Jun. 19, 2024. [Online]. Available: http://arxiv.org/abs/2405.15155

[81] Y. Zhou and O. Tuzel, “VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA: IEEE, Jun. 2018, pp. 4490–4499. doi: 10.1109/CVPR.2018.00472.

[82] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Dec. 10, 2015, *arXiv*: arXiv:1512.03385. Accessed: Jul. 29, 2024. [Online]. Available: http://arxiv.org/abs/1512.03385

[83] S. Casas *et al.*, “DeTra: A Unified Model for Object Detection and Trajectory Forecasting,” Jun. 13, 2024, *arXiv*: arXiv:2406.04426. Accessed: Jun. 15, 2024. [Online]. Available: http://arxiv.org/abs/2406.04426

[84] Kiam Heong Ang, G. Chong, and Yun Li, “PID control system analysis, design, and technology,” *IEEE Trans. Contr. Syst. Technol.*, vol. 13, no. 4, pp. 559–576, Jul. 2005, doi: 10.1109/TCST.2005.847331.

[85] K. Gandhi, J.-P. Fränken, T. Gerstenberg, and N. D. Goodman, “Understanding Social Reasoning in Language Models with Language Models,” Dec. 04, 2023, *arXiv*: arXiv:2306.15448. Accessed: Sep. 15, 2024. [Online]. Available: http://arxiv.org/abs/2306.15448

[86] I. Loshchilov and F. Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts,” May 03, 2017, *arXiv*: arXiv:1608.03983. Accessed: Aug. 23, 2024. [Online]. Available: http://arxiv.org/abs/1608.03983

[87] I. Loshchilov and F. Hutter, “Decoupled Weight Decay Regularization,” Jan. 04, 2019, *arXiv*: arXiv:1711.05101. Accessed: Aug. 23, 2024. [Online]. Available: http://arxiv.org/abs/1711.05101

[88] D. Kraft, “Computing the Hausdorff Distance of Two Sets from Their Signed Distance Functions,” *Int. J. Comput. Geom. Appl.*, vol. 30, no. 01, pp. 19–49, Mar. 2020, doi: 10.1142/S0218195920500028.

[89] N. Hirose, D. Shah, K. Stachowicz, A. Sridhar, and S. Levine, “SELFI: Autonomous Self-Improvement with Reinforcement Learning for Social Navigation,” Mar. 01, 2024, *arXiv*: arXiv:2403.00991. Accessed: Mar. 13, 2024. [Online]. Available: http://arxiv.org/abs/2403.00991

[90] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” Aug. 28, 2017, *arXiv*: arXiv:1707.06347. doi: 10.48550/arXiv.1707.06347.

[91] Y. Li *et al.*, “ActiveSplat: High-Fidelity Scene Reconstruction through Active Gaussian Splatting,” *IEEE Robot. Autom. Lett.*, vol. 10, no. 8, pp. 8099–8106, Aug. 2025, doi: 10.1109/LRA.2025.3580331.

[92] N. Keetha *et al.*, “SplaTAM: Splat, Track & Map 3D Gaussians for Dense RGB-D SLAM,” Apr. 16, 2024, *arXiv*: arXiv:2312.02126. doi: 10.48550/arXiv.2312.02126.

[93] J. Nilsson and T. Akenine-Möller, “Understanding SSIM,” Jun. 29, 2020, *arXiv*: arXiv:2006.13846. doi: 10.48550/arXiv.2006.13846.

[94] J. Li, D. Li, S. Savarese, and S. Hoi, “BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models,” Jun. 15, 2023, *arXiv*: arXiv:2301.12597. doi: 10.48550/arXiv.2301.12597.

[95] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” Jul. 06, 2022, *arXiv*: arXiv:2207.02696. doi: 10.48550/arXiv.2207.02696.

[96] C. Zhang *et al.*, “Faster Segment Anything: Towards Lightweight SAM for Mobile Applications,” Jul. 01, 2023, *arXiv*: arXiv:2306.14289. doi: 10.48550/arXiv.2306.14289.

[97] C. Cao, C. Yu, F. Wang, X. Xue, and Y. Fu, “MVInpainter: Learning Multi-View Consistent Inpainting to Bridge 2D and 3D Editing,” Nov. 19, 2024, *arXiv*: arXiv:2408.08000. doi: 10.48550/arXiv.2408.08000.

[98] Y. Wang, X. He, S. Peng, D. Tan, and X. Zhou, “Efficient LoFTR: Semi-Dense Local Feature Matching with Sparse-Like Speed,” Mar. 11, 2024, *arXiv*: arXiv:2403.04765. doi: 10.48550/arXiv.2403.04765.

[99] S. Garrido, L. Moreno, and D. Blanco, “Exploration of a cluttered environment using Voronoi Transform and Fast Marching,” *Robotics and Autonomous Systems*, vol. 56, no. 12, pp. 1069–1081, Dec. 2008, doi: 10.1016/j.robot.2008.02.003.

[100] K. Yadav *et al.*, “Habitat-Matterport 3D Semantics Dataset,” Oct. 12, 2023, *arXiv*: arXiv:2210.05633. doi: 10.48550/arXiv.2210.05633.

Copyright Acknowledgements