POLITECNICO DI TORINO

MSc in Data Science and Engineering

Select2Plan:
Training-Free ICL-Based Planning through VQA and Memory Retrieval

Supervisors:

Prof. Giuseppe Averta

Prof. Philip Torr

Dr. Daniele De Martini

PhD Tim Franzmeyer

Candidate:

Davide Buoso

October 2024

Declaration of own work

This research was conducted at the University of Oxford, where I was hosted for approximately six months by the Torr Vision Group (TVG). This thesis is the result of a collaborative effort between the Torr Vision Group and the Oxford Robotics Institute. I hereby declare that the work presented in this MSc dissertation has been carried out in full compliance with the University's regulations and requirements.

Davide Buoso

Acknowledgements

First and foremost, I owe everything to my family. Your unwavering support, love, and encouragement have been the foundation upon which all my achievements rest. Without you, none of this would have been possible. You've been my strength through every challenge and my joy in every success.

To my girlfriend, thank you for being with me in the darkest of moments. Your patience, understanding, and constant belief in me have inspired me to become the best version of myself. You gave me the courage to chase my dreams, and with you by my side, I felt truly free to express who I am. You are my greatest motivation, and for that, I am endlessly grateful.

Lastly, I would like to extend my heartfelt thanks to my supervisors. You granted me the opportunity to grow both personally and professionally. You gave me the opportunity to undertake a piece of research in one of the most remarkable universities in the world. The experiences and lessons I've gained under your guidance have been invaluable, and I will carry them with me throughout my life and career. Your mentorship has been an incredible gift.

Abstract

This study explores the potential of off-the-shelf Vision-Language Models (VLMs) for high-level robot planning in the context of autonomous navigation. Indeed, while most of existing learning-based approaches for path planning require extensive task-specific training/fine-tuning, we demonstrate how such training can be avoided for most practical cases. To do this, we introduce Select2Plan (S2P), a novel training-free framework for high-level robot planning which completely eliminates the need for fine-tuning or specialised training. By leveraging structured Visual Question-Answering (VQA) and In-Context Learning (ICL), our approach drastically reduces the need for data collection, requiring a fraction of the task-specific data typically used by trained models, or even relying only on online data. Our method facilitates the effective use of a generally trained VLM in a flexible and cost-efficient way, and does not require additional sensing except for a simple monocular camera. We demonstrate its adaptability across various scene types, context sources, and sensing setups. We evaluate our approach in two distinct scenarios: traditional First-Person View (FPV) and infrastructure-driven Third-Person View (TPV) navigation, demonstrating the flexibility and simplicity of our method. Our technique significantly enhances the navigational capabilities of a baseline VLM of approximately 50% in TPV scenario, and is comparable to trained models in the FPV one, with as few as 20 demonstrations.

Contents

List of Figures

List of Tables

1 Introduction

Traditionally, autonomous systems rely on a rich array of onboard sensors, such as LiDAR, sonar, and stereo cameras, to perceive their environment, make decisions, and navigate safely. These sensors provide a detailed, multi-modal understanding of the surroundings, enabling precise navigation even in complex and dynamic environments. However, the reliance on such comprehensive sensory input presents limitations, particularly in scenarios where weight, cost, or environmental constraints make the use of extensive sensors impractical. Path planning for vehicles is a longstanding problem in robotics, traditionally addressed using model-based or Reinforcement Learning (RL) approaches [1]–[3]. However, methods that directly learn from experience often struggle when confronted with ambiguous or unfamiliar scenarios. Interestingly, recent research has shown that Large Language Models (LLMs) and Vision Language Models (VLMs) demonstrate surprising reasoning capabilities that can be adapted for proposing robot paths in arbitrary scenes [4]. Indeed, these models excel at incorporating common-sense reasoning acquired during their long pretraining phase [5]. This ability is crucial in robotics operations, where the deployment scenario rarely aligns perfectly with the training dataset [6], [7]. While methods like LoRA [8] reduce the computational cost of fine-tuning LLMs and VLMs, they still require domain-specific data, which can be costly to obtain. In parallel, In-Context Learning (ICL) and Retrieval-Augmented Generation (RAG) have shown promising results in scoping the ability of LLMs *at deployment time* with no additional fine-tuning, mitigating these costs.

Our novel framework – Select2Plan (S2P) – combines Visual Question-Answering (VQA) and ICL with VLMs in a training-free manner, showing remarkable flexibility across various scenes, contexts, and setups. More specifically, we formulate the planning problem as a VQA task using visual prompting. A high-level overview of the approach can be observed in 1.1. Inspired by [9] and [10], we generate a set of position candidates in the image space and use them as part of a query mechanism to a VLM, to extract the next robot move. We combine this approach with ICL to enhance the model's reliability: we retrieve similar successful samples and use them, along with the current annotated

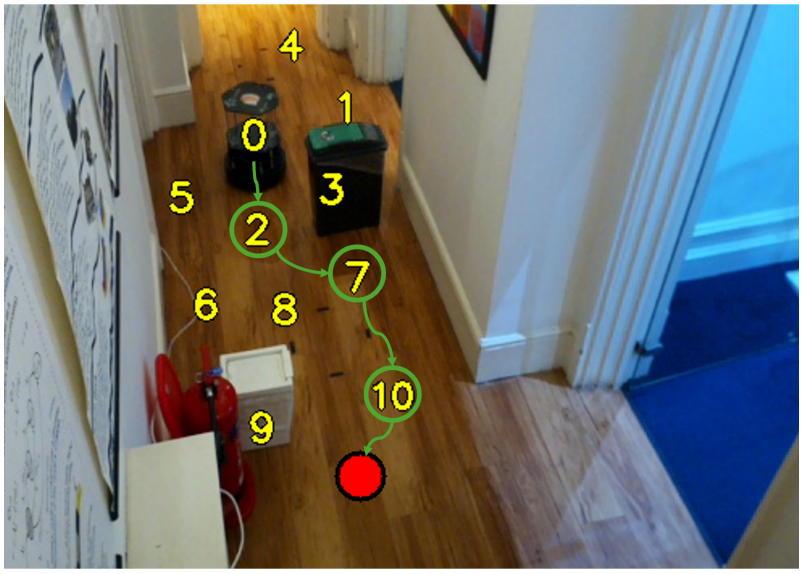


Figure 1.1: High-level demonstration of S2P in a TPV scenario. The robot must reach the red mark from its location, controlled solely via the external camera, shown in the figure. S2P proposes candidate keypoints – in yellow – and draws them into the original image before requesting a feasible trajectory to an off-the-shelf VLM. The latter will output a trajectory – green – as a sequence of keypoints, ideally yielding a trajectory that avoids obstacles – e.g. 3 and 9.

image, as context to support the model's generalization. In this way, we can generate a robust path, which can span multiple planning steps within a single response, in contrast to the iterative approach taken by [9].

We evaluate our system in two different navigation scenarios. The first is a more traditional FPV, where the robot is equipped with a monocular camera and needs to reach specific objects in the scene. The challenge in this case is the sensor's limited view, as the goal object might get out of view while the robot navigates the environment. As a second test-bench, we consider a robot controlled through eye-to-hand visual servoing, as in [11], [12] and as depicted in 1.1. Here, the camera is not physically attached to the robot, and the far viewpoint inherently limits the depth [13] and spatial resolution. However, given the widespread use of CCTV cameras, we believe this approach offers new opportunities and, interestingly, this setup also mirrors the type of data that VLMs are trained on – static RGB images paired with textual descriptions – making these models well-suited for tasks involving external camera navigation. We show how our setup can flexibly adapt to both visual inputs and diverse sources of context, such as videos from the Internet or even human traversal of the scenario.

To summarise, our main contributions are:

1. A framework for planning and navigation using only RGB data, leveraging structured VQA, ICL and retrieval techniques to reduce the task-related data needed to just a handful of episodes.

2. The application of this framework to two separate scenarios: traditional FPV navigation and infrastructure-driven TPV navigation.

3. An extensive analysis of the impact of different sources of in-context examples on the system's overall performance.

Our empirical analysis demonstrates that our approach enhances the navigational abilities of VLMs without requiring further training, and lays the groundwork for more sophisticated and flexible planning in autonomous systems. To the best of our knowledge, our approach is the first that can seamlessly adapt to multiple setups *and* utilise multiple sources of in-context samples.

2 Background

Our research lies at the confluence of robot navigation, planning, Large Language Models (LLMs), Vision Language Models (VLMs), and In-Context Learning (ICL). This review examines the most pertinent literature, highlighting both commonalities and distinctions while emphasizing the novel aspects of our methodology.

2.1 Robotic Navigation: Overview

Conventional approaches to robot navigation tasks, including Object Navigation and Visual Navigation, have predominantly relied on Reinforcement Learning (RL) to develop policies for complex scenarios. The majority of these methods employ deep RL algorithms that process visual inputs to guide robots through various environments. Seminal works such as [1]–[3] utilized deep RL algorithms to train agents capable of navigation using high-dimensional sensory data, effectively leveraging vision for locomotion. While these approaches successfully enabled robots to learn navigation through trial and error, they also addressed challenges related to dynamic environments, object detection, and localization. However, deep RL models generally suffer from limited sample efficiency; their dependence on training in specific scenarios often results in poor generalization to novel, previously unseen environments. Moreover, most RL-based methods typically require extensive training periods and numerous interactions with the environment, rendering them computationally expensive. In response to these challenges, recent research has shifted focus towards transformer-based models for robot navigation. Initially gaining prominence in Natural Language Processing (NLP), transformers have found application in robotics due to their superior ability to model long-range dependencies compared to traditional Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for sequential data. These models have demonstrated enhanced generalization capabilities for navigation policies, particularly in tasks involving the comprehension and processing of spatial and temporal relationships across large-scale environments. A more detailed examination of these models will be presented in the subsequent section.

2.2 Transformers, Large Language and Vision Language Models

2.2.1 Introduction to Transformers and derived models

The transformer architecture, introduced by [14] in the paper "Attention is All You Need", is a radical departure from familiar sequence-based neural networks such as Recurrent Neural Networks (RNNs) and Long-Short Term Memory (LSTM). Transformers depend on a specific mechanism that could be referred to as self-attention, where it can weigh dynamically against parts of the input sequence that are less relevant. This self-attention mechanism proceeds to process all the tokens in parallel, which allows the model to capture long-range dependencies and relationships of context without requiring the processing to be sequential. Fundamentally, the Transformer architecture is based on an encoder-decoder model that separates input encoding and output decoding into two different processes. While the former is normally regarded as transforming input sequence into a vector of fixed length, the latter generates the output sequence from that vector. This architecture allows the training of the model jointly to maximize the conditional log-likelihood of the output given the input for any kind of task, such as machine translation and text generation [15]. Figure 2.1 offers a high-level overview of the model. The encoder consists of a number of layers chained together, with each comprising a multi-head attention mechanism and a feed-forward neural network. With such a design, the encoder is able to extract invariant features from an input sequence while processing all elements in parallel, thus overcoming the unavailability of parallelization inherent in RNNs, which process data in sequence. Each position within the sequence is further augmented not only with word meaning through embedding and position encoding layers but also with positional information, which is combined for further processing. While the encoder primarily generates contextualized representations with a host of useful information, the decoder fundamentally generates sequences based on these contextualized representations received from the encoder. Similar to the encoder, the decoder is also based on multi-head self-attention and feed-forward neural network layers that help the model in coherence, preserving contextual relevance across the generated output.

Attention will be the backbone of the Transformer model, featuring scaled dot-product

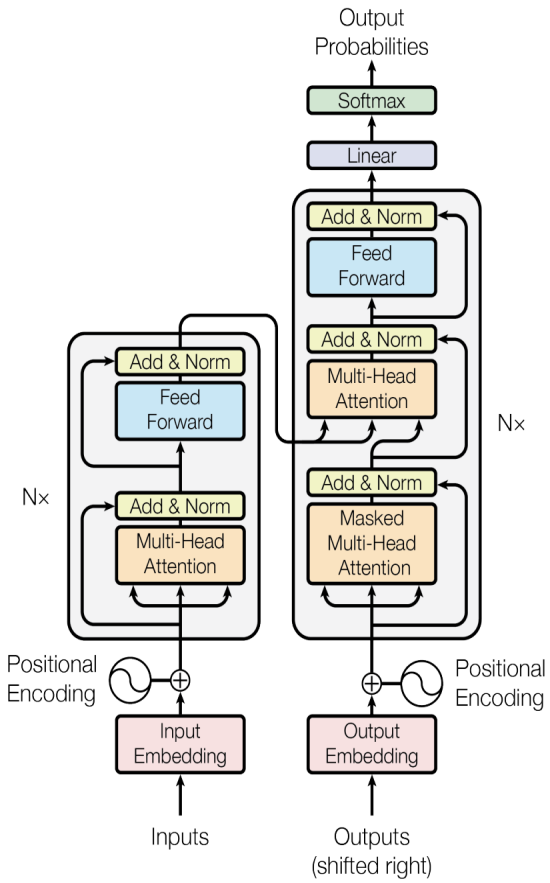


Figure 2.1: Architecture of a Transformer model: The structure consists of an encoder-decoder framework. The encoder processes input sequences by applying multi-head self-attention mechanisms and position-wise feed-forward layers, capturing long-range dependencies. The decoder, similarly structured, generates output sequences while attending to both the input through encoder-decoder attention and its own past outputs. Layer normalization and residual connections are used throughout to stabilize and improve training. *Figure and caption taken from original paper [14].*

attention and multihead attention. Scaled dot-product attention computes attention scores through a query, key, and value mechanism whereas multi-head attention applies the above process in parallel across various representation subspaces. This allows it to jointly pay attention to information from multiple aspects of the input.

The resulting formula to compute attention is reported in Equation 2.1.

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

The Transformer takes advantage of position encoding, which was very helpful in capturing the order of words within a sequence. This enables the model to identify exactly

which position is occupied by each word, when that would often be key to discerning context and meaning in a flow of text. Sinusoidal position encodings are recommended due to their simplicity in being able to capture relative positions and understand lengths of sequences longer than what it was trained with.

Modern LLMs have the transformer architecture and are trained on very large corpora of text data. These include models like OpenAI's GPT-3 [16] and Google's BERT [17], which have raised the bar for state-of-the-art natural language understanding and generation. GPT-3, for example, has 175 billion parameters and is able to generate highly coherent and contextually relevant text with the given prompt. While BERT is an advance to address contextual understanding from both directions of sentences and therefore very impressive for tasks such as question-answering and named entity recognition.

Pre-training for LLMs starts from the training of large corpora to learn representations of language, followed by fine-tuning for a special task in order to adapt these representations to a particular application. The notable size of an LLM combined with enormous training data imparts it with remarkable capability related to generalization across a wide range of language tasks and domains.

Besides the improvement in language processing, through Vision Transformers (ViTs), the transformer architecture has also been extended into the realm of computer vision. Vision Transformers represent a break from traditional convolutional neural networks - CNNs - in an adaptation of this model to image data. Introduced by [18], in Vision Transformers, image patches are considered tokens, just like words in an ordered text sequence. These ViTs divide the image into non-overlapping fixed-size patches, flatten them, and then linearly project these into a lower dimensionality space to obtain patch embeddings. These are combined with positional encodings to retain certain spatial information. The sequence of embeddings, which results from this process, is fed into a standard transformer encoder. The overview of ViT is presented in Figure 2.2. This approach enables the Vision Transformers to capture global relationships of an image by utilizing the self-attention mechanism, thus offering a strong alternative to locally limited receptive fields of CNNs.

Important advantages of Vision Transformers are long-range dependencies in a picture they capture, thus enhancing performance within different vision tasks. ViTs achieved competitive image classification benchmarks for the further extension of this approach to

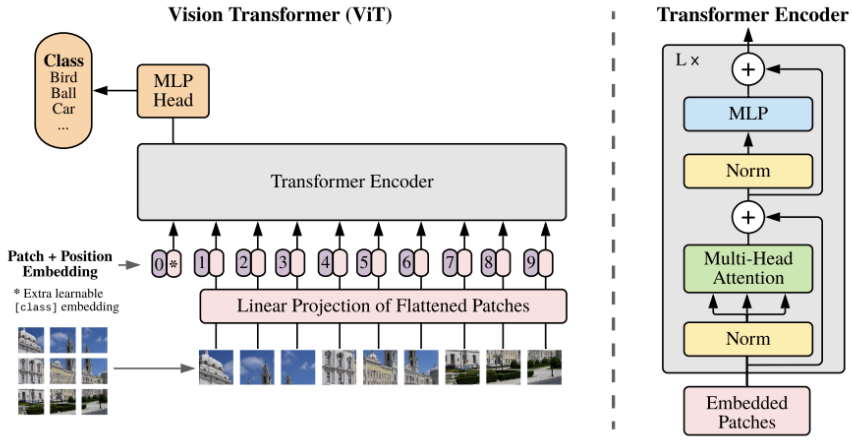


Figure 2.2: An illustration of the Vision Transformer (ViT) architecture: input images are split into fixed-size patches, linearly embedded, and combined with positional encodings. These embeddings pass through multiple transformer layers, with self-attention mechanisms capturing global relationships, before final classification by a fully connected layer. *Figure and caption taken from original paper [18].*

object detection and image segmentation.

The transformer architecture had first revolutionized natural language processing, computer vision, among a few other domains, by outstanding breakthroughs in sequence-to-sequence tasks. Large Language Models, built on transformers, reached the state-of-the-art in understanding and generating human languages. Moreover, Vision Transformers showed promising results by applying transformer principles to image data. Together-the emergence of such wide and challenging tasks across different domains-will testify to the versatility and power of transformers.

These models began to be used more and more in Robotics and in Navigation especially, for their notable adaptability. For instance, [19], introduced an Object Memory Transformer model designed to handle Object Navigation tasks, showcasing improved memory retention and decision-making abilities. This model allows robots to maintain a more robust memory of objects and their locations, resulting in more efficient navigation strategies. [20] further expanded on transformer-based models with Visual Navigation Transformer (ViNT), a framework designed to handle diverse visual navigation tasks by capturing long-term visual features more effectively than traditional RL methods. [21] introduced NoMaD, a navigation framework that leverages transformers to facilitate enhanced spatial awareness and adaptability in unseen environments, underscoring the

model's potential to outperform RL-based systems in generalization and long-term planning.

These latest advancements in transformer-based models mark an important milestone in robot navigation, traditionally plagued by limitations of deep RL methods. Applying their capability for long-range dependency modeling and memory mechanisms, transformer models are emerging as promising models for robot navigation tasks that address requirement scalability, robustness, and generalization across diverse and complex environments.

2.2.2 Evolution of Vision Language Models

Visual Language Models represent the evolution of multimodal learning where models integrate and process both visual and textual data. There was originally an obvious tendency to propose the use of CNNs for the processing of visual data with transformers for textual data analysis. After that, there were more and more proposals focused on the joint training for visual and textual data using two different branches—one for each modality—followed by a network responsible for fusion. Some of the first examples of VLMs were indeed those to have emerged in the mid-2010s, and two of the most successful instances being [22] and [23]. They proved to be successful due to their basic principle, which is the backbone of functionality within the VLMs themselves: enabling the model effectively to communicate between its visual and textual representations by aligning image embeddings from a visual backbone with that of a text backbone. However, the area did not take off as expected then due to the lack of big datasets and powerful architectures. Starting from these rather simpler models, the development of the field lead to huge architectures with billions (and even trillions) of parameters. The companies operating in this domain moved either to proprietary methods or open-source philosophy.

2.2.3 VLM Architectures

Rather recent developments introduced models such as CLIP—Contrastive Language-Image Pretraining [24] by OpenAI. A pretraining inspired by CLIP relies on contrasting an image and its caption, aiming to align both in one embedding space. That was the capability that realized zero-shot classification tasks enabled through the comparison of an image directly with textual labels. Training CLIP on a huge dataset of image-caption

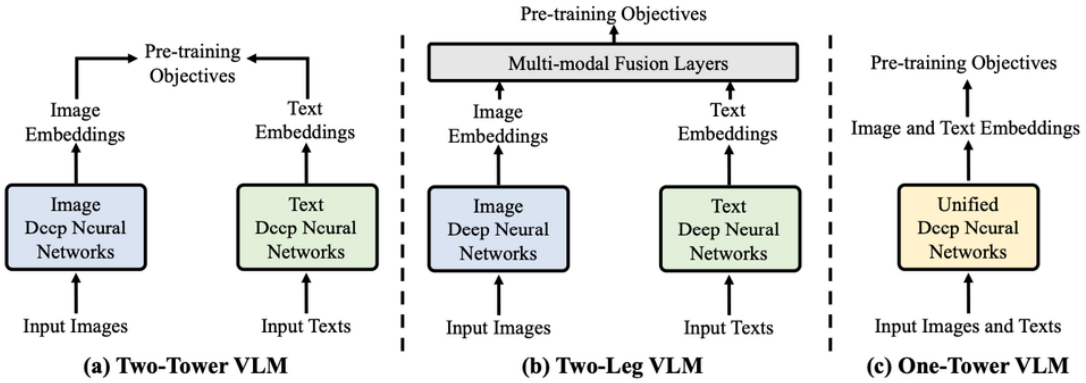


Figure 2.3: Overview of the main VLMs pre-training architectures. *Figure and caption taken from [25].*

pairs enables generalization across a wide array of visual and linguistic tasks without task-specific fine-tuning. Being the architecture public, researchers had the opportunity to start from CLIP and create more and more advanced models.

One of the key aspects of an effective VLM is how to combine image embeddings into the text embedding space. The architectures can usually be divided into three macro categories:

1. Two-Tower VLM where the connection among Vision and Text embeddings is at the final stage. CLIP is the classic example of this approach.

2. Two-Leg VLM where a single LLM takes text tokens along with tokens from Vision Encoder.

3. Unified VLM where the backbone is attending to Visual and Textual inputs at the same time.

The main architectures are reported in Figure 2.3.

In particular, as for techniques for combining visual and textual information, it is possible to identify strategies such as Shallow/Early, Late, and Deep Fusion [25].

Early Fusion These models all have in common that the interaction between visual inputs and language happens early on. From a high level, this approach usually does not process the visual data much before feeding it into a stage that processes text. Thus, it is also known as "shallow" fusion. Given a well-aligned vision encoder, this can easily support multiple image inputs, something even far more advanced models often struggle

with. Because of this, many approaches take the path of early fusion. Two of the major early fusion methods are discussed below.

Vision Encoder This methodology represents one of the more straightforward approaches to integrating visual and linguistic processing. A crucial step in this approach is ensuring compatibility of the outputs of the vision with the input of an LLM, while training only the vision encoder, hence keeping the latter frozen. The architecture is essentially a decoder-only transformer with an additional branch for the image encoder. This is relatively easy to implement, intuitive, and usually does not require the introduction of any additional layers. These architectures share the same final objective as LLMs: maximize the accuracy of next-token prediction. One of these works, [26], extends the training of the vision encoder by prefix tuning. They operate by appending a static token to all visual inputs, which allows the vision encoder to contextually alter its output based on the response of the LLM to the prefix. A short summary of their approach is given in Figure 2.4.

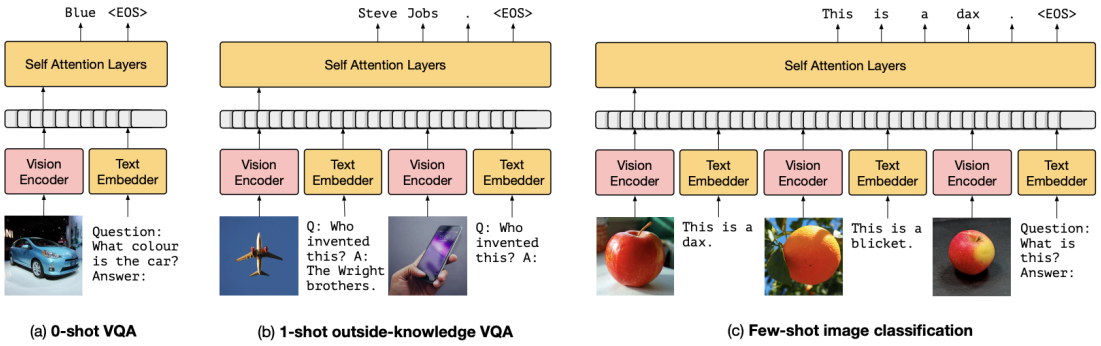


Figure 2.4: Frozen Inference Interface. The figure demonstrates how Frozen supports: (a) Visual Question Answering, (b) Outside-Knowledge Question Answering and (c) Few-Shot Image Classification via In-Context Learning. *Figure and caption taken from original paper [26].*

Vision Projector/Adapter The primary disadvantage of relying exclusively on a vision encoder is the fact that guaranteeing its output will be directly compatible with the LLM is challenging, which restricts potential combinations of vision encoders and LLMs. This can be done in a more flexible way, by introducing an intermediary layer between these two networks that rescales the output of the vision encoder to match the expected input by LLM. You can use a projector in order to align any vision embeddings with

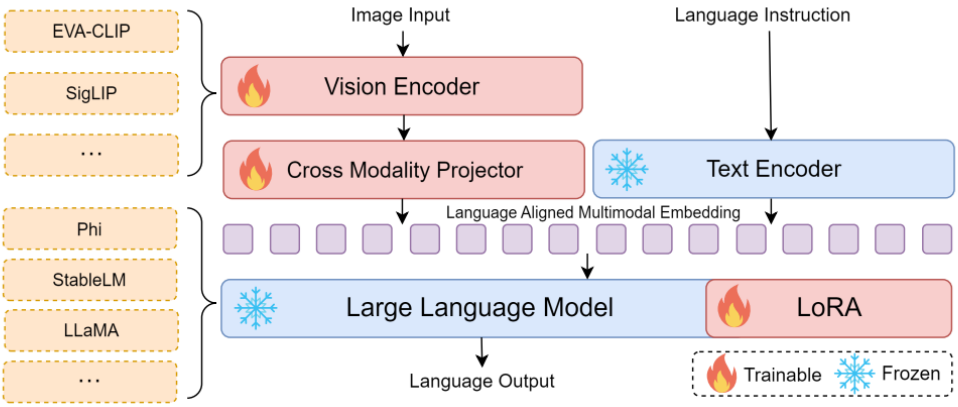


Figure 2.5: Overview of Bunny [29] architecture. It offers a flexible choice of vision encoder and LLM backbone combination, which are aligned through the cross-modality projector.*Figure and caption taken from original paper [29]*

any LLM. That's more flexible than simply training the vision encoder. The other option is freezing both the vision and LLM networks; this speeds up training since adapters are typically small in size. The projectors can be as simple as a multi-layer perceptron (MLP), consisting of several linear layers interspersed with non-linear activation functions. Some examples of models using this approach are:

1. **LLaVa family of models** ([27] [28]) - The architecture, which appears straightforward at first glance, has gained recognition for its focus on training utilizing high-quality synthetic data (and for its performance).

2. **Bunny** [29] - An architecture which supports several vision and language backbones. It uses [8] to train LLMs component in an efficient way. Overview presented in 2.5.

3. **BLIP-2** [30] utilizes a Q-Former [31] as its adapter for stronger grounding of content with respect to images.

4. **DeepSeek-VL** [32] uses instead multiple encoders to preserve both high-level and low-level details in the image. However, their approach is also based on exploiting "Deep Fusion" which we will explore further later in the section.

Late Fusion In these architectures, the vision and text models remain completely separate, with their embeddings only being combined during the loss calculation. Typically,

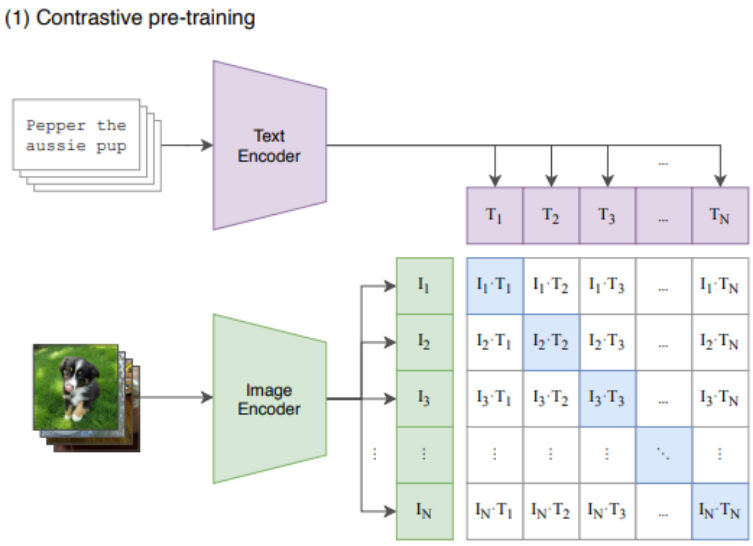


Figure 2.6: Overview of CLIP working. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. *Figure and caption taken from original paper [24]*

this involves the use of contrastive loss and [24] is the main example. Here text and image are encoded separately and are compared via contrastive loss to adjust the encoders. In Figure 2.6 is reported an overview of the architecture.

Deep Fusion These architectures usually attend to image features in the deeper layers of the network allowing for richer cross modal knowledge transfer. Typically the training spans across all the modalities. They, hence, require more time to train but may offer better efficiency and accuracies. Sometimes the architectures are similar to Two-Leg VLMs with LLMs unfrozen:

1. **GroudingDINO** [33] - uses Localization Loss on a cross-modality transformer to perform zero-shot object detection, i.e, predict classes that were not present in training. The architecture overview is reported in Figure 2.7

2. **Chameleon** [34] - treats images natively as tokens by using a quantizer leading to text-vision agnostic architecture.

3. **Flamingo** [35] - The vision tokens are computed with a modified version of ResNET [36] and from a special layer called "Perceiver Resampler", which works as DETR

[37]. It then uses dense fusion of vision with text by cross-attending vision tokens with language tokens using a frozen LLM.

4. **MoE-LLaVa** [38] - uses the Mixture of Experts technique to handle both vision and text tokens. It trains the model in two stages where only the FFNs are trained first and later the LLM.

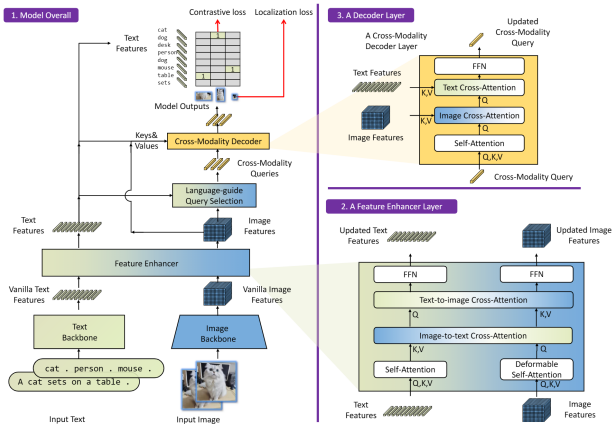


Figure 2.7: The framework of Grounding DINO. In the figure they present: the overall framework, a feature enhancer layer, and a decoder layer in block 1, block 2, and block 3, respectively. *Figure and caption taken from original paper* [33].

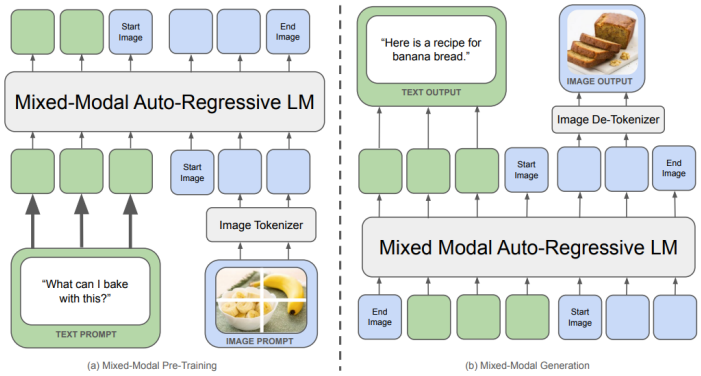


Figure 2.8: Chameleon overview. It represents all modalities: images, text, and code, as discrete tokens and uses a uniform transformer-based architecture that is trained from scratch in an end-to-end fashion on about 10T tokens of interleaved mixed-modal data. As a result, Chameleon can both reason over, as well as generate, arbitrary mixed-modal documents. Text tokens are represented in green and image tokens are represented in blue. *Figure and caption taken from original paper* [34]

2.2.4 VLM Training

Training a VLM is a very complex process that can make use of multiple objectives, each aimed at performance improvement on various tasks. Below, we review some of the common objectives during both training and pre-training of VLMs.

Contrastive Loss The model learns from this objective to bring closer the embedding of related pairs and push further apart the embedding of unrelated pairs. This finds broad application because the gathering of matching pairs (e.g., image-text pairs) is relatively simple, and training can take full advantage of a large number of negative (unrelated) samples. One popular example is CLIP, where they used contrastive loss to align the embeddings of images and text. [39] also use this approach. [40] further enhances this by pre-training the vision encoder first using an image-to-image contrastive loss, which results in better performance when fine-tuned on an image-text task. Another enhancement of the contrastive loss is done by [41], who incorporate image labels and text hashes, called Unified-CL. In contrast, [42] apply two contrastive losses: one on the alignment of images and text and one for the alignment of text.

Generative Loss Generative loss treats the VLM as a generator; it is commonly utilized for tasks such as zero-shot learning and language generation. One of the common forms is Language Modeling Loss, meant to predict the next token in a sequence. [34] takes this even further by employing the same loss to predict image tokens, while [43] applies this across all its tasks. In Masked Language Modeling (MLM), the model learns to predict masked text tokens given their context. It is best exemplified in the models like [44]. Masked Image Modeling (MIM) involves masking parts of an image and training the model on the estimation of these deleted patches. These models using this include [45] amongst several others, while [46] and [47] are a few of the models that use the Masked Autoencoder approach (MAE) [48]. In Figure 2.10 is reported the training and architecture overview of FLAVA. A more integrated approach is Masked Image+Text Modeling, whereby both image and text tokens are masked together, ensuring the model captures cross-domain interactions efficiently. [47] exemplifies this combined method.

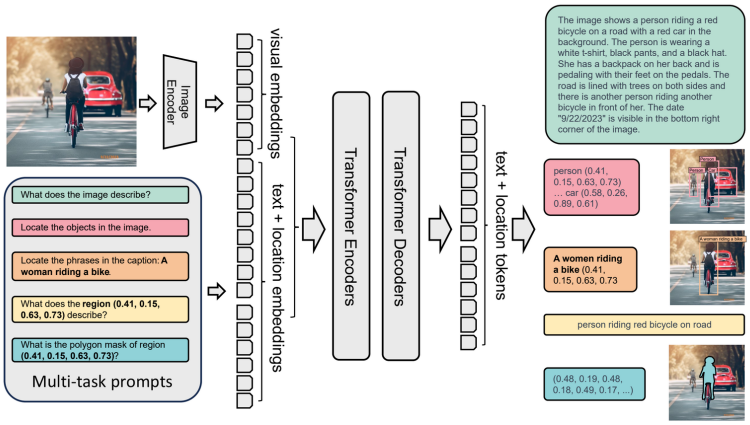


Figure 2.9: Florence-2 Architecture. Florence-2 consists of an image encoder and standard multi-modality encoder-decoder. It is trained on FLD-5B data in a unified multitask learning paradigm, resulting in a generaslist vision foundation model, which can perform various vision tasks. *Figure and caption taken from original paper* [43]

.

Specialized Cross-Modality Alignments For instance, [30] presents an Image-Grounded Text Generation loss for generating text highly consistent with the input image. On the other hand, [45] proposes the Word Patch Alignment method to roughly locate words in the layout of a document. This becomes particularly significant when tasks require the model to discern the spatial arrangement of text and images, for instance, document processing. This goal makes VLMs understand deep relationships between images and texts. These make them accomplish a variety of tasks ranging from captioning, text generation to image understanding.

Training Best-Practices Training VLMs consists of not only choosing appropriate objectives but also best practices to optimize performance such as effective usage of pre-training, fine-tuning, instruction tuning, and handling complicated visual inputs [25].

Pre-training The traditional first step is a process called pre-training, where only a part of the model-for instance, the adapter or projector layer-is trained on large-scale datasets comprising millions of image-text pairs. In all, the goal of this process is to align the image encoder with the text decoder. Most pre-training is unsupervised and based on objectives such as contrastive loss or next-token prediction. One important aspect, though, is the size of the data: enormous datasets ensure that the model will learn general

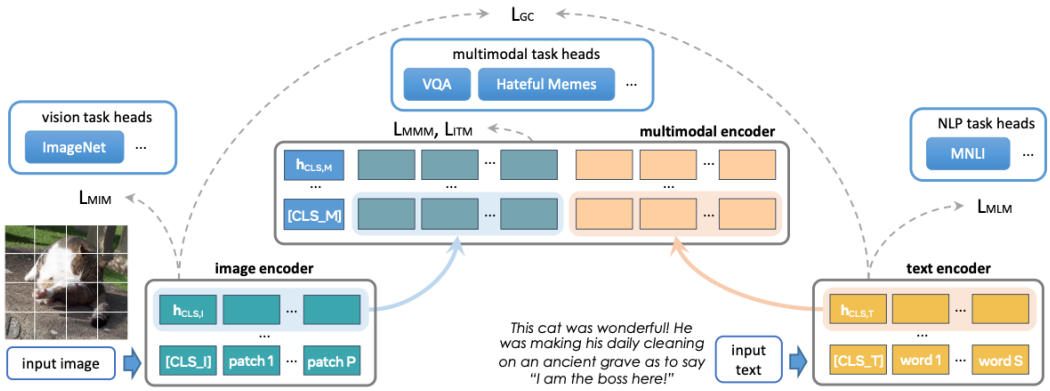


Figure 2.10: FLAVA model overview. The FLAVA model consists of three main components: an image encoder transformer for unimodal image representations, a text encoder transformer for unimodal text processing, and a multimodal encoder transformer that integrates the image and text representations for multimodal reasoning. During pretraining, the model uses masked image modeling (MIM) and masked language modeling (MLM) for individual image and text inputs, while contrastive, masked multimodal modeling (MMM), and image-text matching (ITM) losses are applied to paired image-text data. For downstream tasks, classification heads are added to each encoder for visual recognition, language understanding, and multimodal reasoning tasks. *Figure and caption taken from original paper* [47].

relations between images and text. Text prompts are also often modified to make sure that in this process, the model really captures the context of the image using the language.

Fine-tuning Fine-tuning follows pre-training as the next stage. Again, this will depend on the architecture of the model, but several components may be unfrozen and retrained: the adapter, text, and vision encoders. Specialization allows the model to specialize further at an increased cost regarding complexity-that is, more parameters are now trainable. Fine-tuning is generally slow since it involves a high number of parameters; thus, it usually employs a smaller dataset that is more refined. That would also ensure the data being utilized here is of better quality and task-specific, hence allowing the model to focus on performance improvement over specialized tasks.

Instruction Tuning Instruction tuning tends to happen in a post-training, fine-tuning kind of process for models trying to achieve the status of chatbots. At that point in time, training data is transformed into some type of instructional format, which usually includes a large language model. This comes in handy for adapting the VLM to execute tasks that

require the model to follow instructions-a typical application for conversational AI. That is, this will make it more responsive and capable of handling human-like interaction.

Efficient Training/Finetuning with LoRA Fine-tuning of all the parameters is computationally expensive for a large-scale model. LoRAs are techniques that can be used to make training more efficient by reducing the number of trainable parameters. This train method updates smaller task-specific layers, and not the whole model. This saves much time and computing resources yet still enables fine-tuning on specialized tasks.

The result is that these combined best practices in the training of VLMs yield models that are generalizable yet capable of handling specialized tasks across various domains.

2.2.5 Proprietary Models (Closed Source)

The evolution of VLMs has reached a new pinnacle with the creation of large-scale Vision-Language Models like OpenAI's GPT-4V [49], Google's Gemini [50] and Claude 3 [51], among all. These models represent the state of the art for what concerns VLMs. Given the proprietary nature of these models, direct access to their precise architectural specifications and parameters remains restricted. The GPT-4V [49] is an extension of the GPT-4 architecture that supports visual inputs in addition to textual data. It leverages the powers of large-scale language modeling in GPT-4 by further introducing complex visual-processing mechanisms into its architecture. GPT-4V is trained on large-scale datasets that include, but are not limited to, images with textual descriptions for highly detailed and contextually appropriate descriptions of visual scenes, solving complex visual questions, and performing advanced joint visual-textual interactions. It follows a unified architecture where the visual and textual streams are processed in a shared representation space, thus allowing for seamless integration and mutual enhancement of both modalities.

In April 2024, Google introduced an updated version of Gemini Pro [52], marking a significant advancement in Vision-Language Models (VLMs). This new iteration leverages a highly efficient transformer-based architecture, specifically optimized for multimodal tasks. Gemini Pro 1.5's key innovation lies in its implementation of advanced techniques, particularly Ring Attention, which proves crucial for expanding the model's context window. Ring Attention represents a novel approach to the attention mechanism, designed to balance computational efficiency with the capture of important local depen-

dencies. Unlike traditional full self-attention mechanisms, Ring Attention restricts its focus to a limited range of neighboring positions. This targeted approach significantly reduces computational complexity, making it particularly suitable for tasks involving long sequences where local context holds greater relevance. The essence of Ring Attention can be understood through its formula, which applies a locality constraint to the conventional attention calculation. This modification allows Gemini Pro 1.5 to process an impressive 1 million tokens of context – a substantial increase over its predecessors and competitors. This technique's implementation in Gemini Pro 1.5 not only showcases Google's continued innovation in the field but also highlights the ongoing efforts to enhance AI models' ability to handle and understand vast amounts of contextual information across multiple modalities.

Claude 3 [51] is a suite of three models released in 2024 by Anthropic. The main differences introduced are the use of the Sparse Transformer Architecture: a novel sparse transformer architecture, which introducing sparsity into the attention mechanism, makes it far more efficient. This approach hence significantly reduces the computational complexity and memory requirements, enabling the model to scale to unprecedented sizes while maintaining challenging performance. Another strong point is the introduction of the Reversible Transformer Layers, which allow for efficient back-propagation during training. This innovation allows to further reduction of the memory footprint and enables training on larger datasets, leading to overall better performance.

These models signify the advancement in joint representation learning and sophisticated handling of diverse multimodal data. They demonstrate a deepened understanding and generation capability across both vision and language domains, achieving significant gains in generalization and task performance, thereby setting new benchmarks in the field of Vision-Language Models. A major disadvantage of these models is the requirement for a lengthy and expensive training phase. Consequently, recent literature, including this work, has introduced a new trend: exploiting these models with prompt-engineering techniques to circumvent additional fine-tuning. Another limitation is their closed-source nature, which precludes direct hosting and further study. However, in this work, we transform this constraint into an advantage by utilizing APIs, thus eliminating the need for powerful hardware.

2.2.6 Benchmarks for Vision Language Models

This chapter highlights the most important benchmarks that could be used in the evaluation of VLMs over a wide range of tasks. These benchmarks allow models to be tested on perception, reasoning, and knowledge extraction, among others, ranging from visual question-answering to document-specific challenges.

MMMU (Massive Multi-discipline Multimodal Understanding and Reasoning) First proposed in 2023, MMMU [53] is an 11.5k document complete benchmark that targets generalized evaluation on several domains, which evaluates perception, knowledge, and reasoning. In a zero-shot setting, models are challenged to generate correct answers without fine-tuning or few-shot demonstrations. An improved variant, MMMU-PRO, includes harder questions and removes data points that could potentially be solved just by the text inputs themselves.

MME (Multi-Modal Evaluation) MME [54] is a benchmark created in 2023 that has less than 1000 highly selected images. It consists of 14 subtasks with around 50 images for each and all with yes/no answers. The benchmark covers different aspects of object existence, perception of famous objects/people, and the translation of text. This has its distinctiveness in that none of the examples can be found on the internet, meaning that the VLMs are not biased in their evaluation.

MMStar First proposed in 2024, MMStar [55] is a high-quality subset of 6 VQA datasets, with 1,500 samples in total. They are rigorously filtered such that the question cannot be answered using only the knowledge in the text, the image must be used to produce the response, and the sample cannot be directly recalled from LLM training corpora. This benchmark is more about quality and not the quantity of VLM evaluation.

Math-Vista Math-Vista [56], published in 2023, includes 6.1k documents on mathematics for diverse reasoning types and grade levels. Collected from 31 different sources, it presents questions with multiple-choice or numeric answers in a manner that lightens the load of assessing the VLMs' ability in mathematics.

AI2D (AI2 Diagrams) Proposed in 2016, AI2D [57] contains 15k diagrams related to science understanding. Examples: More than 5000 grade school science diagrams with more than 150,000 rich annotations, with ground truth syntactic parses and over 15,000 associated multiple-choice questions. This benchmark stresses a VLM on its interpretation of intricate visual and textual relations appearing in scientific contexts.

ScienceQA ScienceQA [58], on the other hand, came out in 2022. It contains 21k questions, focused on scientific reasoning, elicited with the Chain of Thought paradigm. It provides the models with an Elaborated Explanation alongside multiple-choice questions, and it offers chat-like skills by presenting multiple texts alongside images to the VLM.

MM-Vet v2 The MM-Vet v2 [59], despite the size being small with 200 questions in size, is one of the most popular benchmarks. It comprehensively measures up recognition, knowledge, spatial awareness, language generation, OCR, and math capabilities. This benchmark evaluates both single-image-single-text situations and chat-like interactions.

VisDial VisDial [60], proposed in 2020, includes 120k images and 1.2M data points by reusing the COCO dataset. It evaluates a VLM chatbot for its response from a sequence of images and text to end Its question in a simulated visual dialog situation.

LLaVA-NeXT-Interleave LLaVA-NeXT-Interleave [61], released in 2024, consists of 17k samples and targets evaluation of a model's ability in settings that require multiple input images. This benchmark combines 9 new and 13 existing datasets including Muir-Bench and ReMI by providing a comprehensive multi-image understanding test.

Other datasets There are various other datasets which further flesh out the landscape for VLM evaluation. SEED, coming in 2023, provides 19k multiple-choice questions both for images and videos. VQA is one of the first datasets in this arena, with 2M data points on generic day-to-day scenarios in 2015. GQA, from 2019, contains 22M samples directed at compositional question answering, since it relates a number of objects in an image. Finally, VisWiz, in 2020, offers another unique perspective with 8k samples-a few generated by the blind-where spoken questions about images and crowdsourced answers are presented. Together, these benchmarks provide a comprehensive framework

for evaluating VLMs across different domains, task types, and modes of interaction. In the continuously evolving area of vision-language modeling, these benchmarks will serve as an active constituent to quantify progress and adopt improvements.

2.2.7 Applications of LLMs/VLMs in Robotics

Recently, large language and vision-language models have emerged as a revolutionary force in several fields, including robotics. These have greatly considered the general perception, interaction, and understanding of the world by robots. With LLMs, such as GPT-4 [49] built by OpenAI, robotics has gone way beyond simple commands or basic natural language processing that characterized previous generations. The model serves to provide reasoning capabilities for robots in developing human-like responses in intuitive and meaningful ways.

LLMs have significantly improved the power of dialogue systems, allowing robots to engage in high-level conversations with humans, to understand subtle instructions by themselves, and to comply with the instructions accordingly. To master complex events, where only vague or incomplete information is available, one has to cope with adaptive responses for given changing human input. LLMs execute multi-and complex-component natural language instructions for tasks, such as instruction execution and planning, at an unnaturally high level of productivity. This, in turn, greatly increases the productivity of robots executing complex, multistep manipulations of real-world objects, especially when these models are used to improve semantic understanding of the world. LLMs also give robots the ability to quickly adapt to new tasks through novel instructions without explicit reprogramming. This flexibility is especially valuable in an environment with continuously changing tasks and requirements. The capabilities of the robots are further enhanced by incorporating VLMs, which fuse visual and textual information to enhance understanding of the environment. For example, given a visual scene, VLMs can enable segmenting and manipulation using textual description. It is an important combination of vision and language for the progress necessary for true human-robot collaboration. The robot should be able to understand the human intention and communicate in a pattern that will allow smooth cooperation between them. Particularly in the big factories or research environments, meant to be shared by both humans and robots, this ability is growing ever so critical. Shared autonomy is important in these settings, where the robot has to

find its balance between autonomous operations and human control. Furthermore, these technologies enable a wide range of supporting applications aimed at letting a robot need spoken language or hand gestures as input while claiming to assist people with disabilities.

LLMs and VLMs have hence become quite the tool for a number of tasks including planning, navigation and manipulation [62], [63].

VLMs have proven very powerful for high-level decision making in robotics because they combine visual perception with language-based reasoning.

In [64] they showed how a pre-trained language model (LLM) can be used in zero-shot situations to get a robot solve complex tasks. [4], [63], [65] also studied how enhance the capabilities of these models in order to perform real-world tasks with zero-shot learning, which is basically using these models in a variety of situations without further training.

In Appendix A.3 ethical issues of using these models in Robotics.

2.3 In-Context Learning and Retrieval-Augmented Generation (RAG)

Recent years have shown new methods, using the advantages of machine learning and Natural Language Processing to enhance the capabilities of models such as LLMs and VLMs. In-Context Learning and, deriving from it, also Retrieval-Augmented Generation-RAG combine the strength of LLMs and retrieval systems to raise state-of-the-art performance on a wide range of tasks. Activities that involve minimum data preparation have recently been gaining momentum for ICL. Recent works like [66] have shown ICL integrated with memory-based retrieval to be an effective solution for robotics applications. These methods commonly rely on a single setup and most of the time require external modules for object recognition or other advanced techniques to extract textual features. In particular, our approach differs in adopting a zero-training pipeline, which leverages Image-Based ICL coupled with VQA to enhance the skills of VLMs further without resorting to other techniques. Moreover, it is enough to collect only a few samples for the accomplishment of the task. This enables our framework to generalize over both setups of FPV and TPV without having specialized sensors or large-scale pre-training. Our model effectively builds navigation plans from Image-Text pairs and generalizes better to changing scenarios and goals, and even completely different setups, compared with previous

methods.

2.3.1 Principles of In-Context Learning

In-Context Learning consists in the general capacity of large language models to adopt the right approach or a possible solution, given only a few examples provided within the same interaction [67]. This makes the concept crucial for understanding how such models as GPT-4, or similar architectures, can perform well on tasks that they were not explicitly trained for. In-Context Learning relies primarily on the ability to use example inputs and outputs in the prompt to guide the model's response. In cases where there is a series of examples or particular context, a user can elicit the model to produce an output relevant to that without the need for further fine-tuning. The main trick here is how to give the model the relevant examples so it is guided toward following a common reasoning or pattern to its responses. For instance, if a model is given several examples of translations from English to French, it then generalizes from those examples to translate completely new sentences. The model induces the pattern or structure from provided examples and applies that pattern or structure to new data. 2.11. It fits these examples to the internal representations until the context is drawn, while this adjustment is temporary and not permanent. It only persists for interaction or session length. The models work by taking the context from a prompt when approaching some new task. One of the key benefits to In-Context Learning, which we exploit in this work, is that it does not require the retraining or finetuning of the model on specific tasks. Rather, large-scale pre-training of the model enables generalization from context given to it. It enables flexibility in how models can be applied to a wide range of tasks. Powerful as it is, there are limits to In-Context Learning. However, the quality and relevance of examples given by the prompt are very sensitive to model performance. There is bound to be some optimum amount of context beyond which effective use will not be feasible, and this may construe performance for tasks requiring higher complexity and subtlety. In general, there is only that much impact on heeding, determined by how well the actual examples in the prompt match the task. Examples that are poorly chosen, or a lack of context, are likely to result in more incorrect or uninterpretable responses.

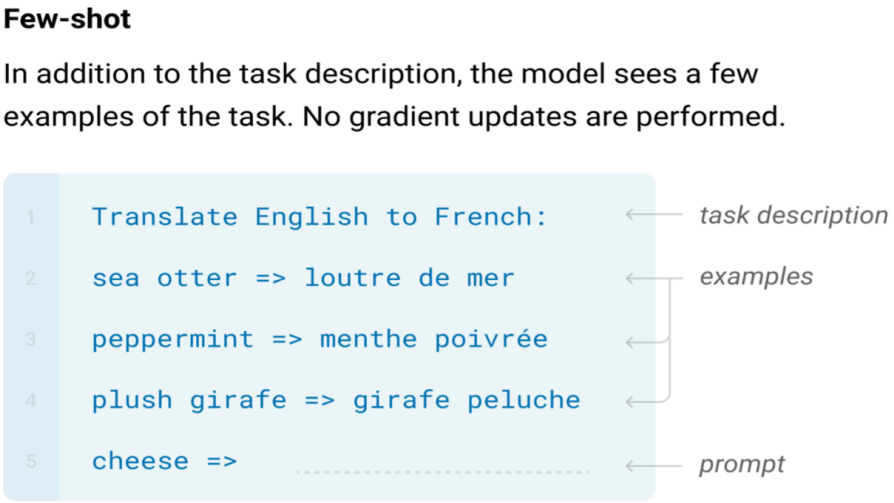


Figure 2.11: In-Context Learning working overview. In-Context Learning is also called Few-Shot Learning, since it provides few samples to the model in order to enhance its effectiveness.

2.3.2 Overview of RAG

Retrieval-augmented generation in the broadest sense can be referred to as the enriching process of generation through some sort of retrieval-based technique. This is done through the integration of external knowledge with the help of retrieval mechanisms into the generation process. Introduced in [68], RAG relies on an external retrieval mechanism that fetches relevant information or documents from a large database. This step in retrieval helps the model access a wider range of knowledge than it could have learned during pre-training. For example, the retrieval system, given some query or prompt, searches in the database or knowledge base for documents or passages related to the input provided. The information returned then enriches the output of its generative model counterpart. Of course, the concept of documents can be taken very broadly. For example, one can use Vision Language Model and retrieve images or text-image combinations. It retrieves relevant documents and then passes the information to a generative model for the final response. The generative model embeds the answers based on the content retrieved, both more generally accurate and contextually richer. Figure 2.12 report the general working of a RAG system. The retrieved information will add context to the generation that might not have been learned during its training; hence, responses are a lot more situated in knowledge from outside, not only with the model's pre-trained data. This will be particularly helpful in question answering, summarization, and information retrieval tasks,

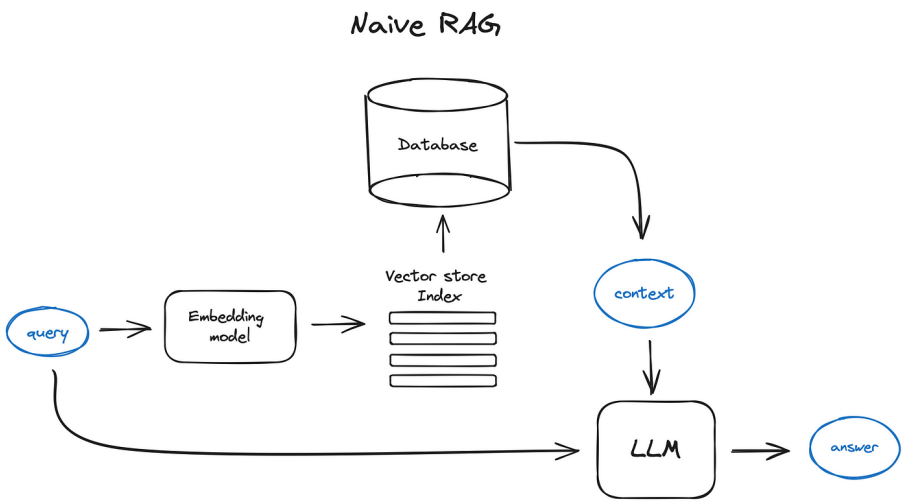


Figure 2.12: RAG Overview. The mechanism is straightforward: given a query (i.e. document or image), an embedding model processes the input and then perform retrieval from the database of items with similar embedding. These items are then added to the context in order to enhance generation of LLMs/VLMs.

where access to external documents can greatly raise the quality of responses generated.

In this architecture, the retrieval and generation are combined in a single pipeline. The retrieval module can be obtained using assorted techniques including BM25 or some dense retrieval methods. In this work, we use the output features of a ViT to retrieve similar context by exploiting the attention mechanism. While RAG offers improvements, there are still several challenges, such as efficient retrieval mechanisms being required and the risk of over-retrieving possibly irrelevant or noisy information. There aspects of balancing retrieval quality with generative capabilities to optimize RAG systems. It has thereby become of the essence that the retrieval system offers relevant and accurate documents in regard to effectiveness pertaining to the overall approach. The quality of generated responses is centered on the quality of the retrieved content. This is the basic reason behind our using a reranking algorithm Maximal Marginal Relevance (MMR) for choosing the context that gives best overall examples.

2.3.3 Maximal Marginal Relevance for Retrieval

Maximal Marginal Relevance (MMR) [69] is a method for enhancing relevance and diversity among retrieved information. One of the challenges in retrieving documents or pieces of information from a large collection is selecting those that are not only relevant

to the query but also diverse in their content. MMR tries to balance the trade-off between relevance and diversity by modulating the ranking of a document for both its relevance to the query and its marginal relevance compared to other retrieved documents. The formula used by MMR to compute the score of a document is reported in Equation 2.2:

$$MMR \stackrel{\text{def}}{=} \arg \max_{D_i \in R \setminus S} \left[\lambda \, \text{Sim}_1(D_i, Q) - (1 - \lambda) \max_{D_j \in S} \text{Sim}_2(D_i, D_j) \right] \tag{2.2}$$

where λ is an parameter balancing diversity and similarity, S is the set of documents already selected, R is the full set of documents (retrieved), D_i is a single document from set R and Q is the query document. The function *Sim* reported in the formula refers to a generic similarity function. MMR operates by combining relevance scores with a penalty for redundancy, striving to maximize the relevance of retrieved documents while minimizing the overlap between them. This results in a more informative and varied set of retrieval results, which is particularly useful in scenarios where diverse perspectives or pieces of information are valuable.

The integration of MMR into the RAG pipeline offers enhanced retrieval systems. This combination is valuable in various applications, including image search engines, multi-modal content retrieval, etc.

2.4 Tasks in Robotic Navigation

Robotic navigation is a multifaceted field, with tasks that can be range in different directions. Below are reported the main navigation tasks:

1. Planning and navigation: it is a fundamental aspect of robotics, involving formulation of strategies that successfully enable a robot to act or reach pre-determined objectives autonomously. The planning role of robotics could be discussed within the context where the robot decides for a sequence of actions interchanging its initial state to a desired goal state, considering constraints in the optimization of performance metrics. When applied to navigation, this means the sequence of actions required to take the robot from any given state to its final state; that is, to reach a target or execute a task in a certain point. Some tasks in which planning is necessary are: Some task where planning is necessary are:

(a) Object Navigation (ObjectNav): In robotics, object navigation entails the process of a robot finding its way to an object in an environment autonomously, using various means that involve perception, mapping, and planning. While general path planning focuses on how to reach a location, object navigation specifically targets identifying and moving to objects of interest.

(b) Point Navigation (PointNav): this could be referred to as a specific type of task in robotics. A robot is put at work to navigate itself toward a, so-called goal position, which in its essence is a point in space. It can normally be in the form of a coordinate: (x, y) or in three dimensions (x, y, z). PointNav is to reach a given point in space, not necessarily due to interaction with objects or specific actions to accomplish when it reaches that point.

(c) Image Navigation (ImageNav) consists in reaching a particular place in its environment based on the given target image taken there. Therein, an image depicting the desired destination is given instead of explicit coordinate or object-based movement instructions, and it takes the cue via sensors and perception capabilities to find that place.

Quite a number of these tasks, naturally rely hugely on robot perception to achieve the purpose.

2. Obstacle Avoidance: this is an important ability of robots, whereby a robot is enabled to perceive and then move around other objects in its surrounding environment, making sure no collisions happen. In this task cameras, LiDAR, and ultrasonic sensors are exploited to identify obstacles and, as a result, make changes on the go in the robot trajectory so that it will be capable of passing through them safely. To achieve the goal with obstacle avoidance by real-time decisions, potential fields [11], vector field histograms, or other reactive control methods are used.

3. Exploration: The independent process in robotics whereby the robot aims to explore an unknown environment by mapping or solving a particular problem. Due to different sensors on a robot, it perceives the surroundings differently; hence, the robot makes a systematic movement to new areas and updating the internal representation or map becomes necessary. The exploration strategy often faces a trade-off between

maximum area coverage, obstacle avoidance, and optimal path to obtain total and efficient mapping of the environment.

3 Methodology

This section details we introduce the general framework developed this work along with the specific declinations for the two deployment scenarios: Third Person View and First Person View. In figure 3.1 is depicted the overview of the framework.

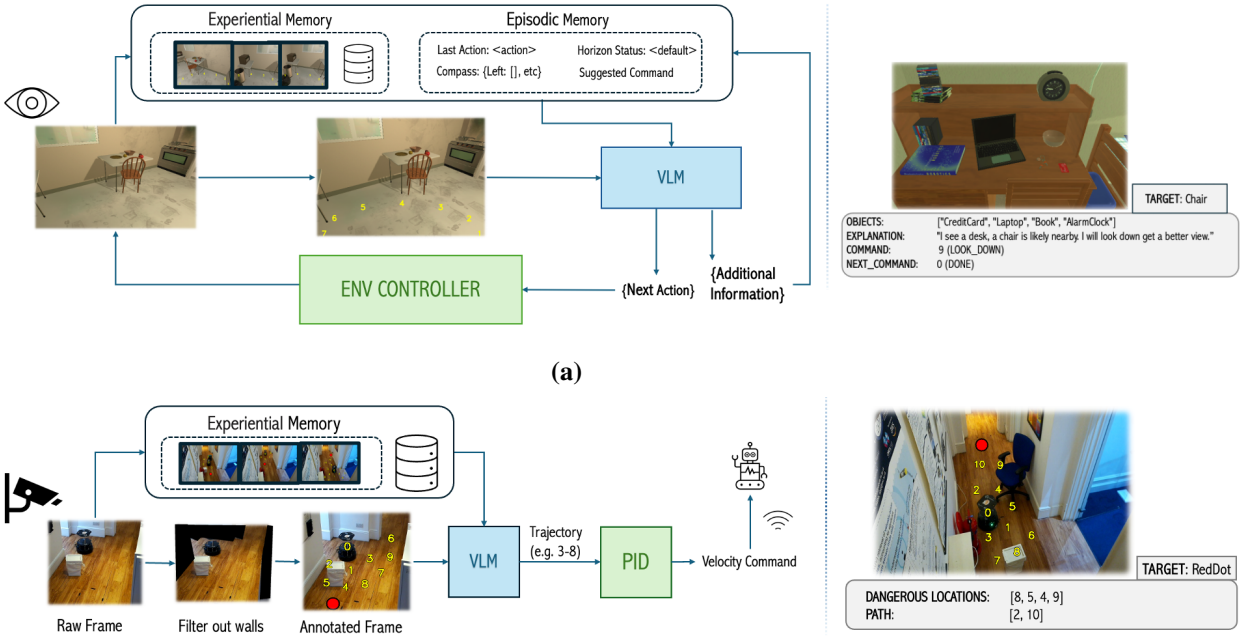


Figure 3.1: Overview of the proposed approach in FPV (a) and TPV (b). The two settings are designed to fit two specific scenarios but share their components. The framework takes a live image from the onboard or a CCTV camera and retrieves similar images from the experiential memory. It is then annotated and passed, with the sampled images and an optional episodic memory, to the VLM to retrieve the next commands to send to the platform and explanations. The main difference is the absence of an Episodic Memory in the TPV setting, where the off-board sensing setup empirically limits its benefits. Alongside the overview, response examples are presented for both setups.

3.1 Framework Architecture

In this section, we present a comprehensive overview of the proposed framework, highlighting its architecture and the individual components that work together to achieve the desired functionality in both deployment scenarios: Third Person View (TPV) and First Person View (FPV). The framework is designed to leverage multimodal data from various sources and seamlessly integrate it to facilitate decision-making and action generation for autonomous systems.

As depicted in Figure 3.1, our approach consists of six key components: an experiential memory, a sampler, an episodic memory, an annotator, a prompt templating engine, and a vision-language model (VLM). These components interact with each other to process real-time data from either onboard sensors or external cameras and generate contextually relevant responses and commands. The framework is designed to be versatile, catering to the specific requirements of both FPV and TPV scenarios while maintaining a shared core architecture.

3.1.1 Experiential Memory

Experiential Memory denotes a set of annotated images providing contextual knowledge to guide theVLM through ICL. Each image in this memory is enriched with annotations that provide useful information, such as the ground truth for a particular situation, information about dangerous points, or explanations in a human-like manner for decisions made. These experiences are vital to the system, where the system may remember relevant past events and make an informed decision in real time, similar to how humans use memory to navigate a new situation based on previously acquired knowledge.

Information gathered from real-world environments can also be used in addition to simulation-generated data to populate the Experiential Memory.

Visual inputs and contextual data are collected by the system in real-time operation of robots, which provides both more natural and random variations in capture such as lighting changes, dynamic objects, and noise in general. These real-world experiences are particularly valuable since they bring the system closer to the operational scenarios it will face with complex challenges. Moreover, the system is able to incorporate external input from, for example, surveillance camera footage or publicly sourced video content,

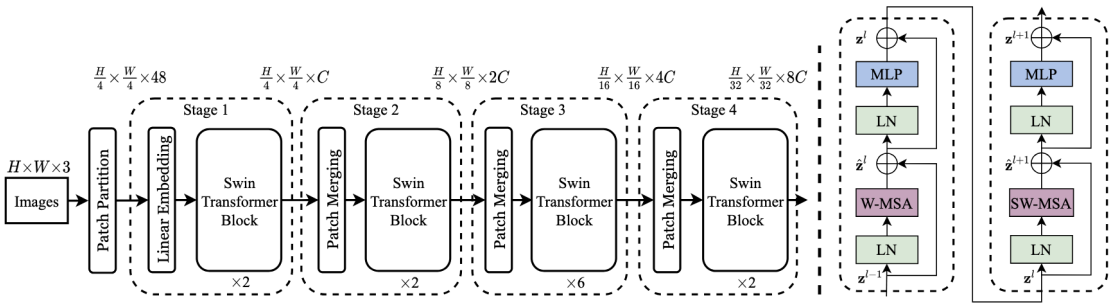


Figure 3.2: Swin Transformer Overview (on the left) and transfomer blocks (on the right). The Swin Transformer Model is a hierarchical model processing images in patches at various resolutions. *Figure and caption taken from original paper* [70].

increasing exposure to scenarios that may not have been seen in the training data, and thereby increasing the robustness of generalization across various tasks. Experiences in the Experiential Memory can be collected in a myriad of ways, and in the results section, we consider a few combinations of data sources. These include data from similar or different environments than the test environment, and examples sourced directly from online footage.

3.1.2 Sampler

The sampler aims to select the most appropriate samples from the Experiential Memory to be presented to the VLM as context. We feed a Vision Transformer (ViT) – a Swin Transformer [70] – with the live camera image to recover similar situations from the Experiential Memory. An overview of the Swin Transformer's architecture is reported in Figure 3.2.

We represent the image – and the Experiential Memory samples – as the average output of the last hidden layer and obtain a feature vector which we employ as the query.

Empirically, we observed that building a diverse context (both similar and different situations) benefits the model's generalisation to the current situation. To balance out the similarity, we incorporated a re-ranking algorithm adapted to our framework into the retrieving process. We employ a Maximal Marginal Relevance (MMR) [69], which aims to reduce redundancy and increase sample diversity according to a combined criterion of query relevance and novelty of information. MMR is defined as follows:

$$\texttt{MMR}(Q,M,C) = \arg\max_{s_i \in M \setminus C} \left[\lambda < s_i, Q > -(1-\lambda) \max_{s_j \in C} < s_i, s_j > \right] \tag{3.1}$$

where Q represents the query image embedding and M the experiential memory. This algorithm operates by iteratively selecting images (samples $s_i \in M$), to add to the context C, based on a trade-off between two factors: the image's relevance – similarity – to the query and the image's dissimilarity from the samples that have already been chosen. The goal is to ensure that each selected image adds new, informative content rather than repeating information. The scalar λ balances this trade-off.

3.1.3 Episodic Memory

Episodic Memory is the information relevant to the current interaction or episode that provides the essential basis for informed navigation decisions. Unlike Experiential Memory, which retains data about past experiences, Episodic Memory deals only with the current context of the present task or situation. This is the memory that captures important information from the current episode, including actions taken by the robot, layouts of environments, objects that have been encountered, and decisions made during the session. The system can ensure relevance and robustness because the focus is on the present situation.

To avoid the VLM from being overprompted with unnecessary information, we intentionally leave out the full raw context of previous responses or interactions that could add noise and confusion. Instead, we use a text-based, simplified scene representation. The distilled format includes some critical information. The simplified structure helps streamline decision-making and focuses on only the most relevant elements of the current episode to keep the VLM on track for solving the task at hand. By continuously updating this memory, the system keeps crystal clear on the context and hence makes proper predictions of the next best action since this provides the system with the opportunity to devise new strategies based on a clear and organized summary of the scene rather than reassessing the entire environment from scratch. In a nutshell, Episodic Memory assumes an active situational record for real-time decision-making: maintaining the VLM updated by means of compact, salient information from the present episode provides it with good performance on immediate reaction and adaptation during the evolution of the scene.

3.1.4 Annotator

Before presenting the live and episodic images to the VLM, we take inspiration from [9] and visually annotate them with the possible actions the model can choose from. These annotations are platform-specific, and are applied as numerical values superimposed on the image frame; we will discuss them more in this section.

3.1.5 Prompt Templating Engine

The Prompt Template Engine lies at the core of the prompt engineering procedure and aims at delivering structured and understandable directives to the VLM. Its major role is to acquire multiple streams of data from the system's Experiential Memory, Episodic Memory, and live inputs from the robot's surroundings and combine them into a coherent and actionable prompt that would get readily processed by the VLM. The prompt serves to organize this information in a clear and digestible format that will enable the VLM to contextualize the information and perform the needed operations efficiently.

It integrates information from three major sources. First, it incorporates pertinent knowledge from the Experiential Memory, which contains episodes or experiences of the past that the VLM can refer to in negotiating the present situation. Second, it includes information from the Episodic Memory that represents the immediate context of the ongoing episode and gets updated in real time as the robot interacts with the environment. Third, the prompt template embeds live input, such as annotated images of the current environment providing visual context and identifying salient features such as obstacles, navigational points, or goal locations. These form a structured prompt combining these data sources that is designed to direct the VLM on how to perceive the current situation and what set of next actions to execute. For example, it may explain to the prompt how the annotated image represents certain actions or decision points about steering directions, object recognition, or obstacle avoidance. It also represents relationships between prior episodes and the current episode, thereby informing the VLM of relations between similar situations it has experienced in the past and the current episode under consideration. These instructions are given in a template format that the VLM can easily parse and respond to, thus ensuring compatibility with existing robotic systems. We further request that the VLM provide its output in machine-readable JSON format so it can easily inter-

face with any control pipeline that may already be deployed on the robotic platform. The JSON may contain a sequence of suggested actions, justifications taken by the actions, and any additional data that may actually be used to execute actually the action. This, in turn, allows for smooth interaction between the VLM and the robot control system: the prompt is an integral part of the decision-making. In Appendix, we present the full template prompts that were used for both deployment scenarios: how the system is adapted to fit different environments and different tasks while consistency in structure and execution is followed.

3.1.6 Vision Language Model and In-Context Learning

The VLM is prompted to select a discrete action from the annotated frame, represented by numerical values, and explain its decision. Depending on the deployment platform, the model may also produce additional outputs, such as identifying actions that could lead to dangerous locations or recognizing objects in the scene. To provide context, we use a *History-Injection* ICL, where a fictitious chat conversation is created. In the chat, episodes retrieved from a memory database are split in *query* (annotated image and prompt) and *answer*, and injected into the model as turns of conversation as question and answer. This yields a multi-turn conversation of k turns, with k equal to the number of ICL samples. Finally, we explicitly ask the model, based on its previous responses, to process a new image.

3.1.7 Controller

Finally, the low-level, platform-specific controller executes the selected action on the platform. In the case of the external camera scenario, the positions of the annotations chosen by the VLM are used as vertices in a piecewise linear path. The robot is then guided along this path using a PD controller measuring cross-track and heading error as demonstrated in [12]. In FPV scenario, the controller is embedded into the environment and teleports the agent to the target location computed on the base of the input command.

Figure 3.3: Example of the view from the robot in the iThor simulator.

3.1.8 Adaptations to the FPV Setting

We designed our framework with the AI2-THOR simulator [71] as the platform, where the discrete robot actions comprise the robot's and camera's movements. In figure 3.3, a frame captured by AI2-Thor simulator is presented. As shown in 3.1a, we enable the VLM to interpret visual annotations that resemble a control overlay inspired by video-game interfaces. We display the numbers 1 to 7 on a semicircle at the bottom of the image, providing rotational control, where 4 is the neutral `MOVE_FORWARD` and the others numbers represent various degrees of rotation. Additionally, the model can select `LOOK_UP` and `LOOK_DOWN` commands, associated with the non-displayed numbers 8 and 9. Lastly, number 0 is associated with `DONE` command to end the episode.

Following a structured procedure, we build the Experiential Memory by manually navigating the robot in the AI2-THOR environment. At each timestamp, the operator selects an appropriate action based on the current visual context – i.e. a command number – and provides a natural language explanation, simulating a "think-aloud" process. For instance, upon observing a microwave on the left, the explanation would state: "A microwave is visible on the left, so the system will steer slightly to the left." The annotated image, the selected command, and the corresponding explanation are inserted in the Experiential Memory. This process aims to imitate human-like physical movements and capture the underlying thought processes that drive these actions.

Finally, we request the VLM a text description of the environment – the list of objects in the frame – and save it in the Episodic Memory. From it, we create a "circular compass", as shown in 3.4, which rotates along with the agent's rotations and can inform the model's decisions since certain objects can be found near affine items or go out of the field of view due to motion.

We also include the last action, the current vertical view status and the previous command list: the VLM at each timestamp is asked not only to provide the current action, but also the next future one, to robustly follow the navigation strategy in act (3.1a).

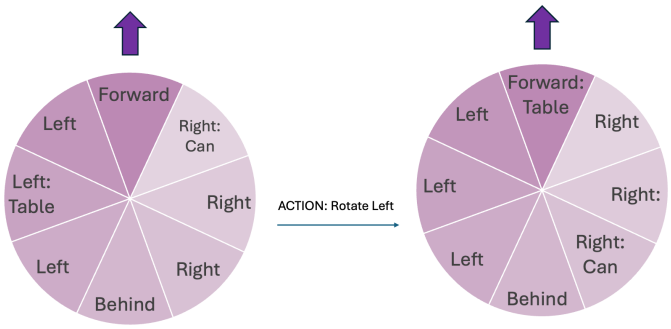


Figure 3.4: The figure depicts a scenario where the agent uses the compass. The compass keeps track of the scene content as the robot rotates, remembering insightful information about the room's layout. For instance, if the agent is looking for a *chair*, it will likely rotate towards where it last saw a *table*, although it is now out of sight.

3.1.9 Adaptations to the TPV Setting

Hardware Setup

The rover used for this project is a Turtlebot3 Burger (Figure 3.5). The TurtleBot3 Burger is a compact, affordable, and versatile open-source mobile robot designed for research, education, and hobbyist applications. Developed by ROBOTIS, it features a modular design, allowing for easy customization and upgrades. Traditionally equipped with a Raspberry Pi for onboard computation and a suite of sensors, including a 360-degree LiDAR and an inertial measurement unit (IMU), the TurtleBot3 Burger is capable of autonomous navigation, mapping, and obstacle avoidance. Its compatibility with the Robot Operating System (ROS) enhances its flexibility, enabling users to implement various algorithms and participate in a vibrant community for collaborative development and learning.

However, in this project we used the rover only with a Wi-Fi module as on-board sensors, to receive command messages and a Raspberry PI to eleborate them.

Figure 3.5: The figure represents the TurtleBot3 Burger. We made use of this robot in our experiments, especially for the data annotation step.

TPV Methodology

We follow the setup described in [11], [12], utilizing a visual servoing system [72], [73] where the robot is controlled remotely through cameras installed in the environment. Security cameras capture raw frames of the area, which are then annotated and combined with samples from the Experiential Memory. This combined input is then processed by the VLM to predict the next set of trajectory points for the robot to follow. Figure 3.1b provides a visual summary of the framework components.

The annotation process for the live frames begins with detecting the robot's position using a YOLO model, which marks the robot's current location with the number 0, making the task independent of the robot's embodiment. During the system's initial setup, a segmentation mask of the scene's floor is created using the Segment Anything model [74] and saved as a binary mask. This step is performed only once during the initial setup phase and, as a result, does not account for any new objects or obstacles that may appear later in the scene.

Once the robot's position is identified, we generate a grid of concentric circles around it, with numbers assigned at equal intervals and increasing radii. These numbers represent potential positions for the robot to move to, but only traversable positions are considered, as determined by filtering with the segmentation mask.

Next, the VLM is tasked with choosing a sequence of points, guiding the robot from its current location to the desired end goal. This goal could be specified as a particular object in the scene or as a predefined safe zone, such as a red circle clear of obstacles, as

depicted in Figure 3.1b.

Optionally, the image can be cropped to focus solely on the labels and the target destination. This reduces the image size processed by the model, thereby lowering computational costs and improving the model's attention to crucial details in the scene.

After the VLM outputs a sequence of points, these points are mapped back to coordinates in the image space. The robot then follows this path using a PID controller [12]. During inference, the VLM typically produces a sequence of three to four points. As the robot successfully tracks the first one or two points, the system updates the sequence dynamically to guide the robot to its target efficiently.

3.2 Experimental Setup

3.2.1 Data Collection and Annotation

The data collection process has been done in two different versions, since we have two different setups.

FPV: Data Collection

Starting from the first person view, we manually navigate the robot through the environments, recording one episode per target object, resulting in a total of just 25 episodes. This database is extremely limited purposefully to demonstrate that even few episodes can establish an effective framework, thereby challenging the generalization capabilities of trained models. The process of collection starts with the operator spawned in a randomly picked scene (e.g. Kitchen1) and a random object (from the objects chosen for "train"). The operator, following its reasoning, at each timestamp chooses a numerical command and afterwards gives the explanation for his choice. This constitutes a human-like sub-optimal ground truth, which composes the system's Experiential Memory and helps the model generate situation-grounded strategies. In Figure 3.6, we present an example of a single step recorded.

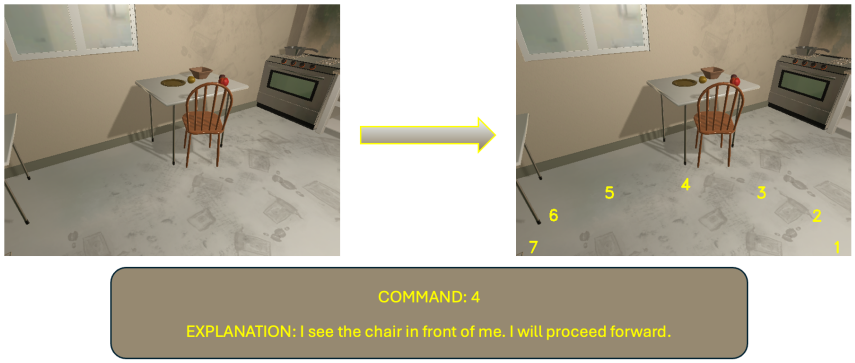


Figure 3.6: Visualization of the decision-making process by the robot. The left image shows the initial scene with the chair in front of the robot. The right image highlights the robot's forward movement based on the command issued (Command: 4), as indicated by the numbered path markers. The explanation box describes the robot's reasoning: "I see the chair in front of me. I will proceed forward," demonstrating how visual inputs and navigation decisions are processed.

TPV: Data Collection

We evaluate our approach on a custom dataset recorded in four rooms of the Oxford Robotics Institute premises – see A.1 – where we collected expert trajectories by teleoperating a TurtleBot3 rover.

Before starting, we load a single picture of each room and using [74], we extract the mask of the floor from each one. This process is executed once, during the initial setup. The objective of this step is to delete all the walls from the picture, which are fixed non-traversable areas. The trajectories recorded encompass a range of difficulty levels, from simple paths with minimal obstacles to more complex routes that include various dynamic and static obstacles, representing real-world navigation challenges. After having saved the pictures, we annotate them following these steps:

1. Identify the position of the rover using a custom Object Detector [75], finetuned from YOLO. This step is not strictly necessary but it greatly helps in the annotation process. We then superimpose the number 0 on the image, in order to make the task embodiment-agnostic, since from now on: there is an *AGENT* in position 0 (even if it's not a robot anymore).

2. Starting from position 0, in the annotation process, we take the next image (one step into the future, since we have a full video recorded) and identify the robot, in the original picture we create concentric circles with center in the position of label

(a) Camera 1: ORI Office #1

(b) Camera 2: Narrow corridor

(c) Camera 3: Wide corridor

(d) Camera 4: ORI Office #2

Figure 3.7: Examples scenes in the TPV scenario. Random obstacles are placed to challenge the planner, e.g. the blue chair. in subfigure 3.7b

0 and the distance between the center and the new position as radius. The numbers chosen as labels are totally random and while we get further from the center, the number of labels per circle increases. We repeat this process 3/4 times for each trajectory.

3. We then use the binary mask collected during the initial setup to delete all the points overlapping with walls. In this way, we obtain a much less noisy annotation.

4. Last, but not least, we use the last timestamp as the target position. Instead of extracting a new circle of points, we use the last location as coordinates for a red circle. In this way, on the picture we obtain the position 0 (start of the trajectory), numeric labels, identifying the path of the human operator, in between and finally the red circle which is the end goal of the rover.

3.2.2 Model Selection: VLM

We compare different VLMs, in order to understand which is better as zero-shot planner. To benchmark this, we use a split (100 images) of the dataset recorded for the TPV scenario. The experiment is built to understand if these models can understand the task at

hand. We keep only the images with 1-step trajectories and ask the model where to go next in order to reach the red circle. This is a much easier version of the final task but is useful to get insightful results. We compared both open-source ([76], [27], [32]) and proprietary models ([50], [49]. We use as main metric the accuracy, since this is basically a classification problem (configured as VQA). Results are reported in Table 3.1.

Model	**Accuracy**
Idefics2	0.347
LLava-Mistral7B	0.58
Gemini Pro 1.0	0.683
Claude Opus	0.78
GPT4V	0.844
Gemini Pro 1.5	0.950

Table 3.1: VLMs results: comparison between the zero-shot abilities of the models in finding the best next step towards the target.

As we can extract from the results, the most promising model is Gemini Pro 1.5 (`gemini-1.5-pro-001`), so we empirically decide to focus on it from now on. Our framework relies mostly on ICL, so the huge context-window of this model fits in our work perfectly.

3.2.3 First Person View

Among the several simulators proposed to facilitate the Embodied Navigation tasks [71], [77], [78], we selected AI2-THOR's ObjectNav task, whose goal is to navigate towards a predefined target object.

We follow the evaluation procedure of [79], using the same metrics and setup, with different object classes for training and testing[1]. We hence compare our approach to [79] [80], [81], [82]. Due to resource constraints, we evaluate our framework on 300 episodes

[1] [79] chose as training objects: *HousePlant, Sink, TableTop, Knife, Fridge, Bowl, Cabinet, Cloth, KeyChain, WateringCan, Bed, Lamp, Book, Chair, LightSwitch, Candle, Painting, Watch, Cabinet, Toilet, SprayBottle* (for us, there is one episode for each one of them in the experiential memory). As test objects they chose: *Toaster, Microwave, Television, LapTop, RemoteControl, CellPhone, Mirror, AlarmClock, Toiletpaper, SoapBottle*.

per scene type, and limit maximum number of steps per episode to 25, thereby increasing the complexity of the task.

3.2.4 Third Person View

We annotate – see 3.1.9 – the images and manually mark the labels overlapping with obstacles or non-traversable locations as *dangerous*. We compare the results of our model against a zero-shot approach on this dataset. We test different Experiential Memories, composed of scenes from the same or different cameras, called scenarios `A` and `D`, respectively – see 3.8. In addition, we will show results with trajectories performed by a human in the same scenes – scenario `H` – simulating the images usually captured by security cameras; this would demonstrate how a person move in an office room, avoiding obstacles such as chairs, boxes, etc. Finally, we will also use short clips of robots from the web [83], [84] – scenario `O` – to validate how general and different from the target scenario the samples in the context can be, while still allowing the model to understand the task and mimic it successfully. In Appendix, we present samples for each context scenario. The

Figure 3.8: Experiential Memories for TPV: Scenario `D` includes experiences from the same environment excluding the inference room, `O` from online videos and `H` from the same environment but with a human as navigator instead of a robot.

main metric to evaluate our system is the Trajectory Score (TS), defined as:

$$\text{TS} = \sum_{i=0}^{N} S_i \frac{Pc_i}{\max(\text{len}(P_i), \text{len}(G_i))} \qquad TS_i \in [0,1] \tag{3.2}$$

where Pc_i is the number of correct predicted points, P_i is the predicted sequence and G_i is the ground truth sequence, and S_i indicates if the selected point is safe, at iteration i. The maximum value of this trajectory score is N, in our case 300, which is also the maximum TS value. The purpose of this metric is to measure to which extent the model is able to

reproduce human-like navigation. Moreover, we measure the number of dangerous points selected during the evaluation process.

$$D = \sum_{i=0}^{N} D_i \quad D_i \in \{0,1\} \tag{3.3}$$

where D_i indicates if at episode i a dangerous point has been selected or crossed, simulating a collision.

To make the evaluation goal-agnostic, we consider as end-goal a red circle, superimposed on the picture and report results averaging three consecutive runs.

4 Results

In this chapter, we provide a detailed analysis of the performance of our proposed framework under different deployment scenarios. We compare the outcomes with state-of-the-art approaches and demonstrate the strengths and limitations of our methodology in both Third Person View (TPV) and First Person View (FPV) settings. The evaluation metrics include Trajectory Score (TS) and Danger Score (D) for TPV, as well as Success Rate (SR) and Success weighted by Path Length (SPL) for FPV. These metrics collectively offer a comprehensive view of the framework's effectiveness, robustness, and generalizability across various experimental conditions.

4.1 Third Person View (TPV) Scenario

Our In-Context Learning (ICL) approach demonstrated significant improvements in the TPV scenario. The highest Trajectory Score (TS) achieved was 270.70 in context scenario *A*, which allows unrestricted retrieval from the database, compared to the baseline zero-shot approach, which scored 147.82. This represents a 54.6% improvement, indicating that our model can effectively leverage contextual information to enhance navigational accuracy and safety. Results are reported in Table 4.1.

In addition to the overall TS improvement, the framework showed a remarkable 38%

Mode	**CL**	**Scenario**	**TS (/300)** ↑	**D (/300)**↓
Zero-Shot	0	-	147.82	76
ICL (Ours)	**10**	**A**	**270.70**	**2**
ICL (Ours)	10	D	247.24	8
ICL (Ours)	10	H	219.58	13
ICL (Ours)	10	O	235.83	16

Table 4.1: TPV results: comparison between the zero-shot and our framework on the same scenario but using different types of context sources (based on Scenario column).

reduction in selecting dangerous points (reported as D in the table), which indicates locations with a high risk of collision. This reduction is crucial for real-world applications, where safety and reliable navigation are paramount. Such performance gains were consistent across different context scenarios, further highlighting the versatility and adaptability of the proposed approach.

4.1.1 Detailed Analysis of Context Scenarios

Scenario *A* (Unrestricted Context)

In scenario *A*, the model had access to the entire database of contextual information. This scenario yielded the highest TS (270.70) and the lowest D (2) scores, illustrating that a rich and unrestricted context significantly boosts the model's ability to predict safe and efficient trajectories. The model effectively utilized diverse data sources to improve decision-making, avoiding hazardous locations and minimizing path deviations.

Scenario `D` (Different rooms Context)

When the context was restricted in scenario `D`, the model still performed exceptionally well, achieving a TS of 247.24 and a D of 8. Although there was a slight decrease in performance compared to scenario `A`, the model's ability to generalize and adapt to a limited context set was evident. This scenario underscores the model's capability to function efficiently even when the available contextual information is limited, demonstrating robustness and flexibility.

Scenario `H` (Human-Driven Trajectories)

Scenario `H` involved using context derived from human-driven trajectories. The TS of 219.58 and D of 13 indicate that the model was able to effectively incorporate human behavioral patterns into its decision-making process. This is a significant finding as it showcases the model's potential to learn from human demonstrations and adapt to diverse navigation styles, potentially useful for personalized navigation systems.

Scenario 0 (Online Video Data)

With scenario 0, which utilized context from online videos, the model achieved a TS of 235.83 and D of 16. This performance, while slightly lower than other scenarios, demonstrates the model's ability to leverage external, non-specific sources of information to improve navigation. This adaptability is particularly valuable for scenarios where real-time contextual data might be scarce, making the approach suitable for dynamic and unpredictable environments.

4.1.2 Implications and Future Directions

The results from the TPV scenario suggest that our ICL-based framework can significantly enhance the safety and efficiency of autonomous navigation systems by leveraging contextual information. Future work could explore integrating additional modalities of context, such as sensor data from the environment or user-provided preferences, to further improve performance.

4.2 First Person View (FPV) Scenario

4.2.1 General Performance Evaluation

In the FPV setup, our framework, referred as Select2Plan (S2P), was evaluated against state-of-the-art methods in a variety of scenarios. The average Success Rate (SR) of 46.16% in known scenes with known objects reflects the model's ability to perform well even with a minimal training dataset. Compared to the best-performing model [79], which was trained on 8 million episodes, S2P required only a fraction of the data, specifically one episode per object type. Despite this, S2P managed to achieve comparable results, highlighting the efficiency of our approach in leveraging pre-trained Vision-Language Models (VLMs) for efficient knowledge transfer.

Known Scenes and Known Objects

In the most favorable scenario, where both scenes and objects were familiar to the model, S2P achieved a SR of 46.16% and an SPL of 28.01%. While this was lower than the top-performing models such as GVSN, which achieved an average SR of 83.73% and

an SPL of 57.03%, it is important to note that S2P was working with significantly less data and with no training at all. This indicates that while extensive training can enhance performance in familiar environments, our approach remains competitive with a much smaller data footprint.

Known Scenes and Novel Objects

In scenarios with known scenes but novel objects, the SR of S2P was 37.75% with an SPL of 24.66%, outperforming the state-of-the-art models in several cases. Notably, the performance in the bathroom setting reached an SR of 54%, surpassing even the best model's performance in this particular scenario. This suggests that S2P's ability to generalize across unseen objects, combined with its minimal data requirements, makes it particularly suited for environments where new objects are frequently encountered.

Novel Scenes and Known Objects

When presented with novel scenes but known objects, S2P achieved an SR of 41.17% and an SPL of 24.50%. It outperformed the best models in certain settings, such as the kitchen and bathroom, where it showed a clear advantage in navigating unfamiliar environments. This suggests that our model can effectively transfer its knowledge from known to novel environments, an essential trait for real-world applications where environmental conditions are often unpredictable.

Novel Scenes and Novel Objects

In the most challenging scenario, involving both novel scenes and novel objects, S2P's performance remained robust, achieving an SR of 38.95% and an SPL of 28.62%. This was significantly higher than the state-of-the-art models, with the closest competitor achieving only 28.67% as Success Rate. S2P's ability to maintain high performance in such a challenging scenario highlights its strong generalization capabilities and reinforces the value of integrating VLMs for navigation tasks.

4.2.2 Implications for Real-World Applications

The FPV results demonstrate that our approach, utilizing a significantly smaller dataset, can achieve competitive performance across a range of challenging scenarios. Results

are reported in Table 4.2. This has profound implications for deploying autonomous navigation systems in real-world environments where comprehensive datasets are often unavailable or costly to obtain.

Eval. set	Method	SR ‖ SPL %				
		Kitchen	**Living room**	**Bedroom**	**Bathroom**	**Average**
Known scenes & Known objects	Random policy	10.40 ‖ 4.80	11.47 ‖ 3.40	13.07 ‖ 8.20	21.60 ‖ 11.13	14.13 ‖ 6.88
	Scene priors	50.67 ‖ 34.27	67.07 ‖ 29.43	69.07 ‖ 22.47	71.33 ‖ 28.73	64.53 ‖ 28.73
	SAVN	46.27 ‖ 38.27	56.93 ‖ 40.60	74.67 ‖ 35.47	81.87 ‖ 46.53	64.93 ‖ 40.22
	VTNET	57.60 ‖ 43.20	68.53 ‖ 45.03	86.53 ‖ 38.53	76.13 ‖ 52.30	72.20 ‖ 44.77
	GVSN	**63.20** ‖ **50.27**	**88.00** ‖ **52.43**	**94.27** ‖ **59.07**	**89.47** ‖ **66.37**	**83.73** ‖ **57.03**
	Ours	45.50 ‖ 31.05	28.50 ‖ 19.0	50.40 ‖ 25.29	60.25 ‖ 36.70	46.16 ‖ 28.01
Known scenes & Novel objects	Random policy	2.93 ‖ 0.66	5.60 ‖ 0.93	8.80 ‖ 2.53	8.13 ‖ 2.03	6.37 ‖ 1.54
	Scene priors	21.07 ‖ 14.55	23.20 ‖ 9.40	19.47 ‖ 12.17	30.53 ‖ 18.47	23.57 ‖ 13.65
	SAVN	17.27 ‖ 8.30	27.20 ‖ 7.60	37.87 ‖ 20.47	32.53 ‖ 16.50	28.72 ‖ 13.22
	VTNET	26.53 ‖ 12.27	49.07 ‖ 23.97	35.87 ‖ 18.63	36.67 ‖ 22.53	37.03 ‖ 19.35
	GVSN	**32.67** ‖ **20.13**	**58.53** ‖ **32.50**	**53.07** ‖ **20.97**	49.07 ‖ 25.60	**48.33** ‖ **24.80**
	Ours	30.00 ‖ 19.01	23.00 ‖ 18.70	44.00 ‖ 32.44	**54.00** ‖ **28.51**	37.75 ‖ 24.66
Novel scenes & Known objects	Random policy	6.00 ‖ 0.87	4.20 ‖ 1.27	3.47 ‖ 0.37	4.67 ‖ 1.30	4.58 ‖ 0.95
	Scene priors	11.60 ‖ 6.23	13.87 ‖ 8.27	17.20 ‖ 10.83	15.07 ‖ 8.40	14.43 ‖ 8.43
	SAVN	26.80 ‖ 10.70	31.33 ‖ 6.00	43.87 ‖ 15.63	21.73 ‖ 8.33	30.93 ‖ 10.17
	VTNET	35.73 ‖ 12.30	40.93 ‖ 13.93	57.87 ‖ 17.83	47.60 ‖ 10.73	45.53 ‖ 13.70
	GVSN	44.13 ‖ 18.50	**48.00** ‖ **27.53**	**68.67** ‖ **19.50**	60.53 ‖ 26.07	**55.33** ‖ 22.90
	Ours	**50.00** ‖ **32.40**	28.70 ‖ 17.85	24.00 ‖ 12.79	**62.00** ‖ **34.94**	41.17 ‖ **24.50**
Novel scenes & Novel objects	Random policy	1.60 ‖ 0.43	3.87 ‖ 0.80	3.60 ‖ 0.33	2.27 ‖ 0.93	2.83 ‖ 0.63
	Scene priors	2.93 ‖ 1.13	10.80 ‖ 3.13	23.07 ‖ 7.60	15.87 ‖ 6.37	13.17 ‖ 4.56
	SAVN	17.33 ‖ 5.97	13.60 ‖ 4.50	25.47 ‖ 5.50	12.27 ‖ 4.37	17.17 ‖ 5.08
	VTNET	26.67 ‖ 7.03	19.47 ‖ 9.03	16.93 ‖ 7.40	16.93 ‖ 7.40	22.43 ‖ 6.79
	GVSN	34.40 ‖ 7.30	17.87 ‖ 8.10	**32.40** ‖ 9.60	30.00 ‖ 6.57	28.67 ‖ 7.89
	Ours	**52.40** ‖ **40.75**	**35.40** ‖ **21.61**	26.00 ‖ **17.48**	**42.00** ‖ **34.65**	**38.95** ‖ **28.62**

Table 4.2: Comparison of SR and SPL across different environments (kitchen, living room, bedroom, and bathroom). The table shows that S2P outperforms trained models in novel scenes, particularly in terms of average SR and SPL, indicating superior generalization across different environments and object configurations with minimal data needed.

5 Discussion and Conclusion

This work presents the use of In-Context Learning combined with a VLM and shows its capability for improving state-of-the-art reasoning and planning of autonomous navigation systems. The results show that our method, without specific training or large datasets, significantly outperforms zero-shot baseline performance. This may prove important in many applications where data is limited. The fact that it can immediately adapt to various and dynamic contexts without extensive data collection and retraining shows its flexibility and scalability.

A key strength of the work is how effectively In-Context Learning allows the model to incorporate new information on the fly and make contextually informed decisions. For instance, the model leveraged the scene and previous trajectories to navigate open spaces it had never seen before without a significant amount of computational resources and time saved due to its reliance on minimal data instead of exhaustive training procedures. This positions our framework as one potential solution for when rapid adaptation may be required, such as robotic exploration or autonomous driving.

By combining various contextual hints, our model significantly reduced navigation errors and achieved higher trajectory scores, especially in challenging situations. This emphasizes the importance of context within the autonomous world: the more detailed the information an agent could gather from the scene, the more advanced its decision-making capabilities would be. Hence, our framework provides the foundation for developing even more intelligent systems for navigation with limited data. While these results are promising, many aspects need further research. The future work could include additional modalities such as audio and depth information that help better understand the context. A multimodal approach would permit the model to navigate more complex environments, such as indoor scenes with moving objects or outdoor scenes with varied terrain. Reinforcement learning and curriculum learning are methods that could further optimize the model to learn from simulated environments in order to enhance its generalization capability to practical scenarios, such as navigation on city roads and search-and-rescue operations.

Another very promising line of investigation involves the study of human-AI collaboration in navigation. Indeed, human feedback or interactive learning will make the model more intuitive to use in shared environments like factories or warehouses for general safety and efficiency. While our experiments are performed in controlled and simulated environments, the eventual goal is to deploy our technology in practical applications. Model evaluation in dynamic, unstructured scenes with unpredictable obstacles and different weather conditions will be crucial in establishing the robustness and reliability. In all, this work sets the foundation for creating scalable, flexible, and contextually aware navigation systems with In-Context Learning using VLMs. Our results show excellent navigation performance can be achieved with very minimal data, making this framework quite attractive for a wide range of applications. As these models become increasingly powerful, applications within complex environments will play a significant role in the development of intelligent and adaptive autonomous systems.

A Appendix

A.1 Prompt Engineering

In this section we present the prompts used during the evaluation of the two modalities under study.

A.1.1 TPV Prompt

In the Third Person View scenario, we use a simple yet effective prompt. We explicitly ask to the model to follow the same reasoning used before, since we inject samples in its history. We ask the model to provide the response in JSON format.

```
"""
Given an image from an external camera depicting a room, locate the robot

labeled with a number 0 and the target location marked with a red circle.
Note that the body of the robot under label 0 is not an obstacle.
Based on your previous analysis, apply the same reasoning on this new scenario.
Determine the optimal sequence of numbered points that the robot should follow
to reach the target without encountering any obstacles.

Reply in this JSON format:
{
DANGEROUS: [<number>, <number>..]
COMMANDS: [<number>, <number>..].
}
"""
```

A.1.2 FPV Prompt

In the First Person View scenario, we use a more articulated prompt. The main difference is that in this setup we need to inform the model about the control overlay and we use the prompt to give the model feedbacks as well. Also in this case, we ask the model to provide the response in JSON format.

```
"""
NAVIGATION MEMORY: still_empty.
Remember your navigation memory is just a hint! You must rely on what you see.

TASK: Navigate to a TARGET_OBJ.
You are given an annotated picture of what you can see from your camera.

LAST ACTION:
no_action.

SUGGESTED ACTION:
no_strategy.

COMMAND SYSTEM GUIDE:
The yellow overlay you see is a command system:

- Command 1: Rotate the view 45° right.
- Command 2: Rotate the view 30° right.
- Command 3: Rotate the view 15° right.
- Command 4: Move forward.
- Command 5: Rotate the view 15° left.
- Command 6: Rotate the view 30° left.
- Command 7: Rotate the view 45° left.
- Command 8 (not displayed): Look up.
- Command 9 (not displayed): Look down.

Tips:
- Use point 4 to explore going forward.
```

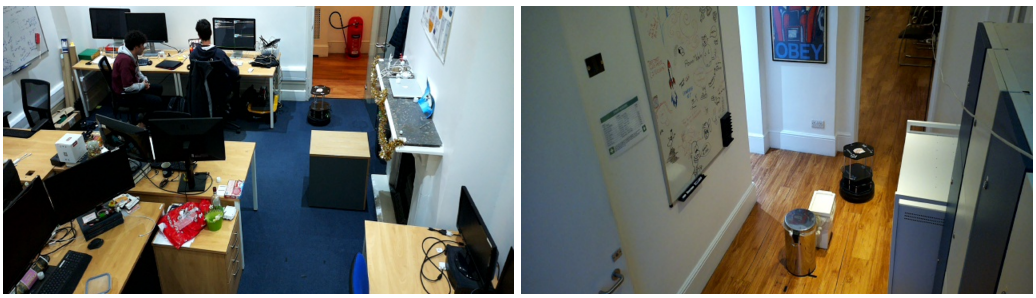
```
- Points 1, 4 and 7 are usually used to explore the room. Try to get into areas where
the likelihood of finding the TARGET_OBJ is higher.
- Points 2, 3, 5, and 6 are typically used to fine-tune your direction towards
the target or to avoid obstacles.

STEPS:
a. Identify objects semantically related to the TARGET_OBJ. Consider whether the target
    is typically found in higher or lower areas and act accordingly.
b. If you are in a closed space, e.g. a corridor, try to move towards open space.
c. If you don't see the TARGET_OBJ in the scene, navigate towards areas where the
    probability of finding the  TARGET_OBJ is higher
    (use objects in the NAVIGATION MEMORY as a reference).
    For example, a cushion is likely found in a bedroom.
d. Keep in mind your body is large, so you need a large turn angle.
    Aim for open areas when avoiding walls.
e. If you are stuck, your priority will be to avoid the obstacles
    going towards an open space.
f. Return also the next command for your future self. In case you
 forget the past.
g. If you are CLOSER THAN 1 METER from the TARGET_OBJ and you are sure of it,
    return command 0 to end the episode.

Reply in this JSON format:
{
    "objects": ["obj1", "obj2", ...],
    "explanation": "explanation",
    "command": "number",
    "next_step_command": "number",
}
"""
```

A.2 Examples of different context samples

In the following sections, we suppose, for simplicity, to be at inference time and with a sample from Camera 1 3.7a.

A.2.1 Scenario D

In Scenario D, we evaluated Camera 1 using samples exclusively from Cameras 2, 3, and 4 to test the model's generalization capability across different perspectives while maintaining consistent operational context and rover configuration. Images were chosen to preserve exact spatial layouts and furniture arrangements across camera viewpoints, ensuring consistent environmental characteristics.



(a) Inference Camera #1 **(b)** Inference Camera #2

Figure A.1: Example of retrieved samples starting from a frame taken by Camera #4. On the left there is a very similar room while on the right a very dissimilar one (thanks to MMR reranking).

Figure A.1 shows two views from Camera 4 with visual similarities to Camera 1, using comparable furniture and color schemes. This setup tests the model's retrieval mechanism for identifying relevant contextual information from diverse sources. Given the high similarities between scenes, we implemented a Maximum Marginal Relevance (MMR) approach as discussed in Section 3. MMR enriches the contextual dataset by varying content without redundancy, stimulating effective decision-making and reinforcing the model's understanding of spatial relationships and navigation dynamics. This approach helps avoid "overfitting" to particular visual features, promoting a more generalized environmental understanding. Scenario D demonstrates the development of robust navigation models through cross-camera evaluations. By using similar images from different cameras, we showcased the model's ability to generalize and adapt to changing visual inputs.

The MMR technique further enhanced context capture and scene interpretation. These findings suggest potential applications in real-world scenarios where autonomous systems must navigate complex environments captured from multiple viewpoints.

A.2.2 Scenario O

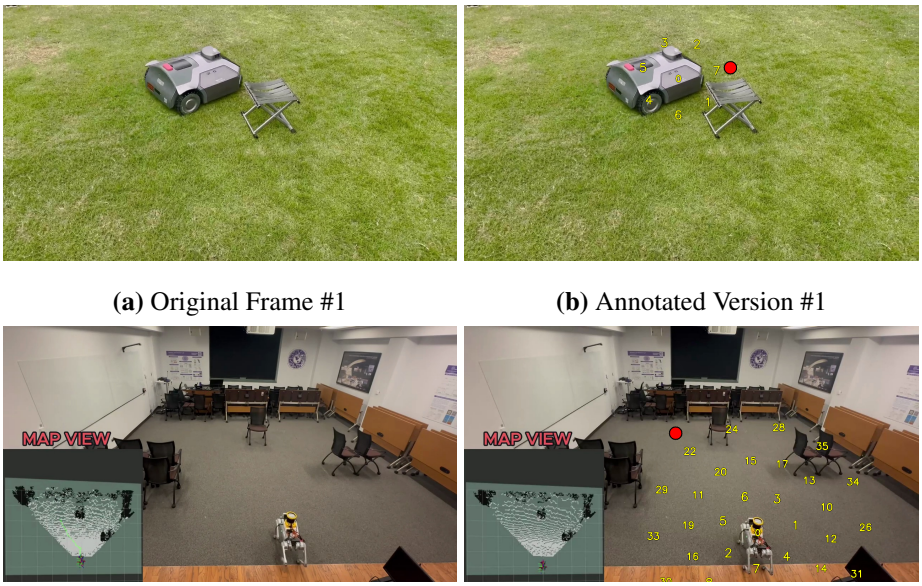
To further demonstrate the flexibility and robustness of our proposed approach, we explored the performance of our model in a unique scenario, labeled as Scenario O. This scenario was specifically designed to test the model's ability to generalize to new environments using real-world data sources very different from the deployment scenaio. For this purpose, we selected two publicly available online videos depicting robots navigating various environments. These videos, sourced from [84] and [83], provided a diverse set of visual contexts and navigation challenges.

From the selected videos, we extracted a total of 30 to 40 frames, capturing different segments of the robots' trajectories. The frames were carefully chosen to include a variety of navigation situations, such as obstacle avoidance and open spaces, to comprehensively test the model's adaptability. Each frame was then automatically annotated to include ground truth about the correct trajectory. These annotations served as the contextual input for our model, simulating a real-world context retrieval scenario. Figure A.2 depicts an example of retrieved samples and annotation result.

A.2.3 Scenario H

In Scenario H, we explored the integration of human-centric contextual information into our navigation system using human activity annotated video sources from CCTV cameras. This approach represents a practical application of our model in environments where visitor behavior influences navigation choices.

We installed CCTV cameras at selected locations to capture real-time human activity footage. The video selections showcased various individuals navigating narrow corridors and avoiding typical obstacles like chairs and desks. This rich contextual information aimed to enhance the model's understanding of human activities within the environment. The generated data from these behavioral observations provided insights into how typical human behavior patterns could inform and constrain navigational choices. This scenario



(a) Original Frame #1 **(b)** Annotated Version #1

(c) Original Frame #2 **(d)** Annotated Version #2

Figure A.2: The figure depicts examples of the original frames extracted by the online footage and their respective annotated versions.

demonstrates the potential for incorporating human activity data to improve autonomous navigation systems in populated environments.

Once the filmed videos were collected, we methodically labelled human movement trajectories as well as points of interest and potential navigational impediments. The formatted footage and annotations would serve as the contextual information for a navigational model to apply human activities to show how human engagement influenced decision-making. Knowledge gained from the processing and labelled annotations of video footage towards human actions and the influences towards navigation, presented possible implications of the model's understanding surrounding evolving situations in a way which made predictions easier within a real-world activity.

The observations, annotations, and framing of the observed dynamic context gave us some sense of how our model behaved to leverage information on intent and predict outcomes based upon human articulated exchange. In terms of navigation accuracy, human behavior contextual information improved navigation choice outcomes significantly. The potential to understand human directional intent was mitigated where the model used human behavioral contextual information as a basis for reduced collision points in enhancing safety during wayfinding. Each input where the model exercised an understanding which displayed human activity or intent allowed for a development in proactive path making

(a) Example Human-Centered Context #1

(b) Example Human-Centered Context #2

Figure A.3: The figure depicts examples of the original frames extracted from CCTV cameras.

choices which might produce enhanced navigation.

Summarizing, Scenario H emphasizes the necessity of embedding data occupying a direct human context in autonomous navigational frameworks. Utilizing footage from CCTV cameras to annotate the context can provide the model with additional structured information related to contexts of human behavior in the model's workspace. Not only does this practice improve navigation performance, it prepares the model's navigation behavior for real-world applications, where the presence and interaction of other humans is an essential element of navigation. Findings presented in this scenario highlight the prospect of advancing more intelligent and adaptive autonomous systems that can navigate a populated space with relative ease and safety.

A.3 Ethical Considerations of VLMs

A.3.1 Biases and Ethical Considerations

Special ethical issues connected with the application of VLM in robotic navigation include bias and discrimination within automated decision-making processes. Similar to many other AI systems, VLMs are able to reflect and amplify the biases involved in their training data. For the most part, a bias arises where unequal treatment based on arbitrary characteristics ultimately leads to discriminatory effects against one group. This form of bias may be in historical data, reflecting inequalities of the past that VLMs could be unaware of solidifying within operational systems.

A.3.2 Impact on Employment

The introduction of automation by technologies such as VLMs is usually associated with changes in employment patterns. Although productivity gains can translate into heightened wealth overall, they often also mean significant labor market disruptions, particularly in those roles that are quite susceptible to automation.

Historical trends tend to point to the shifts in labor, either due to technological changes or otherwise, with estimates as high as 50% reduction of agricultural workers in the UK between the years of 1950 and 1970. As such, technological shifts in productivity and benefits must be carefully considered not only in terms of how benefits are divided but also who may be potentially harmed by said technologies.

A.3.3 Addressing Bias in AI Systems

The efforts at reducing bias from AI or robotic systems are at a very nascent stage. One of the challenges is to create unbiased datasets and a notion of fairness in mathematics that works universally. There are numerous technical approaches to find and reduce bias. But those solutions very often get faced with a lot of restrictions because the sociotechnical systems are very complex, and so is the behavior of humans. These issues have consequently been addressed as different institutional proposals emphasize the need for collaboration between technologists and ethicists in devising frameworks that guarantee the fairness and accountability of AI applications.

A.3.4 Ethical Implications of Automation

The use of VLMs for robotic navigation has ethical implications related to larger issues of justice and equity in society. These debates range from the alarmist views, foreseeing generalized unemployment due to automation, to the optimistic view of those who believe that in the future, productivity increases could generate new types of employment. These are moral dilemmas that policymakers and researchers should consider, factoring in both the immediate effects of technologies on the labor market and long-term consequences in terms of social equity.

References

[1] A. Staroverov, D. A. Yudin, I. Belkin, V. Adeshkin, Y. K. Solomentsev, and A. I. Panov, Real-time object navigation with deep neural networks and hierarchical reinforcement learning, *IEEE Access*, vol. 8 2020, pp. 195 608–195 621, 2020.

[2] K. Zhou, C. Guo, and H. Zhang, "Visual navigation via reinforcement learning and relational reasoning," in *2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation*, 2021, pp. 131–138. DOI: 10.1109/SWC50871.2021.00027.

[3] D. Shah, A. Sridhar, A. Bhorkar, N. Hirose, and S. Levine, *Gnm: A general navigation model to drive any robot*, 2023. arXiv: 2210.03370 [cs.RO]. available from: https://arxiv.org/abs/2210.03370.

[4] Q. Zeng, Q. Yang, S. Dong, *et al.*, *Perceive, reflect, and plan: Designing llm agent for goal-directed city navigation without instructions*, 2024. arXiv: 2408.04168 [cs.AI]. available from: https://arxiv.org/abs/2408.04168.

[5] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, Language models are unsupervised multitask learners, *OpenAI blog*, vol. 1, no. 8 2019, p. 9, 2019.

[6] D. S. W. Williams, M. Gadd, P. Newman, and D. D. Martini, *Masked gamma-ssl: Learning uncertainty estimation via masked image modeling*, 2024. arXiv: 2402.17622 [cs.CV]. available from: https://arxiv.org/abs/2402.17622.

[7] D. S. W. Williams, D. D. Martini, M. Gadd, and P. Newman, Mitigating distributional shift in semantic segmentation via uncertainty estimation from unlabeled data, *IEEE Transactions on Robotics* [online], vol. 40 2024, pp. 3146–3165, 2024. DOI: 10.1109/TRO.2024.3401020.

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

[9] S. Nasiriany, F. Xia, W. Yu, *et al.*, *Pivot: Iterative visual prompting elicits actionable knowledge for vlms*, 2024. arXiv: 2402.07872 [cs.RO]. available from: https://arxiv.org/abs/2402.07872.

[10] A. J. Sathyamoorthy, K. Weerakoon, M. Elnoor, *et al.*, Convoi: Context-aware navigation using vision language models in outdoor and indoor environments, *arXiv preprint arXiv:2403.15637* 2024, 2024.

[11] L. Robinson, M. Gadd, P. Newman, and D. D. Martini, "Robot-relay: Building-wide, calibration-less visual servoing with learned sensor handover networks," in *International Symposium on Experimental Robotics*, Springer, 2023, pp. 129–140.

[12] L. Robinson, D. De Martini, M. Gadd, and P. Newman, "Visual servoing on wheels: Robust robot orientation estimation in remote viewpoint control," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2023, pp. 6364–6370.

[13] D. Zhong, L. Robinson, and D. De Martini, Nerfoot: Robot-footprint estimation for image-based visual servoing, *arXiv preprint arXiv:2408.01251* 2024, 2024.

[14] A. Vaswani, Attention is all you need, *Advances in Neural Information Processing Systems* 2017, 2017.

[15] X. Amatriain, A. Sankar, J. Bing, P. K. Bodigutla, T. J. Hazen, and M. Kazi, Transformer models: An introduction and catalog, *arXiv preprint arXiv:2302.07730* 2023, 2023.

[16] T. B. Brown, Language models are few-shot learners, *arXiv preprint arXiv:2005.14165* 2020, 2020.

[17] J. Devlin, Bert: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv:1810.04805* 2018, 2018.

[18] A. Dosovitskiy, An image is worth 16x16 words: Transformers for image recognition at scale, *arXiv preprint arXiv:2010.11929* 2020, 2020.

[19] R. Fukushima, K. Ota, A. Kanezaki, Y. Sasaki, and Y. Yoshiyasu, *Object memory transformer for object goal navigation*, 2022. arXiv: 2203.14708 [cs.CV]. available from: https://arxiv.org/abs/2203.14708.

[20] D. Shah, A. Sridhar, N. Dashora, *et al.*, *Vint: A foundation model for visual navigation*, 2023. arXiv: 2306.14846 [cs.RO]. available from: https://arxiv.org/abs/2306.14846.

[21] A. Sridhar, D. Shah, C. Glossop, and S. Levine, *Nomad: Goal masked diffusion policies for navigation and exploration*, 2023. arXiv: 2310.07896 [cs.RO]. available from: https://arxiv.org/abs/2310.07896.

[22] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.

[23] S. Antol, A. Agrawal, J. Lu, *et al.*, "Vqa: Visual question answering," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2425–2433.

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

[25] Y. Reddy, *Bridging images and text - a survey of vlms*. available from: https://nanonets.com/blog/bridging-images-and-text-a-survey-of-vlms/.

[26] M. Tsimpoukelli, J. L. Menick, S. Cabi, S. Eslami, O. Vinyals, and F. Hill, Multimodal few-shot learning with frozen language models, *Advances in Neural Information Processing Systems*, vol. 34 2021, pp. 200–212, 2021.

[27] H. Liu, C. Li, Y. Li, and Y. J. Lee, *Improved baselines with visual instruction tuning. corr abs/2310.03744 (2023)*, 2023.

[28] H. Liu, C. Li, Q. Wu, and Y. J. Lee, Visual instruction tuning, *Advances in neural information processing systems*, vol. 36 2024, 2024.

[29] M. He, Y. Liu, B. Wu, *et al.*, Efficient multimodal learning from data-centric perspective, *arXiv preprint arXiv:2402.11530* 2024, 2024.

[30] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," in *International conference on machine learning*, PMLR, 2023, pp. 19 730–19 742.

[31] Q. Zhang, J. Zhang, Y. Xu, and D. Tao, *Vision transformer with quadrangle attention*, 2023. arXiv: 2303.15105 [cs.CV]. available from: https://arxiv.org/abs/2303.15105.

[32] H. Lu, W. Liu, B. Zhang, *et al.*, Deepseek-vl: Towards real-world vision-language understanding, *arXiv preprint arXiv:2403.05525* 2024, 2024.

[33] S. Liu, Z. Zeng, T. Ren, *et al.*, Grounding dino: Marrying dino with grounded pre-training for open-set object detection, *arXiv preprint arXiv:2303.05499* 2023, 2023.

[34] C. Team, Chameleon: Mixed-modal early-fusion foundation models, *arXiv preprint arXiv:2405.09818* 2024, 2024.

[35] J.-B. Alayrac, J. Donahue, P. Luc, *et al.*, Flamingo: A visual language model for few-shot learning, *Advances in neural information processing systems*, vol. 35 2022, pp. 23 716–23 736, 2022.

[36] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV]. available from: https://arxiv.org/abs/1512.03385.

[37] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, *End-to-end object detection with transformers*, 2020. arXiv: 2005.12872 [cs.CV]. available from: https://arxiv.org/abs/2005.12872.

[38] B. Lin, Z. Tang, Y. Ye, *et al.*, Moe-llava: Mixture of experts for large vision-language models, *arXiv preprint arXiv:2401.15947* 2024, 2024.

[39] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 11 975–11 986.

[40] N. Mu, A. Kirillov, D. Wagner, and S. Xie, "Slip: Self-supervision meets language-image pre-training," in *European conference on computer vision*, Springer, 2022, pp. 529–544.

[41] L. Yuan, D. Chen, Y.-L. Chen, *et al.*, Florence: A new foundation model for computer vision, *arXiv preprint arXiv:2111.11432* 2021, 2021.

[42] M. Faysse, H. Sibille, T. Wu, G. Viaud, C. Hudelot, and P. Colombo, Colpali: Efficient document retrieval with vision language models, *arXiv preprint arXiv:2407.01449* 2024, 2024.

[43] B. Xiao, H. Wu, W. Xu, *et al.*, "Florence-2: Advancing a unified representation for a variety of vision tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 4818–4829.

[44] Z.-Y. Dou, A. Kamath, Z. Gan, *et al.*, Coarse-to-fine vision-language pre-training with fusion in the backbone, *Advances in neural information processing systems*, vol. 35 2022, pp. 32 942–32 956, 2022.

[45] Y. Huang, T. Lv, L. Cui, Y. Lu, and F. Wei, "Layoutlmv3: Pre-training for document ai with unified text and image masking," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 4083–4091.

[46] H. Luo, J. Bao, Y. Wu, X. He, and T. Li, "Segclip: Patch aggregation with learnable centers for open-vocabulary semantic segmentation," in *International Conference on Machine Learning*, PMLR, 2023, pp. 23 033–23 044.

[47] A. Singh, R. Hu, V. Goswami, *et al.*, "Flava: A foundational language and vision alignment model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 15 638–15 650.

[48] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 000–16 009.

[49] J. Achiam, S. Adler, S. Agarwal, *et al.*, Gpt-4 technical report, *arXiv preprint arXiv:2303.08774* 2023, 2023.

[50] G. Team, Google. gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, *arXiv preprint arXiv:2403.05530* 2024, 2024.

[51] Anthropic. "The claude 3 model family: Opus, sonnet, haiku anthropic." (2024), available from: `https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618 Model_Card_Claude_3.pdf`.

[52] M. Reid, N. Savinov, D. Teplyashin, *et al.*, Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, *arXiv preprint arXiv:2403.05530* 2024, 2024.

[53] X. Yue, Y. Ni, K. Zhang, *et al.*, *Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. arxiv*, 2023.

[54] C. Fu, P. Chen, Y. Shen, *et al.*, Mme: A comprehensive evaluation benchmark for multimodal large language models, *arXiv preprint arXiv:2306.13394* 2023, 2023.

[55] L. Chen, J. Li, X. Dong, *et al.*, Are we on the right way for evaluating large vision-language models? *arXiv preprint arXiv:2403.20330* 2024, 2024.

[56] P. Lu, H. Bansal, T. Xia, *et al.*, Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts, *arXiv preprint arXiv:2310.02255* 2023, 2023.

[57] A. Kembhavi, M. Salvato, E. Kolve, M. Seo, H. Hajishirzi, and A. Farhadi, "A diagram is worth a dozen images," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, Springer, 2016, pp. 235–251.

[58] P. Lu, S. Mishra, T. Xia, *et al.*, "Learn to explain: Multimodal reasoning via thought chains for science question answering," in *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[59] W. Yu, Z. Yang, L. Ren, *et al.*, Mm-vet v2: A challenging benchmark to evaluate large multimodal models for integrated capabilities, *arXiv preprint arXiv:2408.00765* 2024, 2024.

[60] A. Das, S. Kottur, K. Gupta, *et al.*, "Visual dialog," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 326–335.

[61] F. Li, R. Zhang, H. Zhang, *et al.*, Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models, *arXiv preprint arXiv:2407.07895* 2024, 2024.

[62] G. Zhou, Y. Hong, Z. Wang, X. E. Wang, and Q. Wu, *Navgpt-2: Unleashing navigational reasoning capability for large vision-language models*, 2024. arXiv: 2407.12366 [cs.CV]. available from: https://arxiv.org/abs/2407.12366.

[63] J. Duan, W. Yuan, W. Pumacay, *et al.*, *Manipulate-anything: Automating real-world robots using vision-language models*, 2024. arXiv: 2406.18915 [cs.RO]. available from: https://arxiv.org/abs/2406.18915.

[64] D. Shah, B. Osiński, b. ichter brian, and S. Levine, "Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action," in *Proceedings of The 6th Conference on Robot Learning*, K. Liu, D. Kulic, and J. Ichnowski, Eds., ser. Proceedings of Machine Learning Research, vol. 205, PMLR, 14–18 Dec 2023, pp. 492–504. available from: https://proceedings.mlr.press/v205/shah23b.html.

[65] J. Chen, B. Lin, R. Xu, Z. Chai, X. Liang, and K.-Y. K. Wong, *Mapgpt: Map-guided prompting with adaptive path planning for vision-and-language navigation*, 2024. arXiv: 2401.07314 [cs.AI]. available from: https://arxiv.org/abs/2401.07314.

[66] N. D. Palo and E. Johns, *Keypoint action tokens enable in-context imitation learning in robotics*, 2024. arXiv: 2403.19578 [cs.RO]. available from: https://arxiv.org/abs/2403.19578.

[67] N. Wies, Y. Levine, and A. Shashua, The learnability of in-context learning, *Advances in Neural Information Processing Systems*, vol. 36 2024, 2024.

[68] P. Lewis, E. Perez, A. Piktus, *et al.*, *Retrieval-augmented generation for knowledge-intensive nlp tasks*, 2021. arXiv: 2005.11401 [cs.CL]. available from: https://arxiv.org/abs/2005.11401.

[69] J. Carbonell and J. Stewart, The use of mmr, diversity-based reranking for reordering documents and producing summaries, *SIGIR Forum (ACM Special Interest Group on Information Retrieval)* [online] Jun. 1999, Jun. 1999. DOI: 10.1145/290941.291025.

[70] Z. Liu, Y. Lin, Y. Cao, *et al.*, *Swin transformer: Hierarchical vision transformer using shifted windows*, 2021. arXiv: 2103.14030 [cs.CV]. available from: https://arxiv.org/abs/2103.14030.

[71] E. Kolve, R. Mottaghi, W. Han, *et al.*, *Ai2-thor: An interactive 3d environment for visual ai*, 2022. arXiv: 1712.05474 [cs.CV]. available from: https://arxiv.org/abs/1712.05474.

[72] X. Liang, H. Wang, W. Chen, D. Guo, and T. Liu, Adaptive Image-Based Trajectory Tracking Control of Wheeled Mobile Robots With an Uncalibrated Fixed Camera, *IEEE Transactions on Control Systems Technology* [online], vol. 23, no. 6 Nov. 2015, pp. 2266–2282, Nov. 2015, Conference Name: IEEE Transactions on Control Systems Technology, ISSN: 1558-0865. DOI: 10.1109/TCST.2015.2411627.

[73] X. Liang, H. Wang, Y.-H. Liu, *et al.*, Purely Image-Based Pose Stabilization of Nonholonomic Mobile Robots With a Truly Uncalibrated Overhead Camera, *IEEE Transactions on Robotics* [online], vol. 36, no. 3 Jun. 2020, pp. 724–742, Jun. 2020, Conference Name: IEEE Transactions on Robotics, ISSN: 1941-0468. DOI: 10.1109/TRO.2019.2961052.

[74] A. Kirillov, E. Mintun, N. Ravi, *et al.*, *Segment anything*, 2023. arXiv: 2304.02643 [cs.CV]. available from: https://arxiv.org/abs/2304.02643.

[75] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, *You only look once: Unified, real-time object detection*, 2016. arXiv: 1506.02640 [cs.CV]. available from: https://arxiv.org/abs/1506.02640.

[76] H. Laurençon, L. Tronchon, M. Cord, and V. Sanh, What matters when building vision-language models? *arXiv preprint arXiv:2405.02246* 2024, 2024.

[77] M. Savva, A. Kadian, O. Maksymets, *et al.*, *Habitat: A platform for embodied ai research*, 2019. arXiv: 1904.01201 [cs.CV]. available from: https://arxiv.org/abs/1904.01201.

[78] F. Xia, A. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese, *Gibson env: Real-world perception for embodied agents*, 2018. arXiv: 1808.10654 [cs.AI]. available from: https://arxiv.org/abs/1808.10654.

[79] Z. Wang and G. Tian, Goal-oriented visual semantic navigation using semantic knowledge graph and transformer, *IEEE Transactions on Automation Science and Engineering* 2024, 2024.

[80] W. Yang, X. Wang, A. Farhadi, A. Gupta, and R. Mottaghi, Visual semantic navigation using scene priors, *arXiv preprint arXiv:1810.06543* 2018, 2018.

[81] M. Wortsman, K. Ehsani, M. Rastegari, A. Farhadi, and R. Mottaghi, "Learning to learn how to learn: Self-adaptive visual navigation using meta-learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6750–6759.

[82] H. Du, X. Yu, and L. Zheng, Vtnet: Visual transformer network for object goal navigation, *arXiv preprint arXiv:2105.09447* 2021, 2021.

[83] H. Robotics. "H1 pro obstacle avoidance demo," Youtube. (2023), available from: `https://www.youtube.com/watch?v=FgftdWrSYzM`.

[84] R. Roy. "Unitree Go1 Obstacle Avoidance using Nav2 and 3D SLAM (RTAB MAP)." (), available from: `https://www.youtube.com/watch?v=hL4MTG0u1K0`.