Command-Driven Single-Robot System for Object Detection and Task Execution

AmirHossein Rahimi Bidrouni
MSc Artificial Intelligence

December 15, 2025

Declaration

I declare that I have personally prepared this assignment. The work is my own, carried out personally by me unless otherwise stated and has not been generated using Artificial Intelligence tools unless specified as a clearly stated approved component of the assessment brief. All content remains my own, reflecting my original research, analysis, and writing, in accordance with Aston University guidelines. To ensure the clarity, grammatical accuracy, and academic tone of this work, I used Grammarly to check for grammar errors and enhance sentence clarity.

AmirHossein Rahimi Bidrouni

Abstract

Developing an interactive home robotic system capable of understanding and executing human commands is a key area of research. Specifically, interactive robots that can reliably recognize, validate, and manipulate objects while serving as intelligent user interfaces are essential. In this thesis, a novel modular dual-head architecture was proposed to address these needs. This system integrates a vision-language model (VLM) for conversational scene understanding and a vision-language-action model (VLA) for performing physical tasks, all managed by an intelligent command routing component. The design emphasizes functional specialization, computational efficiency, and ease of maintenance, rather than relying on a single unified AI model. The command router, based on a modified six-layer BERT model(Devlin et al. 2019), classifies user intent (query or action), while the VLM component (Gemma 3n E2B) (Team 2025) handles query processing, and the VLA component (SpatialVLA) (Qu et al. 2025) executes physical tasks. System evaluation showed promising results: the command router achieved high accuracy (93.38%), and the VLM component demonstrated strong performance in real-world environments, with an average accuracy of 80.96% for query validation. However, the VLA component, despite fine-tuning, faced significant challenges in mastering fine motor skills, particularly object grasping, resulting in a low task completion rate of 4% for physical actions. The overall success rate for query commands was 75.6%, whereas for action commands, it was only 3.7%. These findings highlight the necessity for further research to enhance physical manipulation capabilities to develop more effective home assistant robots.

Contents

1 Introduction

In recent years, the concurrent advancements in robotics, computer vision, and natural language processing have opened up new frontiers for the development of intelligent technologies. These systems hold immense potential not only to facilitate the daily lives of the elderly and individuals with disabilities but also to serve as intelligent home assistants for the general public. However, a primary challenge in this domain is the scarcity of interactive robotic systems that can reliably and simultaneously detect, validate, and manipulate objects within an environment, while also acting as an intelligent interface between the user and their home.

Imagine a future in which every home has its own personal robotic assistant, humanoid figure, about the height of a human, gliding along on wheels or walking on two legs. Equipped with articulated arms and advanced sensors, it navigates cluttered rooms with ease, responding to voice or text commands with conversational intelligence. For instance, a user, rushing out the door, might ask, "Where are my keys?" The robot, using its vision-language capabilities, scans the environment and responds, "Your keys are on the table by the entrance." Upon the user's follow-up command, "Bring them to me," the robot employs its arm to pick up the keys and deliver them, saving time and reducing stress. Beyond this, such a robot could fold laundry with precision, wipe down kitchen surfaces, or assist elderly users by fetching medications and engaging in warm conversation to alleviate loneliness. Unlike single-purpose devices, this versatile assistant seamlessly integrates perception and action, enhancing safety, efficiency, and quality of life.

This dissertation presents an intelligent robotic control system that bridges this gap through a novel modular AI architecture for domestic assistance. The system integrates language understanding with physical action execution, where visual perception directly informs robotic behavior. A high-level overview of the system's workflow is illustrated in Figure 1, which depicts the command routing mechanism directing user inputs to specialized modules for query response or task execution.

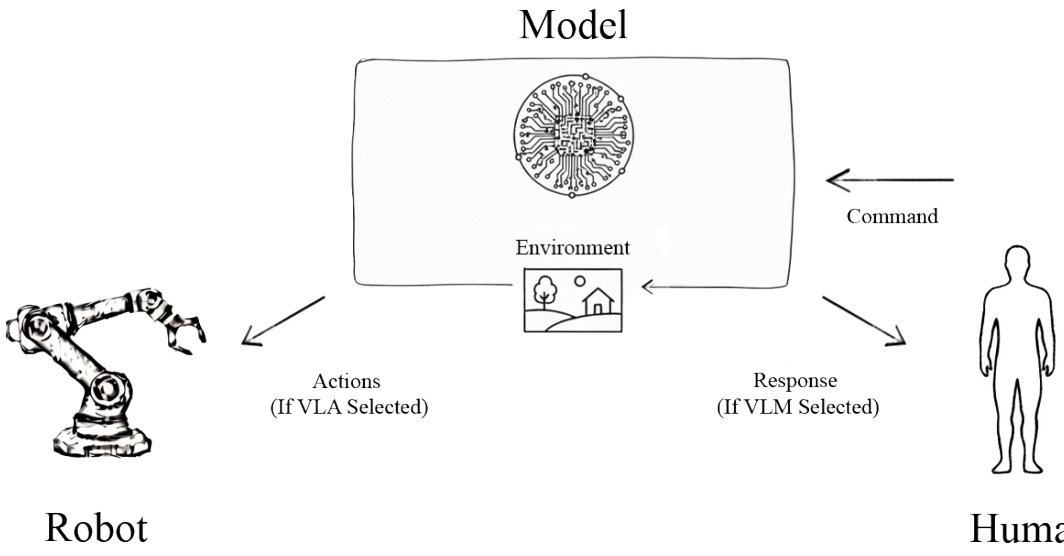


Figure 1: High-level overview of the intelligent robotic control system. User commands are processed by the command router, which directs queries to the VLM head for scene understanding and responses, or action commands to the VLA head for physical task execution.

The initial project concept drew from simpler approaches, such as employing object detection models like YOLO (Redmon et al. 2016) for basic query responses. However, insights from recent advancements in Vision-Language Models (VLMs) and Vision-Language-Action (VLA) models inspired a more sophisticated design, leading to the development of the dual-head architecture detailed in Section 2. This evolution is further explored in Section 5, which outlines the project's progression.

The final architecture of this system is based on an innovative and modular "Dual-Head" design. Instead of employing a single, monolithic, all-purpose AI model, this design divides the system's responsibilities into two highly specialized components. This strategic choice was based on three key principles: functional specialization, computational efficiency, and maintainability.

The workflow of this architecture operates as an intelligent pipeline:

- **Command Routing Module:** At the system's input, an intelligent router, based on a fine-tuned BERT (Devlin et al. 2019) natural language processing model, analyzes the user's command. This module determines whether the command is a query requiring a verbal response or an executive command requiring physical action.
- **Vision-Language Model (VLM) Head for Understanding and Response:** If the command is a query (e.g., "Is the stove light on?"), it is sent to the VLM head. By analyzing the image feed from its camera, this head generates a complete and conversational response.
- **Vision-Language-Action (VLA) Head for Task Execution:** If the command is an executive order (e.g., "Turn off the stove burner"), it is directed to the VLA head. This component is responsible for translating the linguistic command and visual information into a sequence of precise physical movements for the robot arm.

This modular design offers significant advantages. Firstly, by utilizing smaller, more specialized models, the system's computational load is drastically reduced, making it feasible to run on standard, non-specialized home hardware. Secondly, this architecture provides high extensibility and maintainability; each "head" can be independently upgraded with newer models in the future without necessitating a redesign of the entire system.

The primary goal of this work is to develop a computationally efficient system that combines scene understanding and task execution, enabling reliable domestic assistance on standard home hardware. To achieve this, the research addresses the following key questions:

- How can Vision-Language Models (VLMs) and Vision-Language-Action (VLA) models be effectively integrated into a single, modular architecture for domestic robotics?
- What is the performance of such a system in both simulated and real-world environments for query response and physical task execution?
- How does the system balance high-level task comprehension with the challenges of fine-motor control in manipulation tasks?

This dissertation proceeds as follows: Section 2 reviews the foundational and recent literature. Section 3 describes the technical methodology, including the dual-head architecture and the

specific models chosen for each component. Section 4 presents the detailed results of the evaluation for each module and the system as a whole.Section 5 provides an reflection on Project Management, detailing the evolution of the project from its initial concept to its final form. Section 6 provides a holistic discussion of the findings, their implications, and the project's limitations. Finally, Section 7 concludes the dissertation, summarizing the key findings.

2 Literature Review

The development of an interactive domestic robot capable of understanding and acting upon human commands requires integrating advancements in Natural Language Processing (NLP), Computer Vision, and Robotics. This review critically evaluates the literature to inform the architectural and methodological choices of this project, with a focus on creating a computationally efficient system for resource-constrained domestic environments. The analysis is structured around three key themes:

1. the evolution of Vision-Language Models (VLMs) for scene understanding.
2. the emergence of Vision-Language-Action (VLA) models for robotic control.
3. the architectural considerations for a command router to enable a modular, dual-head system.

Central to this project are the requirements of **offline operation for privacy and stability, resource efficiency to run on general-purpose hardware, and the use of open-source models for accessibility and reproducibility**. These principles have profoundly influenced the selection of models and the overall system architecture, guiding the design away from massive, cloud-dependent systems toward a more practical and accessible solution for domestic use.

2.1 Overview of Related Robotic Systems

While this project proposes a specific "Dual-Head" architecture separating query-answering from task execution, it is crucial to situate this approach within the broader landscape of modern robotic systems that also aim to bridge language, vision, and action. Several alternative architectures offer different solutions to the same fundamental challenge: making robots both intelligent and physically competent.

A compelling alternative is the **Dual Process VLA (DP-VLA)** architecture proposed by Han et al. (Han et al. 2024), which is inspired by the dual-process theory of human cognition. Like our system, DP-VLA employs a modular design to enhance efficiency. However, its division of labor is different. DP-VLA separates tasks into a high-frequency "System 1" for real-time motor control and a low-frequency "System 2" for complex reasoning and planning. The large System 2 model (e.g., a VLA like OpenVLA (Kim et al. 2024)) runs only occasionally to generate a guiding "latent feature," while the small System 1 model (e.g., a Behavior Cloning transformer) uses this feature to generate smooth, high-frequency actions. This approach is highly effective for optimizing a single, continuous manipulation task. In contrast, our Dual-Head system creates a higher-level separation: one head is dedicated entirely to non-physical, conversational interaction (VLM), while the other handles the entire action pipeline (VLA). This makes our architecture arguably more specialized for a domestic assistant that must frequently switch between answering questions and performing actions, rather than just optimizing a single action sequence.

Another relevant paradigm is presented by Jiang et al. with **VIMA (Visuomotor Attention agent)** (Jiang et al. 2022). VIMA's core contribution is the unification of a wide spectrum of robot tasks into a single sequence modeling problem using "multimodal prompts" that interleave

text and images. VIMA is designed as a single, generalist transformer-based agent that can perform diverse tasks like one-shot imitation, visual goal reaching, and novel concept grounding, all specified through this flexible prompt interface. While powerful, this "one-model-for-all-tasks" approach represents a different philosophy from our specialized, modular design. A large, monolithic model like VIMA, while generalist, may be less computationally efficient for the specific needs of a domestic assistant, where many user interactions are simple queries that do not require the full power of a complex visuomotor policy. Our system, by routing simple queries to a lightweight VLM, reserves the more powerful (and resource-intensive) VLA only for when it is needed, adhering more strictly to the principle of computational efficiency on consumer hardware.

Finally, the **OmniManip** system. (Pan et al. 2025) offers a solution to one of the most significant challenges in robotics, which this dissertation also identifies: the gap between high-level, commonsense reasoning of VLMs and the low-level precision required for physical manipulation. OmniManip's key innovation is a novel object-centric representation that defines "interaction primitives" (like points and directions) in an object's own canonical 3D space. It uses a VLM not to directly output actions, but to reason about these primitives and define spatial constraints between objects (e.g., "align the teapot's spout axis with the cup's opening axis"). A traditional motion planner then solves these constraints to generate the robot's trajectory. This approach cleverly sidesteps the need to fine-tune a VLM into a VLA, avoiding high data collection costs and potentially improving generalization. While highly innovative, OmniManip's pipeline is complex, relying on multiple components like 6D pose estimators and 3D mesh generators. Our choice to use an end-to-end VLA model, while facing its own challenges in fine-motor control, represents a trade-off in favor of architectural simplicity.

These systems, summarized in Table 1, highlight that there is no single best approach to building a generalist robot. Each architecture makes different trade-offs between modularity, efficiency, and performance. Our Dual-Head system's strength lies in its pragmatic focus on the specific use case of a domestic assistant, where the binary split between understanding queries and executing actions provides a clear, efficient, and maintainable design.

2.2 The Evolution of Vision-Language Models (VLMs) for Scene Understanding

Vision-Language Models (VLMs) are multimodal AI systems that combine computer vision and natural language processing to process visual and textual inputs jointly. They align images and text in a shared embedding space using a dual-encoder setup: a vision encoder (e.g., Vision Transformer (Dosovitskiy et al. 2020)) extracts features from images, while a text encoder (e.g., Transformer) handles language. Fusion occurs via cross-attention or contrastive learning during pre-training on image-text pairs, enabling tasks like visual question answering and scene description. In robotics, VLMs enable semantic environment interpretation for conversational responses.

The ability of a robot to understand its environment and respond to visual queries is fundamentally dependent on the power of its underlying vision-language model. Early approaches

Table 1: Comparison of Robotic System Architectures.

Feature	Our Dual-Head System	DP-VLA (Han et al. 2024)	VIMA (Jiang et al. 2022)	OmniManip (Pan et al. 2025)
Core Architecture	Dual-head, modular. Explicit separation of a VLM for queries and a VLA for actions, managed by a command router.	Hierarchical, dual-process. A large, low-frequency model (L-Sys2) guides a small, high-frequency model (S-Sys1) for a single task.	Monolithic, generalist. A single, large transformer agent designed to handle a wide variety of tasks via a unified multimodal prompt interface.	Hybrid, object-centric. A VLM reasons about object "primitives" to set spatial constraints, which are then solved by a separate motion planner.
Key Innovation	Architectural simplicity and efficiency via functional specialization for a domestic assistant role (answering vs. doing).	Inspired by cognitive science, it optimizes for smooth, real-time motor control by separating high-level planning from low-level execution frequency.	Unifies diverse robotic tasks (imitation, goal-reaching, etc.) into a single sequence modeling problem, maximizing multi-task capability in one model.	Bridges the VLM-to-robot gap without VLA fine-tuning by using VLMs for high-level spatial reasoning on canonical object parts rather than direct action generation.
Suitability for Project Requirements	Explicitly designed for resource efficiency on consumer hardware by using smaller, specialized models and routing tasks appropriately. Open-source models are used throughout.	Highly efficient for continuous action, but its architecture is less focused on handling the mix of conversational queries and action commands needed for a domestic assistant.	A single, large generalist model is likely too computationally intensive for the target hardware. Its monolithic design also conflicts with our project's principle of modularity.	Avoids VLA fine-tuning, which is a major advantage. However, its complex pipeline of multiple models (VLM, pose estimators, 3D generators) may still be resource-intensive.

often treated object detection and language understanding as separate tasks. However, the recent paradigm shift towards end-to-end VLMs has enabled a much richer form of interaction. A comparative overview of leading VLM candidates for this project is shown in Table 2.

Table 2: Comparison of VLM Candidates relevant to this project.

Feature	**SmolVLM 2.2B**[1]	**Qwen2.5 VL 3B**[2]	**Gemma 3n E2B**[3]
Developer	Hugging Face	Alibaba	Google
Published Date	Nov 2024	Jan 2025	Feb 2025
Total Parameters	2.2B	3.75B	5B (1.91B Active)
Context Window	16K	32K	32K
Language Support	1 (English)	29	140

1 2 3

A notable example of this trend is the **Qwen series**, culminating in Qwen2.5-VL (Bai et al. 2025). This model exemplifies the progress in foundational capabilities, demonstrating enhanced visual recognition, precise object localization, and robust document parsing. While powerful, the flagship models are computationally intensive. This highlights a critical trade-off in the field: the balance between state-of-the-art performance and practical feasibility.

This trade-off is directly addressed by research into compact, efficient models. **SmolVLM**, introduced by Marafioti et al. (Marafioti et al. 2025), is a prime example of this counter-trend. Through systematic exploration of architectural configurations and tokenization strategies, the authors demonstrate that smaller, carefully designed models can achieve remarkable performance. Their smallest model, SmolVLM-256M, requires less than 1GB of GPU memory, yet outperforms much larger predecessors. Another strong contender, **Gemma 3n E2B** (Team 2025), offers a unique architecture with Per-Layer Embedding (PLE) caching, which significantly reduces graphical resource usage during inference, along with native support for audio, making it highly extensible. This body of work provides a strong justification for this project's decision to forgo massive, monolithic models in favor of smaller, more efficient architectures.

2.3 The Emergence of Vision-Language-Action (VLA) Models

Vision-Language-Action (VLA) models extend Vision-Language Models (VLMs) by incorporating the ability to translate linguistic and visual inputs into physical actions for robotic control. Built upon pre-trained VLMs, VLA models are fine-tuned on robotic datasets containing action demonstrations, such as joint trajectories or gripper movements. Actions are typically represented as action tokens—discrete or continuous vectors encoding robot motions (e.g., joint angles, velocities, or end-effector positions). These tokens are generated through techniques like flow matching for continuous action distributions or tokenization for discrete movements, integrated into the model's output via fine-tuning. In robotics, VLAs enable language-driven task execution, mapping complex instructions (e.g., "pick up the cup") to precise physical movements, pushing robotics toward generalist, language-driven control.

[1] (Marafioti et al. 2025) [2] (Bai et al. 2025) [3] (Team 2025)

An architectural comparison of the VLA models considered for this project is provided in Table 3. The following paragraphs provide detailed descriptions of these models.

Table 3: Architectural Comparison of VLA Candidates.

Feature	**OpenVLA**[1]	**SpatialVLA** [2]	π_0 **(pi-zero)** [3]
Base VLM	Prismatic-7B [4]	PaliGemma2 [5]	PaliGemma [6]
Vision Encoder	Fused DINOv2 [7] and SigLIP [8] features for combined spatial and semantic understanding.	SigLIP for 2D semantics, enhanced with Ego3D Position Encoding from a predicted depth map for 3D context.	Standard PaliGemma vision components.
Action Generation	Autoregressive discretization of 7D control actions, treated as language tokens.	Autoregressive prediction of spatial action tokens derived from Adaptive Action Grids. Predicts a chunk of T=4 future actions.	Flow matching (a diffusion variant) with a dedicated "action expert" to model continuous action distributions. Supports high-frequency action chunking.
Total Params	7B	3.5B	3.3B
GPU Memory	15GB (bfloat16)	8.5GB (bfloat16)	8GB (bfloat16)

1 2 3 4 5 6 7 8

The concept of a "generalist robot policy" is central to this new paradigm. Black et al. (Black et al. 2024) propose π_0 **(pi-zero)**, a novel architecture built upon a pre-trained VLM. A key innovation in π_0 is the use of a flow matching architecture, which allows the model to represent complex, continuous action distributions. This enables the robot to perform highly dexterous and fluid manipulation skills, such as folding laundry, by fine-tuning on a large and diverse dataset of robot demonstrations. This work demonstrates the immense potential of leveraging Internet-scale semantic knowledge from VLMs and adapting it for complex physical tasks.

Building on this, Kim et al. (Kim et al. 2024) introduced **OpenVLA**, a 7B-parameter open-source VLA trained on nearly a million real-world robot demonstrations. OpenVLA directly fine-tunes a VLM to generate robot actions by treating them as tokens in the language model's vocabulary. Its success demonstrates that a direct, end-to-end approach can yield state-of-the-art results, outperforming even much larger closed-source models. The open-source nature of OpenVLA and its focus on efficient fine-tuning make it a critical resource for the research community and a strong candidate for projects like this one.

[1] (Kim et al. 2024) [2] (Qu et al. 2025) [3] (Black et al. 2024) [4] (Karamcheti et al. 2024) [5] (Steiner et al. 2024) [6] (Beyer et al. 2024) [7] (Oquab et al. 2023) [8] (Zhai et al. 2023)

process injects 3D spatial context into the 2D semantic visual features, thus eliminating the need for specific robot-camera calibration and making it universally applicable across different robot types. Ultimately, this encoding helps the model overcome environmental variations and achieve superior 3D scene spatial understanding and generalizability

However, a key limitation in many VLA models is their reliance on 2D visual information, which can be insufficient for tasks requiring precise 3D interaction. This gap is addressed by Qu et al. (Qu et al. 2025) with **SpatialVLA**. This model explicitly incorporates 3D spatial understanding into the VLA framework. The process used in this method injects 3D spatial context into the 2D semantic visual features, thus eliminating the need for specific robot-camera calibration and making it universally applicable across different robot types. Ultimately, this encoding helps the model overcome environmental variations and achieve superior 3D scene spatial understanding and generalizability. The work of Qu et al (Qu et al. 2025). provides a compelling argument for the importance of 3D spatial representations in robotics and directly informed the selection of SpatialVLA as the VLA head for this project.

2.4 Command Routing for the Dual-Head System

The development of a practical, real-world system requires more than just selecting the most powerful models; it necessitates careful architectural design to balance performance with efficiency. The dual-head architecture of this project, which separates the VLM and VLA functionalities, is a direct response to this need. A critical component of this architecture is the Command Router, which must accurately and efficiently classify user intent.

The choice of an appropriate model for this classification task is crucial. The work of Benayas et al. (Benayas et al. 2024) provides a vital comparative analysis of encoder-only versus decoder-only models for NLU tasks like intent classification. Their research demonstrates that for such tasks, encoder-only models like BERT (Devlin et al. 2019)consistently outperform their larger, decoder-only counterparts. This is because encoder architectures are specifically designed to build a rich, bidirectional representation of the entire input, making them inherently better suited for classification. Furthermore, they found that encoder-only models achieve this superior performance while requiring only a fraction of the computational resources and exhibiting significantly lower latency. This study provides a strong, evidence-based justification for the selection of a BERT-based model (Devlin et al. 2019) for the Command Router in this project, as it ensures both high accuracy and the real-time responsiveness required for a seamless user experience.

In summary, the literature provides a clear trajectory for the development of intelligent robotic systems. The evolution from task-specific models to generalist VLMs and VLAs has opened new possibilities. However, the practical application of these models, particularly in a domestic setting, requires a focus on efficiency and modular design. This project builds upon these insights by integrating a state-of-the-art, spatially-aware VLA, a resource-efficient VLM, and a high-performance encoder-based router into a novel dual-head architecture.

3 Methodology

The methodology for this project is designed as a modular, multi-stage pipeline to develop an interactive robotic system capable of both understanding user queries and executing physical tasks. The architecture is intentionally designed to be practical for a domestic assistance scenario, balancing advanced capabilities with computational efficiency. This chapter details the overall system architecture and the specific design and implementation of each core component.

3.1 Guiding Principles for Model Selection

The selection of models for the proposed dual-head architecture was guided by the three core requirements established in the Literature Review (Section 2): offline capability, resource efficiency, and open-source availability. This section outlines the practical implementation of these principles.

- **Offline Capability** was achieved by prioritizing models with compact architectures that eliminate dependency on cloud-based processing, ensuring privacy and operational stability.
- **Resource Efficiency** was quantified by targeting models that operate within the computational constraints of general-purpose hardware, specifically requiring less than 16GB of GPU memory for inference.
- **Open-Source Availability** was ensured by selecting models with publicly accessible weights and codebases, facilitating reproducibility and future extensions.

3.2 System Architecture: A Dual-Head Approach

To meet the project's dual requirements of verbal validation and physical action, a **dual-head architecture** was designed. This approach diverges from a single, monolithic model in favor of two specialized "heads": a Vision-Language Model (VLM) for scene understanding and a Vision-Language-Action (VLA) model for manipulation. A diagram of this architecture is shown in Figure 2. This design was motivated by several key principles:

- **Functional Specialization:** By decoupling the tasks, each model can be optimized for its specific function. The VLM head is trained to excel at static scene interpretation and visual question answering, while the VLA head is focused on the more complex dynamics of physical interaction and motor control. This specialization allows for higher performance in each domain compared to a single generalist model that must compromise between different objectives.
- **Computational Efficiency:** A single, monolithic model capable of performing both complex reasoning and precise robotic control would be exceptionally large and computationally expensive. The dual-head architecture is inherently more resource-efficient because it allows for the use of models with computational demands matched to their

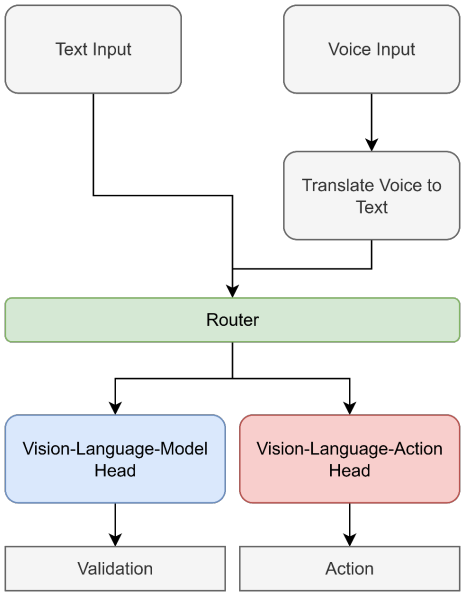


Figure 2: High-level overview of the dual-head system architecture. User commands are routed to either the VLM head for validation or the VLA head for task execution.

specific function. For instance, the simple classification required for command routing can be effectively handled by a lightweight encoder model (such as the 6-layer BERT used in this work), while the more resource-intensive VLA head is reserved exclusively for the complex task of physical manipulation. This targeted allocation of computational resources makes the system more feasible to run on the general-purpose hardware found in domestic environments.

- **Modularity and Maintainability:** This design offers significant advantages from a software engineering perspective. Each head functions as an independent module. This means the VLA model, for instance, could be upgraded to a more advanced architecture in the future without requiring the entire system to be retrained. This modularity simplifies development, testing, and future improvements.

The system's workflow, detailed in the following subsections, begins with a user command (either text or voice) being processed by a **Command Routing Module**, which intelligently directs the command to the appropriate head—either the VLM for queries or the VLA for action requests.

3.3 Command Pre-processing and Routing

This initial module is responsible for interpreting the user's intent. It processes raw user input and classifies it to ensure the correct robotic capability is activated.

3.3.1 Command Input

The system is designed to be flexible, accepting commands through two modalities:

1. **Voice Input:** Spoken commands are transcribed into text using the **Whisper model** (Radford et al. 2023), known for its good accuracy in speech-to-text conversion.

2. **Text Input:** Users can also input commands directly as text through the system's interface.

3.3.2 The Routing Classifier

Once the command is in text format, it is passed to a classification model to determine the user's intent.

An **encoder-only**, Transformer-based model, **BERT (Bidirectional Encoder Representations from Transformers)** (Devlin et al. 2019), was chosen for this task. As discussed in the literature review 2, encoder-only architectures are purpose-built for tasks requiring a deep understanding of the input text, such as intent classification (Benayas et al. 2024). They tend to outperform decoder-only models in classification while being significantly more computationally efficient. The specific model used was `bert-base-multilingual-uncased` for its broad language support. To further optimize for efficiency, the standard 12-layer architecture was reduced to **6 layers**, which was found to provide optimal accuracy for this task, as detailed in the evaluation (Section 4.2).

The classification task was intentionally designed as a binary problem to simplify the system's core logic. User inputs are categorized into one of two classes:

- **Class 0 (Non-Action/Query):** A broad, "catch-all" class for all other inputs, including validation questions, general conversation, and out-of-scope requests (e.g., "Is the water glass on the table?", "Hello, how are you?", "What's the weather like?").
- **Class 1 (Action):** For commands requiring physical robot action (e.g., "Bring me the water glass").

This binary design is robust because any command that is not a direct, executable instruction is routed to the VLM head. The VLM, being a powerful generative model, is well-equipped to handle such queries gracefully. For example, if a user asks an unanswerable question like, "Do you think my wife likes my new shirt?", the VLM can generate a helpful, context-aware response such as, "As a robot, I can't offer opinions on fashion, but I can tell you what I see." This approach leverages the strengths of each component: the router performs a simple, efficient classification, while the VLM handles the complexity of nuanced human conversation.

Due to the lack of a publicly available dataset for this specific routing task, a large-scale synthetic dataset was generated. This was a critical part of the project, ensuring the router could be trained to be robust and generalizable. The generation process was as follows:

1. **Prompt Engineering for Template Generation:** The Gemma 3n model (Team 2025) (the same model used for the VLM head) was prompted with carefully crafted instructions (e.g., "Design several questions for picking up an object from a table") to generate a diverse set of command templates.

2. **Template Categories:** The dataset was designed to cover five main linguistic styles for both Action and Non-Action classes to ensure robustness against real-world user speech: Formal, Colloquial, Grammatically Incorrect, Non-Question Commands, and General Conversation.

3. **Programmatic Expansion:** Each template contained an ‘object‘ placeholder, which was systematically replaced with items from a predefined list of common household objects. This approach programmatically expanded the dataset to cover a wide range of scenarios and, more importantly, enhanced the model’s ability to **generalize**. By training on varied objects within the same sentence structures, the model learns to recognize the underlying intent of the command rather than overfitting to specific keywords.

Due to the lack of publicly available datasets tailored for this specific routing task at the time of this report, a synthetic dataset was created. The dataset was designed to be comprehensive, covering a wide variety of linguistic styles to ensure the router’s robustness in real-world scenarios. With the assistance of the Gemma 3n model (Team 2025) (the same model used for the VLM head) and carefully crafted prompts (e.g., "Design several questions for picking up an object from a table"), a diverse set of command templates was generated.

The dataset has a template-based structure where each command contains an ‘object‘ placeholder. This placeholder is systematically replaced with items from a predefined list of various common household objects. This approach serves two critical purposes: first, it programmatically expands the dataset to cover a much wider range of scenarios than would be feasible with manual annotation. Second, and more importantly, it enhances the model’s ability to **generalize**. By training on varied objects within the same sentence structures, the model learns to recognize the underlying intent of the command (e.g., the action of "picking up") rather than overfitting to specific object names. This ensures the model focuses on the semantic structure of the request, not just keywords.

The dataset encompasses five main categories for both Action (Class 1) and Non-Action/ Query (Class 0) commands:

- **Formal Questions:** Structurally formal commands (e.g., Class 0: "Do you detect a {object}?"; Class 1: "Please pick up the {object}.").

- **Colloquial Questions:** Informal, everyday language (e.g., Class 0: "Hey, you see a {object}?"; Class 1: "Just nab the {object} for me, aight?").

- **Grammatically Incorrect Sentences:** Commands with common grammatical errors to test robustness (e.g., Class 0: "The {object}, it’s on the table, right?"; Class 1: "Get me that {object}, you can?").

- **Non-Question Commands:** Imperative or declarative statements (e.g., Class 0: "Tell me if you see the {object}."; Class 1: "Get that {object} for me.").

- **General Conversation:** Common conversational phrases, relevant only to Class 0 to handle non-task-oriented interactions (e.g., "Hello, how are you?").

The final dataset was split into training, validation, and test sets, as detailed in Table 4. The validation and test sets were made of comparable size to ensure a reliable evaluation of the model's performance.

Table 4: Split of the synthetic dataset for command classification.

Dataset Split	Sample Count
Training	≈129,000
Validation	≈49,000
Test	≈7,000
Total	**185,000**

3.4 The VLM Head for Query Validation

This head is activated for "Non-Action/Query" commands and is responsible for answering user questions based on the robot's camera feed.

Based on the project's requirements for a high-accuracy model that can run efficiently on resource-constrained hardware, and informed by the comparative analysis in the literature review (Table 2), **Gemma 3n E2B** (Team 2025) was selected. This decision was based on several unique advantages that align closely with the project's long-term goals:

- **Resource Efficiency via PLE Caching:** The Gemma 3n architecture (Team 2025) utilizes a technique known as PLE (Per-Layer Embedding) caching. This allows the model to cache approximately 2.55B of its parameters, significantly reducing the graphical resource usage during inference. This feature makes it exceptionally well-suited for systems with limited resources.
- **Superior Multimodality and Extensibility:** Unlike the other candidates, Gemma 3n E2B (Team 2025) natively supports not only image and text but also audio inputs. This inherent capability provides a clear path for future enhancements, such as enabling more natural, voice-based interactions with the robot, a topic further explored in the Discussion section 6.
- **Extensive Language Support:** With support for 140 languages, the model offers a robust foundation for developing a system that can be adapted for a global user base.

The selection of Gemma 3n E2B (Team 2025) reinforces the project's commitment to creating a private, stable, and extensible offline system.

3.5 The VLA Head for Task Execution

This head is the most complex component of the system, responsible for translating a user's action command and visual input into a sequence of physical robot movements.

Based on a comprehensive analysis of the models presented in the literature review (Table 3), **SpatialVLA** (Qu et al. 2025) was selected as the VLA head. While all three candidates represent the state-of-the-art, SpatialVLA offered the most compelling combination of innovative features, performance, and practical feasibility for this project. The key justifications are as follows:

1. **Superior Spatial Awareness:** Unlike models that primarily rely on 2D image features, SpatialVLA is explicitly designed for **3D spatial understanding**. As detailed in (Qu et al. 2025), it uses techniques like *Ego3D Position Encoding* by integrating a depth map (predicted by a ZoeDepth model) to construct a 3D point cloud. This provides the model with crucial depth information, which is critical for the precision required in manipulation tasks like grasping and placing objects. This architectural choice directly addresses the project's core challenge of accurate object interaction in a physical space.

2. **Efficiency through Future Action Prediction:** A standout feature of SpatialVLA is its action chunking mechanism. As stated in (Qu et al. 2025), "the SpatialVLA policy predicts a chunk of T=4 future actions... and executes the ensemble actions before predicting the next chunk." This allows the system to achieve a higher effective control frequency (approx. 20Hz) with fewer inference calls, reducing the long-term computational load on the system. This is a significant advantage for a resource-constrained domestic robot.

3. **Superior Zero-Shot Performance:** The ultimate goal of a generalist model is to perform well without task-specific training. According to the benchmark results published in (Qu et al. 2025), SpatialVLA demonstrates superior zero-shot capabilities. In the "SimplerEnv Google Robot(Li et al. 2024)" task, SpatialVLA achieved an average Visual Matching success rate of **71.9%**, outperforming both π_0 (Black et al. 2024) (70.1%) and, most notably, OpenVLA (Kim et al. 2024) (27.7%). This strong baseline performance indicates the model's robust potential and its ability to generalize effectively from its pre-training data.

These factors, combined with its lower resource requirements compared to OpenVLA (Kim et al. 2024), made SpatialVLA (Qu et al. 2025) the most logical and promising choice for the VLA component of this project.

3.5.1 Fine-Tuning for the Target Task

To adapt the general-purpose SpatialVLA model for this project's specific needs, a fine-tuning stage was implemented. This process was conducted within a custom-built simulation environment designed to replicate a common domestic manipulation scenario.

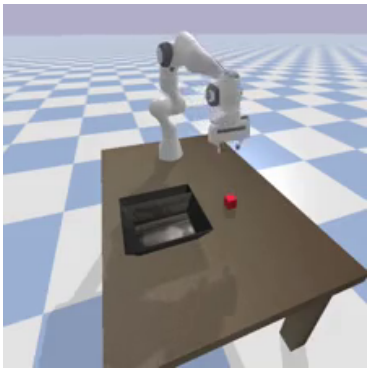
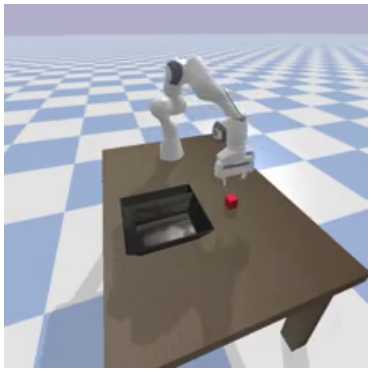
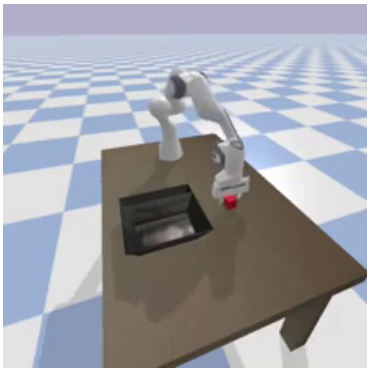
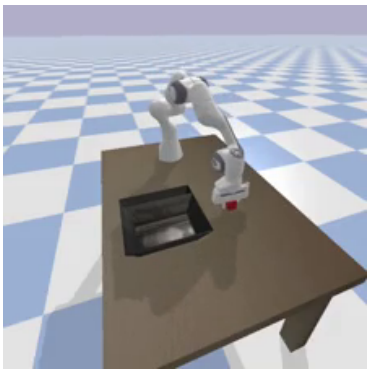
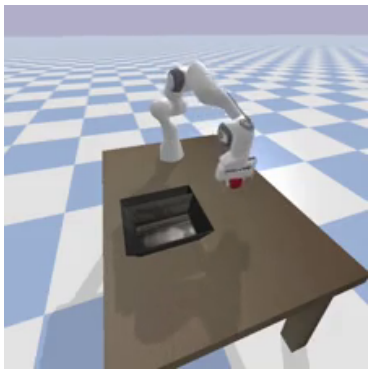
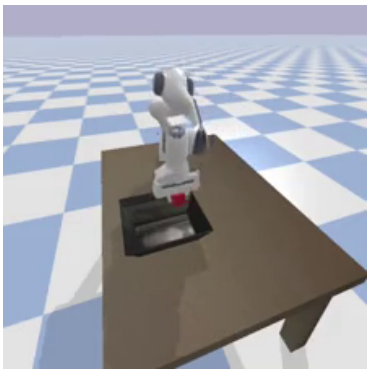
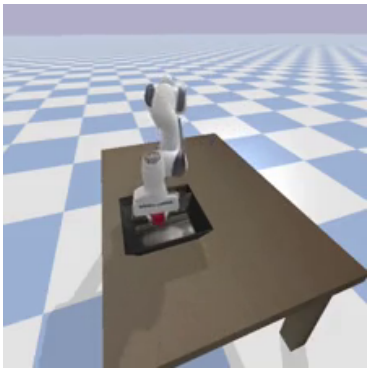
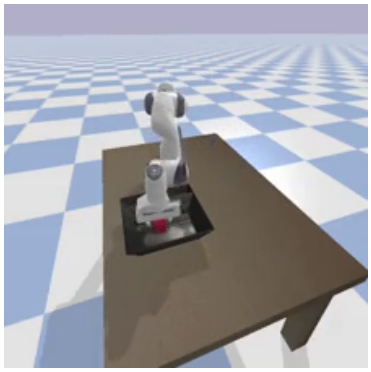
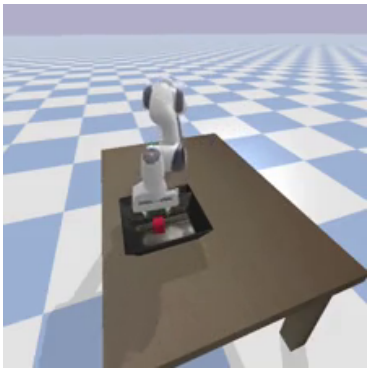
1. **Environment and Task Definition** The environment was developed using the **PyBullet physics simulator** (Coumans & Bai 2016). It consists of a tabletop scene featuring a static **Franka Emika** robotic arm, a cube, and a tray. The core task for the robot is a pick-and-place operation: to grasp the cube and place it inside the tray. To ensure the model learns a generalizable skill rather than memorizing specific locations, the environment introduces randomization in every training and evaluation episode: the positions of the cube and the tray on the table are randomized, and the color of the cube is also randomly changed.

2. **Performance of the Pre-Trained Model** Initial tests of the pre-trained, general-purpose SpatialVLA model (Qu et al. 2025) in this newly defined environment revealed a significant performance gap. The model struggled to perform the task reliably, a common challenge when transferring generalist models to specific, unseen environments. This observation necessitated the collection of a task-specific dataset for fine-tuning.

3. **Dataset Collection** To train the SpatialVLA model (Qu et al. 2025), a dataset of **100 demonstration episodes** was generated programmatically in a PyBullet simulation (Coumans & Bai 2016) using a scripted expert policy. A Franka Panda arm performed a pick-and-place task, moving a cube to a tray, with randomized cube and tray positions (x, y $\in$ [0.1, 0.3] or [-0.3, -0.1], z = 0.7 for cube, 0.63 for tray) and cube colors (RGBA [0.2, 0.8]) for diversity. The expert policy controlled the arm's six DoF (x, y, z for position; fixed orientation with yaw aligned to cube-tray direction) via predefined waypoints: moving above the cube, grasping (z - 0.015), lifting, moving to the tray, and releasing. Each episode recorded observations (224x224 RGB images, 7D proprioception: position, orientation, gripper state), 7D actions (position delta, zeroed orientation delta, gripper command), and a random language prompt (e.g., "place the cube in the tray"). Data was saved in TensorFlow Datasets (TFDS) format, compatible with SpatialVLA's fine-tuning via Low-Rank Adaptation (LoRA) (Hu et al. 2022). Generating 100 episodes required careful tuning of waypoints and physics parameters, ensuring robust data despite simulation challenges like cube settling.

4. **Results and Validation** After fine-tuning, the model showed a dramatic improvement in task comprehension and execution. It successfully learned the sequence of the task: navigating to the cube, moving to the tray's location, and executing the release command. However, a small but consistent error in grasping precision remained; the gripper would close *near* the cube rather than directly on it. While this error prevented full task completion, this experiment successfully validated that the SpatialVLA architecture (Qu et al. 2025) can be effectively adapted to a new, specific task with only a modest amount of fine-tuning data, proving the viability of the chosen approach.

Table 5: Sample episode from the dataset used for fine-tuning SpatialVLA (Qu et al. 2025). The sequence of nine images illustrates the pick-and-place task (moving a cube to a tray) with the language prompt: “place the cube in the tray”. Each image corresponds to a waypoint in the expert policy, capturing key stages such as approaching the cube, grasping, lifting, and placing in the tray.

Step 1	Step 2	Step 3
		
Step 4	Step 5	Step 6
		
Step 7	Step 8	Step 9
		

4 Results and Analysis

This chapter presents the comprehensive evaluation of the system's core components and its overall end-to-end performance. The experiments were designed to quantify the effectiveness of each module and to diagnose the primary bottlenecks in the system's functionality.

4.1 Experimental Setup

All experiments, including model fine-tuning and evaluation, were conducted on a single machine, reinforcing the project's goal of running on consumer-grade hardware. The specifications of the system are as follows:

- **Laptop:** Asus ROG Zephyrus G15
- **CPU:** AMD Ryzen™ 9 6900HS
- **GPU:** NVIDIA® GeForce RTX™ 3070 Ti Laptop GPU (8GB VRAM)
- **RAM:** 16GB DDR5
- **Operating System:** Windows 11

This setup is representative of a high-end personal computer, not a specialized research server, which is crucial for evaluating the feasibility of the proposed system in a domestic setting.

4.2 Command Router: Evaluation and Fine-Tuning

The Command Router is a critical component that enables the dual-head architecture to function. It must accurately classify the user's intent to invoke either the VLM or VLA head. As established in the methodology, a BERT-based model (Devlin et al. 2019) was selected for its proven effectiveness in NLU classification tasks (Benayas et al. 2024). This section details the evaluation process for optimizing this router.

4.2.1 Layer Pruning for Efficiency

To adhere to the principle of resource efficiency, an evaluation was conducted to find the optimal number of Transformer layers for the BERT model (Devlin et al. 2019). The goal was to identify the minimum number of layers required to achieve maximum accuracy on our custom dataset, thereby minimizing both latency and memory usage. The full 12-layer 'bert-base-multilingual-uncased' model was systematically pruned, and the accuracy was evaluated at each step. The results are presented in Table 6.

The evaluation indicates that for this specific binary classification task, a reduced architecture with just **six layers** achieves the same level of accuracy as the full 12-layer model. Increasing the number of layers beyond six yielded no performance benefit while increasing computational cost. Therefore, a 6-layer configuration was selected as it provides the optimal trade-off between performance and efficiency.

Table 6: Accuracy of the BERT-based router vs. number of Transformer layers.

Total Layers	Test Accuracy (%)
2	86.12
4	90.32
6	**93.38**
8	92.40
10	91.94
12	91.82

4.2.2 Multilingual Performance and Transfer Learning

To build a more generalizable model, the multilingual capabilities of the chosen BERT model (Devlin et al. 2019) were evaluated. An experiment was conducted to measure the effect of **transfer learning** across languages. The model was incrementally fine-tuned on English (EN), Persian (FA), German (DE), and Spanish (ES) data. After each stage, its performance was tested on these languages, as well as on two unseen languages: French (FR) and Russian (RU). The results are shown in Table 7.

Table 7: Cross-lingual classification accuracy (%) of the router.

Training Data	EN	FA	DE	ES	FR (Unseen)	RU (Unseen)
EN only	**93.38**	79.44	79.90	79.38	80.59	82.03
EN, FA	90.26	**89.92**	82.03	83.47	85.71	**92.63**
EN, FA, DE	90.50	86.69	**86.00**	77.71	84.91	92.28
EN, FA, DE, ES	92.86	87.50	78.23	**88.94**	**91.24**	87.15

The results in Table 7 vividly illustrate the surprising effectiveness of multilingual models, a phenomenon often referred to as "zero-shot cross-lingual transfer" (Wu & Dredze 2019). When trained only on English, the model performs well on English but struggles to generalize to other languages. However, as data from more languages (Persian, German, Spanish) are added to the fine-tuning process, a significant improvement is observed not only in the seen languages but, crucially, also in the completely unseen languages (French and Russian).

This cross-lingual transfer is possible because the `bert-base-multilingual-uncased` model was pre-trained on a massive corpus from 104 languages. As explained by Wu and Dredze (Wu & Dredze 2019), this process allows the model to develop a language-agnostic, or "interlingua," representation space. This is facilitated by two main factors: (1) a shared subword vocabulary (WordPiece), where tokens for similar concepts may overlap across languages (e.g., numbers, proper nouns, or loanwords), and (2) the deep, hierarchical structure of the Transformer, which learns to abstract away from surface-level lexical features towards more general semantic and syntactic patterns. When we fine-tune this model on our classification task, it learns the task's fundamental structure (distinguishing an action from a query) within this shared representation space. This learned skill is not tied to English vocabulary alone; rather, it is a higher-level capability that can be applied to any language the model has been pre-trained on. Therefore, training on a diverse set of four languages forces the model to learn more robust,

language-independent features for the classification task, which ultimately benefits all languages in its pre-trained repertoire.

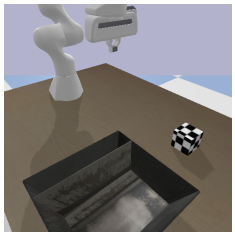
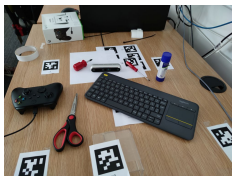
4.3 VLM Head: Qualitative and Quantitative Evaluation

The performance of the selected **Gemma 3n E2B** model was evaluated on its primary task: answering user queries based on visual input. Given the generative nature of the model, which produces free-form text responses, a purely automated metric is insufficient. Therefore, a hybrid evaluation approach combining qualitative and quantitative methods was employed to form a holistic assessment of the model's capabilities. For all tests, a consistent system role was provided to the model to prime its responses for the assistive robotics context: *"You are a helpful assistant for a user operating a mechanical hand. Answer questions clearly and provide relevant information based on user needs. If a mechanical hand is visible in the image, ignore it and focus on describing or analyzing other parts of the image as requested."*

4.3.1 Qualitative Evaluation

A qualitative assessment was performed using a set of 20 images captured from the Aston university lab environment and two images captured from pybullet simulation (Coumans & Bai 2016), representing typical domestic scenes. Because the correctness of a generated sentence can be subjective and context-dependent, a human evaluation approach was deemed essential. This method allows for a nuanced judgment of response quality, considering not just factual accuracy but also helpfulness, clarity, and relevance. Table 8 shows a sample of these evaluations, demonstrating the model's ability to handle both positive and negative existence queries, as well as general scene descriptions.

Table 8: Sample qualitative evaluation of the VLM head.

Image	User Command	Model Response
	What do you see on the table?	On the table, I see a rectangular metal container with a slightly open top. To the right of the container, there is a small, black and white checkered cube. Above the table, a white robotic arm with a gripper is positioned, seemingly looking down at the cube and the container.
	I can't find glue do you see it? Answer with Yes or No.	Yes. There is a blue tube of what appears to be adhesive or glue on the desk.
	What do you see on the table?	On the table, I see a few items: **A piece of white paper** with the word **SURESH** printed in red letters. **A small wooden cube**. **Some pieces of white tape** arranged in a somewhat rectangular shape. **A small stack of brown cardboard pieces** in the upper left corner. **A gray surface** that appears to be a table. There are also some blurry dark shapes in the upper corners, which might be part of the mechanical hand or other objects out of focus.
	I can't find glasses do you see it? Answer with Yes or No.	Yes.

4.3.2 Quantitative Evaluation

To provide a more structured and replicable measure of performance, a quantitative test was conducted across three different environments. For each environment, the model's accuracy was tested on two fixed prompt templates targeting different objects: "I can't find {object}, do you see it?" and "What do you see on the table?". The accuracy for each response was scored as correct (1) or incorrect (0) by a human evaluator, allowing for the calculation of an average

success rate. The results are summarized in Table 9.

Table 9: Quantitative accuracy (%) of the VLM head across different environments.

Environment	**Avg. Accuracy (%)**
Env 1 (PyBullet (Coumans & Bai 2016) Sim)	100
Env 2 (Lab Setup A)	82.9
Env 3 (Lab Setup B)	79.02

The results in Table 9 indicate exceptional performance in the controlled PyBullet simulation, where the model achieved a perfect 100% accuracy. This demonstrates the model's fundamental capability to understand the scenes and queries when presented with clean, synthetic data. As expected, a performance drop was observed when transitioning to real-world environments. In Lab Setup A, the accuracy was 82.9%, and in the slightly more cluttered Lab Setup B, it was 79.02%. This modest degradation is characteristic of the 'sim-to-real' gap and can be attributed to real-world complexities such as variable lighting, reflections, and subtle object textures not present in the simulation. Nevertheless, an average accuracy of over 80% in real-world settings confirms that the Gemma 2-2B model is a robust and effective choice for the VLM head.

4.4 VLA Head: Fine-Tuning and Performance Analysis

The VLA head, powered by **SpatialVLA** (Qu et al. 2025), was evaluated on its ability to execute a pick-and-place task in a PyBullet (Coumans & Bai 2016) simulation. This evaluation was designed to rigorously assess two key aspects: (1) the performance improvement gained by fine-tuning a generalist model on a specific task, and (2) the model's ability to learn the semantic sequence of a task versus the low-level motor skills required for execution. The evaluation compared the performance of the original, pre-trained SpatialVLA model (Qu et al. 2025) against a version that was fine-tuned specifically for this environment.

4.4.1 Fine-Tuning Procedure

The fine-tuning was conducted in a simulated environment containing a table with a cube and a tray placed at random positions. The color of the cube was also randomized to improve model generalization. The model was fine-tuned over 100 episodes using Low-Rank Adaptation (LoRA) (Hu et al. 2022), a parameter-efficient fine-tuning (PEFT) (Xu et al. 2023) technique. LoRA (Hu et al. 2022) was chosen as it allows for adapting large models on resource-constrained hardware by avoiding the need to retrain all of the model's parameters. Instead, it injects small, trainable "adapter" matrices into the model's layers. The behavior of these adapters is primarily controlled by two key hyperparameters:

- **LoRA Rank (r):** This determines the capacity, or the number of trainable parameters, in the adapter matrices. A higher rank allows for more expressive adaptations but increases the computational cost.

- **LoRA Alpha (α):** This acts as a scaling factor for the adaptations. A common and effective practice is to set alpha equal to the rank or twice the rank to provide a stable starting point for training.

For this project, a rank of 16 was chosen as it represents a widely-used value that offers a strong balance between model expressiveness and computational efficiency. Accordingly, alpha was also set to 16, following the common heuristic. The full list of key hyperparameters was:

- LoRA Rank: 16
- LoRA Alpha: 16
- Target Modules: Linear Layers
- Precision: bfloat16
- Epochs: 5
- Learning Rate: 1e-5 with a cosine scheduler
- Weight Decay: 0.01

During training, the model's accuracy on the validation set, which measures the correctness of the predicted action tokens, increased substantially from 20% to 87% after 5 epochs. This steep learning curve indicates that the model was successfully adapting to the specific patterns of the new task.

4.4.2 Performance Evaluation

To deconstruct the model's behavior, its performance was measured against a sequence of four distinct sub-task checkpoints over 100 new, unseen test episodes. In each episode, the initial positions of the cube and tray, as well as the cube's color, were randomized to ensure the evaluation tested the model's generalization capabilities. The prompt was always: "Put the cube in the tray". This checkpoint-based approach allows for a granular analysis of where the model succeeds and fails.

1. **Checkpoint 1 (Reach):** Tests the model's ability to visually locate the target object and plan a path. Success is defined as the gripper reaching a position within 1cm of the cube.
2. **Checkpoint 2 (Release over Tray):** Tests the model's understanding of the goal location. Success is defined as the gripper moving above the tray and opening.
3. **Checkpoint 3 (Grasp):** Tests the fine-motor skill of grasping. Success is defined as the gripper successfully closing on and lifting the cube.
4. **Checkpoint 4 (Task Completion):** The final goal, requiring all previous steps to succeed in sequence. Success is defined as the cube being placed inside the tray.

The results of this evaluation are presented in Table 10.

Table 10: Success rate (%) of VLA models at different task checkpoints. Results are averaged over 100 randomized test episodes.

Checkpoint	Original Model	Fine-tuned Model
1: Reach Cube	0%	64%
2: Release over Tray	0%	78%
3: Grasp Cube	0%	9%
4: Task Completion	0%	4%

The results in Table 10 clearly demonstrate the profound impact of fine-tuning. The original, general-purpose model failed at all checkpoints, indicating its inability to generalize to this specific task without further training. In contrast, the fine-tuned model shows a remarkable ability to understand the task's semantic structure, successfully navigating to the cube (64%) and then to the tray (78%) in a high percentage of trials.

However, the most critical insight comes from the sharp performance drop in the physical interaction stages. The model only managed to successfully grasp the cube in 9% of trials, leading to an overall task completion rate of just 4%. This stark contrast reveals a crucial distinction between learning the semantic sequence of a task (the "what to do") and mastering the fine-grained motor precision required for physical interaction (the "how to do it"). While fine-tuning was highly effective at teaching the model the task's logic, it was insufficient to overcome the inherent challenges of learning precise, contact-rich manipulation skills from a limited number of demonstrations.

4.5 Overall System Performance Analysis

The end-to-end performance of the robotic system is a product of the accuracy of its individual components operating in sequence. For a user command to be successfully executed, both the Command Router and the corresponding VLM or VLA head must perform correctly. An error at any stage will result in a system-level failure from the user's perspective. This dependency makes it crucial to analyze the combined probability of success.

This relationship can be modeled academically as a conditional probability. The overall success rate of the system for a given type of command (Action or Query) is the accuracy of the router multiplied by the accuracy of the relevant head, given that the command was routed correctly. This calculation provides a realistic, albeit conservative, estimate of the system's end-to-end reliability. It represents the **worst-case scenario** or a **lower bound** on performance, as it assumes no error recovery mechanisms are in place.

The formula for the overall system success rate is:

$$P(\text{System Success}) = P(\text{Router Correct}) \times P(\text{Head Correct}|\text{Router Correct}) \quad (1)$$

Using the peak performance values obtained from our evaluations:

- Router Accuracy (6-layer, EN-only): **93.38%** (from Table 6)
- VLM Head Accuracy (Avg. Real-World): **80.96%** (average of 82.9% and 79.02% from

Table 9)

- VLA Head Accuracy (Task Completion): **4%** (from Table 10)

The estimated end-to-end success rates are therefore:

- **For a Query Command (VLM path):**

$$P(\text{Query Success}) = 0.9338 \times 0.8096 \approx 0.756 \text{ or } \mathbf{75.6\%} \tag{2}$$

- **For an Action Command (VLA path):**

$$P(\text{Action Success}) = 0.9338 \times 0.04 \approx 0.037 \text{ or } \mathbf{3.7\%} \tag{3}$$

Analysis of Overall Performance This analysis reveals a significant disparity between the two main functionalities of the system. The query-validation path demonstrates a reasonably high end-to-end success rate of 75.6%, suggesting that the robot can be a reliable source of information for users. However, the action-execution path shows a very low success rate of only 3.7%. This highlights that the primary bottleneck in the system's overall performance is the VLA head's inability to master fine-motor skills, specifically grasping. Even with a highly accurate router, the low success rate of the final action component drastically reduces the system's utility for physical tasks. This underscores the critical importance of improving the physical manipulation capabilities to make the system a truly effective domestic assistant.

5 Reflection on Project Management

This chapter details the project management approach used during this project, from the initial idea to the final product. Good project management was key to handling the complexities of combining advanced robotics with modern AI models. The approach was a mix of clear initial planning, making flexible changes when new challenges appeared, and careful risk management. This story explains how the project's goals changed over time, how the work was carried out, and the key decisions that led to the final system.

5.1 Initial Planning and Strategic Evolution

The project started with a basic plan that set out timeframes for key steps, including a literature review, setting up the environment, developing the models, and evaluation. While this plan provided a starting point, the project's technical direction changed in an important way based on what was learned during the early research phase.

The original idea was to use a simple object detection model, like YOLO (Redmon et al. 2016), to answer user questions about whether an object was present. This method would have given simple yes/no answers. However, deeper research showed the powerful potential of modern Vision-Language Models (VLMs)(Zhang et al. 2024). It became clear that a VLM could offer a much better and more helpful user experience by not only confirming an object's presence but also giving descriptive, conversational answers. This realization led to the first major change in direction.

At the same time, looking into recent progress in robotics showed the rise of Vision-Language-Action (VLA) models. These models were a major breakthrough in robotic control, able to turn high-level language commands directly into physical actions. The decision was made to take on this more challenging and modern approach by using a VLA model for the task execution part of the project.

This strategic change from a single object detector to a more advanced **dual-head architecture** (a VLM for validation and a VLA for action) was an important and deliberate decision. It was made to give the project more technical depth, a greater chance for innovation, and to make sure it was up-to-date with the latest research. This change in goals and design was discussed and agreed upon with the project supervisor, and the project's success criteria were updated for this new, harder goal. This early change set the stage for a more rewarding project with more technical depth.

5.2 Project Execution and Monitoring

A practical, hands-on approach was used to manage the project's daily tasks and track its progress. Instead of formal project management software, a system of personal notes was used to keep a detailed and updated list of tasks. This list worked like a personal backlog, containing completed tasks, tasks yet to be done, bugs and technical issues that were found, and specific needs for testing. This simple method provided a flexible yet organized way to track progress and change priorities as needed on a daily and weekly basis.

Code management was handled by using and building on top of existing public code from repositories like GitHub. For example, the fine-tuning of the SpatialVLA (Qu et al. 2025) model was based on the official code provided by its authors. This was a difficult task, as the original code was designed for a multi-GPU Linux system and had to be carefully changed to run on a single-GPU Windows system. This process of learning from, adapting, and adding to modern code was a central part of the development work.

Regular meetings with the project supervisor were the main way to formally track progress and make key decisions. These meetings were important checkpoints to discuss technical problems, confirm strategic choices, and make sure the project stayed on track with its academic goals.

5.3 Managing Technical Challenges and Deviations

The execution phase of the project involved several major technical challenges that needed flexible problem-solving and led to delays in the original timeline.

A key part of the architecture, the Command Router, needed a dataset to be trained to classify user commands. A critical early challenge was that no public dataset was available for this specific task. To solve this problem, a creative decision was made to generate a custom synthetic dataset. As described in Section 4.2, the Gemma 3n E2B model—the same one chosen for the VLM head—was used with well-designed prompts to create a large and varied dataset. This new approach not only solved the immediate problem but also showed an efficient way of using the project's own tools to help with its own development.

The practical implementation of the VLM and VLA heads created the biggest technical difficulties. This included an ongoing challenge of managing software dependencies, which involved dealing with complex and often conflicting library versions for different models. There were also platform-specific problems, such as not being able to build Flash Attention on Windows. This prevented a possible performance improvement and meant that a slower alternative had to be used.

The biggest challenge was adapting the SpatialVLA model (Qu et al. 2025). The official fine-tuning code needed major changes to move from a Linux, multi-GPU setup to a single-GPU, Windows system. This required re-configuring the core code and using parameter-efficient fine-tuning (PEFT) (Xu et al. 2023) with Low-Rank Adaptation (LoRA) (Hu et al. 2022) to make the training process possible to run on the available hardware.

As planned, the pre-trained SpatialVLA (Qu et al. 2025) model was first tested in a custom PyBullet simulation (Coumans & Bai 2016). Its performance was very poor, as it failed to follow even the simplest commands. This was a turning point. The solution was to fine-tune the model, for which 100 demonstration episodes were carefully collected.

The fine-tuning process led to a surprising and interesting result: the model became very good at understanding the *sequence* of the task (like reaching for the cube, then moving to the tray) but consistently failed at the fine-motor skill of grasping. This discovery was a key moment

in the project. It changed the focus of the evaluation from trying to achieve full task completion to analyzing and documenting this important difference between learning a task's logic and mastering its physical execution.

These challenges, especially those related to the complex VLA model, caused the "Environment setup, model selection, and fine-tuning" phase to take about **two weeks** longer than planned. This significant delay was managed by making later tasks more efficient and changing their order, which allowed the project to be finished within the overall deadline.

5.4 Risk Management in Practice

The project started with an awareness of potential risks, two of which actually happened and needed to be actively managed.

The first, and most expected risk, was the **high computational requirement** of large AI models. This risk was managed from the start by carefully selecting the models, with a focus on balancing performance with efficiency. This strategy can be seen in the choice of a smaller 6-layer BERT (Devlin et al. 2019) for the router and, importantly, the selection of SpatialVLA (Qu et al. 2025) (3.5B parameters) over much larger options like OpenVLA (7B parameters) (Kim et al. 2024). This decision was a direct and successful way to reduce the risk of the system being too demanding for the available hardware.

The second, more serious risk that happened was the **lack of prior experience with the details of complex VLA models**. The steep learning curve needed to understand the SpatialVLA (Qu et al. 2025) architecture, adapt its code, and diagnose its subtle performance issues was the main reason for the two-week delay. This risk was handled by spending dedicated time studying research papers and documentation, along with a continuous cycle of coding, testing, and fixing bugs. The final result of this process was a deep analysis of the model's strengths and weaknesses, which turned a project risk into an important research discovery.

5.5 Conclusion

The management of this project was a case of flexible planning and overcoming technical problems. While an initial plan provided a basic structure, the project's final success came from the ability to change direction based on new research opportunities (like adopting VLMs) and to methodically solve unexpected technical problems (like the VLA fine-tuning issues). The delay was managed by solving problems and carefully adjusting the schedule. The final system came not from a strict plan, but from a flexible and responsive management style that accepted complexity and turned challenges into the project's most important findings.

6 Discussion

This project set out to design and implement a modular, dual-head robotic system for domestic assistance, balancing advanced AI capabilities with computational efficiency. The evaluation of this system yielded a set of nuanced findings that warrant a holistic discussion, reflecting not only on the results themselves but also on the project's journey and its implications for the broader field of assistive robotics.

The Success of Modularity and Efficiency The dual-head architecture proved to be a highly effective design choice. The Command Router, with its pruned 6-layer BERT model, achieved high accuracy (93.38%) while remaining computationally lightweight, confirming that specialized encoder-only models are superior for such NLU tasks. Similarly, the VLM head (Gemma 3n-E2B) performed robustly, providing helpful and accurate responses with over 80% accuracy in real-world settings. These successes validate the core hypothesis that a modular design using smaller, specialized models is a viable and resource-efficient strategy for domestic robotics, avoiding the high computational cost of a single, monolithic model.

The "What to Do" vs. "How to Do It" Dilemma The most significant academic contribution of this work emerged from the VLA head's performance. The fine-tuned SpatialVLA model demonstrated a strong ability to learn the semantic and sequential logic of the pick-and-place task—successfully reaching for the cube (64%) and moving to the tray (78%). This indicates that modern VLAs can effectively learn "what to do" from a surprisingly small number of demonstrations. However, the drastic drop in performance for the grasping phase (9% success) highlights a profound challenge: learning the fine-motor skills for physical interaction, or "how to do it." This discrepancy suggests that the high-level, semantic knowledge transferred from large pre-trained models does not easily translate into the low-level, continuous control needed for precise manipulation. This finding is critical, as it points to a fundamental gap in current VLA models: they excel at reasoning but struggle with the physics of interaction.

Reflection on Project Evolution and Challenges In retrospect, the project's most insightful outcome was a direct result of its most significant challenge. The initial plan to use a simple object detector evolved into a more ambitious dual-head architecture based on emerging research. While this increased complexity, it positioned the project at the forefront of current methodologies. The unforeseen difficulty in fine-tuning the SpatialVLA model, which caused a two-week delay, was initially a setback. However, the meticulous analysis of its failure to grasp became the centerpiece of this dissertation's contribution. This journey underscores a key aspect of research: unexpected results and technical hurdles often lead to the most valuable insights.

Limitations and Future Work: Bridging the Gap to a Generalist Assistant The primary limitation of this work is the low end-to-end success rate for action tasks (3.7%), which is a direct consequence of the VLA's grasping deficiency. This bottleneck prevents the system from being a fully effective physical assistant. Reflecting on the grander vision of a household assistant

painted in the introduction, this work clarifies what is truly missing: the bridge from high-level reasoning to low-level physical competency.

Simply "working harder" by collecting massive datasets for a single, general-purpose VLA may not be the most efficient path forward. The "what to do" vs. "how to do it" dilemma suggests a more structured approach is needed. A more promising strategy, which aligns with this project's modular philosophy, would be to develop a hierarchy of models. In such a system, a high-level VLM could act as a task planner, while a router directs commands not to a single VLA, but to a suite of smaller, task-specific VLAs, each expertly trained on a particular skill (e.g., a "grasping expert," a "wiping expert," or a "pouring expert").

A tangible and achievable first milestone on this path would be to focus on mastering a single, fundamental skill. For instance, developing a VLA model exclusively for robust grasping of various objects under different conditions would be a significant contribution. This "grasping expert" could then be integrated as the first specialized module in a larger, routed architecture. This approach breaks down the monumental challenge of creating a generalist robot into a series of manageable, solvable problems, paving a more practical road toward the capable domestic assistants of the future. Further research should also prioritize sim-to-real transfer and the integration of richer sensory data, such as tactile feedback, to give these models the physical grounding they currently lack.

7 Conclusion

This dissertation successfully designed, implemented, and evaluated a novel dual-head robotic system that integrates natural language understanding with physical action for domestic assistance. The project's core contribution is its demonstration of a modular and computationally efficient architecture that leverages specialized AI models for distinct tasks—a robust approach for resource-constrained environments.

The system proved highly effective in its conversational and scene-understanding capabilities. However, the evaluation of its physical task execution revealed a critical insight and the main finding of this work: a profound gap between a model's ability to learn the high-level semantic sequence of a task versus the fine-motor skills required for physical interaction. While fine-tuning on a small dataset was sufficient to teach the robot "what to do," it was markedly insufficient for teaching it "how to do it." This distinction highlights a key challenge for the next generation of Vision-Language-Action models. Ultimately, this work validates the promise of modular AI architectures in robotics while clearly identifying the mastery of physical interaction as the pivotal frontier for future research in creating truly capable domestic assistants.

References

Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., Zhong, H., Zhu, Y., Yang, M., Li, Z., Wan, J., Wang, P., Ding, W., Fu, Z., Xu, Y., Ye, J., Zhang, X., Xie, T., Cheng, Z., Zhang, H., Yang, Z., Xu, H. & Lin, J. (2025), 'Qwen2.5-vl technical report', *arXiv preprint arXiv:2502.13923* .

Benayas, A., Sicilia, M. A. & Mora-Cantallops, M. (2024), 'A comparative analysis of encoder only and decoder only models in intent classification and sentiment analysis: Navigating the trade-offs in model size and performance', *Language Resources and Evaluation* pp. 1–24.

Beyer, L., Steiner, A., Pinto, A. S., Kolesnikov, A., Wang, X., Salz, D., Neumann, M., Alabdulmohsin, I., Tschannen, M., Bugliarello, E. et al. (2024), 'Paligemma: A versatile 3b vlm for transfer', *arXiv preprint arXiv:2407.07726* .

Black, K., Brown, N., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B. et al. (2024), 'π0: A vision-language-action flow model for general robot control. corr, abs/2410.24164, 2024. doi: 10.48550', *arXiv preprint ARXIV.2410.24164* .

Coumans, E. & Bai, Y. (2016), 'Pybullet, a python module for physics simulation for games, robotics and machine learning'.

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2019), Bert: Pre-training of deep bidirectional transformers for language understanding, *in* 'Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)', pp. 4171–4186.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S. et al. (2020), 'An image is worth 16x16 words: Transformers for image recognition at scale', *arXiv preprint arXiv:2010.11929* .

Han, B., Kim, J. & Jang, J. (2024), 'A dual process vla: Efficient robotic manipulation leveraging vlm', *arXiv preprint arXiv:2410.15549* .

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W. et al. (2022), 'Lora: Low-rank adaptation of large language models.', *ICLR* **1**(2), 3.

Jiang, Y., Gupta, A., Zhang, Z., Wang, G., Dou, Y., Chen, Y., Fei-Fei, L., Anandkumar, A., Zhu, Y. & Fan, L. (2022), 'Vima: General robot manipulation with multimodal prompts', *arXiv preprint arXiv:2210.03094* **2**(3), 6.

Karamcheti, S., Nair, S., Balakrishna, A., Liang, P., Kollar, T. & Sadigh, D. (2024), Prismatic vlms: Investigating the design space of visually-conditioned language models, *in* 'Forty-first International Conference on Machine Learning'.

Kim, M. J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., Rafailov, R., Foster, E., Lam, G., Sanketi, P. et al. (2024), 'Openvla: An open-source vision-language-action model', *arXiv preprint arXiv:2406.09246* .

Li, X., Hsu, K., Gu, J., Pertsch, K., Mees, O., Walke, H. R., Fu, C., Lunawat, I., Sieh, I., Kirmani, S. et al. (2024), 'Evaluating real-world robot manipulation policies in simulation', *arXiv preprint arXiv:2405.05941* .

Marafioti, A., Zohar, O., Farré, M., Noyan, M., Bakouch, E., Cuenca, P., Zakka, C., Allal, L. B., Lozhkov, A., Tazi, N. et al. (2025), 'Smolvlm: Redefining small and efficient multimodal models', *arXiv preprint arXiv:2504.05299* .

Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A. et al. (2023), 'Dinov2: Learning robust visual features without supervision', *arXiv preprint arXiv:2304.07193* .

Pan, M., Zhang, J., Wu, T., Zhao, Y., Gao, W. & Dong, H. (2025), Omnimanip: Towards general robotic manipulation via object-centric interaction primitives as spatial constraints, *in* 'Proceedings of the Computer Vision and Pattern Recognition Conference', pp. 17359–17369.

Qu, D., Song, H., Chen, Q., Yao, Y., Ye, X., Ding, Y., Wang, Z., Gu, J., Zhao, B., Wang, D. et al. (2025), 'Spatialvla: Exploring spatial representations for visual-language-action model', *arXiv preprint arXiv:2501.15830* .

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C. & Sutskever, I. (2023), Robust speech recognition via large-scale weak supervision, *in* 'International conference on machine learning', PMLR, pp. 28492–28518.

Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. (2016), You only look once: Unified, real-time object detection, *in* 'Proceedings of the IEEE conference on computer vision and pattern recognition', pp. 779–788.

Steiner, A., Pinto, A. S., Tschannen, M., Keysers, D., Wang, X., Bitton, Y., Gritsenko, A., Minderer, M., Sherbondy, A., Long, S. et al. (2024), 'Paligemma 2: A family of versatile vlms for transfer', *arXiv preprint arXiv:2412.03555* .

Team, G. (2025), 'Gemma 3n'.
URL: *https://ai.google.dev/gemma/docs/gemma-3n*

Wu, S. & Dredze, M. (2019), 'Beto, bentz, becas: The surprising cross-lingual effectiveness of bert', *arXiv preprint arXiv:1904.09077* .

Xu, L., Xie, H., Qin, S.-Z. J., Tao, X. & Wang, F. L. (2023), 'Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment', *arXiv preprint arXiv:2312.12148* .

Zhai, X., Mustafa, B., Kolesnikov, A. & Beyer, L. (2023), Sigmoid loss for language image pre-training, *in* 'Proceedings of the IEEE/CVF international conference on computer vision', pp. 11975–11986.

Zhang, J., Huang, J., Jin, S. & Lu, S. (2024), 'Vision-language models for vision tasks: A survey', *IEEE transactions on pattern analysis and machine intelligence* **46**(8), 5625–5644.