Politecnico di Torino

Politecnico di Torino

Master's Degree in Computer Engineering

A.y. 2025/2026

Graduation Session March 2026

Scene-graphs based robotic intelligence for visually impaired people support in domestic environments

Supervisor:
Prof. Giuseppe Averta

Candidate:
Francesca Porcelli

Abstract

For individuals with visual impairments, the absence of visual feedback can make everyday household activities more difficult and potentially unsafe, underscoring the need for assistive technologies that offer reliable spatial awareness and accessible forms of interaction.

This thesis presents a mobile robotic assistive system designed to support visually impaired users in domestic environments. The system is deployed using the TRON1 robot operating in a wheeled configuration, which provides stable indoor mobility while maintaining an elevated, human-like sensing perspective well suited to household spaces. This embodiment allows the robot to observe and reason about the environment from viewpoints that closely reflect how people naturally interact with objects and rooms.

The system brings together autonomous perception, semantic scene understanding, voice-based interaction, and task-level reasoning within a unified and modular architecture. The robot builds detailed three-dimensional representations of indoor environments, which are enriched with semantic information and organized into a scene graph capturing rooms, objects, and their spatial relationships. This structured representation acts as a cognitive map, enabling high-level reasoning, navigation, and natural grounding of language in the physical world.

Voice-based interaction serves as the primary interface between the user and the robot, recognizing speech as a natural and accessible communication modality for visually impaired individuals. By relying on spoken commands and verbal feedback rather than visual interfaces, the system allows users to request assistance, ask questions about their surroundings, and receive meaningful contextual information in a hands-free and intuitive manner. Building on this foundation, the system supports assistive tasks such as object finding, spatial environment description, out-of-place and hazard detection, and food label scanning.

This work demonstrates how the integration of embodied semantic perception with voice-centered interaction can deliver practical and intuitive domestic assistance, contributing to greater autonomy, safety, and situational awareness for visually impaired users.

Table of Contents

List of Figures

Chapter 1

Introduction

Visual perception plays a fundamental role in everyday interaction with domestic environments, enabling people to navigate spaces, locate objects, and assess potential hazards with minimal cognitive effort. For individuals with visual impairments, the absence or reduction of visual feedback significantly alters this interaction, often making routine household activities complex, time-consuming, and potentially unsafe. Despite advances in accessibility technologies, many domestic environments remain challenging to navigate independently, highlighting the need for assistive solutions that can provide spatial awareness, contextual understanding, and intuitive interaction.

Recent progress in robotics, computer vision, and artificial intelligence has opened new possibilities for developing intelligent assistive systems capable of operating autonomously in indoor settings. In particular, mobile robots equipped with advanced perception and interaction capabilities offer the potential to act as embodied assistants, bridging the gap between physical environments and accessible information. Unlike static assistive devices or purely software-based solutions, robotic systems can actively explore their surroundings, perceive the environment from multiple viewpoints, and physically position themselves to support user requests.

However, effective domestic assistance requires more than basic mobility or obstacle avoidance. Homes are inherently semantic spaces, characterized by rooms, furniture, objects, and their relationships, which are often referenced using natural language. For an assistive robot to be useful to visually impaired users, it must be able to reason about this semantic structure, understand high-level user intents, and communicate information in an accessible and natural manner. This necessitates the integration of autonomous navigation, semantic scene understanding, and voice-based interaction within a unified system.

This thesis addresses these challenges by presenting the design and implementation of a mobile robotic system for domestic assistance aimed at visually impaired

users. The proposed system combines autonomous navigation, visual perception, semantic environment representation, and natural language interaction to support everyday household tasks. By enabling the robot to explore and understand its environment, interpret spoken user commands, and provide meaningful verbal feedback, the system seeks to enhance independence, safety, and situational awareness in domestic settings.

The remainder of this chapter introduces the broader context and motivation for this work. Section 1.1 reviews recent developments in assistive robotics for domestic environments. Section 1.2 discusses the specific challenges faced by visually impaired individuals at home. Section 1.3 motivates the use of semantic understanding and voice interaction as key enablers for accessible robotic assistance. Section 1.4 outlines the target use cases that guide the system design.

1.1 Assistive Robotics in Domestic Environments

Recent advances in robotics and artificial intelligence have enabled the development of assistive robotic systems capable of operating in domestic environments, with the goal of supporting individuals in their daily activities and increasing their level of independence. In particular, assistive robotics has emerged as a promising solution for visually impaired people, who often face significant challenges within their own homes due to limited access to spatial information, object localization, and environmental awareness. Common domestic tasks such as finding objects, understanding room layouts, identifying potential hazards, or accessing product information can become complex and cognitively demanding without visual feedback.

Several research efforts have demonstrated that mobile robots can effectively assist visually impaired users by providing navigation support and contextual information in indoor environments. Existing systems typically focus on wayfinding, obstacle detection, and guidance through verbal or auditory feedback, showing that semantic spatial information and voice-based interaction can substantially improve safety and orientation during indoor navigation [1, 2]. These approaches highlight the importance of combining autonomous navigation with accessible human–robot interaction to address real-world domestic needs.

However, domestic assistance extends beyond navigation alone. A home environment is inherently dynamic and semantically rich, requiring an assistive robot to reason about rooms, objects, and their relationships in order to support higher-level tasks. For visually impaired users, this includes not only being guided through space, but also understanding where objects are located, whether items are misplaced or potentially dangerous, and accessing information that is typically obtained visually, such as expiration dates or ingredient lists on product packaging. These requirements motivate the adoption of semantic environment representations and

natural voice interaction as key components of an effective domestic assistive robot.

In this context, the work presented in this thesis focuses on the design of a mobile robotic system for domestic environments that combines autonomous navigation, semantic scene understanding, and voice-based interaction to assist visually impaired users in everyday household scenarios.

1.2 Challenges Faced by Visually Impaired People at Home

Visually impaired individuals encounter a wide range of challenges in domestic environments that go beyond simple mobility and significantly impact independence and safety. Without reliable visual feedback, basic tasks such as navigating indoor spaces, locating objects, and understanding the spatial layout of a home can become cognitively demanding and anxiety-inducing, requiring substantial effort and planning to perform safely and efficiently. Traditional home automation systems, while offering potential support, often present accessibility barriers because they are designed primarily for sighted users, with interfaces and interactions that rely heavily on visual cues and lack accessible alternatives, such as meaningful auditory feedback, tactile or speech-based controls [3].

In addition, although voice assistants and smart devices have proliferated in recent years, studies on their impact indicate that current technological solutions do not fully meet the needs of people with visual impairments, as many smart home applications and interfaces remain difficult to navigate without sighted assistance or compatible accessibility features [3]. Moreover, general smart home systems tend to assume visual engagement for tasks such as appliance operation, device configuration, and menu navigation, which can hinder independence and increase reliance on external help [3]. These challenges are compounded by the absence of contextual environmental information — such as the location of obstacles, the spatial arrangement of rooms, or the presence of potentially hazardous objects — which vision typically provides effortlessly.

Together, these limitations highlight the need for assistive solutions that integrate robust spatial awareness, accessible interaction modes, and real-time contextual feedback to support visually impaired users in their daily domestic activities.

1.3 Motivation for Semantic Understanding and Voice Interaction

The limitations of traditional assistive technologies and smart home interfaces underscore the need for more advanced interaction paradigms in domestic robotics.

While early assistive systems have demonstrated the potential of programmed behaviors or simple guidance, they often lack the ability to interpret high-level user intent or provide rich contextual feedback about the environment. For example, emerging research in open-vocabulary mobile manipulation highlights that enabling a robot to understand and act upon natural language commands in a home environment requires the integration of perception, language understanding, navigation, and manipulation capabilities, thereby motivating semantic representations that link language to world knowledge [4]. Such semantic understanding allows robots to go beyond fixed task scripts and interpret user requests in terms of objects and spatial relations within the environment.

In parallel, voice interaction has become a key enabler for accessible human–robot interaction, particularly for users who cannot rely on visual interfaces due to disability. Studies on voice assistant usage show that speech interfaces can significantly improve users' ability to interact with technology in a hands-free and intuitive manner, increasing task efficiency and user satisfaction, and making robotics systems more accessible to people with a range of abilities [5]. For assistive robots in domestic contexts, voice interaction provides a natural channel for users to express their needs, ask for spatial information, and receive feedback without the cognitive load and accessibility barriers associated with conventional graphical user interfaces. Furthermore, research incorporating verbal and semantic feedback into assistive navigation systems demonstrates that combining navigation with voice cues and semantic context can improve both guidance quality and user experience [6].

Taken together, these insights motivate the adoption of semantic environment representations and robust voice interaction in assistive domestic robots. Semantic representations enable robots to reason about objects, rooms, and their relationships in a way that closely maps to human language and conceptualization, while voice interaction provides an accessible and intuitive interface for users to interact with the system, especially where visual feedback is limited or unavailable.

1.4 Target Use Case

The primary goal of this work is to develop a robotic system capable of providing domestic assistance to visually impaired users, helping them maintain autonomy, safety, and situational awareness within their homes. In this context, the robot is envisioned as an intelligent assistant that complements the user's abilities by offering support in everyday tasks that are otherwise difficult or time-consuming without visual input. Key functionalities of the system are driven by typical domestic scenarios that pose challenges for visually impaired individuals.

First, the robot assists in **object finding**, enabling users to locate household

items such as personal belongings, kitchenware, or frequently used tools by understanding their spatial context and guiding the user through the environment. Second, the system supports **spatial awareness**, providing verbal descriptions of room layouts, object locations, and relative positions within the home, which helps users form a mental map and navigate safely and efficiently. Third, it contributes to **safety monitoring** by identifying hazards or misplaced objects—such as obstacles, spills, or potentially dangerous items—and notifying the user through accessible feedback. Finally, the robot facilitates access to **product information**, including details such as expiration dates and ingredients on packaged goods, which are typically obtained visually, thus enabling informed and independent decision-making in domestic routines.

These scenarios collectively define the scope of the system and illustrate how semantic understanding of the environment and natural voice interaction can be leveraged to address real-world needs of visually impaired individuals in domestic settings.

Chapter 2

Related Work and Background

The development of an assistive robotic system for visually impaired users requires the integration of multiple research areas, including mobile robotics, visual perception, semantic environment representation, voice-based human–robot interaction, and autonomous navigation. Each of these fields has evolved substantially in recent years, producing a wide range of approaches that address specific subproblems in isolation. However, effective domestic assistance emerges only when these components are combined into a coherent, embodied system capable of perceiving, reasoning, and interacting within real-world environments.

This chapter reviews the state of the art and foundational concepts that underpin the system presented in this thesis. Rather than providing an exhaustive survey of each domain, the focus is on identifying key ideas, representative systems, and design trends that directly inform the architectural and methodological choices adopted in later chapters. Particular emphasis is placed on approaches that address accessibility, semantic understanding, and interaction in domestic settings, as these aspects are critical for supporting visually impaired users.

The chapter begins by examining prior work on assistive robots for visually impaired users, with an emphasis on navigation assistance, safety support, and voice-based interaction in indoor environments. This review highlights both the potential and the limitations of existing systems, motivating the need for richer semantic representations and tighter integration between perception and interaction. The discussion then moves to scene graphs and semantic environment representations, tracing how advances in perception, object recognition, and spatial reasoning enable robots to construct structured models of domestic environments that support high-level reasoning and explanation.

Subsequent sections focus on voice-based human–robot interaction, reviewing the

components of modern speech-based interfaces and their role in accessible robotic systems. Particular attention is given to grounding language in perception and environment models, which is essential for enabling robots to respond meaningfully to situated user queries. Finally, the chapter reviews robotic navigation and mapping techniques relevant to indoor domestic environments, discussing how reliable localization, planning, and motion execution serve as enabling subsystems for higher-level assistive tasks.

By consolidating insights from these research areas, this chapter establishes the conceptual and technical background for the system architecture described in the following chapter. The reviewed literature motivates the use of semantic scene representations, voice-based interaction, and tightly integrated navigation as key building blocks for domestic assistive robots, and provides the foundation for the design choices presented in this thesis.

2.1 Assistive Robots for Visually Impaired Users

Assistive robotics has been explored as a means to support users with sensory impairments, particularly in domestic and indoor environments where independence and safety are critical. A significant portion of existing work focuses on navigation assistance and accident prevention for visually impaired users. For instance, robotic systems have been proposed to detect obstacles and hazardous situations in real time and to alert users through auditory feedback, demonstrating the potential of mobile robots to improve safety during indoor mobility [7]. Similarly, assistive solutions based on artificial intelligence and speech recognition have been developed to guide visually impaired users by providing verbal cues and environmental information, highlighting the importance of voice-based interaction as an accessible communication channel [8].

Beyond navigation-focused systems, research in assistive and social robotics has investigated the deployment of domestic robots for long-term home assistance, primarily targeting elderly users. Field studies with social robots deployed in real homes emphasize the importance of usability, reliability, and user acceptance, showing that robots can support daily activities and provide meaningful assistance when integrated into domestic routines [9]. Although these systems are not specifically designed for visually impaired users, they provide valuable insights into human–robot interaction, long-term autonomy, and the challenges of operating in unstructured home environments.

Recent advances in general-purpose domestic robotics further demonstrate the potential of combining perception, language understanding, and navigation in home settings. The HomeRobot framework, for example, addresses open-vocabulary mobile manipulation by grounding natural language commands in visual perception

and spatial reasoning, enabling robots to operate flexibly in complex domestic environments [4]. While such approaches are not explicitly accessibility-driven, they highlight the relevance of semantic representations and language grounding for enabling high-level interaction in the home.

In parallel, studies on smart home technologies and voice assistants reveal both opportunities and limitations for visually impaired users. Accessibility analyses show that many home automation systems remain difficult to use due to visually oriented interfaces and limited semantic feedback, while longitudinal studies on voice assistants indicate that speech-based interaction can significantly improve accessibility but often lacks contextual understanding of the environment [3, 5]. More recently, surveys on visually impaired assistance with large language and vision models underline the growing interest in leveraging semantic and multimodal understanding to provide richer, more flexible support [10]. These works collectively suggest that effective assistive robots for visually impaired users require not only navigation and voice interaction, but also a structured semantic understanding of the domestic environment to support complex, context-aware tasks.

2.1.1 Perception in Robotic Environments

Perception plays a fundamental role in mobile robotics, as it enables a robot to acquire information about its surrounding environment and to interact with it in a meaningful and autonomous manner. In domestic settings, perception is particularly critical due to the unstructured, dynamic, and cluttered nature of home environments, where furniture, objects, and people frequently change position. Without reliable perception capabilities, a mobile robot would be limited to basic navigation and obstacle avoidance, lacking the ability to understand and reason about the environment at a semantic level.

Modern mobile robots rely on a combination of heterogeneous sensors to perceive their surroundings. Among the most commonly used sensors are RGB cameras and RGB-D cameras, which provide visual information in the form of color images and, in the case of RGB-D sensors, depth measurements aligned with the visual data. Depth sensors, such as structured-light cameras or time-of-flight sensors, enable the estimation of three-dimensional geometry, allowing the robot to perceive distances, shapes, and spatial layouts of the environment. These sensors are widely adopted in indoor robotics due to their relatively low cost and their suitability for short-range perception in confined spaces. In many robotic platforms, visual sensing is further complemented by inertial sensors, wheel odometry, or LiDAR measurements, allowing the system to fuse multiple sensing modalities and obtain a more robust and consistent estimate of the environment.

From a functional perspective, robotic perception can be broadly divided into geometric perception and semantic perception. Geometric perception focuses on

the spatial structure of the environment and is primarily concerned with tasks such as obstacle detection, free-space estimation, localization, and mapping. Techniques such as Simultaneous Localization and Mapping (SLAM) allow a robot to construct metric or topological maps of its environment while estimating its own pose within it . These representations are essential for safe navigation but typically do not encode information about the identity or meaning of objects within the environment.

Semantic perception, on the other hand, aims to enrich geometric representations with high-level information about objects, rooms, and categories. This includes recognizing objects such as tables, chairs, or appliances, identifying room types such as kitchens or bedrooms, and associating semantic labels with regions of the environment. Advances in computer vision and deep learning have significantly improved the ability of robots to perform semantic perception through tasks such as object detection, semantic segmentation, and instance segmentation [11]. By incorporating semantic information, robots can move beyond purely reactive behaviors and reason about the environment in a way that is closer to human understanding.

In recent years, research in robotic perception has increasingly focused on integrating geometric and semantic information into unified environment representations. This integration is typically achieved through multi-stage perception pipelines in which sensor data is first processed to extract low-level geometric features, followed by higher-level semantic interpretation. For instance, a robot may first build a geometric map using SLAM, and subsequently apply object detection or segmentation models to associate semantic labels with regions or objects within that map. The fusion of these outputs produces semantic maps that encode both spatial structure and object-level meaning.

Scene graphs extend this idea by representing the environment as a structured graph that explicitly captures entities and their relationships. In a scene graph representation, nodes correspond to objects, places, or landmarks, while edges encode spatial or semantic relationships such as *on*, *inside*, *next to*, or *in-room*. Each node may also contain attributes describing object properties, such as category labels, visual features, or textual descriptions. Compared to traditional semantic maps, scene graphs emphasize relational structure and interpretability, enabling robots to reason about complex environments in a way that more closely resembles human spatial understanding.

The construction of scene graphs in robotic systems typically follows a multi-stage process. First, perception modules detect and segment objects from sensor data, producing candidate entities together with geometric information such as position and bounding volume. Second, these entities are localized within a global reference frame using depth information and robot pose estimates obtained from SLAM or visual odometry. Third, spatial relationships between objects are inferred based on geometric proximity, containment, or support relations. Finally, the

resulting nodes and edges are assembled into a graph structure that can be queried, updated, and used for reasoning.

This perception-to-graph pipeline enables robots to move from raw sensor observations to structured knowledge representations that support higher-level reasoning, natural language interaction, and task planning. The integration of geometric mapping, object-level perception, and relational reasoning therefore forms the foundation for semantic environment representations such as scene graphs, which are discussed in greater detail in the following sections.

2.1.2 Object Detection and Recognition

Object detection is a fundamental task in robotic perception that aims to identify and localize objects within a visual scene. Unlike image classification, which assigns a single label to an entire image, object detection operates at the instance level, detecting multiple objects and estimating their positions in the scene. In mobile robotics, object detection enables a robot to recognize relevant elements of the environment, such as furniture, household items, or appliances, which are essential for interaction, navigation, and task execution in domestic settings.

The typical output of an object detection system consists of a set of bounding boxes, each associated with a semantic class label and a confidence score. The bounding box defines the spatial extent of the detected object in the image, while the class label represents the object category, such as *chair*, *table*, or *cup*. The confidence score expresses the detector's certainty about the prediction and is commonly used to filter unreliable detections. This structured output allows detected objects to be directly integrated into higher-level representations and decision-making pipelines.

Modern object detection systems are primarily based on convolutional neural networks (CNNs), which exploit the spatial structure of images through convolutional filters that slide across the input and extract local patterns such as edges, textures, and shapes. By stacking multiple convolutional layers, the network learns hierarchical feature representations, where early layers capture low-level visual features and deeper layers encode abstract semantic concepts. Pooling operations and nonlinear activations further enable invariance to small transformations while reducing spatial dimensionality.

In a typical CNN-based object detector, the backbone acts as a feature extractor, converting the input image into high-dimensional feature maps. These maps are then processed by a *neck* module, often implemented using a Feature Pyramid Network (FPN), which aggregates information across spatial scales to detect objects of varying sizes. Finally, a *detection head* predicts both class probabilities and bounding box coordinates for each object. Object localization is typically formulated as a regression problem, predicting bounding box parameters (center coordinates,

width, height) relative to predefined anchors or reference points, with a joint optimization of classification and localization losses (e.g., cross-entropy for classes and Smooth L1 or IoU-based loss for boxes).

Two-stage detectors, such as Faster R-CNN [12], first generate a set of region proposals likely to contain objects and then refine these proposals through classification and bounding-box regression. This approach typically achieves high accuracy but at the cost of increased computation. In contrast, single-stage detectors perform detection and classification in a single pass, trading a small loss in accuracy for significantly faster inference.

YOLO (You Only Look Once) YOLO is a canonical example of a single-stage object detector [13]. Unlike two-stage approaches, YOLO divides the input image into a grid and directly predicts bounding boxes and class probabilities for each cell in one evaluation of the network. This design enables real-time performance, making it well-suited for mobile robotic systems where low latency and efficient computation are critical. YOLO's architecture typically includes a CNN backbone for feature extraction, followed by fully convolutional layers that simultaneously output a fixed number of bounding boxes and associated class probabilities. Over the years, YOLO has evolved through multiple versions (YOLOv2, YOLOv3, YOLOv4, YOLOv8), improving accuracy, speed, and robustness while maintaining the real-time advantage that is crucial for robotics applications.

Advanced Architectures and Open-Vocabulary Detection

Recent advances include transformer-based detectors, such as DETR, which replace traditional region proposal mechanisms with attention-based architectures capable of modeling global relationships in the image. These models simplify the detection pipeline but often require large-scale training datasets and significant computational resources.

Another important direction is open-vocabulary object detection, which leverages vision-language models trained on large-scale image-text datasets. These systems can detect objects based on natural language descriptions rather than being limited to a fixed set of categories, making them particularly valuable for domestic robots where object variety is high and cannot be fully enumerated in advance [14].

Limitations

Despite their effectiveness, object detection methods exhibit several limitations when considered in isolation. Detections are typically independent and do not encode spatial or semantic relationships between objects. Additionally, outputs are local observations tied to individual sensor frames, providing no global representation of the environment. Consequently, object detection alone is insufficient for

higher-level reasoning, spatial queries, or natural language interaction that require understanding object relationships and scene context.

2.1.3 Semantic Segmentation and Scene Understanding

Semantic segmentation is a core task in visual perception that aims to assign a semantic label to each pixel of an image. Unlike object detection, which provides coarse localization through bounding boxes, semantic segmentation produces a dense, pixel-level understanding of the scene. This enables a robot to distinguish between different regions of the environment, such as floors, walls, furniture, and objects, with a much finer spatial resolution. In domestic environments, this level of detail is particularly valuable due to the presence of clutter, overlapping objects, and complex spatial arrangements.

From a theoretical perspective, semantic segmentation can be interpreted as a dense prediction problem in which the goal is to perform pixel-wise classification over the entire image. Early deep learning approaches for segmentation were based on Fully Convolutional Networks (FCNs), which replaced the fully connected layers of traditional convolutional neural networks with convolutional layers, allowing the network to produce spatially structured outputs [15]. This architectural modification enables the network to process images of arbitrary size while generating dense prediction maps in which each pixel corresponds to a class probability.

Many modern segmentation architectures follow an encoder–decoder structure. In this framework, the encoder progressively extracts high-level semantic features from the input image through convolutional and pooling layers, reducing the spatial resolution while increasing the abstraction level of the representation. The decoder then reconstructs a dense prediction map by gradually upsampling the feature maps and combining them with higher-resolution features from earlier layers. This process allows the network to recover fine spatial details that may have been lost during downsampling.

A closely related task is instance segmentation, which extends semantic segmentation by differentiating between individual instances of the same object category. While semantic segmentation labels all pixels belonging to a class (e.g., *chair*) with the same label, instance segmentation assigns a unique identifier to each object instance. One of the most influential architectures for instance segmentation is Mask R-CNN, which extends the Faster R-CNN object detection framework by adding a parallel branch that predicts a binary segmentation mask for each detected object [16]. This architecture simultaneously performs object detection, classification, and pixel-level segmentation for each instance.

Training segmentation networks typically involves pixel-wise loss functions that compare the predicted label distribution for each pixel with the corresponding

ground truth annotation. The most common formulation is the categorical cross-entropy loss applied independently to each pixel, although alternative losses such as Dice loss or Intersection-over-Union (IoU) loss are often used to better handle class imbalance and improve boundary accuracy.

Compared to object detection, segmentation-based approaches provide several advantages. First, pixel-level predictions allow a more accurate estimation of object shape and spatial extent, which is essential for tasks such as navigation in tight spaces, manipulation, and obstacle avoidance. Second, segmentation preserves contextual information by explicitly modeling the spatial arrangement of different semantic regions within the scene. For example, distinguishing between navigable floor areas and obstacles enables safer and more efficient path planning in indoor environments.

In the context of 3D scene understanding, segmentation plays a particularly important role because it enables a more precise association between visual observations and geometric information. When segmentation masks are combined with depth measurements, it becomes possible to isolate the three-dimensional points that belong to a specific object or semantic region. This facilitates the reconstruction of semantically labeled point clouds and improves the accuracy of object localization in the global coordinate frame. Furthermore, instance-level segmentation provides a natural mechanism for defining nodes in a scene graph representation, where each segmented object can be associated with semantic attributes and spatial relationships. As a result, semantic and instance segmentation constitute a fundamental component in pipelines that aim to bridge low-level perception with higher-level scene representations and reasoning capabilities.

Segment Anything Model (SAM) The Segment Anything Model (SAM) is a recent advancement in segmentation research that introduces an *open-set, promptable* segmentation framework [17, 18]. Unlike traditional models that are restricted to a fixed set of classes, SAM is trained on a massive dataset of over 1 billion masks, enabling it to generalize to arbitrary objects without explicit retraining. SAM operates by accepting flexible prompts—points, boxes, or free-form masks—and producing high-quality segmentation masks for the indicated region.

At a theoretical level, SAM is built upon a transformer-based architecture, which allows it to model long-range spatial dependencies and integrate multi-scale context. The model decouples the *image encoder* from the *prompt encoder* and *mask decoder*, providing a flexible interface for interactive segmentation. The image encoder extracts a global feature representation of the input image, while the prompt encoder embeds user-provided cues into a compatible feature space. The mask decoder then integrates both streams to produce a binary segmentation mask for each query. This design allows SAM to achieve robust generalization to unseen objects, including those not present in the training set.

SAM is particularly relevant for robotic perception in domestic environments. By combining SAM with depth or multi-view data, robots can generate precise 3D masks for arbitrary objects, enabling on-the-fly scene understanding, object manipulation, and semantic mapping without the need for extensive per-task training. Its open-set nature also complements open-vocabulary detection and language-guided perception, supporting flexible human–robot interaction and reducing the need for exhaustive labeling of domestic object categories [19].

2.1.4 Spatial Relationships and Scene Graph Representation

Understanding an environment requires not only the recognition of individual objects, but also the ability to reason about the spatial and semantic relationships that exist among them. Spatial relationships such as *on*, *inside*, *next to*, *near*, or *behind* play a fundamental role in how humans describe and interpret their surroundings, and have long been recognized as a key component of scene understanding. In domestic environments, these relations are essential to convey meaningful information, for instance in statements such as "the mug is on the table in the kitchen", where the semantics emerge from the interaction between objects, rooms, and their relative spatial arrangement rather than from isolated object detections.

Spatial relationships can be broadly categorized into metric and topological relations. Metric relations rely on precise quantitative measurements, such as distances and angles, and are commonly used for low-level tasks including navigation and obstacle avoidance. Topological and qualitative relations, on the other hand, abstract away from exact geometry and describe how entities are arranged or connected with respect to one another. These qualitative relations are particularly important for semantic reasoning and natural language interaction, as they align more closely with how humans describe space and how language-conditioned models reason about visual scenes [20].

The explicit modeling of spatial relationships is crucial for higher-level cognitive capabilities in robotics, including task planning, scene understanding, and human–robot interaction. Without relational representations, a system cannot answer spatial queries, explain object locations, or interpret instructions that rely on spatial concepts. This limitation is especially critical in assistive scenarios for visually impaired users, where verbal explanations constitute a primary communication channel. Recent works in open-vocabulary and foundation-model-based 3D scene understanding further highlight the need for representations that go beyond object-centric labels and enable semantic reasoning across unseen categories and concepts [21, 22].

Scene graphs provide a structured and expressive representation that integrates

objects, their attributes, and their spatial and semantic relationships into a unified model. Formally, a scene graph can be defined as a directed graph $G = (V, E)$, where the set of nodes V represents entities such as objects, rooms, or landmarks, and the set of edges E encodes relationships between these entities. Each node may also contain attributes describing semantic properties such as category, color, material, or state, while edges capture interactions and spatial relations between pairs of entities [23, 24].

In many computer vision formulations, scene graphs are represented as triplets of the form (*subject*, *predicate*, *object*), for example (*cup*, *on*, *table*). This triplet representation naturally aligns with the structure of natural language and facilitates the integration of perception systems with language-based reasoning modules. Scene graph generation therefore typically involves three interconnected tasks: object detection, attribute prediction, and relationship prediction. Relationship prediction is particularly challenging because it requires the model to infer interactions between objects based on their spatial configuration, contextual cues, and semantic compatibility.

While early work on scene graphs focused primarily on two-dimensional image representations, recent research has extended this concept to three-dimensional environments. In robotics applications, 3D scene graphs combine geometric information obtained from depth sensors or SLAM systems with semantic labels derived from visual perception modules. This allows objects to be anchored in a global coordinate frame and enables reasoning about spatial relationships that persist across multiple viewpoints. As a result, 3D scene graphs can be interpreted as semantic cognitive maps that bridge low-level perception and high-level reasoning.

Vision–language models (VLMs) have recently emerged as powerful tools for enriching scene graph representations with semantic knowledge. These models are trained on large-scale image–text datasets and learn joint embeddings that align visual and linguistic representations. As a result, they can associate visual entities with natural language descriptions, infer semantic attributes, and generate contextual captions describing objects and their relationships. In scene graph generation pipelines, VLMs can therefore be used to augment purely geometric representations with semantic labels, attribute descriptions, and relational predicates extracted from visual-language reasoning.

The integration of VLMs into scene understanding systems also enables open-vocabulary reasoning, allowing robots to recognize and describe objects that were not explicitly included in the training dataset. Instead of relying on a fixed set of predefined categories, these models can interpret objects through descriptive language prompts, making them particularly suitable for domestic environments where object diversity is large and unpredictable. Furthermore, language-conditioned reasoning enables more natural interaction with users, as spatial queries or instructions expressed in natural language can be directly mapped onto the relational

structure of the scene graph.

Compared to traditional semantic maps, which typically associate labels with spatial regions or grid cells, scene graphs emphasize relational structure and interpretability. This distinction has motivated their adoption in both computer vision and robotics, where scene graphs are used as semantic cognitive maps that support explanation, planning, and interaction [25, 26]. Recent advances further explore learning scene graphs directly from visual data, transitioning from pixel-level representations to graph-structured abstractions that better support reasoning and open-vocabulary generalization [20]. In the context of domestic assistive robotics, scene graphs therefore constitute a natural and powerful representation for semantic understanding and human-centered interaction.

ConceptGraphs

A recent example of this paradigm is the ConceptGraphs framework, which introduces an open-vocabulary 3D scene graph representation designed specifically for robotic perception and planning [27]. ConceptGraphs integrate geometric reconstruction, visual perception, and language-conditioned models to construct a structured representation in which objects, their attributes, and spatial relationships are organized into a unified graph.

In this framework, objects are first detected and segmented from RGB-D observations and associated with three-dimensional geometric representations obtained through mapping or SLAM systems. Vision–language models are then used to assign semantic descriptions to these entities, enabling open-vocabulary recognition that is not restricted to a fixed set of predefined categories. As a result, the system can identify and describe previously unseen objects by leveraging semantic similarity in the joint visual–language embedding space.

The resulting graph structure encodes objects as nodes and spatial or semantic relationships as edges, allowing the robot to reason about interactions between entities in a global 3D coordinate frame. This representation supports higher-level reasoning tasks such as spatial querying, goal-directed navigation, and task planning. For example, the robot can interpret instructions such as locating an object relative to another or searching for items that satisfy certain semantic properties. By combining geometric consistency with language-conditioned semantics, ConceptGraphs demonstrate how open-vocabulary perception can be integrated into structured scene representations that are suitable for robotic decision-making.

2.2 Voice-Based Human–Robot Interaction

Voice-based human–robot interaction (HRI) represents a natural and intuitive communication paradigm that enables users to interact with robotic systems

using spoken language. Unlike graphical or tactile interfaces, voice interaction supports hands-free operation and does not require visual attention, making it particularly suitable for domestic and assistive robotics. In such contexts, speech-based interaction lowers the barrier to use, enhances accessibility, and supports more natural and socially acceptable forms of human–robot communication.

Building on these motivations, modern voice-based HRI systems are designed to translate spoken input into meaningful robotic behavior through a structured processing pipeline.

Voice-based HRI systems typically adopt a modular processing pipeline that transforms raw audio input into executable actions and verbal responses. The interaction loop generally begins with wake-word detection, which enables continuous listening while limiting unnecessary computation and unintended activations. Once activated, the user's speech is processed by an automatic speech recognition (ASR) module, which converts the acoustic signal into a textual representation. This text is then analyzed by a natural language understanding (NLU) component to infer user intent and extract relevant semantic information.

While speech recognition and language understanding enable the interpretation of user input, effective interaction also requires managing the temporal and contextual structure of communication.

The dialogue management module governs the conversational flow, handling turn-taking, confirmations, clarifications, and contextual reasoning based on both interaction history and system state. Finally, text-to-speech (TTS) synthesis generates spoken responses, closing the interaction loop and allowing the robot to communicate feedback, explanations, or guidance to the user. This modular architecture reflects the dominant design paradigm adopted by both commercial voice assistants and research-oriented HRI systems, as it enables clear separation of concerns and facilitates independent development and evaluation of individual components [28].

Although this pipeline resembles that of traditional voice assistants, robotic systems impose additional requirements that fundamentally differentiate embodied interaction from purely virtual conversational systems.

In robotic applications, voice interaction systems must go beyond purely conversational capabilities and be tightly integrated with perception, world modeling, and action execution. In particular, NLU and dialogue management often require grounding linguistic expressions in the physical environment, such as associating object names with perceived entities or resolving spatial references relative to the robot's pose and surroundings [29, 30]. This grounding is essential for commands involving navigation, object manipulation, or environment queries, and distinguishes robotic voice interaction from conventional virtual assistants.

This tight coupling between perception, computation, and communication further influences architectural decisions related to where and how speech processing is

performed.

Another important architectural consideration concerns the trade-off between on-device and cloud-based processing. Cloud-based solutions typically offer access to large-scale speech and language models and achieve high recognition accuracy, but introduce latency, reliance on network connectivity, and privacy concerns. In contrast, on-device processing offers lower latency, improved robustness, and stronger privacy guarantees, but is constrained by computational resources. As a result, many modern robotic systems adopt hybrid architectures that balance local and remote processing to optimize performance, responsiveness, and privacy [31, 32].

With this system-level perspective in place, the following subsections examine the individual components of the voice interaction pipeline in more detail, starting from low-level audio activation mechanisms.

2.2.1 Wake-Word Detection and Voice Activation

Wake-word detection, also known as keyword spotting, is the process of identifying a predefined activation phrase that enables a voice-based interaction system. It represents the first stage of most voice interaction pipelines and allows users to initiate interaction in a hands-free and natural manner. Typical activation phrases include expressions such as “Hey Robot” or “Ok Assistant,” which are designed to be easy to remember and unlikely to occur unintentionally during everyday conversation.

From an architectural perspective, two main approaches to wake-word activation can be distinguished. The first approach performs direct wake-word detection on the audio stream using lightweight keyword spotting models that operate on raw audio signals or low-level acoustic features. These models are specifically trained to recognize a small set of predefined phrases and are optimized for low computational cost, enabling continuous listening with minimal energy consumption and latency [33]. An alternative strategy relies on continuous speech transcription, where an ASR system continuously converts audio into text and triggers activation when the wake word appears in the transcription. Although conceptually straightforward, this strategy is generally impractical for real-world robotic systems due to its high computational cost, increased latency, and significant privacy concerns. Continuously transcribing all ambient speech requires substantial processing resources and may expose sensitive information. For these reasons, most modern voice assistants and robotic platforms adopt direct audio-based wake-word detection and activate full speech recognition only after successful keyword detection.

Beyond architectural efficiency, wake-word detection also plays a key role in determining how users physically and cognitively interact with the robot. Wake-word activation offers clear advantages over push-to-talk interaction paradigms,

which require users to press a physical button or interact with a graphical interface. While push-to-talk systems eliminate the risk of false activations, they reduce interaction naturalness and accessibility. This limitation is particularly critical in domestic assistive robotics, where users may have limited mobility or sensory impairments. For visually impaired users, locating and operating a physical control can be inconvenient or infeasible, making hands-free voice activation a fundamental requirement for effective interaction [3].

Despite its importance, reliable wake-word detection remains challenging in real-world environments. False positives may occur when background sounds or speech resemble the activation phrase, while false negatives can prevent the system from responding when expected. These issues are exacerbated in domestic environments characterized by background noise, reverberation, multiple speakers, and overlapping speech. Achieving robust wake-word detection under these conditions remains an active area of research, as reliability at this stage directly influences user trust and overall system usability [34].

OpenWakeWord Framework

Wake-word detection can be formally described as a binary sequence classification problem, where the system must determine whether a short segment of an incoming audio stream contains the target activation phrase. The raw audio waveform $x(t)$ is first transformed into a time-frequency representation suitable for deep learning models. A common choice is the mel-scaled spectrogram, computed as:

$$S_{mel}(f,t) = \sum_{k} |X(k,t)|^2 \cdot H_{mel}(f,k)$$

where $X(k,t)$ is the short-time Fourier transform of the signal, and $H_{mel}(f,k)$ is the mel filter bank mapping linear frequency bins k to perceptually motivated mel bins f [35]. The mel spectrogram emphasizes frequencies most relevant to human hearing and reduces dimensionality, providing an effective input for convolutional neural networks.

OpenWakeWord implements this transformation using an ONNX-based version of PyTorch's MelSpectrogram, allowing efficient computation across heterogeneous devices. The spectrogram is then processed by a shared feature extraction backbone, consisting of a series of convolutional blocks. These layers capture local temporal and spectral patterns, generating a high-level embedding that encodes the acoustic characteristics of speech independent of speaker identity or background noise [36]. The backbone benefits from extensive pretraining on large-scale speech corpora, providing robust generalization to previously unseen wake-word phrases.

The extracted embeddings are fed into a classification head, which maps the audio features to a binary label indicating the presence or absence of the wake

word. In openWakeWord, this head can be implemented as a simple fully connected network or a lightweight recurrent model, integrating temporal context over short audio windows. Formally, for an embedding h_t at time t, the probability of detecting the wake word is:

$$P(\text{wakeword}|h_t) = \sigma(W h_t + b)$$

where W and b are learnable parameters, and σ is the sigmoid activation function. The model is trained with binary cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\Big[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)\Big]$$

By separating feature extraction from classification, frameworks like openWakeWord leverage pretrained acoustic embeddings, allowing the classifier to be trained with relatively small datasets—including synthetic audio—without compromising performance [36]. Convolutional layers provide invariance to minor temporal shifts, while recurrent or context-aware classifiers reduce false positives caused by transient background noise.

In practice, wake-word detection forms the first critical stage of voice-based interaction. Its reliability directly impacts user experience: excessive false positives create unintended activations, while false negatives prevent interaction. By combining perceptually motivated signal processing, pretrained convolutional embeddings, and lightweight temporal classifiers, modern systems like openWakeWord achieve low-latency operation and robust performance in noisy domestic environments.

Once the wake word has been reliably detected, the system proceeds to automatic speech recognition and natural language understanding, forming the next stages of the voice interaction pipeline.

2.2.2 Speech-to-Text and Automatic Speech Recognition

Automatic Speech Recognition (ASR) refers to the process of converting spoken language into a textual representation that can be processed by a computational system. In voice-based human–robot interaction, ASR constitutes a critical component, as it provides the linguistic input required for intent recognition, dialogue management, and task execution. The accuracy and robustness of the ASR module directly affect the usability and effectiveness of the entire interaction pipeline.

Historically, ASR technology has evolved through several methodological paradigms. Early systems were based on probabilistic models such as Hidden Markov Models (HMMs) combined with Gaussian Mixture Models (GMMs), which modeled sequences of acoustic features as stochastic processes. These approaches required extensive feature engineering—typically using Mel-Frequency Cepstral Coefficients

(MFCCs) or similar handcrafted spectral features—and often struggled with variability in speech patterns, accents, and acoustic conditions.

The introduction of deep learning marked a significant shift in ASR research. Deep neural networks (DNNs), and later recurrent architectures such as Long Short-Term Memory (LSTM) networks, were used to model acoustic representations directly from spectro-temporal features. More recent advances have embraced end-to-end architectures, particularly those based on sequence-to-sequence models with attention mechanisms and transformer-based encoders, which learn to map raw acoustic features to text sequences without requiring separate phonetic or lexical modeling stages.

These modern approaches allow ASR systems to capture long-range temporal dependencies in speech, improving both recognition accuracy and robustness under noise and reverberation—conditions typical of domestic environments where assistive robots operate. Despite these advances, ASR in real-world robotic contexts remains challenging due to background noise, spontaneous speech patterns, overlapping conversations, and diverse speaker accents and inflections.

In addition to modeling considerations, architectural decisions regarding where ASR processing is performed also influence system performance. Cloud-based ASR systems typically achieve higher accuracy due to access to powerful models trained on massive datasets, but they introduce concerns related to latency, dependency on network connectivity, and privacy. On-device ASR, in contrast, offers lower latency and stronger privacy guarantees but is constrained by computational and memory limitations. As a result, many modern robotic systems adopt hybrid architectures that dynamically balance local and remote processing to optimize performance, responsiveness, and data protection [31].

GPT-4o Transcribe: Modern Large-Model Speech Recognition

Contemporary ASR development has seen the integration of large multimodal and foundation models, which unify audio, text, and sometimes visual information within a single neural framework. OpenAI's GPT-4o Transcribe represents a recent iteration of this trend, where a general-purpose multimodal model family is extended to support high-quality speech-to-text transcription. The GPT-4o Transcribe model is specifically tailored for speech recognition tasks and is accessible through the OpenAI API via the `audio/transcriptions` endpoint [37].

GPT-4o Transcribe and its lighter variant, GPT-4o mini Transcribe, outperform traditional ASR backends such as Whisper in both word error rate and language recognition accuracy across diverse benchmarks, reflecting significant progress in end-to-end speech modeling [38][37]. These models accept raw audio input and produce a textual transcript, leveraging the same transformer-based architecture that underlies the broader GPT-4o family. The large context window (e.g., 16,000

tokens) allows the model to maintain extended context during transcription, which is particularly valuable for complex or long utterances.

Unlike traditional modular ASR pipelines that separate acoustic modeling, language modeling, and decoding, models like GPT-4o Transcribe implicitly integrate these components through unified representation learning. During inference, the model processes the acoustic input and directly produces textual output, benefiting from vast pretraining on multimodal corpora that include both audio and text. Furthermore, these models support additional features such as optional prompts that can provide contextual biasing to improve transcription quality in specialized domains or conversational scenarios .

Despite these advances, the integration of large model–based ASR into robotic systems must consider practical tradeoffs. Although models like GPT-4o Transcribe offer state-of-the-art recognition performance, they still require careful handling of latency (especially in streaming or real-time settings) and resource costs. Hybrid approaches that combine local wake-word detection and buffering with cloud- or edge-assisted transcription balance responsiveness with accuracy, making them well suited for voice interaction in assistive robots.

Given its central role, errors at the ASR stage can have cascading effects. In robotic systems, ASR errors propagate through the interaction pipeline, potentially leading to incorrect intent inference or unsafe actions. Reliable speech-to-text processing is therefore essential for grounding user commands and queries in perception and action, and for enabling robust interaction between humans and embodied agents. Accurate transcription alone, however, is insufficient; speech must also be segmented and detected reliably in continuous audio streams, typically through voice activity detection and integrated signal processing modules that precede transcription.

2.2.3 Voice Activity Detection

Voice Activity Detection (VAD) is the process of identifying segments of an audio signal that contain human speech, distinguishing them from silence, background noise, or other non-speech sounds. In voice-based human–robot interaction, VAD serves as a critical preprocessing stage that enables efficient computation and improves the reliability of downstream modules such as wake-word detection and automatic speech recognition (ASR) [39]. By accurately segmenting speech from non-speech, VAD ensures that computationally intensive models are only invoked when relevant audio is present, reducing energy consumption, latency, and the risk of spurious activations.

Traditional Approaches Early VAD methods relied on signal-level heuristics and simple feature extraction. Common techniques include energy thresholding,

zero-crossing rate analysis, and spectral features such as spectral entropy or spectral flux [39]. While computationally inexpensive, these approaches are highly sensitive to variations in environmental noise, reverberation, and microphone characteristics. They also typically require hand-tuned thresholds, which limit generalization across different acoustic environments.

Machine Learning and Deep Learning Approaches Modern VAD systems increasingly leverage machine learning and deep neural networks to improve robustness and generalization. Neural architectures, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer-based models, operate on time-frequency representations of the audio signal, such as mel-spectrograms or MFCCs, to capture both spectral and temporal speech patterns. These models can learn complex acoustic patterns that distinguish speech from non-speech even under noisy conditions, achieving higher accuracy compared to traditional methods. Some approaches integrate VAD with downstream tasks like ASR or wake-word detection, allowing joint optimization and improved end-to-end performance.

Silero VAD One state-of-the-art example is Silero VAD, a pre-trained, enterprise-grade voice activity detector designed for robust performance across diverse acoustic conditions [40]. Silero VAD operates on raw audio or mel-spectrogram representations and uses a lightweight neural network backbone to produce frame-wise speech probabilities. Its design emphasizes low latency, high accuracy, and adaptability to noisy and reverberant environments, making it suitable for real-time applications in domestic robotics.

Silero VAD offers several compelling advantages that make it particularly well-suited for real-time, voice-based human–robot interaction. First, it is pre-trained and ready to use, meaning it can be deployed directly without the need for extensive retraining, and it supports both short- and long-form audio inputs. This allows developers to integrate the model quickly into diverse applications. In addition, Silero VAD is designed for cross-device efficiency, with optimizations for both CPU and GPU execution, enabling real-time performance even on resource-constrained embedded platforms. Its architecture also facilitates seamless integration with other components of the voice pipeline, such as wake-word detection and automatic speech recognition, allowing for end-to-end voice processing. Finally, the system demonstrates strong robustness, having been evaluated on multiple quality metrics and shown to outperform many traditional and lightweight VAD solutions, particularly in challenging acoustic environments characterized by noise, reverberation, or overlapping speech.[40].

The underlying workflow of Silero VAD can be summarized as follows:

- The raw waveform is preprocessed and optionally converted into a mel-spectrogram.
- The neural network produces frame-level speech probability scores.
- Post-processing, such as smoothing or thresholding, generates contiguous segments of speech activity suitable for downstream modules.

By providing reliable segmentation of speech from non-speech, VAD forms the foundation upon which higher-level speech understanding, natural language processing, and voice-based human–robot interaction can be built. In robotic systems, robust VAD is particularly important for domestic environments, where overlapping conversations, background music, and reverberation present significant acoustic challenges.

2.2.4 Natural Language Understanding

Natural Language Understanding (NLU) is the component of voice-based human–robot interaction responsible for extracting semantic meaning from transcribed user utterances. Traditionally, task-oriented dialogue systems treat NLU as a combination of intent detection and slot filling, where the model identifies the user's goal and extracts relevant parameters such as object names, locations, or attributes [41].

In robotic contexts, however, NLU must accommodate the embodied and situated nature of the agent. Commands not only need to be semantically parsed, but also grounded in actionable plans for navigation, manipulation, or environment interaction, while queries often require reasoning over internal representations of the robot's world, such as semantic maps or scene graphs [42]. This necessitates a deep integration between language understanding, perception, and planning.

Foundations of Modern Language Models At the core of modern NLU systems are large language models (LLMs), which represent a theoretical shift from traditional rule-based or shallow machine learning approaches to contextual, representation-based reasoning. LLMs are typically built on the transformer architecture, which replaces recurrent or convolutional sequence modeling with self-attention mechanisms that allow each token in an input sequence to attend to every other token. This enables the model to learn long-range dependencies and capture rich contextual relationships within text.

Formally, the transformer's self-attention layer computes a weighted sum of value vectors V for each query vector Q, where the weights are derived from the scaled dot product of Q with key vectors K:

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

where d_k is the dimensionality of the key vectors. Stacking multiple layers of multi-head attention and position-wise feedforward networks yields deep representations that capture lexical, syntactic, and semantic structure.

These deep contextual embeddings allow LLMs to generalize across tasks with minimal task-specific fine-tuning, supporting powerful capabilities like few-shot learning,semantic inference, and long-range context tracking. In robotic NLU, this means that a model can interpret novel, complex, or underspecified commands by leveraging patterns learned from large corpora rather than relying solely on preprogrammed grammar rules.

LLMs Grounded in Perception and Action A critical challenge in robotic NLU is the grounding problem: linking abstract linguistic symbols to real entities, perceptions, and actions in the environment. Language is inherently ambiguous and often relies on contextual and perceptual assumptions; for example, interpreting phrases like "move the object on the table" requires access to spatial and perceptual information. To address this, modern NLU systems integrate LLMs with structured representations such as scene graphs, allowing semantic parses to be grounded in object instances, spatial relations, and environment models [43]. This integration achieves semantic coherence between language and perception, enabling robots to interpret and execute tasks expressed in natural language reliably.

GPT-4o and Its Architecture OpenAI's GPT-4o [44]represents a recent milestone in large multimodal and speech-capable language models. Like its predecessors in the GPT family, GPT-4o is built upon a transformer backbone with hundreds of billions of parameters, trained on a mixture of text, code, and potentially multimodal inputs. The model learns to predict the next token in a sequence, capturing statistical patterns of language, discourse structures, and semantic associations in a high-dimensional representation space. This next-token prediction objective implicitly trains the model to perform tasks ranging from translation and summarization to logical inference and instruction following.

What distinguishes GPT-4o from earlier models is its multimodal integration and expanded context window. Through specialized architecture extensions and pretraining regimes, GPT-4o can ingest and jointly reason over text, audio, and potentially visual modalities, making it particularly effective for embodied systems that combine speech inputs with perceptual data. The large context capacity enables the model to maintain extended conversational history, improving coherence and contextual relevance in dialogue.

In the context of robotic NLU, GPT-4o's architecture allows the system to interpret user commands by leveraging both linguistic and situational context. Rather than treating language understanding as isolated intent classification, GPT-4o can perform compositional reasoning, manage dialogue context across multiple turns, and integrate semantic scene representations provided by perception modules. This makes it possible to handle complex, multi-step instructions and follow-up queries that depend on prior context—a key requirement for natural, conversational interaction with robots.

2.2.5 Text-to-Speech and Verbal Feedback

Text-to-Speech (TTS) synthesis transforms system responses, typically generated as text, into spoken audio. In robotic systems, TTS is essential not only for providing feedback, but also for delivering explanations, guidance, and situational awareness in a manner that is accessible, intuitive, and natural for users. For visually impaired users in particular, speech output constitutes the primary interface modality, enabling the robot to communicate spatial descriptions, task results, and interactive dialogue without reliance on visual displays.

Foundations of TTS Modern TTS systems are grounded in a combination of deep learning and signal modeling. The core theoretical objective of TTS is to learn a mapping from symbolic text input $T = (t_1, t_2, \ldots, t_n)$ to a time-domain acoustic waveform $x(t)$ that is both intelligible and natural sounding. This involves modeling the complex dependencies between phonetic, prosodic, and temporal aspects of human speech.

Classical TTS architectures, including concatenative synthesis and parametric systems such as Hidden Markov Model (HMM)-based speech synthesis, rely on pre-recorded or statistically modeled speech units. While these methods can produce intelligible speech, they often suffer from unnatural prosody and limited variability.

In neural TTS, the training objective typically combines spectrogram reconstruction loss (e.g., mean squared error between predicted and target spectrograms) with waveform reconstruction or perceptual losses that encourage naturalness and intelligibility. By jointly optimizing across these representations, deep TTS models can generate expressive, human-like speech that adapts to content, context, and even speaker style.

GPT-4o-mini-TTS Model In the context of this thesis, text-to-speech synthesis is implemented using OpenAI's GPT-4o-mini-TTS model, a dedicated TTS variant derived from the broader GPT-4o architecture. GPT-4o-mini-TTS leverages a transformer-based backbone that has been pretrained on large corpora of paired

text and speech data, enabling it to generalize across linguistic contexts and produce high-quality, natural speech output [45].

Unlike traditional decoupled pipelines (text → phonemes → spectrogram → waveform), GPT-4o-mini-TTS operates in an integrated end-to-end manner where the model learns to jointly represent both linguistic structure and acoustic realization. This integration allows the model to capture not only phonetic detail, but also contextual and semantic cues that influence prosody — for example, emphasis and intonation driven by sentence meaning or dialogue context.

The GPT-4o-mini-TTS model supports a wide range of voices, languages, and expressive styles, making it suitable for assistive robotics where clarity, intelligibility, and user preference matter. Because the model is based on transformers with large context windows, it can maintain coherence across extended utterances, enabling fluid explanations and multi-sentence responses without unnatural breaks or repetition.

From an architectural perspective, GPT-4o-mini-TTS inherits key advantages of transformer models:

- **Self-Attention Encoders:** Capture long-range dependencies in text, enabling prosody modulation based on global sentence structure.
- **Multi-Head Contextualization:** Allow the model to integrate semantic meaning, dialogue context, and even situational cues when generating speech output.
- **Shared Latent Representations:** Because GPT-4o models are trained on both language and multimodal tasks, the TTS variant benefits from shared representations that encode semantic grounding useful for robotic interaction.

Unlike legacy TTS systems that require separate phonetic or prosodic modules, GPT-4o-mini-TTS learns these aspects as part of a unified objective, resulting in speech that is both fluent and contextually appropriate.

Accessibility Beyond output quality, verbal feedback plays a critical role in accessibility and user trust. For visually impaired users, TTS is not merely a channel for robot responses; it is the primary conduit for conveying spatial descriptions, task results, and interactive dialogue. Clarity, naturalness, and responsiveness in TTS directly influence how users perceive the robot's competence and empathy [46].

By integrating GPT-4o-mini-TTS into the voice interaction pipeline, the system can generate explanations and guidance that align with human expectations of conversational speech. This includes appropriate pacing, emphasis on important

information, and the ability to produce multi-sentence narratives that maintain coherence over time.

Despite its strengths, TTS integration presents practical challenges. Latency must be managed carefully in real-time systems to avoid perceptual lag. Additionally, synthesizing speech at the right volume, intelligibility, and emotional tone requires careful selection of model parameters and potential customization of voice profiles for specific users.

Nevertheless, modern neural TTS systems such as GPT-4o-mini-TTS provide a theoretically grounded, high-quality solution for verbal feedback in assistive robotics, bridging the gap between text-based responses and natural, human-like speech.

2.2.6 Dialogue Management, Assistive Interaction, and Open Challenges

Dialogue management is the component responsible for orchestrating the flow of conversation, determining when the system should respond, ask for clarification, or maintain context across multiple turns. In domestic assistive scenarios, multi-turn dialogue is particularly critical: users often refine their requests, ask follow-up questions, or seek confirmations to ensure the robot has correctly understood their intent.

To be effective in real-world environments, dialogue systems must handle uncertainty gracefully. Noise, spontaneous speech, and diverse user behaviors can introduce ambiguity at every stage, from speech recognition to intent inference. Robust dialogue management addresses these challenges by incorporating clarification strategies, context tracking, and error recovery mechanisms. For example, a system might politely confirm a user's request before acting or ask for additional information if the input is unclear. These capabilities are essential for maintaining trust and ensuring that interaction feels natural rather than frustrating or rigid.

In assistive robotics, effective dialogue management directly impacts user autonomy and confidence. For visually impaired users, voice is often the primary—or only—means of interacting with the robot. Tasks such as querying object locations, receiving verbal descriptions of the environment, or responding to safety alerts all rely on a dialogue system that can understand, remember, and act upon contextually rich instructions. However, many existing voice assistants are limited by their lack of grounding in the robot's perceptual and spatial understanding, making it difficult to respond accurately to situational or environment-specific queries.

Bridging this gap requires a deep integration of dialogue management with perception and structured knowledge representations such as scene graphs. By linking language understanding with spatial and semantic knowledge, robots can provide more precise, explainable, and meaningful responses, transforming abstract

commands into grounded actions. Looking forward, the field faces several exciting open challenges: enabling personalization and long-term adaptation, ensuring safe and predictable behavior when leveraging large language models, and designing interaction paradigms that are both natural and inclusive for users with diverse abilities. Solving these challenges will be key to making voice-based human–robot interaction genuinely effective, intuitive, and empowering in domestic assistive contexts.

2.3 Robotic Navigation and Mapping

Robotic navigation in indoor domestic environments requires an embodied agent to repeatedly answer two coupled questions: *(i) where am I?* and *(ii) how do I safely reach a target?* In assistive robotics, these questions are not purely geometric: navigation must be reliable in the presence of people, furniture, and clutter, and it must remain predictable and comfortable for the user (e.g., avoiding sudden accelerations, keeping safe distances, and preferring wider passages when possible). These requirements push navigation beyond "shortest path" behavior toward *socially compliant* and *risk-aware* motion planning, where conservatism and stability often matter as much as efficiency.

From a systems perspective, navigation is typically built as a layered pipeline: mapping and localization provide a consistent reference frame for motion, global planning produces a collision-free route in a static or slowly changing representation, and local control executes this route while reacting to dynamic obstacles. In ROS 2, these responsibilities are commonly implemented using the Navigation 2 (Nav2) stack, which emphasizes modularity, lifecycle-managed nodes, and Behavior-Tree-based task orchestration [47, 48]. This structure is particularly useful in assistive settings, because it makes it possible to insert explicit safety behaviors (e.g., stop-and-wait, replan, back up, request help) and to integrate higher-level semantic goals (e.g., "go to the kitchen table" rather than a raw coordinate) when paired with semantic mapping and scene representations.

2.3.1 Simultaneous Localization and Mapping (SLAM)

Simultaneous Localization and Mapping (SLAM) is a foundational problem in mobile robotics that seeks to enable a robot to build a map of an unknown environment while simultaneously estimating its own pose within that map. Formally, let the robot's pose at time t be denoted as $\mathbf{x}_t$, and the map representation as $\mathbf{m}$. SLAM aims to recursively estimate the joint posterior distribution:

$$p(\mathbf{x}_{1:t}, \mathbf{m} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t})$$

where $\mathbf{z}_{1:t}$ represents the sequence of sensor measurements (e.g., LiDAR scans, camera images) and $\mathbf{u}_{1:t}$ are the control inputs or odometry measurements. This joint estimation problem is challenging due to several factors: sensor noise, non-linear motion models, environmental ambiguities, and the need to maintain global consistency over time.

Core Components of SLAM

SLAM systems can be conceptually decomposed into three primary components:

- **1. Motion and Sensor Models:** The motion model describes how the robot's pose evolves over time given control inputs. For wheeled robots, a common choice is the differential-drive kinematic model, which predicts incremental changes in pose with associated uncertainty. The sensor model describes the likelihood of observing a measurement $\mathbf{z}_t$ given the current pose and map, i.e., $p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{m})$. Accurate models are critical for probabilistic inference, as they determine how sensor observations constrain pose and map estimates.

- **2. State Estimation:** Early SLAM approaches employed recursive Bayesian filtering. The Extended Kalman Filter (EKF) assumes Gaussian distributions and linearizes the motion and sensor models to propagate uncertainty. Particle filters, represent the posterior using a set of weighted particles, allowing multimodal distributions that better capture ambiguity in highly uncertain or feature-sparse environments. More recently, graph-based methods formulate SLAM as a non-linear least-squares optimization problem, where nodes represent robot poses and landmarks, and edges encode spatial constraints derived from sensor observations.

- **3. Map Representation:** Maps can be classified as metric or topological. Metric maps store geometric information, such as occupancy grids for 2D environments or voxel grids and point clouds for 3D mapping. Topological maps abstract the environment as a connectivity graph of locations or landmarks. Hybrid approaches combine metric precision with topological efficiency. For assistive robotics in indoor domestic environments, 2D occupancy grids often strike a practical balance, providing sufficient detail for navigation while remaining computationally tractable [49].

Challenges in Domestic Environments

While SLAM is theoretically well-understood, domestic settings present unique challenges. Homes are dynamic: furniture may be moved, doors opened or closed, and humans or pets create moving obstacles. Environments are often repetitive,

with corridors or similar-looking rooms causing data association ambiguities. Additionally, cluttered scenes introduce occlusions, complicating landmark extraction and loop closure detection. These factors increase the risk of map inconsistencies and pose jumps if the system overestimates the certainty of its observations.

Sensor Modalities and Fusion

Modern SLAM systems leverage multiple sensor modalities to improve robustness:

- **LiDAR-based SLAM:** Provides high-precision geometric measurements invariant to lighting conditions. 2D LiDAR is common for indoor robots, whereas 3D LiDAR enables richer scene understanding.
- **Visual SLAM (vSLAM):** Uses monocular or RGB-D cameras to extract visual features (e.g., ORB, SIFT, or learned descriptors) for pose estimation. Depth sensors can resolve scale ambiguities in monocular systems.
- **Sensor Fusion:** Combining LiDAR, RGB-D cameras, IMUs, and wheel odometry allows for complementary strengths: IMUs provide motion constraints, cameras offer rich feature information, and LiDAR ensures geometric accuracy.

Sensor fusion is often formalized using probabilistic filters or optimization-based frameworks that incorporate multi-modal measurements into a consistent factor graph or Bayesian estimator.

A critical aspect of SLAM is detecting loop closures—recognizing when the robot revisits a previously mapped area. Loop closure provides strong constraints that reduce accumulated drift and improve global map consistency. In graph-based SLAM, loop closures are added as additional edges connecting current poses to previously visited poses, and optimization refines all poses simultaneously. In dynamic domestic environments, loop closure must be robust to perceptual aliasing (e.g., visually similar rooms) and changes in object placement.

Practical Implementations

In the ROS 2 ecosystem, `slam_toolbox` exemplifies a modern SLAM implementation that supports both online mapping and long-term localization. Its pose-graph representation, incremental optimization, and multi-session support make it particularly suitable for service robots operating in homes, where a robot must transition from initial exploration to daily operational tasks without restarting the mapping process [50].

Overall, SLAM in domestic assistive robotics requires the combination of robust state estimation, adaptive sensor fusion, semantic integration, and careful handling of dynamic and ambiguous environments to achieve reliable, real-time localization and mapping.

2.3.2 Extended Kalman Filter SLAM (EKF-SLAM)

The Extended Kalman Filter (EKF) is one of the earliest and most well-known probabilistic approaches to Simultaneous Localization and Mapping (SLAM). EKF-SLAM estimates both the robot's pose and the positions of landmarks in the environment while accounting for uncertainty in motion and sensor measurements.

In this framework, the robot maintains a joint state representation that includes its current pose (position and orientation) together with the estimated locations of observed landmarks. Along with this state estimate, the algorithm maintains a covariance matrix that models the uncertainty of the estimates and the correlations between the robot and the landmarks in the map.

EKF-SLAM operates recursively through two main steps: prediction and update. During the prediction step, the robot estimates its new pose based on its motion model and control inputs, such as wheel odometry. This prediction also increases the uncertainty of the estimate due to noise in the motion process. In the update step, sensor observations of landmarks are incorporated to correct the predicted state. By comparing the expected observation with the actual measurement, the algorithm adjusts both the robot pose and the landmark positions to improve the overall map consistency.

One important aspect of EKF-SLAM is that it maintains correlations between all landmarks and the robot pose. This joint representation allows information gathered from one observation to influence the entire map, improving overall consistency.

EKF-SLAM offers several advantages for robotic applications. It provides a principled probabilistic framework, supports real-time incremental updates, and produces explicit uncertainty estimates for both localization and mapping. These properties made EKF-SLAM a foundational method in early mobile robotics research.

However, EKF-SLAM also presents several limitations. The computational cost increases significantly as the number of landmarks grows, since the covariance matrix expands with the map size. Additionally, the method relies on linear approximations of non-linear motion and observation models, which can introduce errors in complex scenarios. EKF-SLAM can also be sensitive to incorrect data association, where observations are mistakenly matched to the wrong landmarks.

Despite these limitations, EKF-SLAM remains a useful baseline approach for indoor robotic systems operating in moderately sized environments. In assistive domestic robotics, it can provide reliable localization when combined with modern perception techniques such as object detection and semantic mapping, enabling robots to navigate and interact with structured indoor spaces [49].

2.3.3 Path Planning and Navigation in Domestic Assistive Robotics

Navigation in mobile robotics is typically decomposed into a hierarchical control architecture consisting of *global planning* and *local execution*. The global planner computes a feasible path from the robot's current pose to a goal location, often using graph-based search algorithms such as Dijkstra's or A* on a *global costmap*. The global costmap encodes the static and known occupancy information of the environment, including obstacles, walls, and forbidden regions, allowing the planner to optimize for shortest or safest routes while avoiding collisions [51, 47].

Local Planning and Obstacle Avoidance The local planner operates at a higher frequency than the global planner and is responsible for executing velocity commands while reacting to dynamic obstacles, humans, or changes in the environment. This is enabled through a *local costmap*, a moving window around the robot that incorporates real-time sensor data from LiDAR, RGB-D cameras, or ultrasonic sensors. The local costmap inflates obstacles by a configurable radius to account for robot geometry and uncertainty, and assigns higher traversal costs near obstacles to encourage safe paths.

Trajectory generation algorithms, such as the Dynamic Window Approach (DWA) or Model Predictive Control (MPC), sample candidate velocity commands and evaluate them against the local costmap to maximize safety, smoothness, and progress toward the global path [51, 52]. Each candidate trajectory is scored according to objective functions that typically include:

- **Obstacle cost:** Penalty for proximity to obstacles or high-cost regions in the local costmap.
- **Goal alignment:** Incentive for trajectories that reduce distance to the next global waypoint.
- **Smoothness:** Preference for continuous and feasible velocity profiles respecting kinematic constraints.

Costmap Construction and Update In Nav2, costmaps are represented as 2D occupancy grids with configurable resolution. Each cell contains a cost value representing traversability: free space, unknown, or obstacle. The local costmap merges static map data with dynamic sensor readings using layered plugins, including obstacle layers, inflation layers, and voxel layers for 3D sensing. This layered architecture allows the robot to react promptly to moving obstacles while preserving awareness of static structures. The global and local costmaps are synchronized continuously to ensure that high-level plans remain feasible under local disturbances [47, 48].

Behavior Trees and Execution Orchestration Nav2 employs Behavior Trees (BTs) to coordinate complex navigation tasks, providing modular and interpretable logic for sequencing planners, controllers, and recovery behaviors. Recovery behaviors—such as clearing the local costmap, backtracking, or requesting assistance—are critical in domestic environments where robots may encounter narrow passages, furniture rearrangements, or occluded pathways. BTs facilitate the integration of perception, planning, and execution in a way that is transparent, auditable, and adaptable to assistive scenarios [47].

2.3.4 Integration with Assistive Tasks

In assistive scenarios, navigation is rarely an end in itself. It is a *subsystem* that enables higher-level user goals such as fetching objects, guiding a person through the home, or providing spatial descriptions. Consider a spoken request like "Bring me the milk." Even if the voice interface produces a correct intent, the robot must ground the target (milk) to a location hypothesis (e.g., kitchen refrigerator), select an appropriate navigation goal (a reachable pose near the refrigerator), and execute the motion safely. This pipeline couples three abstractions: (i) *task-level semantics* (what the user wants), (ii) *environment structure* (where relevant places and objects are), and (iii) *motion-level feasibility* (how to move without collisions).

A tight integration between navigation and semantic representations helps reduce user burden. Rather than requiring users to specify coordinates or to manually label maps, the robot can use semantic layers (e.g., room categories, object landmarks, and relations) to choose meaningful intermediate waypoints and to explain its actions. This is also where scene graphs become actionable: they can provide a structured interface between perception and planning, allowing the robot to reason over relations such as *in-room*, *near*, or *on-top-of* while still resolving navigation into metric goals.

Assistive robotics also imposes additional constraints on motion behavior. For visually impaired users, safety and predictability are paramount: the robot should avoid surprising maneuvers, communicate intent early (e.g., "I will pass on your left"), and prefer conservative obstacle avoidance when uncertainty is high. From a software-stack viewpoint, these requirements translate into explicit recovery behaviors, conservative speed limits in proximity to people, and careful handling of localization uncertainty (e.g., pausing when pose confidence drops rather than continuing blindly). Behavior-Tree-based orchestration in Nav2 is well suited to expressing these policies as explicit, auditable logic [47, 48].

Finally, the domestic environment changes over time. This motivates lifelong mapping, periodic map updates, or multi-session SLAM workflows that can adapt without requiring a full remap. Tooling designed for multi-session mapping and robust operation in dynamic spaces is therefore directly relevant for long-term

assistive deployments [50, 53].

Chapter 3

System Overview and Architecture

This chapter presents an overview of the robotic system developed in this thesis, describing both the physical platform and the software architecture that enable assistive functionalities in domestic environments. The goal of this chapter is to provide a unified view of the system, clarifying how perception, navigation, semantic reasoning, and voice-based interaction are integrated into a coherent assistive framework. Rather than focusing on algorithmic details, which are addressed in later chapters, this section emphasizes the overall design choices and data flow that allow the robot to operate autonomously and interact naturally with visually impaired users.

The proposed system is built around a mobile robotic platform designed for indoor environments and equipped with visual, depth, and audio sensors. Through autonomous exploration, the robot acquires geometric and visual information about the environment, which is processed to generate a semantic representation in the form of a scene graph. This representation serves as a central knowledge structure that supports high-level reasoning, navigation, and interaction. Voice-based communication acts as the primary interface between the user and the robot, enabling users to issue commands, ask questions, and receive explanations about their surroundings in a natural and accessible manner.

From a system-level perspective, the architecture follows a modular design in which perception, navigation, scene graph construction, voice interaction, and task execution are implemented as distinct but interconnected components. This separation of concerns improves robustness, scalability, and interpretability, while allowing each module to be independently extended or replaced. At the same time, the modules are tightly integrated through well-defined interfaces to ensure low-latency interaction and consistent grounding of language in the physical environment.

The remainder of this chapter is structured as follows. Section 3.1 introduces the robotic platform used for domestic assistance, including its embodiment, sensors, and design considerations for indoor and assistive use. Section 3.2 provides a high-level overview of the system architecture and data flow, illustrating how information flows from environment exploration to semantic understanding and user feedback. Finally, Section 3.3 describes the main software modules and their roles within the overall system, offering a clear conceptual map that prepares the reader for the detailed technical chapters that follow.

3.1 Robotic Platform for Domestic Assistance

The assistive system developed in this thesis is deployed on the **TRON1 robotic platform**, a bipedal humanoid robot designed for research and development in indoor environments. TRON1 features a humanoid upper body combined with a modular lower body that supports both bipedal and wheeled configurations. For the purposes of this work, the **wheeled configuration** was selected in order to improve stability, simplify navigation, and ensure safe and predictable motion in domestic settings.

Unlike fully legged locomotion, which introduces additional control complexity and stability challenges, the wheeled base enables smooth and continuous motion across flat indoor surfaces such as tiles, wood, and carpet. This design choice is particularly important in assistive scenarios, where the robot is expected to operate in close proximity to users and household furniture.

The **embodiment and height** of the robot play a crucial role in domestic assistance. In contrast to low-profile mobile robots, the humanoid form factor of TRON1 allows its sensors to be positioned at a height comparable to that of everyday objects such as tables, shelves, kitchen counters, and wall-mounted appliances. This elevated sensing perspective improves environmental awareness and facilitates the perception of objects that are relevant to daily activities.

From a semantic and interaction standpoint, the humanoid embodiment supports more natural spatial reasoning and verbal descriptions aligned with human perspectives, such as references to objects being “on” or “next to” furniture. Furthermore, humanoid embodiment has been shown to positively influence social acceptance and intuitiveness in human–robot interaction, which is particularly important in long-term domestic and assistive deployments.

Figure 3.1: The TRON1 robotic platform

3.1.1 Visual Sensing Hardware

To support perception, mapping, and task execution, the robotic platform relies on an RGB-D sensing system capable of providing both high-resolution visual information and reliable depth measurements. In this work, all visual perception tasks are performed using a **ZED 2i stereo camera**, which serves as the primary sensing modality for the robot.

The ZED 2i is a stereo vision camera developed by Stereolabs that combines two synchronized RGB sensors with onboard processing to estimate depth through stereo triangulation. Unlike structured-light or active infrared depth cameras, the ZED 2i computes depth purely from stereo image pairs, enabling robust operation in a wide range of indoor lighting conditions and at larger distances. The sensor supports high-resolution RGB capture, wide field-of-view imaging, and real-time depth estimation, making it particularly suitable for mobile robotics applications that require both scene understanding and spatial awareness.

In the proposed system, the ZED 2i is used throughout multiple stages of the perception pipeline. During the **environment exploration phase**, the camera provides synchronized RGB images, depth maps, and camera pose estimates through the ZED SDK. These data streams are combined to reconstruct dense point clouds of the surrounding environment, which form the geometric foundation for the

Figure 3.2: ZED 2i stereo camera

semantic scene representation used throughout the system. The reconstructed point clouds allow the robot to reason about the spatial structure of rooms, detect objects, and establish spatial relationships between entities.

Beyond environment reconstruction, the ZED 2i is also used during task execution for visual perception tasks such as object inspection, hazard detection, and product label scanning. The RGB stream provides high-quality visual input for vision-language models and object detection algorithms, while the associated depth information allows the system to estimate object positions within the environment and maintain spatial consistency with the scene graph representation.

The use of a single, versatile RGB-D sensor simplifies the hardware setup while maintaining sufficient perceptual capability for both large-scale mapping and close-range visual analysis. Through the integration of the ZED SDK, the system benefits from real-time depth estimation, camera tracking, and point cloud generation, which are essential components for building a coherent and spatially grounded assistive robotic system.

3.1.2 LiDAR Sensor

To support reliable navigation and obstacle avoidance, the robotic platform is equipped with a **Livox Mid-360 LiDAR**. This sensor provides real-time distance measurements of the surrounding environment, enabling the robot to perceive nearby obstacles and maintain safe navigation within indoor spaces.

The Livox Mid-360 is a compact 360-degree LiDAR sensor designed for robotic and autonomous applications. It employs a non-repetitive scanning pattern that progressively increases spatial coverage over time, generating dense point measurements of the environment. The sensor offers a wide field of view and stable depth measurements while maintaining a compact form factor suitable for integration on mobile robotic platforms.

In the proposed system, the LiDAR sensor is primarily used for robot navigation and obstacle detection. The distance measurements produced by the Mid-360 allow the navigation stack to identify obstacles such as walls, furniture, or other objects in the robot's path. These measurements support real-time motion planning and safe navigation within the environment.

It is important to note that the LiDAR is not used for large-scale 3D environment reconstruction in this work. Instead, geometric scene reconstruction and semantic mapping are performed using the RGB-D sensing capabilities of the ZED 2i stereo

Figure 3.3: Livox Mid-360 LiDAR sensor mounted on the robotic platform.

camera. The LiDAR therefore complements the visual perception pipeline by providing robust and reliable spatial measurements specifically for navigation tasks.

The combination of LiDAR-based navigation and vision-based scene understanding allows the system to maintain safe mobility while simultaneously performing higher-level perception tasks such as object detection and environment analysis.

3.1.3 Audio Interface

Voice interaction is a central component of the assistive system. For audio input and output, the robot uses a **Jabra Speak 530** device, which functions as both a microphone array and a speaker. This device was selected for its low cost, ease of integration, and reliable audio quality in indoor environments.

The Jabra 530 supports far-field voice capture, enabling robust wake-word detection and speech recognition without requiring the user to be positioned close to the robot. At the same time, it provides clear audio output for synthesized speech, allowing the robot to deliver verbal feedback, explanations, and guidance. Using a single device for both audio input and output simplifies the hardware setup and contributes to a compact and practical assistive configuration.

Figure 3.4: Jabra Speak 530 used as a combined microphone and speaker.

3.1.4 Computing Hardware

All perception, mapping, and reasoning components of the system are executed on an onboard embedded computing platform based on the **NVIDIA Jetson Orin NX**. This device provides high-performance GPU-accelerated computing capabilities while maintaining a compact size and low power consumption suitable for mobile robotics.

The Jetson Orin NX integrates a powerful GPU architecture optimized for artificial intelligence workloads, enabling efficient execution of deep learning models used throughout the perception pipeline. Tasks such as object detection, visual scene analysis, point cloud processing, and vision-language inference require substantial computational resources, which are provided by the embedded GPU and multi-core CPU of the device.

In the proposed architecture, the Jetson Orin NX serves as the central processing unit for the robotic system. It manages data streams from the onboard sensors, including the ZED 2i stereo camera and the Livox Mid-360 LiDAR, and executes the algorithms responsible for environment reconstruction, scene graph generation, and high-level reasoning. The platform also handles communication with the robot control system and the voice interaction interface.

The use of an embedded AI computer allows the system to perform complex perception and reasoning tasks directly on the robot without relying on remote servers. This design improves responsiveness, reduces communication latency, and enhances system reliability, which are critical requirements for assistive robots operating in domestic environments.

3.2 System Architecture Overview

The overall architecture of the assistive system is designed to integrate perception, semantic reasoning, task execution, and voice-based interaction into a coherent pipeline. Figure **??** shows the high-level architecture, where data flows from environment exploration to task execution and user feedback.

The system operates as a structured pipeline composed of four tightly connected functional components:

1. **Navigation:** The robot autonomously moves within the domestic environment to support exploration and goal-directed motion. Using onboard localization and mapping capabilities, the robot traverses rooms, reaches target locations, and positions itself to acquire informative viewpoints. Navigation is not only used to reach user-specified destinations, but also to actively support perception-driven tasks such as environment inspection and out-of-place object detection.

2. **Perception and Scene Understanding:** Visual data acquired during navigation are processed to detect objects, segment surfaces, and extract semantic attributes. Depth information and camera poses are used to reconstruct the spatial layout of the environment, while detected entities and their relationships are integrated into a structured semantic representation. This representation captures objects, rooms, and landmarks together with their spatial and semantic relationships, enabling reasoning about the environment at a human-interpretable level.

3. **Voice-Based Interaction:** The user interacts with the system through spoken language. Voice input is continuously monitored through wake-word detection and voice activity detection, followed by speech recognition and language understanding. User requests are interpreted as navigation goals, queries about the environment, or task commands. The system responds through synthesized speech, providing guidance, explanations, and feedback throughout the interaction.

4. **Task-Level Reasoning and Execution:** Interpreted user intents are mapped to high-level assistive tasks such as navigation assistance, room summarization, out-of-place object detection, or food label scanning. Task execution coordinates navigation, perception, and semantic reasoning to achieve the requested goal, while adapting to the current environment state. Throughout execution, the system maintains an interaction loop with the user by providing timely verbal feedback.

The modular design ensures that each component can be developed, tested, and improved independently, while the integration of perception, reasoning, and voice interaction enables robust and intuitive assistive functionality. In particular, the separation between exploration, scene understanding, and task execution allows the robot to operate efficiently, even in dynamic domestic environments.

From a functional perspective, the software architecture of the assistive system is organized around a set of tightly connected components that collectively enable perception, semantic understanding, autonomous motion, and voice-based interaction. These components are responsible for navigation and exploration, visual perception and scene understanding, semantic representation through scene graphs, natural language interaction, and task-level reasoning and execution. **Navigation**, **Perception**, **Scene Graph Construction**, **Voice Interaction**, and **Task Execution**. Each module encapsulates specific functionalities while communicating through well-defined interfaces, ensuring modularity, scalability, and maintainability.

3.2.1 Navigation

Navigation provide the system with autonomous mobility within domestic environments. Using the wheeled TRON1 platform and the ROS 2 Navigation Stack (Nav2), the robot performs localization, mapping, global and local path planning, obstacle avoidance, and recovery behaviors.

This functionality supports both goal-directed motion and autonomous exploration. In response to explicit user commands, such as requests to reach a specific object or location, navigation is guided by semantic information derived from the scene representation. In other cases, navigation enables exploration-driven behaviors, such as moving through the environment to acquire multiple viewpoints for perception tasks.

In particular, navigation plays a key role in the out-of-place object detection task, where the robot autonomously traverses the room to capture images from different positions. In this context, navigation is not driven by a predefined semantic goal, but rather by coverage and viewpoint diversity, enabling robust visual analysis of the environment.

3.2.2 Perception and Scene Understanding

Perception and scene understanding transform raw sensory input into a structured semantic representation of the environment. Visual data from the ZED 2i camera are processed to extract geometric, visual, and semantic information, including object detections, instance segmentation, depth measurements, and spatial layouts.

During exploration, RGB-D data are fused to generate dense point clouds that capture the three-dimensional structure of the environment. Object detection and segmentation provide instance-level information, which is associated with spatial coordinates through depth and camera pose estimation.

This information is organized into a scene graph representation, where nodes correspond to objects, rooms, or landmarks, and edges encode spatial and semantic relationships such as containment, proximity, or support. Additional attributes, including object category, color, or state, are attached to nodes when available. The resulting scene graph acts as a semantic cognitive map that supports reasoning, querying, and task execution.

3.2.3 Voice-Based Interaction

Voice-based interaction provides the primary communication channel between the user and the robot, enabling natural language commands, queries, and verbal feedback.

- **Wake Word Detection:** Continuously listens for the activation phrase and triggers the voice pipeline.
- **Voice Activity Detection (VAD):** Identifies segments containing speech, ensuring efficient and low-latency processing.
- **Speech-to-Text (STT):** Transcribes user commands using GPT-4o-mini-transcribe for accurate, streaming transcription.
- **Natural Language Understanding (NLU) and Task Classification:** Determines which of the four tasks the user intends to execute based on the transcribed text.
- **Text-to-Speech (TTS):** Generates verbal responses through GPT-4o-mini-TTS or pre-recorded audio, providing guidance, feedback, or explanations.

This also manages conversational context, language detection, and follow-up queries without requiring the wake word to be repeated, enabling a more fluid interaction for visually impaired users.

3.2.4 Task-Level Reasoning and Execution

Once a user command has been interpreted, the system performs task-level reasoning to determine the appropriate sequence of actions. Task execution coordinates navigation, perception, semantic information, and voice feedback to carry out the requested operation.

1. **Navigation:** Moves the robot to a specific object or location identified in the scene graph, e.g., "approach the table."
2. **Room Summary Queries:** Answers user questions about the environment, such as object locations or room descriptions.
3. **Out-of-Place Object Detection:** Assists the user in identifying misplaced or hazardous items by autonomously exploring the room, capturing multiple images, and performing multi-view visual analysis using a vision model.
4. **Food Label Scanning:** Guides the user to position a product, captures images with the Zed camera, performs OCR and text analysis, and answers queries about product information.

Each task relies on the semantic and geometric representation provided by the Scene Graph module and is executed in coordination with the Navigation and Perception modules. The Voice Interaction module ensures that the user receives immediate, natural feedback throughout task execution.

Chapter 4

Scene Graph Generation

The generation of the 3D scene graph follows a multi-stage pipeline that integrates geometric reconstruction, visual perception, and semantic reasoning. The process begins with environment exploration and spatial mapping using the ZED 2i stereo camera, which produces a dense 3D representation of the environment.

From the captured RGB-D data, objects are detected and segmented using deep learning models, while vision-language models enrich these detections with semantic descriptions. The geometric and semantic information is then fused to localize objects in the global coordinate frame. Finally, nodes and edges are constructed to produce a structured scene graph that encodes objects and their relationships.

The overall pipeline consists of the following stages:

1. Environment exploration and spatial mapping
2. Semantic perception (detection and segmentation)
3. 3D object localization and fusion
4. Scene graph construction
5. Graph-based semantic reasoning

4.1 Environment Exploration, Spatial Mapping, and Localization

To acquire a detailed 3D representation of the environment, the robot performs exploratory traversal of each room while capturing data with the ZED 2i stereo camera. Using the ZED SDK, RGB images are captured from the left camera at HD720 resolution and 15fps, a frame rate that balances image quality with computational efficiency on embedded platforms such as Jetson devices. Depth

frames are generated simultaneously using the SDK's Neural depth mode, providing high-quality depth estimation suitable for mapping and dense point-cloud generation.

4.1.1 Spatial Mapping Overview

Spatial mapping, also called 3D reconstruction, is the process of creating a digital model of the physical world. It allows a device to perceive real-world geometry in three dimensions, enabling collision avoidance, motion planning, and semantic understanding. The ZED continuously scans its surroundings, fusing new depth and positional information into a single, evolving spatial map.

The ZED SDK supports two types of spatial maps: meshes and fused point clouds. A mesh encodes the scene as a set of watertight triangles, optionally textured from captured RGB frames. A fused point cloud represents the environment as a collection of colored 3D points, providing a dense and metric reconstruction. Spatial mapping parameters such as resolution (1–12cm) and range (2–20m) can be adjusted at initialization to balance detail, coverage, and computational efficiency. High-resolution maps provide finer geometric detail, while lower resolutions reduce memory footprint.

During acquisition, the SDK maintains a continuous, fused point cloud, incrementally incorporating new frames to refine the 3D model. The resulting map is aligned to a fixed World Frame, ensuring global consistency. If Area Memory is enabled, previously saved maps can be reloaded and localized within the same physical space, supporting long-term mapping and repeatable interactions.

4.1.2 Positional Tracking Overview

Positional tracking provides real-time estimates of the camera's position and orientation in 3D space (6DoF: X, Y, Z translation; roll, pitch, yaw rotation). The ZED SDK combines stereo visual odometry with inertial measurements from the camera's IMU in a Visual-Inertial SLAM (VSLAM) pipeline. This integration enables robust motion estimation even during rapid movement, in low-texture areas.

Each camera pose includes position, orientation (as a quaternion), linear and angular velocities, and additional metadata such as timestamp and tracking confidence. These poses ensure that depth measurements from consecutive frames are accurately aligned, producing a globally consistent 3D reconstruction. The combination of positional tracking and spatial mapping forms the metric backbone for downstream semantic annotation, object localization, and scene graph construction.

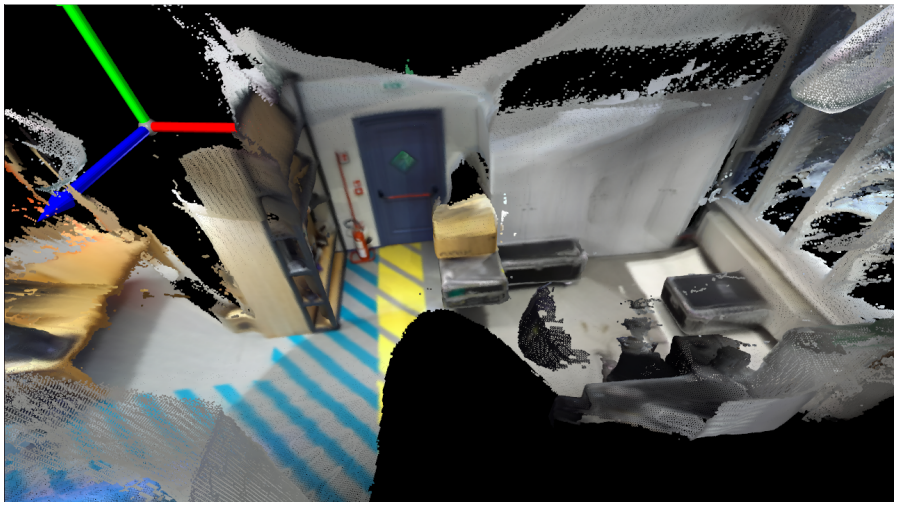

Figure 4.1: Global point cloud reconstruction generated after the exploration phase.

4.1.3 Data Output and Processing

At the end of traversal, the per-frame point clouds generated during spatial mapping are merged into a single global point cloud, representing the fully scanned environment. Camera intrinsics and extrinsics are also saved to guarantee geometric fidelity during post-processing or integration with other perception modules. The fused 3D reconstruction encodes both geometry and color, providing a rich foundation for semantic reasoning and task execution without relying on navigation in this phase.

4.2 Semantic Perception

While spatial mapping and localization provide a metric reconstruction of the environment, assistive reasoning requires an additional semantic layer that identifies objects, their properties, and their relationships. Semantic perception is therefore responsible for transforming raw RGB-D data into structured, meaningful entities that can be integrated into a 3D scene graph.

In this work, semantic perception is implemented as a multimodal pipeline that combines object detection, instance segmentation, and vision–language reasoning. This pipeline constitutes the baseline system for 3D scene graph generation and is designed to serve as a foundation for future extensions, including open-vocabulary recognition and more advanced semantic inference. Its primary objectives are to reliably detect and segment objects in complex indoor scenes, enrich visual perception with high-level semantic information, and produce consistent inputs for scene graph construction.

4.2.1 Object Detection

The first stage of the semantic perception pipeline focuses on identifying object instances within RGB frames acquired during exploration. Object detection is performed using the YOLOv8 architecture, selected for its strong balance between accuracy, inference speed, and robustness in indoor environments. YOLOv8 processes each frame and outputs a set of bounding boxes, each associated with a predicted object class and a confidence score. This stage establishes the initial semantic layer of the environment by identifying candidate objects that may later be localized in 3D space. Although detection is performed in the image plane, the resulting object hypotheses provide the structural anchors required for subsequent instance segmentation, depth association, and semantic reasoning. In practice, YOLOv8 demonstrates strong performance even in challenging indoor scenarios with variable lighting, reflective surfaces, and partial occlusions. Its real-time processing capability allows integration into robotic exploration pipelines without significant latency, enabling downstream tasks such as object tracking and interactive manipulation.

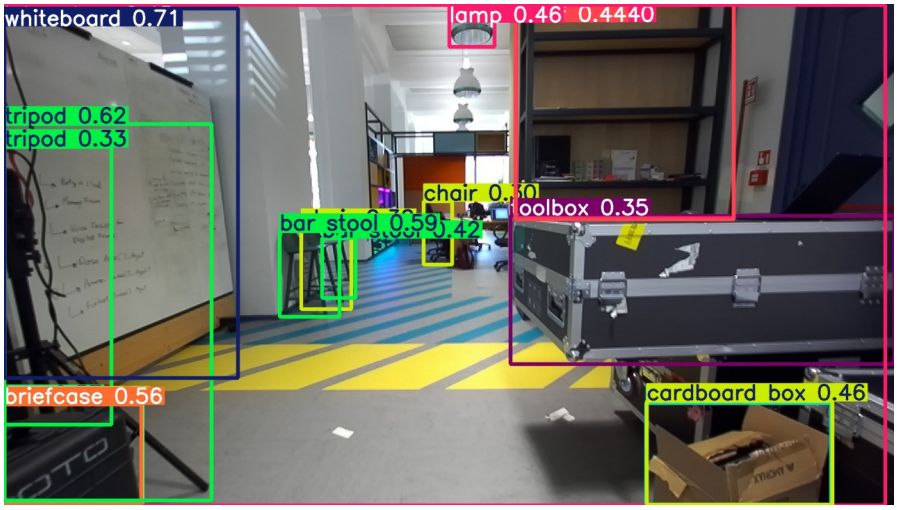

Figure 4.2: Example of object detection results in an indoor scene.

4.2.2 Instance Segmentation

To obtain precise object boundaries and enable accurate 3D localization, the pipeline performs instance segmentation for each detected object. Bounding boxes produced by the detector are used as prompts for the Segment Anything Model (SAM) and its lightweight variants. Instance segmentation yields pixel-accurate masks for each object, allowing the system to isolate object regions even in the presence of clutter, occlusions, or overlapping instances. These masks play a critical role in semantic perception: by restricting depth back-projection to object-specific pixels, they significantly improve the accuracy of 3D object position estimates. Furthermore, segmentation ensures that semantic attributes and relations are grounded in well-defined physical entities rather than coarse image regions. Beyond 3D localization, instance segmentation facilitates advanced reasoning tasks such as object affordance analysis, manipulation planning, and relational scene understanding. By leveraging SAM's robust segmentation capabilities, the system can generalize across a wide range of object types and scenes, maintaining high fidelity in both crowded and visually complex environments.

Figure 4.3: Pixel-accurate instance segmentation masks over detected objects.

4.2.3 Multi-View Object Fusion

Objects may be detected multiple times across different viewpoints during exploration. To ensure consistency, these observations are fused into a single object representation. Detections are associated across frames based on spatial proximity and overlap between their projected 3D bounding boxes. If two detections fall within a predefined distance threshold, they are considered observations of the same physical object and merged into a single node representation. This fusion process reduces duplicate detections and improves the robustness of object localization by averaging positional estimates across multiple observations.

4.2.4 Vision–Language Semantic Reasoning

Beyond object identity and geometry, understanding a domestic environment requires semantic interpretation of object roles and spatial relationships. To this end, the pipeline incorporates a multimodal vision–language model to perform semantic reasoning over visual inputs. The model receives RGB images, depth information, detected object classes, and segmentation masks, and produces structured semantic descriptions of the scene.

This reasoning stage extracts spatial relations such as *on top of*, *next to*, or *inside*, and contextual information describing how objects relate to one another within the environment. In addition, it generates natural-language descriptions that remain consistent with visual evidence, improving interpretability and alignment with human conceptualization of space.

The output of this stage is not treated as free-form text, but rather as structured

semantic information—objects, attributes, and relations—that can be directly integrated into a graph-based representation. This semantic grounding is particularly important for voice-based interaction, as it allows natural language queries to be mapped to concrete entities and relations in the environment.

4.2.5 3D Object Localization and Fusion

Semantic and geometric information are fused to localize objects in three-dimensional space. Using instance masks, depth measurements, and camera pose estimates, each detected object is back-projected from image space into the world reference frame to compute its 3D position and spatial extent.

To perform this transformation, pixels belonging to an object mask are converted from image coordinates into 3D points using the camera intrinsic parameters and depth measurements. For each pixel (u, v) with depth value d, the corresponding 3D point (X, Y, Z) in the camera coordinate frame is computed as:

$$X = \frac{(u - c_x)d}{f_x}, \quad Y = \frac{(v - c_y)d}{f_y}, \quad Z = d$$

where f_x and f_y represent the focal lengths of the camera, and (c_x, c_y) denotes the principal point. These points are subsequently transformed into the global world coordinate frame using the camera pose estimated by the positional tracking module.

Since the same object may be observed from multiple viewpoints during the exploration phase, observations across frames are fused to improve localization accuracy and reduce noise. This multi-view fusion step allows the system to consolidate repeated detections into a single, consistent object representation in the global map.

Through this process, objects are no longer treated as isolated detections in individual frames but become persistent entities anchored in a shared coordinate system. The resulting set of localized objects, enriched with semantic attributes and spatial information, forms the direct input to the scene graph construction process described in the following section.

4.3 Scene Graph Construction

The system constructs a **3D Scene Graph** from the segment-based map produced by Zed camera and object captions generated by LLaVA. The scene graph provides a structured and queryable representation of the environment, encoding objects as nodes and spatial or semantic relationships as edges.

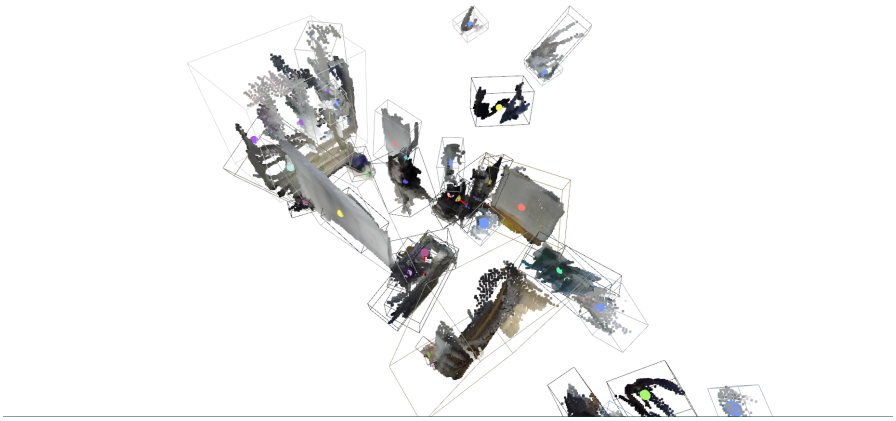

Figure 4.4: Example of the 3D scene graph representation generated by the system.

4.3.1 Semantic Perception

The semantic perception module constructs a structured representation of the environment by organizing detected objects and their relationships into a scene graph. This graph captures both geometric and semantic information, enabling higher-level reasoning and downstream tasks such as object querying, relation prediction, and interactive planning. Our pipeline leverages the ConceptGraph framework [27] for robust representation and reasoning over complex scenes.

4.3.2 Node Construction

Each node in the scene graph corresponds to a *segment* detected in the map. Nodes store the following information:

- **Object tag**: the refined label produced by GPT-4 based on LLaVA captions.
- **Bounding box metadata**: the 3D bounding box center and extent of the object.
- **Captions and possible tags**: textual descriptions extracted from LLaVA and refined by GPT-4.

Nodes with insufficient observations (fewer than a configurable minimum number of views) or invalid GPT responses are removed during preprocessing, ensuring the scene graph is built only from reliable object detections.

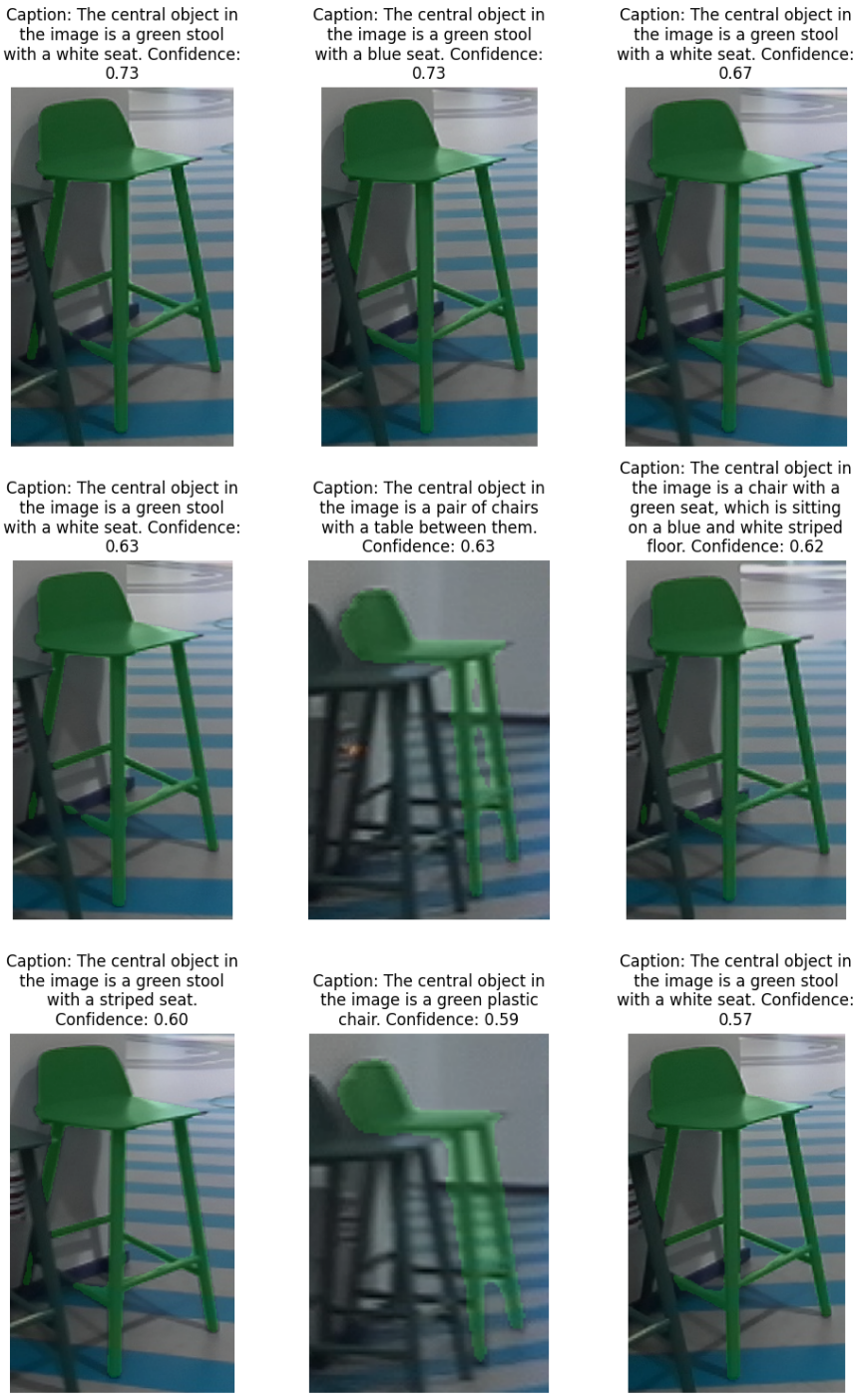

Figure 4.5: Example of generated captions for the object at the center of an image

4.3.3 Edge Construction

Edges in the scene graph represent spatial or semantic relationships between objects and are computed as follows:

1. A **weighted adjacency matrix** is computed using 3D bounding box overlaps between objects.
2. For each connected component of overlapping objects, a **minimum spanning tree (MST)** is extracted to define candidate edges.
3. For each candidate edge, GPT-4 is queried with the bounding boxes and object tags of the two nodes to predict the *object relation*. Valid relations include:
   - `a on b`: object a is commonly placed on top of object b
   - `b on a`: object b is commonly placed on top of object a
   - `a in b`: object a is commonly placed inside object b
   - `b in a`: object b is commonly placed inside object a
   - `a next to b`: object a and object b are adjacent horizontally
   - `none of these`: if none of the above best describe the relationship
4. Edges corresponding to `none of these` are discarded, and the remaining relations are added as edges in the scene graph.

4.3.4 Attributes

Nodes store minimal attributes required for reasoning and downstream tasks:

- **Object category/tag**: GPT-4 refined label.
- **Bounding box geometry**: center and extent in 3D space.
- **Captions**: textual descriptions from LLaVA and GPT-4.

Additional visual features extracted by LLaVA are saved separately for potential downstream tasks but are not directly included in the graph structure.

4.3.5 Graph Pruning and Output

Segments that fail validation checks or have low confidence are removed before constructing the graph. The final scene graph contains:

- Nodes representing valid objects in the environment.

- Edges encoding predicted spatial or semantic relationships between objects.

The graph is stored in a JSON summary containing node IDs, bounding boxes, object tags, and captions for interpretability and downstream integration.

4.4 Scene Graph as a Semantic Cognitive Map

The constructed 3D scene graph serves as a **semantic cognitive map** of the domestic environment, providing a compact, interpretable, and queryable representation that bridges low-level perception and high-level assistive functionality. Rather than storing raw sensory data or dense geometric reconstructions, the scene graph abstracts the environment into persistent semantic entities and their relationships, grounded in three-dimensional space.

From a functional perspective, the scene graph enables the system to reason about the environment in terms that are meaningful both to the robot and to visually impaired people. Objects are represented as distinct entities with semantic labels and spatial extents, while relations such as *on*, *in*, or *next to* encode how these entities are organized in the physical world. This abstraction allows the robot to answer spatial queries, describe scenes verbally, and support task execution without direct reliance on raw visual input at interaction time.

4.4.1 Queryability and Reasoning Support

The graph-based representation supports efficient querying over objects and their relations. Queries such as locating a specific object, identifying objects contained within a region, or retrieving objects adjacent to a given landmark can be resolved directly on the graph structure. This capability is fundamental for assistive tasks including object finding, room-level descriptions, and hazard inspection, where reasoning over spatial relationships is required.

Because each node is anchored to metric 3D information, graph queries can be seamlessly translated into spatial actions, such as guiding the robot toward an object or explaining its location relative to other entities. This tight coupling between symbolic structure and physical space is essential for grounding language and actions in the real environment.

4.4.2 Interpretability for Assistive Interaction

An important advantage of the scene graph representation is its inherent interpretability. Nodes, edges, and attributes correspond to human-understandable concepts, enabling the system to generate explanations and descriptions that align

with how people reason about space. This property is crucial for assistive interaction with visually impaired individuals, as it allows the robot to communicate information about the environment in clear and meaningful terms.

By functioning as a semantic cognitive map, the scene graph forms the central knowledge structure that connects perception, reasoning, and interaction. In the following chapters, this representation is leveraged to support assistive tasks and voice-guided interaction, demonstrating how structured semantic understanding enables practical and accessible robotic assistance in domestic environments.

Chapter 5

Voice-Based Interaction and Command Understanding

5.1 Introduction

Voice-based interaction represents the primary and, in many cases, the only accessible communication channel between the robot and visually impaired people. Unlike graphical or tactile interfaces, spoken language enables hands-free, natural, and immediate interaction, making it particularly suitable for assistive scenarios in domestic environments. For this reason, the voice interaction system is designed to be robust, low-latency, and privacy-aware, ensuring that users can reliably communicate with the robot and receive timely feedback.

In this chapter, we describe the multimodal voice interaction system developed in this work, focusing on how spoken input is captured, interpreted, and transformed into structured task intentions or semantic queries. At this stage, voice interaction is fully decoupled from navigation and physical execution: its role is to understand user intent and produce structured representations that can later drive robot behavior.

5.1.1 High-Level Architecture

The voice interaction system follows a layered architecture in which each component performs a well-defined function and communicates through explicit interfaces. Audio input is processed incrementally, enabling continuous listening with minimal computational overhead. Once a command is recognized and interpreted, the system outputs either a structured query to the scene graph or a task intention that will be executed by downstream modules.

5.2 System Architecture

5.2.1 Audio Trigger and Capture Layer

The audio trigger and capture layer is responsible for activating the interaction pipeline and acquiring the user's spoken commands. This layer combines wake-word detection, Voice Activity Detection (VAD), and audio recording to ensure responsiveness while minimizing unnecessary processing.

Wake-Word Detection

Wake-word detection is implemented using **openWakeWord**, an open-source wake-word detection framework that operates directly on streaming audio without requiring speech transcription. The activation phrase used in this system is *"Hey computer"*. Processing audio directly at the signal level provides two key advantages: low latency and improved privacy, as no speech content is transmitted or stored prior to activation.

Operating entirely on-device, openWakeWord allows the system to remain in a passive listening state until the wake word is detected, significantly reducing computational load and avoiding continuous speech analysis. This design choice is particularly important in assistive contexts, where privacy and trust are essential.

Performance metrics for wake-word detection, including precision, recall, and false activation rate, are reported in Table 5.1.

Table 5.1: Performance metrics of the "Hey computer" wake-word model used for system activation.

Metric	Value
Accuracy	0.783
Recall	0.570
False Positives per Hour	5.22
Training Examples	25,000
Training Steps	500,000
False Activation Penalty	5,000

Voice Activity Detection

Once the system is activated, voice activity detection is performed using **Silero VAD**, an open-source neural VAD model optimized for real-time applications. Silero VAD continuously monitors the audio stream and determines when speech

is present based on configurable confidence thresholds and minimum duration constraints.

Audio recording begins when speech is detected and continues until silence is observed. This mechanism ensures that complete utterances are captured while avoiding unnecessary delays or truncation. The use of VAD also contributes to reducing latency and computational overhead, as downstream modules are only activated when relevant speech input is available.

5.2.2 Speech-to-Text Layer

Captured audio is transcribed using the `gpt-4o-transcribe` model. This Speech-to-Text (STT) module was selected for its high accuracy on short commands and spontaneous speech, as well as its ability to operate in a streaming configuration. Transcription is performed immediately after recording ends, producing text that serves as the input to the natural language understanding pipeline.

Low transcription latency is essential in assistive interaction, as delayed responses can reduce user confidence and disrupt conversational flow.

5.2.3 Natural Language Understanding and Task Classification

The transcribed text is processed by a Natural Language Understanding (NLU) module that extracts the semantic intent of the command and classifies it into one of the supported task categories. This classifier operates using confidence-based criteria and fallback strategies to ensure robust behavior even when commands are ambiguous or partially specified.

The system supports four task categories:

1. Scan
2. Summary
3. Out-of-place objects
4. Navigation

At this stage, no physical action is executed. Instead, the output of this module is a structured task intention that determines how subsequent reasoning and execution modules will operate.

5.2.4 Response Generation Layer

Once the task has been identified, a response generation module produces an appropriate textual reply. This module integrates the user's command, the classified task, and the conversational context to generate coherent and informative responses. Depending on the task, the response may provide immediate feedback, request clarification, or prepare the user for the next interaction step.

5.2.5 Text-to-Speech Output

Verbal feedback in the system is generated using the `gpt-4o-tts` model. To determine the most suitable configuration for interactive use, two different synthesis strategies were experimentally evaluated: chunk-based generation and streaming generation.

In the chunk-based configuration, the model generates the entire audio response before playback begins. While this approach ensures that the complete audio signal is available prior to output, it introduces additional latency because the user must wait for the full generation process to finish.

In contrast, the streaming configuration allows audio playback to begin as soon as the first speech tokens are produced. This approach significantly reduces perceived latency, as the user can begin hearing the response while the remainder of the audio is still being generated. Such behavior is particularly beneficial for conversational interaction, where responsiveness strongly affects usability.

To compare the two approaches, we conducted a series of tests measuring generation time, playback duration, and total response time. The results are summarized in Table 5.2. The experiments show that while playback duration remains similar for both configurations, streaming significantly reduces the time required to start the response. In particular, the average generation time decreases from approximately 2.08 seconds in the chunked configuration to 0.55 seconds in streaming mode, resulting in a noticeable reduction in total response latency.

Table 5.2: Comparison between chunked and streaming Text-to-Speech generation using `gpt-4o-tts`.

Metric	**Chunked**	**Streaming**
Average Generation Time (s)	2.083	0.553
Average Playback Time (s)	6.281	6.016
Average Total Time (s)	8.363	6.569
Minimum Total Time (s)	8.170	5.699
Maximum Total Time (s)	8.682	6.891

These results confirm that the streaming configuration provides a faster and

more responsive interaction, making it more suitable for assistive scenarios where voice serves as the primary communication channel. Reduced response latency improves conversational flow and helps minimize cognitive load for visually impaired users during interaction.

After producing a voice response, the system does not immediately return to wake-word listening. Instead, the Voice Activity Detection (VAD) module remains active for a predefined time window, allowing the user to continue speaking without repeating the wake word. This design choice supports more natural multi-turn conversations and reduces the effort required for continued interaction.

5.3 Mapping Spoken Commands to Semantic Queries

Spoken commands are mapped to structured queries over the scene graph or to task-level intentions. For example:

- "Where is the bottle?" is translated into a query over object nodes.
- "Is there something dangerous?" triggers an action to check the environment
- "Approach the table" results in target object identification

This abstraction layer ensures that language understanding remains independent from execution, improving modularity and robustness.

5.4 Latency, Privacy, and Edge–Cloud Considerations

Latency is a critical factor in voice-based assistive systems, as delayed feedback can compromise usability and safety. For this reason, wake-word detection and VAD are executed entirely on-device, eliminating network delays and preserving user privacy. Speech transcription, language understanding, and response generation leverage cloud-based models to achieve higher accuracy and flexibility, while streaming techniques mitigate latency.

This hybrid edge–cloud approach balances responsiveness, computational efficiency, and privacy. In contexts where sound is the only communication channel, such trade-offs are essential to ensure reliable and trustworthy interaction.

Chapter 6

Task Execution

This chapter describes how the proposed assistive system translates high-level user requests into concrete actions executed by the robot in the physical environment. While previous chapters focused on perception, semantic representation, and voice-based interaction, the emphasis here is on *task execution*, where natural language commands are grounded in navigation, sensing, and decision-making.

Task execution is driven by the combination of spoken user input and the semantic scene graph constructed during autonomous exploration. Natural language provides goal-level intent, while the scene graph supplies structured spatial knowledge about rooms, objects, and their relationships. By reasoning jointly over these two sources of information, the robot is able to determine what action should be performed and where it should take place.

The system supports multiple assistive behaviors, including navigation to objects, answering environment-related questions, detecting out-of-place or hazardous items, and scanning food labels. All tasks follow a common execution pattern: user intent is inferred from speech, relevant semantic information is retrieved from the scene graph, and the robot executes the required actions while maintaining continuous verbal and auditory feedback. This design ensures that the robot operates autonomously while remaining transparent, predictable, and accessible for visually impaired users.

6.1 Navigation to a Target Object

Navigation to a target object is the most fundamental task supported by the system and serves as a building block for higher-level assistive behaviors. The objective is to guide the robot—and implicitly the user—to a specific object or location in the environment based on a spoken command such as “approach the fire hydrant” or “take me to the table.”

From a robotics perspective, this task belongs to the broader class of *goal-directed navigation problems*, where the robot must interpret a high-level objective, localize the target in its internal representation of the environment, and compute a safe trajectory to reach it. In the proposed system, this process integrates three distinct components: semantic reasoning, geometric transformation, and autonomous navigation. The combination of these layers allows natural language instructions to be translated into reliable physical actions.

6.1.1 From Natural Language to Semantic Navigation Goals

The navigation process begins with the interpretation of a natural language query issued by the user. The transcribed speech is processed using a prompt-based reasoning pipeline built on GPT-4o. Instead of directly generating low-level motion commands, the language model is provided with two key inputs:

- the user's spoken query, and
- the current semantic scene graph representing the environment.

This design follows the principle of *language grounding*, in which symbolic language expressions are mapped onto entities and relations present in the robot's internal world model. The scene graph provides a structured semantic representation of the environment, where nodes correspond to objects and edges encode spatial or semantic relationships. By reasoning over this graph, the language model can resolve references in the user's query and identify the corresponding object instance.

Conceptually, a command such as

"Approach the fire hydrant"

is translated into a symbolic navigation goal of the form:

$$\text{Go to } (\texttt{fire\_hydrant}, x = x', y = y', z = z')$$

where (x', y', z') correspond to the estimated position of the object stored in the scene graph reference frame.

This intermediate symbolic representation separates semantic reasoning from motion execution. Rather than directly producing motor commands, the system first resolves the linguistic intent into a structured goal that can be interpreted by the robot's navigation stack. This abstraction layer improves robustness to linguistic variability and allows the same navigation infrastructure to support multiple forms of natural language commands.

6.1.2 Coordinate Frame Transformation

The scene graph maintains object locations in its own reference frame, which is distinct from the global coordinate frame used by the robot's navigation system. In robotic systems, multiple coordinate frames are typically used to represent spatial relationships between sensors, maps, and objects. Managing these frames consistently is essential for ensuring that information generated by different subsystems remains geometrically consistent.

To bridge the gap between semantic reasoning and physical navigation, the system defines a transformation between the scene graph frame and the global world frame used by the robot. Let $\mathbf{p}_g$ denote the position of an object in the scene graph frame. The corresponding position in the world frame $\mathbf{p}_w$ can be computed using a rigid-body transformation:

$$\mathbf{p}_w = R\mathbf{p}_g + \mathbf{t}$$

where R is a rotation matrix representing the orientation difference between frames and $\mathbf{t}$ is a translation vector representing the displacement between their origins.

This transformation is published within the ROS 2 transform (TF) system, which maintains a dynamic tree of coordinate frames and enables consistent spatial reasoning across all components of the robot. Once transformed into the world frame, the object position becomes a valid navigation goal that can be consumed by the robot's motion planning algorithms.

This design cleanly separates semantic reasoning from geometric navigation: the scene graph acts as an abstract cognitive map, while the navigation stack operates purely in metric space.

6.1.3 Localization, Mapping, and Obstacle Avoidance

Safe and autonomous motion in domestic environments is achieved through the combined use of SLAM Toolbox and the ROS 2 Navigation Stack (Nav2). These components implement the core algorithms required for localization, mapping, and motion planning.

Simultaneous Localization and Mapping (SLAM) addresses the fundamental robotics problem of estimating a robot's pose while simultaneously constructing a map of the environment. Formally, SLAM aims to estimate the posterior distribution:

$$P(x_t, m \mid z_{1:t}, u_{1:t})$$

where x_t represents the robot pose at time t, m denotes the map of the environment, $z_{1:t}$ are sensor observations, and $u_{1:t}$ are control inputs. SLAM Toolbox

implements efficient graph-based SLAM algorithms that maintain a pose graph representing the robot's trajectory and refine it using optimization techniques.

The resulting map is represented as a 2D occupancy grid, where each cell encodes the probability that the corresponding region of space is occupied by an obstacle. This representation enables efficient path planning and collision checking.

On top of this localization layer, Nav2 provides the full navigation pipeline, including:

- global path planning,
- local trajectory generation,
- dynamic obstacle avoidance,
- recovery behaviors.

Global planning typically relies on graph search algorithms such as A* or Dijkstra's algorithm to compute a feasible path through the occupancy grid. Local planning then refines this path into smooth velocity commands while respecting the robot's kinematic constraints and reacting to dynamic obstacles detected by onboard sensors.

The layered costmap architecture used in Nav2 combines static map information with real-time sensor observations, enabling the robot to adapt to changes in the environment while maintaining safe navigation behavior.

Together, SLAM Toolbox and Nav2 allow the system to continuously transform high-level navigation intents derived from natural language into safe, executable motion commands.

6.1.4 Auditory Guidance for Visually Impaired Users

Throughout navigation, the robot provides continuous auditory feedback to support visually impaired users. Human–robot interaction research shows that non-visual feedback modalities are essential for accessible assistive robotics systems. In environments where visual cues cannot be relied upon, auditory signals provide a natural channel for conveying spatial and behavioral information.

In addition to verbal confirmations and status updates, the system emits a non-verbal acoustic cue while the robot is moving. This sound acts as an auditory beacon that allows the user to follow the robot's trajectory, maintain awareness of its position, and safely approach the target object together with the robot.

Continuous auditory feedback also improves user trust and situational awareness. By providing real-time information about the robot's actions and movement state, the system reduces uncertainty and helps users build an intuitive mental model of the robot's behavior.

By combining spoken feedback with continuous auditory cues, the system reduces cognitive load while enhancing spatial awareness. This multimodal feedback strategy ensures that navigation remains transparent, predictable, and accessible from the moment a natural language command is issued to the moment the target object is reached.

6.2 Environment Question Answering

Environment Question Answering enables the robot to respond to user queries about the surrounding space by leveraging the semantic and spatial information stored in the scene graph. Unlike navigation tasks, which result in physical motion, this functionality focuses on *explanatory assistance*: identifying rooms, locating objects, and describing spatial relationships in a way that is meaningful and accessible to visually impaired users.

This capability is particularly important in domestic environments, where users may need to quickly understand the layout of a room or the position of specific objects without physically navigating to them.

6.2.1 Room-Level Spatial Summarization

During exploration, the robot constructs a scene graph that encodes both semantic labels (e.g., `couch`, `table`, `door`) and metric information (3D coordinates and extents) for all detected entities. For each room, this information is used to generate a high-level spatial summary that captures:

- the type of room (e.g., living room, kitchen),
- the main objects present in the room,
- their approximate positions within the room.

To produce human-readable descriptions, the system converts the structured scene graph into a textual representation listing objects together with their coordinates and semantic attributes. This representation is then provided to a large language model, which generates a concise spatial summary of the room. The summary combines object labels and relative positioning to convey spatial awareness, for example describing which objects are central, near walls, or close to each other.

Summaries are generated once per room and cached, allowing the system to reuse them efficiently during subsequent interactions.

6.2.2 Object Location Queries and Relational Reasoning

When a user asks a question such as:

> "Where is the couch?"

the system interprets the query as a request for spatial information rather than an action command. The referenced object is identified within the scene graph, and its position is analyzed relative to:

- the room reference frame, and
- nearby or semantically relevant objects.

Rather than reporting raw coordinates, the system performs relational reasoning over the scene graph to generate descriptions that align with human spatial understanding. For example, the robot may explain that the couch is *along the left wall*, *in front of the coffee table*, or *near the window*. This abstraction from metric data to relational language is essential for usability, as absolute coordinates are not meaningful to end users.

6.2.3 Voice-Based Spatial Feedback

Answers to environment-related questions are delivered through synthesized speech. The system prioritizes clarity and conciseness, ensuring that responses provide enough spatial context without overwhelming the user. By combining semantic labels with relative spatial descriptions, the robot enables users to build a mental map of the environment over time.

This question-answering mechanism complements navigation-based assistance: users can either ask the robot to explain where objects are or request to be guided to them physically. Together, these interaction modes provide flexible and intuitive access to spatial information, reinforcing the role of the scene graph as a shared representation between perception, reasoning, and natural language interaction.

6.3 Out-of-Place and Hazard Detection

The third assistive task supported by the system focuses on identifying objects that are either misplaced or potentially hazardous for a visually impaired user. Unlike navigation and environment question answering, which rely primarily on the semantic scene graph as a relatively static representation of the environment, this task emphasizes *dynamic, perception-driven analysis* of the current room state.

Domestic environments are inherently dynamic: objects may be moved, left on the floor, or temporarily placed in unsafe locations. As a result, hazards such as

tripping obstacles, sharp objects, or items left out of place may not be accurately captured by the scene graph constructed during initial exploration. To address this limitation, the system adopts an active perception strategy that prioritizes real-time visual inspection over static semantic memory.

6.3.1 Active Perception Through Exploratory Navigation

When the user requests an out-of-place or hazard detection task (e.g., "Is there anything dangerous in this room?"), the robot initiates an exploratory behavior rather than querying the scene graph directly. The robot autonomously navigates a short loop around the room

During this motion, the robot relies on its localization, mapping, and obstacle avoidance capabilities to safely traverse the space while maintaining a suitable distance from furniture and users. As it moves, the robot captures multiple RGB images of the room from different angles and positions. This multi-view acquisition strategy improves robustness by reducing occlusions and enabling better spatial understanding of object placement.

6.3.2 Visual-Language Model–Based Hazard Recognition

The collected images are processed by a vision–language model using a safety-oriented prompt. Rather than performing explicit object detection or relying on predefined hazard categories, the model is asked to reason holistically about the scene. By analyzing multiple images simultaneously, the model can:

- identify objects that appear misplaced (e.g., items left on the floor or blocking walkways),
- recognize potentially dangerous objects (e.g., sharp tools, spilled liquids, exposed cables),
- reason about spatial relationships using natural descriptors such as "near the door" or "in front of the sofa."

This approach allows the system to flexibly adapt to a wide range of household environments and object configurations without requiring task-specific visual models. The vision–language model effectively acts as a high-level safety assessor, interpreting visual evidence in the context of everyday domestic risks.

6.3.3 Verbal Hazard Communication and User Awareness

Once hazards or misplaced items are identified, the system communicates the findings to the user through synthesized speech. Responses are designed to be

concise, spoken-style, and spatially informative. For example, instead of merely stating that an obstacle exists, the robot explains where it is located relative to prominent landmarks in the room.

This verbal feedback enables the user to take appropriate action, such as avoiding a specific area, removing an obstacle, or asking the robot for further assistance. By combining active exploration, visual reasoning, and natural language explanations, the system provides a practical and user-centered solution for maintaining safety in dynamic domestic environments.

6.4 Food Label Scanning and Product Understanding

Figure 6.1: Example of food label scanning execution

Food label scanning enables the robot to assist users—particularly visually impaired individuals—in accessing detailed product information that is typically conveyed through small, densely packed text. This task focuses on extracting, organizing, and explaining textual and semantic information from product packaging, such as ingredients, allergens, expiration dates, and nutritional values, while supporting natural, conversational queries.

Unlike navigation or environment question answering, this task is object-centric and requires close-range, high-quality visual input. To achieve robust and accessible label understanding, the system combines guided perception, multi-frame acquisition, classical computer vision filters, open-vocabulary vision models, and large language models for reasoning and interaction.

6.4.1 Guided Camera Framing with Vision-Language Models

The scanning pipeline begins with real-time camera framing guidance provided by a GPT-based vision model. As the user holds or positions the product in front of the camera, the model continuously analyzes the live visual stream and generates spoken, actionable feedback such as:

> "Move the object closer," "Tilt the package slightly," "Shift a bit to the left."

This step is critical for accessibility, as it replaces visual alignment cues with auditory instructions. By ensuring that the label is centered, fully visible, and well illuminated before capture, the system significantly improves downstream text extraction quality. This guided interaction also reduces user frustration and minimizes the number of failed scans.

6.4.2 Multi-Frame Acquisition for Robust Coverage

Once the product is properly positioned, the system captures multiple frames of the label from slightly different viewpoints. This multi-frame acquisition strategy addresses common challenges in real-world product scanning, including:

- reflections and glare on glossy packaging,
- curved or cylindrical surfaces,
- partial occlusions caused by the user's hand,
- very small or dense text regions.

By observing the label from multiple angles, the system increases the likelihood that all relevant information appears clearly in at least one frame.

Frame Similarity Filtering Using pHash

To avoid redundant processing and unnecessary computational cost, the system employs perceptual hashing (pHash) to compare newly captured frames against previously stored ones. Perceptual hashing generates a compact representation of an image that preserves its visual structure while remaining robust to minor variations in illumination, compression artifacts, and small geometric transformations.

Unlike cryptographic hash functions, which change completely with even a single pixel modification, perceptual hashes map visually similar images to similar hash values. This property makes them suitable for detecting near-duplicate frames.

The pHash algorithm typically operates by transforming the image into the frequency domain using the Discrete Cosine Transform (DCT). Low-frequency coefficients are then used to generate a binary hash representing the global visual structure of the image. The similarity between two frames can be evaluated by computing the Hamming distance between their hashes:

$$d_H(h_1, h_2) = \sum_{i=1}^{N} \mathbb{1}(h_1^i \neq h_2^i) \tag{6.1}$$

where h_1 and h_2 are the hash vectors and N is the hash length. A small Hamming distance indicates high visual similarity.

If the similarity between two frames exceeds a predefined threshold, the newly captured frame is considered redundant and discarded. Only frames that differ sufficiently in viewpoint or visual appearance are retained. This filtering process ensures that each stored frame contributes new visual information while maintaining computational efficiency.

Image Quality Control via Laplacian Variance

In addition to similarity filtering, image sharpness is evaluated using a Laplacian variance metric, which serves as a widely used focus measure in computer vision. The Laplacian operator computes the second spatial derivative of an image, highlighting regions of rapid intensity change that correspond to edges and fine structural details.

Formally, the Laplacian of an image $I(x, y)$ is defined as:

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \tag{6.2}$$

Sharp images typically contain a higher number of strong edges and fine details, resulting in larger variations in the Laplacian response. Conversely, blurred images exhibit smoother intensity transitions and therefore lower variance in the Laplacian output.

To quantify this effect, the system computes the variance of the Laplacian response:

$$\sigma_L^2 = \mathrm{Var}(\nabla^2 I) \tag{6.3}$$

Frames with Laplacian variance below a predefined threshold are classified as blurry and discarded. This filtering step is critical for maintaining OCR accuracy, as motion blur and defocus significantly degrade the ability of text recognition models to identify characters.

By combining Laplacian-based blur detection with pHash similarity filtering, the system produces a refined set of frames that are both visually distinct and

sufficiently sharp. This curated collection of images forms a reliable basis for subsequent stages of text extraction, semantic parsing, and product information retrieval.

6.4.3 Object-Aware Text Localization with YOLOv8

Before performing text extraction, the system applies a YOLOv8-based object detection model to identify the bounding box of the product within the captured frame. This preprocessing step isolates the product from the surrounding environment and ensures that subsequent vision-language processing operates only on the relevant region of the image.

In unconstrained environments, captured images often contain significant background clutter, including shelves, other products, or surrounding objects. Processing the entire frame with a vision-language model may introduce irrelevant visual context that can negatively affect text interpretation. By detecting the product region first, the system reduces this noise and constrains the extraction process to the visual area most likely to contain label information.

Object detection can be formulated as a prediction task in which the model estimates the spatial location of objects within an image. Given an input image I, the detector produces a set of bounding boxes

$$B = \{b_1, b_2, ..., b_n\} \tag{6.4}$$

where each bounding box b_i is parameterized by its center coordinates (x_i, y_i) and spatial dimensions (w_i, h_i). Each bounding box is also associated with a confidence score representing the probability that the region contains the target object.

In this system, the detector is trained to identify the product as a whole rather than specific label regions. The bounding box with the highest confidence is selected and used to crop the corresponding region from the original frame. This cropped image serves as the input to the vision-language model responsible for text interpretation.

YOLOv8 belongs to the family of single-stage object detectors, which perform localization and classification in a single forward pass through the network. This architecture allows the system to achieve real-time performance while maintaining high localization accuracy. The model uses a convolutional backbone to extract hierarchical visual features, followed by feature aggregation layers that combine multi-scale information to improve detection robustness.

Once the product bounding box is obtained, the system extracts the corresponding image region:

$$I_{crop} = I[x_i - w_i/2 : x_i + w_i/2,\ y_i - h_i/2 : y_i + h_i/2] \tag{6.5}$$

This cropped region contains the product and its label while excluding most background elements. The resulting image is then passed to the vision-language model for text extraction.

By restricting the visual input to the detected product region, the system significantly reduces irrelevant context and improves the reliability of the subsequent text interpretation process. This step effectively increases the signal-to-noise ratio of the visual input, allowing the vision-language model to focus exclusively on the product packaging and the textual information it contains.

6.4.4 Open-Vocabulary Text Extraction with GPT Vision

All validated frames are processed using a GPT-based vision-language model capable of open-vocabulary text interpretation. Unlike traditional optical character recognition (OCR) systems, which typically rely on explicit character segmentation and fixed recognition vocabularies, modern vision-language models jointly encode visual and textual information within a shared representation space. This allows the system to interpret diverse product labels without relying on predefined templates or rigid layout assumptions.

In this paradigm, the image is converted into a sequence of visual embeddings representing spatial regions or patches of the image. These embeddings are processed by a transformer-based architecture that models relationships between visual features and language tokens. Through large-scale multimodal pretraining, the model learns to associate visual patterns with textual descriptions and semantic concepts.

To ensure reliability, the prompting strategy used for the vision-language model is intentionally **highly conservative**. The model is explicitly instructed to extract only the text that is clearly visible in the image and to avoid guessing or inferring missing information. If any part of the text is partially occluded, blurred, or unreadable, the model is instructed to either ignore that portion or explicitly mark it as uncertain rather than attempting to reconstruct it.

This conservative prompting strategy is particularly important given the intended application of the system. The platform is designed to assist visually impaired users in accessing critical product information, including ingredients, allergens, and expiration dates. In such contexts, hallucinated or incorrectly inferred information could lead to serious consequences for user health and safety. By enforcing strict extraction rules, the system prioritizes factual accuracy and traceability over completeness.

The model therefore focuses on extracting only verifiable textual content from the product packaging, including:

- ingredient lists and allergen warnings,

- nutritional tables and serving information,
- expiration dates and storage instructions,
- product descriptions, brand names, and manufacturer details.

Text extracted from multiple frames is aggregated and deduplicated to form a unified representation of the product label. Because different viewpoints may reveal different parts of the packaging, combining information across frames increases overall coverage while preserving reliability. During aggregation, redundant or semantically equivalent text segments are removed using similarity-based filtering to produce a clean and consolidated representation of the label content.

This multi-frame aggregation process further increases robustness by ensuring that information is retained only when it has been explicitly observed in at least one frame, thereby maintaining the system's conservative extraction policy.

6.4.5 Knowledge Base Construction and Incremental Memory

The consolidated label text is appended to a raw text store that serves as the basis for building a semantic knowledge base. To enable efficient semantic retrieval, the text is segmented into smaller units that preserve contextual meaning while remaining suitable for embedding-based indexing. Typical segmentation strategies include sentence-level or paragraph-level chunking.

Each text segment is converted into a vector representation using a sentence embedding model. These embeddings map textual content into a high-dimensional vector space where semantically similar pieces of information lie close to each other according to a distance metric such as cosine similarity.

Formally, given a text segment t_i, the embedding function $f(\cdot)$ produces a vector representation:

$$\mathbf{e}_i = f(t_i) \tag{6.6}$$

Semantic similarity between two segments t_i and t_j can then be computed using cosine similarity:

$$\text{sim}(\mathbf{e}_i, \mathbf{e}_j) = \frac{\mathbf{e}_i \cdot \mathbf{e}_j}{\|\mathbf{e}_i\|\|\mathbf{e}_j\|} \tag{6.7}$$

These embeddings are indexed using the FAISS library, which enables efficient nearest-neighbor search in high-dimensional vector spaces. FAISS employs approximate nearest-neighbor algorithms that significantly reduce retrieval latency while maintaining high recall.

While the FAISS index itself is rebuilt from scratch after each update to prevent stale entries, the underlying text store grows incrementally. Each new scan enriches the existing knowledge rather than overwriting it, resulting in progressively more complete product representations over time.

This incremental memory design allows the system to accumulate information across multiple scans and products, effectively building a continuously expanding semantic repository of product knowledge.

6.4.6 Retrieval-Augmented Reasoning and Question Answering

When the user asks questions such as:

> "Does this contain nuts?" "What is the expiration date?" "Is this suitable for a low-sodium diet?"

the system employs retrieval-augmented generation (RAG). In this paradigm, language model reasoning is grounded in externally retrieved knowledge rather than relying solely on the model's internal parameters.

Given a user query q, the system first converts the query into an embedding vector:

$$\mathbf{e}_q = f(q) \tag{6.8}$$

The FAISS index is then used to retrieve the k most semantically similar text segments from the knowledge base. These retrieved segments serve as contextual evidence for the language model. The language model receives both the user query and the retrieved context as input and generates a response conditioned on this information.

This retrieval step serves two important purposes. First, it grounds the generated response in factual content extracted directly from the product label. Second, it reduces the risk of hallucination by constraining the model to reason over relevant evidence.

Each answer is accompanied by a confidence score derived from retrieval relevance and contextual consistency. If the confidence falls below a predefined threshold or if critical information is missing from the retrieved context, the system automatically requests a re-scan and guides the user to capture additional frames.

This feedback mechanism ensures that the system maintains a high level of reliability and prevents the generation of unsupported or speculative responses. By tightly coupling visual perception, semantic retrieval, and language reasoning, the pipeline provides an interpretable and trustworthy interface for accessing product information.

6.4.7 Performance Evaluation

The performance of the label scanning pipeline was systematically evaluated by scanning a diverse set of consumer products and querying a predefined set of questions for each item. The evaluation focused on extracting safety-critical and commonly requested information:

- product name,
- ingredients,
- expiration date,
- whether the product is expired.

For each product, ground truth answers were manually annotated based on careful inspection of the physical label. The system's responses were then compared against the ground truth using semantic similarity rather than exact string matching. A BERT-based sentence embedding model was used to compute similarity scores between predicted and expected answers, allowing robust evaluation despite variations in phrasing.

Table 6.1 reports the average semantic similarity aggregated across all evaluated products.

Question	**Average Similarity**
Is it expired?	0.932
What are the ingredients?	0.800
What is the expiration date?	0.930
What is the product name?	0.842

Table 6.1: Average semantic similarity scores for food label queries.

The system achieves an overall mean similarity score of **0.876**, indicating a high degree of accuracy and reliability. Performance is strongest for expiration-related queries, which typically involve short and well-structured text, while ingredient lists exhibit slightly lower scores due to their length and variability.

Despite these strong results, the system remains sensitive to challenging lighting conditions, low-resolution imagery, and highly curved or reflective packaging. These factors can reduce text clarity and lead to incomplete extraction. The guided framing stage and automatic re-scan mechanism help mitigate these limitations by prompting users to acquire additional visual data when confidence in the extracted information is low.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis presented the design and implementation of an assistive robotic system intended to support visually impaired users in domestic environments. The central goal of the work was to demonstrate how modern advances in robotic perception, semantic scene understanding, and voice-based human–robot interaction can be integrated into a unified assistive framework capable of performing meaningful daily-life tasks.

The proposed system combines a mobile humanoid robotic platform, an RGB-D perception pipeline, semantic scene graph construction, and natural language interaction. Together, these components enable the robot to build a structured representation of its environment and to interact with users through intuitive spoken commands. By grounding language in a spatially consistent semantic map, the system can transform high-level user requests into executable actions while maintaining transparency and interpretability.

From a system architecture perspective, the work introduced a modular design composed of several tightly connected subsystems, including navigation, perception, scene graph construction, voice interaction, and task-level reasoning. This modular structure allows each component to be independently developed and extended while maintaining efficient communication between modules.

A key contribution of the thesis is the use of a *semantic scene graph* as a cognitive representation of the environment. By combining geometric reconstruction with semantic perception and vision–language reasoning, the system constructs a graph-based representation where nodes correspond to objects and edges encode spatial or semantic relationships. This representation serves as the central knowledge structure used for navigation, querying, and task execution. In contrast to purely geometric maps, the scene graph allows the robot to reason about the environment

in terms that are meaningful to humans, such as objects being located *on*, *next to*, or *inside* other objects.

Another important aspect of the system is the integration of natural language interaction. Voice-based communication enables visually impaired users to interact with the robot through spoken commands and questions, eliminating the need for visual interfaces. This conversational interface allows users to request navigation assistance, ask questions about their surroundings, detect potential hazards, and obtain information about products.

To demonstrate the practical capabilities of the proposed architecture, the thesis implemented four assistive tasks: navigation to target objects, environment question answering, out-of-place object detection, and food label scanning. These tasks highlight the versatility of the system and illustrate how semantic perception, navigation, and language understanding can be combined to support real-world assistive scenarios. In particular, the food label scanning task demonstrates how vision–language models can be used to extract product information in a reliable way, which is essential when assisting visually impaired users.

7.2 Limitations

Despite the encouraging results obtained in this work, several limitations remain that highlight opportunities for further research and development.

One important limitation concerns the generation of the semantic scene graph. In the current implementation, the scene graph is constructed through an exploration phase that processes captured images offline or in batch mode. As a result, scene graph updates are not performed fully in real time. While this approach simplifies processing and ensures accurate semantic reasoning, it limits the system's ability to dynamically update its understanding of the environment as new observations become available.

Another limitation is related to computational requirements. Several components of the system rely on large vision–language models and cloud-based language processing services. Although these models provide powerful reasoning capabilities, they introduce latency and depend on network connectivity. For assistive applications in real homes, reducing this dependence and improving edge-based inference capabilities would be beneficial.

Perception robustness also remains a challenge. Although the system combines object detection, segmentation, and multi-view reasoning, recognition performance may still degrade in cluttered environments, under poor lighting conditions, or when objects are heavily occluded. Furthermore, the accuracy of the scene graph depends on reliable object detection and spatial localization, which may vary across different environments.

Finally, while the voice-based interaction pipeline enables natural communication, conversational interaction is currently limited to relatively short task-oriented dialogues. More advanced dialogue management and long-term contextual memory could further improve the naturalness and usefulness of the interaction.

7.3 Future Work

Several directions for future research can extend and improve the capabilities of the proposed system.

A first important direction is the development of **real-time semantic scene graph generation**. Instead of building the scene graph only after an exploration phase, future systems could incrementally update the graph as the robot navigates through the environment. This would allow the robot to continuously refine its semantic understanding and respond to dynamic changes such as moved objects or newly introduced items.

Another promising direction involves **active perception and adaptive exploration**. Rather than passively collecting visual data, the robot could actively choose viewpoints that maximize information gain for semantic perception. Such strategies could improve object detection accuracy, reduce uncertainty in scene graph construction, and enable more efficient environment understanding.

Future work could also explore **edge deployment of vision–language models**. Advances in lightweight multimodal models and efficient inference techniques may allow more components of the system to run directly on the robot. This would reduce latency, improve privacy, and increase robustness in environments with limited connectivity.

An additional research direction concerns **improved human–robot interaction**. More advanced dialogue management systems could allow the robot to maintain longer conversations, remember previous interactions, and adapt responses based on user preferences. Multimodal feedback strategies, combining auditory cues with haptic or spatial guidance, could further enhance accessibility for visually impaired users.

Finally, future research could investigate **additional assistive tasks** that build on the existing semantic perception framework. Examples include medication identification, household object retrieval, daily activity reminders, or collaborative object search. Expanding the set of tasks would further demonstrate the potential of semantic robotic assistants in real domestic settings.

7.4 Final Remarks

Assistive robotics has the potential to significantly improve independence and quality of life for visually impaired individuals. By integrating perception, semantic reasoning, and natural language interaction, robots can act not only as physical assistants but also as interpreters of the surrounding environment.

The work presented in this thesis demonstrates an important step toward such systems by combining modern vision, language, and navigation technologies into a coherent assistive robotic platform. While significant challenges remain, continued progress in semantic perception, multimodal reasoning, and human–robot interaction will bring assistive robotic systems closer to practical deployment in everyday homes.

Bibliography

[1] Shuijing Liu, Aamir Hasan, Kaiwen Hong, Chun-Kai Yao, Justin Lin, Weihang Liang, Megan A. Bayles, Wendy A. Rogers, and Katherine Driggs-Campbell. *Designing a Wayfinding Robot for People with Visual Impairments*. 2023. arXiv: 2302.09144 [cs.RO]. URL: https://arxiv.org/abs/2302.09144 (cit. on p. 2).

[2] Ike Obi, Ruiqi Wang, Prakash Shukla, and Byung-Cheol Min. *Robot Patrol: Using Crowdsourcing and Robotic Systems to Provide Indoor Navigation Guidance to The Visually Impaired*. 2023. arXiv: 2306.02843 [cs.RO]. URL: https://arxiv.org/abs/2306.02843 (cit. on p. 2).

[3] Gabriela AA de Oliveira, Otavio de Faria Oliveira, Stenio de Abreu, Raphael W de Bettio, and André P Freire. «Opportunities and accessibility challenges for open-source general-purpose home automation mobile applications for visually disabled users». In: *Multimedia Tools and Applications* 81.8 (2022), pp. 10695–10722 (cit. on pp. 3, 8, 19).

[4] Sriram Yenamandra et al. *HomeRobot: Open-Vocabulary Mobile Manipulation*. 2024. arXiv: 2306.11565 [cs.RO]. URL: https://arxiv.org/abs/2306.11565 (cit. on pp. 4, 8).

[5] Alessandro Diogo Vieira, Higor Leite, and Ana Vitória Lachowski Volochtchuk. «The impact of voice assistant home devices on people with disabilities: A longitudinal study». In: *Technological Forecasting and Social Change* 184 (2022), p. 121961. ISSN: 0040-1625. DOI: https://doi.org/10.1016/j.techfore.2022.121961. URL: https://www.sciencedirect.com/science/article/pii/S0040162522004826 (cit. on pp. 4, 8).

[6] Chen-Lung Lu et al. «Assistive Navigation Using Deep Reinforcement Learning Guiding Robot With UWB/Voice Beacons and Semantic Feedbacks for Blind and Visually Impaired People». In: *Frontiers in Robotics and AI* Volume 8 - 2021 (2021). ISSN: 2296-9144. DOI: 10.3389/frobt.2021.654132. URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2021.654132 (cit. on p. 4).

[7] Praveena B, C. Vimala Josphine, Nithya Jenev J, Anitha G, Chairma Lakshmi K R, and Vaishnavi G. «Robotic Accident Prevention and Alert System for Visually Impaired». In: *2023 International Conference on Disruptive Technologies (ICDT)*. 2023, pp. 672–675. DOI: 10.1109/ICDT57929.2023.10150923 (cit. on p. 7).

[8] S Abhishek, Harsha Sathish, Arvind Kumar K, and Anjali T. «Aiding the Visually Impaired using Artificial Intelligence and Speech Recognition Technology». In: *2022 4th International Conference on Inventive Research in Computing Applications (ICIRCA)*. 2022, pp. 1356–1362. DOI: 10.1109/ICIRCA54612.2022.9985659 (cit. on p. 7).

[9] Matteo Luperto et al. «Integrating Social Assistive Robots, IoT, Virtual Communities and Smart Objects to Assist at-Home Independently Living Elders: The MoveCare Project». In: *International Journal of Social Robotics* 15.3 (2023), pp. 517–545. DOI: 10.1007/s12369-021-00843-0 (cit. on p. 7).

[10] Yi Zhao, Yilin Zhang, Rong Xiang, Jing Li, and Hillming Li. *VIALM: A Survey and Benchmark of Visually Impaired Assistance with Large Models*. 2024. arXiv: 2402.01735 [cs.CL]. URL: https://arxiv.org/abs/2402.01735 (cit. on p. 8).

[11] Tianfei Zhou, Fatih Porikli, David J. Crandall, Luc Van Gool, and Wenguan Wang. «A Survey on Deep Learning Technique for Video Segmentation». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.6 (2023), pp. 7099–7122. DOI: 10.1109/TPAMI.2022.3225573 (cit. on p. 9).

[12] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. «Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks». In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper_files/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf (cit. on p. 11).

[13] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. «You Only Look Once: Unified, Real-Time Object Detection». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016 (cit. on p. 11).

[14] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. *Open-vocabulary Object Detection via Vision and Language Knowledge Distillation*. 2022. arXiv: 2104.13921 [cs.CV]. URL: https://arxiv.org/abs/2104.13921 (cit. on p. 11).

[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. «Fully Convolutional Networks for Semantic Segmentation». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015 (cit. on p. 12).

[16] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. «Mask R-CNN». In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (cit. on p. 12).

[17] Alexander Kirillov et al. «Segment Anything». In: *arXiv preprint arXiv:2304.02643* (2023) (cit. on p. 13).

[18] Meta AI Research. *Segment Anything 2 (SAM2)*. `https://ai.meta.com/research/sam2/`. 2023 (cit. on p. 13).

[19] Ultralytics. *Segment Anything Model (SAM) Documentation*. `https://docs.ultralytics.com/models/sam/`. 2024 (cit. on p. 14).

[20] Rongjie Li, Songyang Zhang, Dahua Lin, Kai Chen, and Xuming He. «From Pixels to Graphs: Open-Vocabulary Scene Graph Generation with Vision-Language Models». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2024, pp. 28076–28086 (cit. on pp. 14, 16).

[21] Krishna Murthy Jatavallabhula et al. *ConceptFusion: Open-set Multimodal 3D Mapping*. 2023. arXiv: `2302.07241 [cs.CV]`. URL: `https://arxiv.org/abs/2302.07241` (cit. on p. 14).

[22] Songyou Peng, Kyle Genova, Chiyu "Max" Jiang, Andrea Tagliasacchi, Marc Pollefeys, and Thomas Funkhouser. «OpenScene: 3D Scene Understanding With Open Vocabularies». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 815–824 (cit. on p. 14).

[23] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R. Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. «3D Scene Graph: A Structure for Unified Semantics, 3D Space, and Camera». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019 (cit. on p. 15).

[24] Ue-Hwan Kim, Jin-Man Park, Taek-jin Song, and Jong-Hwan Kim. «3-D Scene Graph: A Sparse and Semantic Representation of Physical Environments for Intelligent Agents». In: *IEEE Transactions on Cybernetics* 50.12 (2020), pp. 4921–4933. DOI: `10.1109/TCYB.2019.2931042` (cit. on p. 15).

[25] Antoni Rosinol, Andrew Violette, Marcus Abate, Nathan Hughes, Yun Chang, Jingnan Shi, Arjun Gupta, and Luca Carlone. «Kimera: From SLAM to spatial perception with 3D dynamic scene graphs». In: *The International Journal of Robotics Research* 40.12-14 (2021), pp. 1510–1546. DOI: 10.1177/0278364 9211056674. eprint: https://doi.org/10.1177/02783649211056674. URL: https://doi.org/10.1177/02783649211056674 (cit. on p. 16).

[26] Nathan Hughes, Yun Chang, and Luca Carlone. *Hydra: A Real-time Spatial Perception System for 3D Scene Graph Construction and Optimization*. 2022. arXiv: 2201.13360 [cs.RO]. URL: https://arxiv.org/abs/2201.13360 (cit. on p. 16).

[27] Qiao Gu et al. *ConceptGraphs: Open-Vocabulary 3D Scene Graphs for Perception and Planning*. 2023. arXiv: 2309.16650 [cs.RO]. URL: https://arxiv.org/abs/2309.16650 (cit. on pp. 16, 52).

[28] Steve Young, Milica Gašić, Blaise Thomson, and Jason D. Williams. «POMDP-Based Statistical Spoken Dialog Systems: A Review». In: *Proceedings of the IEEE* 101.5 (2013), pp. 1160–1179. DOI: 10.1109/JPROC.2012.2225812 (cit. on p. 17).

[29] Stefanie Tellex et al. «Understanding natural language commands for robotic navigation and mobile manipulation». In: *Proceedings of the AAAI Conference on Artificial Intelligence* (2011) (cit. on p. 17).

[30] Rohan Paul and Ronald Arkin. «Efficient Grounding of Abstract Spatial Concepts for Natural Language Interaction with Robots». In: *ACM Transactions on Intelligent Systems and Technology* (2016) (cit. on p. 17).

[31] Nicholas D. Lane, Sourav Bhattacharya, Petko Georgiev, Claudio Forlivesi, and Fahim Kawsar. «An Early Resource Characterization of Deep Learning on Wearables, Smartphones and Internet-of-Things Devices». In: *Proceedings of the 2015 International Workshop on Internet of Things towards Applications*. IoT-App '15. Seoul, South Korea: Association for Computing Machinery, 2015, pp. 7–12. ISBN: 9781450338387. DOI: 10.1145/2820975.2820980. URL: https://doi.org/10.1145/2820975.2820980 (cit. on pp. 18, 21).

[32] Jian Chen et al. «Deep learning for speech recognition on mobile devices». In: *IEEE Signal Processing Magazine* (2019) (cit. on p. 18).

[33] Y. Wang et al. «A survey on keyword spotting systems». In: *IEEE Transactions on Audio, Speech, and Language Processing* (2020) (cit. on p. 18).

[34] Peng Cheng and Utz Roedig. «Personal Voice Assistant Security and Privacy—A Survey». In: *Proceedings of the IEEE* 110.4 (2022), pp. 476–507 (cit. on p. 19).

[35] *MelSpectrogram — Torchaudio Transforms Documentation*. https://docs.pytorch.org/audio/main/generated/torchaudio.transforms.MelSpectrogram.html. Accessed: 2026-03. PyTorch Project, 2024 (cit. on p. 19).

[36] D. Scripka, T. Marrinan, A. Krall, D. Flynn, A. Brakke, M. Johnson, and K. Chinen. «openWakeWord: An Open Source Wake Word Dataset and Models». In: *arXiv preprint* abs/2002.01322 (2020). Open source wake word dataset and CNN+RNN models. URL: https://arxiv.org/abs/2002.01322 (cit. on pp. 19, 20).

[37] *GPT-4o Transcribe Diarize Model | OpenAI API*. https://platform.openai.com/docs/models/gpt-4o-transcribe-diarize. Accessed: 2026-03. OpenAI, 2024 (cit. on p. 21).

[38] *GPT-4o Transcribe Model | OpenAI API*. https://platform.openai.com/docs/models/gpt-4o-transcribe. Accessed: 2026-03. OpenAI, 2024 (cit. on p. 21).

[39] Ke Tan et al. «A survey on deep learning-based voice activity detection». In: *IEEE Access* (2020) (cit. on pp. 22, 23).

[40] Silero Team. *Silero VAD: pre-trained enterprise-grade Voice Activity Detector (VAD), Number Detector and Language Classifier*. https://github.com/snakers4/silero-vad. Version 1.2.0. 2024 (cit. on p. 23).

[41] Gokhan Tur and Dilek Hakkani-Tur. «Towards spoken language understanding systems». In: *MIT Press* (2011) (cit. on p. 24).

[42] Dipendra Misra et al. «Mapping instructions and visual observations to actions with reinforcement learning». In: *Robotics: Science and Systems* (2018) (cit. on p. 24).

[43] Stefanie Tellex et al. «A survey of approaches to grounding natural language in robotics». In: *Foundations and Trends in Robotics* (2020) (cit. on p. 25).

[44] OpenAI. «GPT-4o: Multimodal Large Language Model». In: *arXiv preprint arXiv:2303.08774* (2023). https://arxiv.org/abs/2303.08774 (cit. on p. 25).

[45] *GPT-4o mini TTS Model | OpenAI API*. https://developers.openai.com/api/docs/models/gpt-4o-mini-tts. OpenAI, 2025 (cit. on p. 27).

[46] Holly Yanco and Jill Drury. «A Taxonomy for Human-Robot Interaction». In: *Proceedings of the AAAI Fall Symposium on Human-Robot Interaction* (2002) (cit. on p. 27).

[47] Steve Macenski, Francisco Martín, Ruffin White, and Jonatan Ginés Clavero. «The Marathon 2: A Navigation System». In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020. DOI: 10.1109/IROS45743.2020.9341207 (cit. on pp. 29, 33, 34).

[48] *Nav2 Documentation: Getting Started*. Open Navigation LLC / Nav2 Community. URL: https://docs.nav2.org/getting_started/index.html (visited on 01/25/2026) (cit. on pp. 29, 33, 34).

[49] Hamid Taheri and Zhao Chun Xia. «SLAM; definition and evolution». In: *Engineering Applications of Artificial Intelligence* 97 (2021), p. 104032. DOI: 10.1016/j.engappai.2020.104032 (cit. on pp. 30, 32).

[50] Steve Macenski and Ivona Jambrecic. «SLAM Toolbox: SLAM for the dynamic world». In: *Journal of Open Source Software* 6.61 (2021), p. 2783. DOI: 10.21105/joss.02783 (cit. on pp. 31, 35).

[51] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. «The dynamic window approach to collision avoidance». In: *IEEE Robotics & Automation Magazine* 4.1 (1997), pp. 23–33 (cit. on p. 33).

[52] Jur van den Berg, Michiel Snoek, and Mark Overmars. «Dynamic window approach to collision avoidance». In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2008, pp. 1501–1507 (cit. on p. 33).

[53] Julio A. Placed, Jared Strader, Henry Carrillo, Nikolay Atanasov, Vadim Indelman, Luca Carlone, and José A. Castellanos. «A Survey on Active Simultaneous Localization and Mapping: State of the Art and New Frontiers». In: *IEEE Transactions on Robotics* 39.3 (2023), pp. 1686–1705. DOI: 10.1109/TRO.2023.3248510 (cit. on p. 35).