Introducing ATHENA: Autonomous Task Handling and Execution Agent by AdaRobotics@Home team

1st Leonardo C. Neves
Ada Robotics
São Paulo, Brazil
neves.leonardoc@gmail.com

***Abstract*—This paper introduces ATHENA, an Autonomous Task Handling and Execution Agent developed by the AdaRobotics@Home team for the RoboCup@Home competition. The RoboCup@Home competition poses a significant challenge with its requirement for robots to perform a diverse range of complex tasks within domestic environments. The Athena robot represents a notable advancement in this domain through its innovative use of Large Language Models (LLMs) for high-level task management. Athena employs a sophisticated agent-based architecture within a Plan and React framework, comprising distinct roles for planning and execution. The Planning Agent utilizes advanced LLMs to interpret complex directives and generate detailed action plans, while the Execution Agent implements these plans by interfacing with the ROS 2 system. Athena integrates state-of-the-art LLMs such as Gemini 1.5 Flash for real-time processing and Phi3 Small for offline contexts, ensuring robust performance in both online and constrained environments. This integration is facilitated by Langchain and LangGraph libraries, which bridge high-level language models with low-level robotic control systems, providing structured command interpretation and real-time visualization of interactions. The implementation of this framework offers enhanced task management, real-time adaptability, offline functionality, and efficient integration. By leveraging cutting-edge language models and robust integration tools, Athena is equipped to meet the complex demands of RoboCup@Home with increased flexibility and effectiveness.**

***Index Terms*—Service Robot, LLM Agents, Ada Robotics.**

I. INTRODUCTION

The RoboCup@Home competition, established in 2006, is a premier event dedicated to advancing the field of domestic robotics. It challenges teams to develop autonomous robots capable of performing a wide range of household tasks, emphasizing advancements in human-robot interaction, computer vision, behavior analysis, and artificial intelligence. This competition, serves as a platform for showcasing innovative solutions that address real-world domestic challenges [1].

In the context of the 2024 RoboCup@Home competition, AdaRobotics has adopted a novel approach by focusing on the integration of advanced architectural frameworks within their robotic systems. This paper delves into the architectural strategies employed by AdaRobotics, elucidating the design and integration of key components crucial for addressing the multifaceted challenges posed by the competition. The emphasis is on illustrating how these architectural components contribute to the robot's capability to operate effectively within dynamic and unpredictable home environments.

Identify applicable funding agency here. If none, delete this.

The RoboCup@Home competition encompasses a range of challenges, such as object manipulation, activity management, and responsive interactions with users. To tackle these challenges, AdaRobotics has developed a comprehensive architectural framework that integrates state-of-the-art technologies for sensor fusion, task planning, and control. This framework is designed to support sophisticated data processing and decision-making, enabling the robot to perform tasks with high precision and adaptability. The system architecture includes modules for object recognition, environment mapping, and user interaction, each playing a crucial role in enhancing the robot's overall functionality [2].

AdaRobotics' architectural strategies aim to advance the field of domestic robotics by addressing key challenges through innovative design and integration. The approach seeks to establish new standards for task automation and interaction within domestic settings, setting a benchmark for future developments in the field.

This paper provides an in-depth examination of the architectural approaches employed by AdaRobotics in the RoboCup@Home competition. It focuses on the design and integration of key components, highlighting their relevance and potential impact. The discussion will center on the architectural framework and its alignment with the competition's objectives, without presenting specific results or detailed applications.

II. DEVELOPMENT ENVIRONMENT ROS2

The Robotic Operating System (ROS) is a comprehensive framework designed to facilitate the development of robotic software. It provides a suite of tools and libraries that support a wide range of functionalities, from basic hardware manipulation to complex behavior control. ROS2 represents a significant evolution from its predecessor, ROS1, introducing several enhancements that streamline the development process for robotic systems [3].

A. Architecture of ROS2 Humble

ROS2 builds upon the foundational concepts of ROS1 while incorporating substantial improvements in its architecture. Key features of ROS2 include:

- DDS-based Middleware: Unlike ROS1, which relies on custom middleware for inter-node communication, ROS2 employs the Data Distribution Service (DDS) standard.

DDS provides robust, real-time communication capabilities, facilitating seamless data exchange between distributed components and enhancing the scalability and reliability of robotic systems.
- Real-Time Capabilities: ROS2 introduces real-time support, which is crucial for applications requiring deterministic behavior. This feature allows for the development of robots that can perform time-sensitive tasks with high precision and predictability.
- Improved Security: ROS2 integrates advanced security features, including encryption, authentication, and access control, to protect data integrity and prevent unauthorized access. This is particularly important in scenarios involving sensitive or critical operations.
- Enhanced Flexibility and Modularity: ROS2 emphasizes modularity and component-based design, enabling developers to create reusable and interchangeable software components. This approach promotes better code organization and facilitates the integration of diverse functionalities.

B. *Differences Between ROS1 and ROS2*

The transition from ROS1 to ROS2 involves several key differences:

- Middleware: ROS1 uses a custom communication protocol, whereas ROS2 adopts DDS, offering more flexibility and better support for real-time and scalable applications [4].
- Real-Time Support: ROS1 lacks built-in real-time capabilities, which are introduced in ROS2, enabling more precise and predictable control in time-sensitive tasks.
- Security: ROS2 provides enhanced security features compared to ROS1, addressing critical concerns related to data protection and system integrity.
- Client Libraries: ROS2 supports multiple client libraries, including C++, Python, and Rust, whereas ROS1 primarily supports C++ and Python. This broadens the accessibility and usability of ROS2 across different programming environments.

C. *Advantages for AdaRobotics and Other Teams*

ROS2 offers several advantages that benefit both AdaRobotics and other teams participating in the RoboCup@Home competition:

Scalability and Flexibility: The DDS-based middleware and modular architecture enable teams to develop scalable and adaptable robotic systems. This flexibility is crucial for addressing the diverse challenges presented in domestic robotics tasks.

- Enhanced Development Tools: ROS2 provides a range of modern development tools and libraries, facilitating the integration of advanced functionalities such as Large Language Models (LLMs) and real-time data processing. This support accelerates the development cycle and enhances the overall performance of robotic systems.
- Community and Ecosystem: The widespread adoption of ROS2 by the robotics community ensures a rich ecosystem of libraries, tools, and resources. This collaborative environment fosters innovation and allows teams to leverage existing solutions and contribute to ongoing research.

In conclusion, ROS2 represents a significant advancement in robotic software frameworks, offering robust features that support the development of sophisticated and reliable robotic systems. Its architectural improvements and enhanced capabilities make it a valuable asset for teams like AdaRobotics and others engaged in the RoboCup@Home competition.

III. Robot Vision

Effective robotic vision systems are critical for the performance of autonomous robots, especially in dynamic and diverse domestic environments. In the AdaRobotics @Home project, we employ several state-of-the-art computer vision algorithms to achieve comprehensive object recognition, human pose estimation, face recognition, image segmentation, and environmental reasoning.

A. *YOLO v8 for Object Recognition*

YOLO v8 (You Only Look Once version 8) is an advanced object detection framework designed for real-time processing. The architecture builds upon previous YOLO versions with several key improvements:

- Single-Stage Detection: YOLO v8 operates as a single-stage detector, meaning it performs object detection in one pass through the network. This design minimizes computational overhead and maximizes processing speed.
- Enhanced Backbone Network: YOLO v8 employs an upgraded backbone network for feature extraction, enhancing the model's ability to capture fine-grained details and complex object features. Anchor Boxes and Prediction Head: The model utilizes refined anchor boxes and a sophisticated prediction head to improve accuracy in object localization and classification across various scales [5].

B. *YOLO Pose for Human Markers*

YOLO Pose extends the YOLO architecture to address human pose estimation, which involves detecting and tracking human body keypoints:

- Pose Estimation Network: YOLO Pose incorporates a dedicated pose estimation network that predicts the locations of key body joints. This network operates in parallel with the object detection network to provide real-time pose information.
- Joint Localization: The model outputs heatmaps for each keypoint, allowing precise localization of human joints and enabling accurate tracking of body movements and gestures [6][7].

C. VGGFace2 for Face Recognition

VGGFace2 is a deep convolutional network designed for facial recognition, known for its high performance in generating facial embeddings:

- Deep Convolutional Architecture: VGGFace2 employs a deep convolutional neural network (CNN) architecture with multiple layers to capture hierarchical features from facial images.
- Embedding Generation: The model produces high-dimensional embeddings for each face, which are used to measure similarity and perform recognition tasks. These embeddings are robust to variations in lighting, expression, and pose [8].

D. SAM2 for Image Segmentation

SAM2 (Segment Anything Model version 2) is a cutting-edge image segmentation framework that provides detailed object and region delineation:

- Contextual Analysis: SAM2 integrates contextual reasoning to enhance segmentation accuracy, enabling the model to understand and segment complex scenes with overlapping or closely spaced objects [9][10].
- Semantic Segmentation: SAM2 utilizes advanced convolutional layers and segmentation algorithms to segment objects and regions within an image. The model distinguishes between different elements, providing a pixel-wise segmentation map.

E. Gemini 1.5 Flash for Image Description and Environmental Reasoning

Gemini 1.5 Flash is a sophisticated model for generating natural language descriptions and performing contextual reasoning about images:

- Image Captioning: Gemini 1.5 Flash generates descriptive text for visual content by combining visual and textual information. The model uses a combination of convolutional neural networks (CNNs) and transformers to produce detailed and contextually relevant descriptions.
- Contextual Understanding: The model incorporates advanced reasoning capabilities to infer relationships between objects and predict interactions, providing a comprehensive understanding of the environment [11][12].

IV. Voice Recognition and Synthesis

Voice interaction is a pivotal component in enabling effective communication between humans and robots. In the AdaRobotics @Home project, we utilize advanced technologies for both voice recognition and synthesis to ensure accurate and natural interactions.

A. Voice Recognition

For voice recognition, we deploy two primary systems to cater to different needs:

- Google Speech-to-Text API
  The Google Speech-to-Text API provides a powerful online solution for automatic speech recognition (ASR). This API supports a wide range of languages and is highly accurate due to its extensive training on diverse speech data [13]. Key features include:
  - Multi-Language Support: The API supports numerous languages, making it versatile for global applications.
  - Real-Time Processing: It processes speech in real time and provides text transcriptions with high accuracy.
  - Online Dependency: The API requires an internet connection, as it relies on Google's cloud infrastructure for speech processing.

  In the AdaRobotics @Home project, we use this API through a ROS package integrated with the SOX protocol to handle audio input and return transcriptions and confidence scores for each recognized phrase.
- Fast Whisper
  To ensure offline capability, especially in scenarios where an internet connection may be unreliable, we incorporate Fast Whisper for local ASR. Fast Whisper is a lightweight model designed for efficient offline speech recognition [14]. Its features include:
  - Offline Operation: Fast Whisper operates locally on the robot, eliminating the need for continuous internet access.
  - Adaptability: The model can be fine-tuned for specific vocabulary or accents, improving recognition accuracy in varied environments.

  This combination of online and offline ASR solutions provides robustness and flexibility in different operational contexts.

B. Voice Synthesis

For voice synthesis, we utilize the Python library Piper, which allows for the fine-tuning of pre-trained text-to-speech (TTS) models to generate natural and expressive speech outputs [15]. The architecture of Piper includes:

Model Architecture: Piper uses a deep learning model based on Tacotron 2 and WaveGlow. Tacotron 2 is a sequence-to-sequence model that converts text into mel-spectrograms, which are then synthesized into audio using WaveGlow, a flow-based generative network [16].

Fine-Tuning: We fine-tuned a pre-trained voice model using a custom dataset with a female voice. This customization allows the robot to produce speech outputs with a specific tone and style, enhancing user interaction.

Piper's architecture ensures high-quality, natural-sounding speech synthesis, making the robot's responses more engaging and human-like.

V. ROBOT NAVIGATION

Autonomous navigation is a fundamental capability for robots operating in dynamic and unstructured environments. For a robot to navigate independently, it must effectively map its surroundings, determine its position within the environment, and plan optimal routes while avoiding obstacles. This involves a combination of sensor data acquisition, real-time processing, and path planning algorithms.

A. Navigation Stack: Nav2

In the AdaRobotics @Home project, we utilize the Navigation2 (Nav2) stack, which is a ROS 2 package designed to provide advanced capabilities for autonomous robot navigation. Nav2 builds upon the original ROS navigation stack with several enhancements:

- Modular Architecture: Nav2 features a modular design that includes components for localization, path planning, and obstacle avoidance. This modularity allows for flexible configuration and integration with various sensor setups.
- Lifecycle Management: The stack incorporates lifecycle management for nodes, improving robustness and reliability in dynamic environments. This feature ensures that navigation processes can be started, stopped, and reconfigured as needed.
- Improved Algorithms: Nav2 supports advanced algorithms for global and local path planning, including Dijkstra's algorithm and the Timed Elastic Band (TEB) planner, which enhance the robot's ability to navigate complex environments with dynamic obstacles [17].

B. Sensor: RPLIDAR A1

To facilitate navigation, the AdaRobotics @Home robot employs the RPLIDAR A1, a 360-degree laser rangefinder known for its precision and reliability in generating 2D maps:

- Laser Scanning: The RPLIDAR A1 uses a laser to scan its surroundings, providing a 360-degree view with a range of up to 12 meters and a resolution of 0.45 degrees. This high-resolution scanning capability enables the robot to detect obstacles with great accuracy.
- Data Acquisition: The laser sensor collects distance measurements and converts them into a point cloud that is used to create detailed environmental maps. This data is essential for real-time SLAM (Simultaneous Localization and Mapping) operations.
- Obstacle Detection: The RPLIDAR A1 helps identify and avoid obstacles by providing a continuous stream of distance data, allowing the robot to dynamically adjust its path and navigate around obstacles.

VI. MANIPULATORS

In the AdaRobotics @Home project, we developed a custom manipulator designed to perform a range of tasks with precision and efficiency. The manipulator, Figure 1, features three degrees of freedom (DOF) plus base rotation, and it is constructed from a combination of 3D-printed components and aluminum profiles. This design choice balances cost, performance, and ease of construction, making it accessible for other teams to replicate.

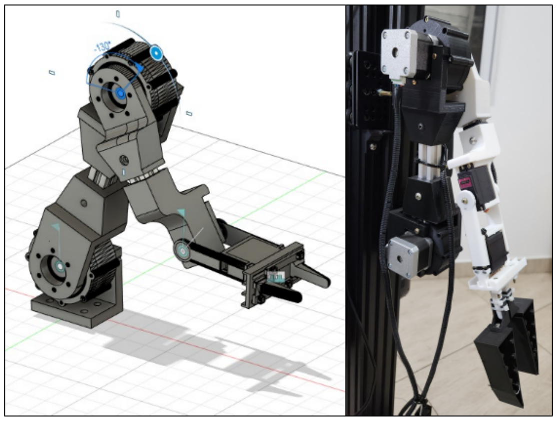

Fig. 1. Manipulator developed for the Athena robot.

A. Mechanical Design

The manipulator consists of:

- Degrees of Freedom: The manipulator incorporates three rotational joints, allowing it to perform a variety of movements similar to those of a human arm. Additionally, the base rotation provides further maneuverability.
- Motors and Actuators: The manipulator is powered by two NEMA 17 stepper motors with planetary gear reductions, which provide precise control over the movement of the arm. Two MG996 servos are used for additional actuation. The stepper motors are selected for their high torque and accuracy, while the servos offer responsive and reliable motion control.
- Construction Materials: All structural components are fabricated using a combination of 3D-printed parts and aluminum profiles. The use of 3D printing allows for rapid prototyping and customization, while aluminum profiles ensure durability and strength. The overall design allows for a payload capacity of 0.4 kg, making it suitable for handling lightweight objects.

B. Kinematics and Control

- Kinematic Analysis: The kinematic model of the manipulator is designed to replicate the functionality of a human arm. This involves both forward and inverse kinematic calculations to determine the position and orientation of the manipulator's end effector. The forward kinematics compute the end effector's position based on the joint angles, while the inverse kinematics determine the required joint angles for a desired end effector position. For inverse kinematics, we utilized the Python library ikpy, which provides efficient algorithms for solving inverse kinematic problems in robotic arms.
- Control System: The manipulator is controlled using a CNC shield in conjunction with an Arduino microcontroller. The CNC shield provides precise control over the stepper motors, while the Arduino handles the coordination of motor movements and integrates feedback from the actuators. The control algorithms are designed

to ensure smooth and accurate movements, allowing the manipulator to perform complex tasks with high precision.

- Modular Design: The manipulator's modular design allows for easy assembly and modification. The 3D-printed components and aluminum profiles are designed to be easily interchangeable, enabling quick adjustments and upgrades. This modularity also reduces manufacturing costs and simplifies maintenance.

C. Practical Considerations

The manipulator's design emphasizes cost efficiency and ease of construction. By utilizing 3D printing and aluminum profiles, it proves to be more cost-effective compared to commercially available alternatives. This approach not only reduces expenses but also makes the manipulator accessible to other teams and research groups, thereby fostering collaboration and innovation within the robotics community. Additionally, the design is optimized for ease of assembly. The use of widely available materials and straightforward construction techniques ensures that other teams can replicate or adapt the design with minimal effort.

VII. High-Level Task Management with LLM Agents

In the RoboCup@Home competition, where robots must perform a wide array of complex tasks within domestic environments, the integration of Large Language Models (LLMs) into robotic systems represents a significant advancement. The Athena robot employs a sophisticated agent-based architecture utilizing LLMs to manage high-level tasks. This section delves into the detailed architecture, functionalities, and implementation of the Athena robot's LLM-based agent system.

A. Agent Architecture and Framework

The Athena robot employs a dual-agent system organized into a Plan and React framework. This approach delineates the roles of planning and execution, enhancing both efficiency and adaptability in task management.

- Plan and React Agent Framework:
  The Plan and React architecture is designed to segregate high-level strategic planning from the low-level task execution. This hierarchical setup comprises two primary agents:
  - Planning Agent: This agent is responsible for high-level task planning and strategy formulation. It utilizes the LLM's advanced language capabilities to interpret complex commands and generate detailed action plans. The planning agent's role includes analyzing the requirements of a given task, constructing a sequence of actions, and preparing the necessary instructions for execution.
  - Execution Agent: The execution agent operates at a lower level, tasked with implementing the plans devised by the planning agent. This agent interfaces directly with the ROS 2 system to control the robot's actuators, sensors, and other hardware components. It translates high-level plans into specific commands that drive the robot's actions, ensuring that the task is performed as intended.
- LLM Models:
  - Gemini 1.5 Flash: For online scenarios, Athena utilizes the Gemini 1.5 Flash model. This LLM is known for its state-of-the-art language comprehension and generation capabilities. It provides real-time processing, enabling the robot to handle dynamic, unpredictable environments. The Gemini 1.5 Flash model enhances Athena's ability to interpret and act upon complex, context-dependent commands, facilitating more sophisticated interactions and problem-solving.
  - Phi3 Small: In offline contexts, where internet connectivity might be limited, Athena relies on the Phi3 Small model. While smaller and less resource-intensive than Gemini 1.5 Flash, Phi3 Small retains robust language processing abilities suitable for local task execution. It ensures that the robot remains operational and effective even without access to online resources, allowing continuous performance in constrained environments.

B. Integration with ROS 2

The integration of LLMs with ROS 2 is facilitated through Langchain and LangGraph libraries, which bridge the gap between high-level language models and low-level robotic control systems.

- Langchain: Langchain serves as the foundational framework for linking LLMs with ROS 2 services. It provides a structured interface for translating natural language commands into executable actions within the ROS 2 ecosystem. Langchain supports the creation of robust pipelines that handle command interpretation, action planning, and communication with ROS 2 nodes. By utilizing Langchain, the Athena robot can seamlessly execute high-level tasks and interact with various ROS 2 services, enhancing its overall functionality.
- LangGraph: LangGraph builds upon Langchain by offering a graphical representation of the interactions between the LLMs and ROS 2 services. It provides a visual tool for monitoring and analyzing the communication flows, state transitions, and task executions. LangGraph's visualization capabilities enable real-time tracking of the agent's performance, allowing for efficient debugging, optimization, and adjustment of the task management processes. This tool is crucial for ensuring that the robot's actions align with the planned strategies and adapting to any unforeseen changes in the environment.

C. Practical Implementation and Benefits

The implementation of the Plan and React framework, coupled with advanced LLMs, offers several key benefits for the Athena robot in the RoboCup@Home competition:

- Enhanced Task Management: The separation of planning and execution allows for a more structured approach to task management. The planning agent can develop comprehensive strategies, while the execution agent ensures precise implementation of these strategies.
- Real-Time Adaptability: The use of Gemini 1.5 Flash enables real-time decision-making and adaptation, allowing Athena to respond effectively to dynamic and unpredictable situations.
- Offline Functionality: The Phi3 Small model provides reliable offline capabilities, ensuring that the robot can continue to perform tasks even in the absence of an internet connection.
- Efficient Integration: Langchain and LangGraph facilitate seamless communication between the LLMs and ROS 2, enhancing the robot's ability to manage complex tasks and interact with its environment efficiently.

In summary, the Athena robot's, Figure 2, deployment of LLM-based agents and advanced integration frameworks represents a significant advancement in high-level task management. By leveraging cutting-edge language models and robust integration tools, Athena is equipped to handle the complex and diverse requirements of the RoboCup@Home competition with increased flexibility and effectiveness.

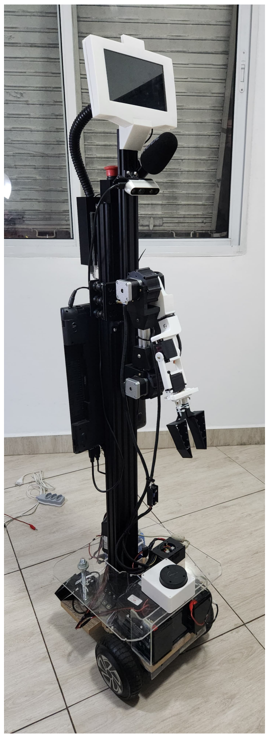

Fig. 2. Athena Robotic Plataform

References

[1] M. Matamoros, V. Seib, R. Memmesheimer, and D. Paulus, "Robocup@ home: Summarizing achievements in over eleven years of competition," in *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pp. 186–191, IEEE, 2018.

[2] M. Matamoros, S. Viktor, and D. Paulus, "Trends, challenges and adopted strategies in robocup@ home," in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pp. 1–6, IEEE, 2019.

[3] A. Bonci, F. Gaudeni, M. C. Giannini, and S. Longhi, "Robot operating system 2 (ros2)-based frameworks for increasing robot autonomy: A survey," *applied sciences*, vol. 13, no. 23, p. 12796, 2023.

[4] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018.

[5] M. Hussain, "Yolo-v1 to yolo-v8, the rise of yolo and its complementary nature toward digital manufacturing and industrial defect detection," *Machines*, vol. 11, no. 7, p. 677, 2023.

[6] D. Maji, S. Nagori, M. Mathew, and D. Poddar, "Yolo-pose: Enhancing yolo for multi person pose estimation using object keypoint similarity loss," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2637–2646, 2022.

[7] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5693–5703, 2019.

[8] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*, pp. 67–74, IEEE, 2018.

[9] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.

[10] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[11] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, and J. Y. et. all., "Gemini: A family of highly capable multimodal models," 2024.

[12] Z. Wang, R. Shen, and B. Stadie, "Solving robotics problems in zero-shot with vision-language models," 2024.

[13] B. Iancu, "Evaluating google speech-to-text api's performance for romanian e-learning resources," *Informatica Economica*, vol. 23, no. 1, pp. 17–25, 2019.

[14] C. Graham and N. Roll, "Evaluating openai's whisper asr: Performance analysis across diverse accents and speaker traits," *JASA Express Letters*, vol. 4, no. 2, 2024.

[15] "Piper." https://rhasspy.github.io/piper-samples/. Acessado em 23 de agosto de 2023.

[16] I. Elias, H. Zen, J. Shen, Y. Zhang, Y. Jia, R. Skerry-Ryan, and Y. Wu, "Parallel tacotron 2: A non-autoregressive neural tts model with differentiable duration modeling," *arXiv preprint arXiv:2103.14574*, 2021.

[17] S. Macenski, F. Martin, R. White, and J. Ginés Clavero, "The marathon 2: A navigation system," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.