Enabling Immersive Indoor Navigation and Control Through Augmented Reality With Computer Vision

M. Pasan Sanjaya Fernando
Department of Electrical Engineering
University of Moratuwa
Sri Lanka
fernandomps.19@uom.lk

A. G. Tharindu Gimras
Department of Electrical Engineering
University of Moratuwa
Sri Lanka
gimrasagt.19@uom.lk

J. M. Budshan P. Jayasundara
Department of Electrical Engineering
University of Moratuwa
Sri Lanka
jayasundarajmbp.19@uom.lk

D. P. Chandima
Department of Electrical Engineering
University of Moratuwa
Sri Lanka
chandimadp@uom.lk

***Abstract*—This project integrates computer vision, Augmented Reality (AR), and cloud-based communication to create a real-time 3D map of the robot's surroundings. The outcome is an engaging and immersive control experience for users in indoor robot monitoring and controlling. Furthermore, combining an image semantic segmentation model with the Microsoft Kinect V2 depth camera introduces a novel method for achieving 1cm accuracy in estimating the distance to a particular object. This precise depth estimation is a significant benefit for tasks that require accurate object identification, segmentation, and depth information to generate a virtual representation of the real-world environment of a robot. We have enabled remote communication with the robot's environment by incorporating AWS cloud services, guaranteeing minimal delay. The commercial viability of this project can be identified in a wide range of applications that find the system useful due to its easy-to-use interface, remote access capability, and precise object rendering capabilities. Examples of such applications include surgical robotics, instructional robotics platforms in classrooms, and automated guided vehicles in warehouses.**

***Keywords*—augmented reality, computer vision, robot monitoring system, cloud integration, 3d object rendering**

I. Introduction

Augmented Reality(AR) is a technology that combines the real-world environment with virtual worlds by overlapping digital graphics on the observing environment. Combining AR with robotics gives the user more insight into the robot's environment. Frameworks like Unity AR Foundation offer developers easy access to AR technology. The Unity AR foundation supports core features from several AR development platforms, such as ARKit (iOS), ARCore (Android), MetaQuest, HoloLens, Apple Vision Pro, and Magic Leap.

This paper offers an AR robot navigation and real-time object rendering system (ARoBotX) that provides an immersive user experience. The main goal of this paper is to dynamically render a user-interactive AR map of the robot's indoor navigation environment. The overall system (ARoBotX) we implemented can be remotely controlled by authorized users. Users are authorized by the AWS platform by defining policies for each service. That allows the user to easily access the robot's navigation environment from anywhere in the world. This paper suggests using this system in warehouses, indoor classrooms, and scientific labs will help scientists and store-keepers, teachers, students, etc.

Previous studies on AR-based robot navigation systems mainly use on-robot or on-body marker-based AR techniques [1]. In addition, previous studies on robot navigation have mainly relied on SLAM technology for map rendering without utilizing computer vision to understand the real-time environment as the robot moves [2]. As a novelty, this paper combines AR and computer vision to create a virtual environment of the robot's physical environment. To render the objects in the AR map, the AR application should be aware of the object's exact 3D coordinates concerning the robot. Most prior studies used LiDAR-based depth estimation techniques [3], ultimately relying on point cloud data optimization techniques, which are often limited by sparsity and struggle to accurately reconstruct fine details. Our research addresses these limitations by introducing a computer vision-based approach that utilizes semantic segmentation with color frames and depth frames obtained using a Microsoft Kinect V2 camera to achieve highly accurate and dense depth estimation, even for complex scenes. This is suitable for estimating the depth of spatially distributed objects which is limited by the existing bounding box middle-point distant estimation technique for depth estimation. To increase the depth estimation accuracy, here uses an image semantic segmentation model [4] with a depth camera for the depth estimation task. From that, they are achieving a higher accuracy than other depth estimation techniques became possible. Building a user-friendly AR application to augment the robot's environment and enable instant command execution is expected from this overall approach.

Here is a guide for this paper. Section II discusses previous work on AR-based robot navigation interfaces and robot map

IEEE Xplore: https://doi.org/10.1109/MERCon63886.2024.10688929

rendering strategies. Section III explains the methodology of this system. Section IV shows the results obtained, and Section V shows the conclusion of this paper. The Section VI paper wraps up the future work that we expect to be related to this area.

II. Related Work

Many prior types of research are related to robot monitoring and navigation interfaces and robot map rendering strategies. Reviewing them will explain how this paper filled that research gap.

A. Augmented Reality-based robot navigation and monitoring interfaces.

Mark Fiala [5] developed a system to control multiple robots through a marker-based AR interface. Mark Fiala enables users to see the path data and status of the robot with the surrounding working area. In future work, the paper expects to create a markerless computer vision-based AR system for augmentations when no robot is in view.

An indoor navigation robot controlling mobile application created by Austin [6] in 2019. That paper evaluates the effectiveness of using an AR mobile application for robot control. They implemented a JAQL system using the Google ARCore platform on Android and a Pixel 2 phone. That Pixel 2 phone is attached to the robot for autonomous navigation and indoor localization. From that, that paper expects to replace devices used for robot navigation and localization with a mobile device camera with the support of AR. In contrast to these marker-based methods, our approach constructs a 3D representation of the robot's environment without relying on specific markers. By utilizing computer vision techniques, a depth camera, and localized robot coordinates obtained through SLAM algorithms, we achieve a more flexible and robust system for indoor robot control and visualization.

B. Robot's Environment rendering strategies

The proposed AR system for robot navigation, also known as ARBIN (Huang [7], 2020), does not use AR technology for environment rendering itself. Use a smartphone camera to take a live video stream of the robot's workspace. Therefore, a virtual model representing the robot overlays the video feed, which the smartphone's touchscreen displays as a reference. However, this approach does not provide accurate AR map rendering, even though it allows the user to have a clear view of the environment. AR map rendering employs a method of placing markers in the environment. These markers trigger a virtual map that can display the robot's location, target objects, and obstacles when a user views the markers through an AR headset. However, AR map rendering can avoid markers by implementing a virtual map directly in the user's real-world view through the headset while highlighting the robot as well as the actions of the robot within the augmented environment.

C. MiRob robot-based previous research projects

As the discussion of the service-based robot projects concerning MIRob is concerned, several important points can be identified. The paper titled "MIRob: The "An Intelligent Service Robot that Learns from Interactive Discussions while Handling Uncertain Information in User Instructions" by Muthugala [8] presents MIRob, a robot that shows how finite state intention modules can be used to handle user interactions and learn from them. To support communication and task performance in this approach MIRob uses an architecture composed of voice recognition, uncertain information understanding, and a Robot Experience Model (REM). The robot has Pioneer 3DX as the base, Cyton Gamma 300 for the manipulator, and highly developed sensory and processing systems; thus, it is a precedent for service robots. Other related studies have also pointed out the need for sound interaction management systems and knowledge models that can be adjusted accordingly to accommodate the role of service robots in environment that are dynamic and are populated with human beings. All such advancements, in unison, underscore the importance of incorporating efficient interaction paradigms and learning methodologies in the design of intelligent service robots.

III. Methodology

Within the AR application, the user is required to position the virtual robot's navigation floor plane in a real-world setting, such as on a table, using a device equipped with the ARoBotX program. Any AR-supported device, such as an Android or iOS smartphone, a Microsoft HoloLens device, Oculus, etc., can be used to install the program. The program predetermines the robot's navigation floor plan based on its chosen environment. Once the user installs the robot's navigation floor, they can observe the virtual robot's movement on the virtual navigation floor plane, mirroring the real robot's actual movement. Simultaneously, the augmented reality (AR) program displays specific virtual objects on the virtual map, taking into account the robot's real-time surroundings.

For this task, we used the " MiRob " robot, which uses Pioneer 3-DX as the base. The Microsoft KinectV2 camera was at the top of the MiRob for object identification and depth estimation. MiRob has a localizing, navigation, and object avoidance system implemented using SLAM from prior studies.

The ARoBotX system prioritizes four key functionalities.

1) Real-time synchronization between the virtual and actual robot movements
2) Precise object positioning and rendering in the AR interface
3) Reducing the error of virtual robot positioning and rotation with the actual robot
4) Creating a user-friendly AR application for a better user experience

Fig.1 1 shows the system architecture of this ARoBotX system. Here, we mainly focus on the implementation of

Augmented Reality for an easy-to-use interface and computer vision for object position estimation. For a better understanding, let us get an idea of the system architecture, robot workspace, and core main parts.

This ARoBotX system consists of 6 main core parts.

1) MiRob robot with custom server and SLAM navigation system
2) Robot commanding and telemetry system with cloud integration
3) Image segmentation and depth estimation system with cloud integration
4) KinectV2 for rendering the robot's surroundings
5) AWS IoT for connecting the AR application with MiRob
6) AR application for providing an interface for robot navigation and robot's environment

A. Hardware Platform

MiRob is an intelligent service robot that can be operated in domestic environments. MiRob consists of Pioneer3-DX, a two-wheel, two-motor differential drive robot. Pioneer 3-DX has SONAR, one battery, wheel encoders, a microcontroller with ARCOS firmware, and the Pioneer SDK advanced mobile robotics software development package. MiRob is built on the Pioneer 3DX mobile platform with a differential drive system having 500-tick encoders for accurate control of motion. The robot is capable of reaching speeds up to 1. 2 m/s and it can carry loads of up to 17 kg. For navigation and safety, MIRob has front/rear sonar sensors of 8 pieces each with a range of 10cm to 5m, and collision detection bumpers. Also, there is an inbuilt gyroscope that helps in navigation and correcting the movement of the drone. To boost the computer vision component of the MIRob, a Kinect v2 camera is set on a stand at a height of one meter from the ground. Kinect v2 possesses better motion and depth sensing which allows the robot to identify objects in front of it and gives detailed spatial data for better interaction and movements.

B. Workspace

For the robot's testing workspace, a square-shaped floor with 30.48 cm x 30.48 cm tiles is selected. The size of that navigation floor is 365.76 cm x 274.32 cm. We used a simple and tile-filled floor to compare the location of the virtual robot with the actual robot to get the placement error easily. In Fig. 2, we can see the exact image for the robot testing workspace. In the testing workspace, the robot moves once or twice to render the map with objects (tables and boxes).

C. MiRob robot with custom server and SLAM navigation system

MiRob is an intelligent service robot that can be operated in domestic environments. This research project used the MiRob robot's existing capability of path planning and obstacle avoidance.

MiRob mobile robotics software development package includes ARNL, Aria, Mapper3Basic, MobileEyes, MobileSim, and ARCOS3. This implementation used only ARNL from

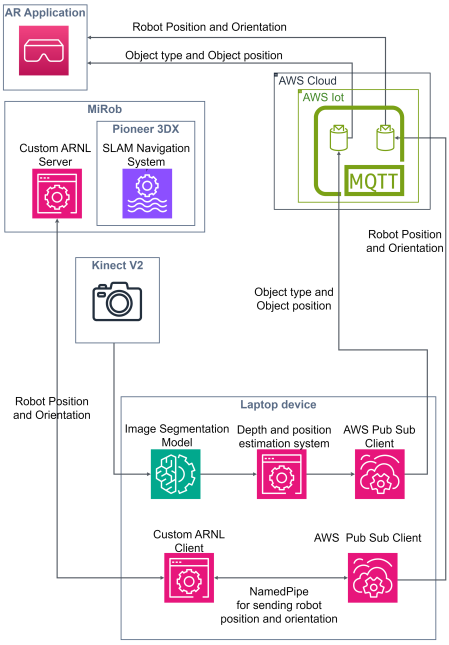

Fig. 1. System Architecture

Fig. 2. Actual image of robot's testing navigation floor

that package to create a custom server on the MiRob robot's computer. ARNL is a software development kit for localizing and intelligently navigating mobile robots. The custom server gives access to the robot's position and rotation info and forwards the robot's info back and forth from a TCP port to a serial port. A portable wifi router is connected to create a bridge network between the MiRob and the laptop device. From the IP address of the wifi router, external devices in the same subnet can connect to MiRob to control the robot and get robot information remotely.

D. Robot Commanding and Telemetry System with Cloud Integration

The ARNL software development kit creates a custom client that connects the laptop device to MiRob's custom ARNL server. To create custom client code, we used C++03 as the

TABLE I
NAMED PIPES APPROACH VS SOCKETS APPROACH

	Named Pipes	Server Approach
Latency	Low	High
Throughput	High	Low
Resources Usage	Low	High

programming language. The custom client sends the wander command to the robot to move freely in its environment. When the MiRob moves to that custom client robot position, it receives orientation information. The NamedPipe inter-process communication (IPC) method connects the custom and AWS pub-sub clients. A Named Pipe is a bi-directional communication method that reads and writes data from a shared pipe. In table I, we can see why we used the named pipes approach instead of sockets here.

As low latency is a crucial factor in this project, we chose NamedPipe because of its high-performance communication. AWS pub-sub client subscribed to a topic on the AWS IoT platform. The AWS pub-sub client, written using NodeJS, updates the robot position and rotation data in a 1-second time interval to the Cloud (AWS IoT).

E. Image segmentation and depth estimation system with Cloud Integration

The robot utilises a Microsoft Kinect V2 camera, positioned at a height of 1.2 meters. This camera, which is equipped with both color and depth cameras, can simultaneously collect visual characteristics of the environment and distance information. A bespoke Python program, utilizing regularly employed libraries for computer vision tasks, captures these image streams concurrently. After capturing the images, we input the color image frames into a pre-trained image semantic segmentation model, specifically designed to detect and isolate objects in the image. This model is trained using the semantic segmentation model parameters of Yolov8.

Fig. 3 demonstrates the trained semantic segmentation model's ability to detect and separate objects accurately. Here, a box-shaped object is well isolated from the background. The pixel coordinates of the mask result in a map that pinpoints the object's exact location within the frame. Here, we store the segmented object's pixel coordinates in a NumPy array for later processing and analysis.

We use the depth frame to extract depth data from each pixel. To enhance the precision of determining the robot's average distance from the chosen target, we utilize the computed predicted depth, which is obtained by considering the average. The resulting value is then published to the cloud by the IoT Message Broker for AWS.

While the depth sensor does deliver significant distance information, it is incapable of furnishing an all-encompassing and definitive viewpoint. Determining the precise three-dimensional position of an object continues to be a challenging endeavor, especially when the object is not within the robot's

Fig. 3. Segmenting a box-shaped object to isolate it from the background

Fig. 4. Designated narrow blue frame for object analysis

visual range. Furthermore, to bridge this distance, it is imperative to compute the object's angle relative to the robot's primary orientation.

To address these constraints, we focus our robot's viewpoint on a smaller, more particular region of the visual frame. If at least 90% of the object's pixel coordinates fall within this stated area, we consider it appropriate for inclusion in the Augmented Reality (AR) map.

While the trained model can identify objects across the entire image frame, we considered only the depth values corresponding to objects within this narrow frame (Fig. 4) for further processing. This approach prioritizes objects closer to the robot's field of view, simplifying the subsequent 3D coordinate calculation process.

F. AWS IoT for connecting the AR application with MiRob

The main bridge between the MiRob and the AR application here is AWS IoT. AWS IoT is a managed cloud platform that connects devices securely with cloud applications. In AWS IoT, there are two main topics.

1) Topic for robot info transmission.
2) Topic for detected object position info transmission.

As the first step, AWS IoT thing set up and then generated the X509 certificates from that. After that, a lightweight MQTT client, uPLibrary, will connect the AR application with the AWS IoT message broker using the AWS endpoint and previously generated certificates.

In the laptop device end, there are two AWS pub sub-clients. Those two AWS pub sub-clients are connected to the AWS IoT message broker. Then, those two clients transmit the data via MQTT messages. By subscribing to both topics, the AR application receives the robot's real-time location info and detects the object's position data. Here, we needed to ensure that this approach gives the user a seamless integration between the robot, mobile application, and AWS IoT Core.

G. AR application

Unity AR foundation is used to develop the AR application. Unity AR foundation supports many device-related AR-supported tool kits like AR Core, AR Kit, Magic Leap, Windows Mixed Reality, and HoloLens. Because of that, its cross-platform support capability is the main thing we considered here. Here, we propose a markerless AR-based method because the user can place the map on any horizontal plane in the real world without QR codes, images, symbols, or tags. It maximizes the user's flexibility in using this app and enables more intuitive and immersive interactions.

As for the guidance for the user to use the AR application, a user-friendly user interface (UI) is created. When the user starts the AR application, the user interface (UI) guides the user to point the mobile camera to a visually clear surface (Figure 5 (a)).

We use Blender, a free open-source 3D computer graphics software, to design the robot testing workspace floor. It is critical to choose a proper scale for both the real-world robot testing map and the virtual robot navigation map. As a result, we scale the virtual map from the actual workspace to the virtual unity space using a 1:0.7 ratio. In the actual workspace, the table has a height of 85 cm. We have predefined the width and depth of the table, as well as the width, depth, and height of the box. When the user places the map, 3D models of the MiRob workspace map with the MiRob are rendered on the surface (Figure 5 (a)).

Here, the AR application subscribed to two topics (for robot info transmission and object position info transmission) using a lightweight MQTT client named uPLibrary.Networking.M2Mqtt. When the application gets the robot placement info and object position coordinates, those coordinates are scaled down by 1:0.0007. Two main algorithms are used to handle the position of the Unity robot (Algorithm 1) and the rendering box object position (Algorithm 2).

With the mobile camera movement, detected surfaces are generated (Figure 5 (a)). The user can place the map by tapping on the place of the screen. After that, the user can see the virtual robot's movement (Figure 5 (c)) on the virtual map. Meanwhile, the user can see the object rendering on the virtual map (Figure 5 (d)). At the initial start of the AR application, the user can see the "Point camera to a surface" message, and then when the surfaces get generated, the message changes to "Place map by tapping on a surface." After placing the map, the message disappears. This UI implementation makes this application more user-friendly and easier to use.

Algorithm 1 Robot Movement Algorithm

Require: MQTT parameters
Ensure: Robot position syncs with the MQTT messages
  Establish MQTT connection
  Subscribe to MQTT topic for robot info transmission
  while Application is running **do**
    Receive robot info from MQTT broker
    Decode robot info
    Extract robot info as $(x, z, rotation)$
    Find robot object in the Unity scene
    Scale and set robot position as $(x*0.0007, 0, z*0.0007)$
    Scale and set robot rotation as $rotation * 0.0007$
  end while

Algorithm 2 Object Rendering Algorithm

Require: MQTT parameters
Ensure: Object position syncs with the MQTT messages
  Establish MQTT connection
  Subscribe to MQTT topic for object position transmission
  while Application is running **do**
    Receive object position info from MQTT broker
    Decode object position info
    Extract object position info as (x, z, y)
    Find box model from Unity prefabs
    Multiply each x, y, z by 0.0007 scale
    Instantiate box model at the scaled position
  end while

IV. Results and Discussion

The system enables real-time robot environment localization and augmentation from remote locations with minimal latency, facilitating interaction over a cloud interface. Experiments have been conducted in an artificially created domestic environment inside the laboratory facility for verification of the capabilities of the ARoBotx. As an initial step, MiRob needed to be placed on the starting point which is same as the starting point of the robot in the AR map. AR application users can augment the robot's initial environment (AR map), which consists of only the floor plan and the 3D virtual annotation of the MiRob. Then in the real world, When the server in the MiRob and the robot controlling code (client) in

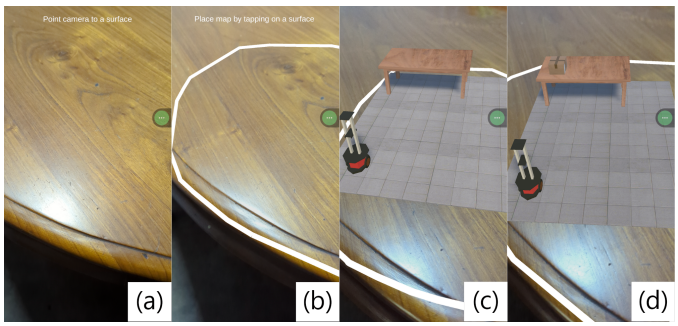

Fig. 5. States of the AR Application

the laptop starts, the MiRob robot will wander in the testing area randomly. The movement of this robot is shown on the AR map in real time.

TABLE II
DIFFERENCE OF TWO EACH CONSECUTIVE POSITION AND ROTATION DATA

Robot consecutive x coordinate difference	4-6 mm
Robot consecutive z coordinate difference	4-6 mm
Robot consecutive rotation difference	0.8-1.3 degrees

When the robot detects and identifies an object with the help of the Kinect V2 camera and the depth estimation model, the MiRob robot will stop for 2 seconds. Simultaneously the depth estimation model will estimate the distance to that particular object and publish the data packet consists of the type of the identified object (i.e Box, Table, Laptop etc.) and the average distance to that object to the cloud. AR application will receive these data from the cloud to calculate the object's 3D coordinates with respect to the robot and render the identified object in the AR map. After the time delay, the robot will wander in the testing area like before. Because the Robot Commanding and Telemetry System updates the robot position and rotation data every 1 second, there are differences between two consecutive positions and rotations.

As the robot monitoring interface, Unity provides a user-friendly approach and allows users to interact with MiRob without any advanced learning curve. In figures 5 (a), (b) (c) and (d) can see the AR application approach that we have presented to improve the user experience in AR robot monitoring systems compared to .

The implemented semantic segmentation model combines with the depth estimation model achieves an accuracy of up to 1 cm precision when estimating the distance to an object from the Kinect. When mapping the pixel coordinates of the image frame to the depth frame, the frame size difference results in the inclusion of depth values from areas outside the identified object in the depth array. This error can be identified in Figure 6. However, with the average method, we were able to achieve up to 1 cm of precision in depth estimation. This approach eliminates the complexity of common LiDAR-based depth estimation methods that often rely on point cloud data optimization techniques [3], which are inherently limited by sparsity and struggle to accurately reconstruct fine details in real time.

V. CONCLUSION

As the robot monitoring interface, Unity provides a user-friendly approach and allows users to interact with MiRob without any advanced learning curve. In figures 5 (a), (b) (c) and (d) can see the AR application approach that we have presented to improve the user experience in AR robot monitoring systems. Updating robot info via the Robot Commanding and Telemetry System is not a problem because the difference gap between two consecutive positions and rotation data is minimal. Also, achieving a higher accuracy of 93% segmenting box-shaped objects and having a minimal distance

Fig. 6. Difference between the image frame and the depth frame after masking the object

error of 1 cm is comparable to previous robot monitoring and navigation systems. Also, having a low latency and user-friendly AR application reduces the user's cognitive load.

VI. FUTURE WORKS

We restricted our testing and scripting to mobile devices running Android and iOS that have augmented reality capabilities. We have not run any scripts or tests on this system for augmented reality headsets like Magic Leap, Microsoft Hololens, etc. Thus, we anticipate the creation of new systems that are compatible with AR glasses. In addition, we expect that the use of hand gestures will enhance user interaction with the robot. For instance, when the user sketches a route on the augmented reality (AR) map, the physical robot should possess the capability to traverse it. In addition, our goal is to utilize hugging face transformers such as OWL VIT to take advantage of its pre-trained capabilities for extracting features in order to improve the classification work. By integrating this information into the semantic segmentation model, we anticipate enhancing its precision in outlining different objects.

REFERENCES

[1] R. Suzuki, A. Karim, T. Xia, H. Hedayati, and N. Marquardt, "Augmented reality and robotics: A survey and taxonomy for ar-enhanced human-robot interaction and robotic interfaces," in *CHI Conference on Human Factors in Computing Systems*, CHI '22, ACM, Apr. 2022.

[2] Z. Makhataeva and H. A. Varol, "Augmented reality for robotics: A review," *Robotics*, vol. 9, no. 2, 2020.

[3] K. Park, S. Kim, and K. Sohn, "High-precision depth estimation using uncalibrated lidar and stereo fusion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, pp. 321–335, 2020.

[4] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3523–3542, 2022.

[5] M. Fiala, "A robot control and augmented reality interface for multiple robots," in *2009 Canadian Conference on Computer and Robot Vision*, pp. 31–36, 2009.

[6] A. Corotan and J. J. Z. Irgen-Gioro, "An indoor navigation robot using augmented reality," in *2019 5th International Conference on Control, Automation and Robotics (ICCAR)*, pp. 111–116, 2019.

[7] B.-C. Huang, J. Hsu, E. T.-H. Chu, and H.-M. Wu, "Arbin: Augmented reality based indoor navigation system," *Sensors*, vol. 20, no. 20, 2020.

[8] G. W. Malith Manuhara, M. A. V. J. Muthugala, and A. G. B. P. Jayasekara, "Design and development of an interactive service robot as a conversational companion for elderly people," in *2018 Moratuwa Engineering Research Conference (MERCon)*, pp. 378–383, 2018.