ASTES

*Advances in Science, Technology and Engineering Systems Journal*
*Vol. 9, No. 4, 51-59 (2024)*
www.astesj.com
*Special Issue on Computing, Engineering and Multidisciplinary Sciences*

**ASTES Journal**
**ISSN: 2415-6698**

# Integrating Speech and Gesture for Generating Reliable Robotic Task Configuration

Shuvo Kumar Paul*, Mircea Nicolescu, Monica Nicolescu

*Department of Computer Science and Engineering, University of Nevada, Reno, 89557, USA*

ARTICLE INFO

*Article history:*
*Received: 24 April, 2024*
*Revised: 17 July, 2024*
*Accepted: 25 July, 2024*
*Online: 02 August, 2024*

*Keywords:*
*Task Configuration*
*Robotic Task*
*Gesture Recognition*

ABSTRACT

*This paper presents a system that combines speech and pointing gestures along with four distinct hand gestures to precisely identify both the object of interest and parameters for robotic tasks. We utilized skeleton landmarks to detect pointing gestures and determine their direction, while a pre-trained model, trained on 21 hand landmarks from 2D images, was employed to interpret hand gestures. Furthermore, a dedicated model was trained to extract task information from verbal instructions. The framework integrates task parameters derived from verbal instructions with inferred gestures to detect and identify objects of interest (OOI) in the scene, essential for creating accurate final task configurations.*

## 1. Introduction

The rapid advancement of robotics, automation, and artificial intelligence has ignited a revolution in robotics. While industrial robots have proliferated over the past few decades, there's been a recent surge in the integration of robots into our daily lives. This shift has led to a significant change in robotics research focus, moving from industrial applications to service robots. These robots now serve as assistants in various tasks such as cooking, cleaning, and education, among others. Consequently, this transformation has redefined the role of human users, evolving them from primary controllers to collaborative teammates, fostering increased interaction between humans and robots.

While robots can autonomously handle tasks in certain scenarios, human interaction is often necessary. Unlike industrial robots that perform repetitive tasks, service robots are designed to engage with humans while carrying out their functions. In such contexts, it's crucial for interactions to feel natural and intuitive.

To achieve this, interaction components should mirror those commonly observed in human-to-human interactions. Human interactions typically involve gestures, gaze, speech, and facial expressions. While speech effectively conveys complex information, gestures can indicate direction, location within a scene, and common task-specific actions. Combining speech and gestures enhances the interaction experience by enabling intuitive communication and conveying meaningful commands.

In this work, our focus was on integrating pointing and four distinct hand gestures with verbal interactions. We developed a neural network model to extract task parameters from verbal instructions, utilizing a dataset of 60,769 annotated samples. For recognizing pointing gestures, we employed AlphaPose to capture skeletal joint positions and calculated the forearm's angle and length ratio to determine the pointing direction. Additionally, we predicted the Object of Interest (OOI) based on the shortest distance from the pointing direction vector. Finally, we identified four common hand gestures—bring, hold, stop, and point—using hand landmarks from Google's Mediapipe and trained a 3-layer fully connected neural network for gesture recognition. This integration not only enhances natural interaction but also gathers crucial additional information and context, thereby aiding in disambiguating and inferring missing task parameters. By combining speech with gestures, our system enhances the richness and clarity of interactions, which is essential for service robots designed to assist in everyday tasks. To this effect, the following are our contributions:

1. Multimodal Integration: Unlike existing approaches that often rely on a single mode of interaction, our research integrates pointing gestures, four distinct hand gestures, and verbal instructions. This multimodal integration is crucial for creating interactions that are more natural and intuitive, closely mirroring how humans communicate with each other.

2. Enhanced Task Parameter Estimation: By combining verbal commands with gestures, our system is able to disambiguate

*Corresponding Author: Shuvo Kumar Paul, 1664 N Virginia St, Reno, NV 89557 & shuvokumarp@unr.edu

and infer missing task parameters more effectively. This leads to more accurate and reliable task configurations, which is a significant advancement in the field of human-robot interaction.

3. Real-time Processing: Our framework operates in real-time, managing multiple inputs concurrently. This capability is vital for practical applications where timely and responsive interactions are required.

4. Experimental Validation: We conducted experiments to validate our approach, demonstrating its efficacy in generating reliable task configurations. Our results show that the integration of gestures and verbal instructions significantly improves the system's performance in real-world applications.

Our work introduces a framework that seamlessly integrates multiple forms of communication. The ability to interpret and combine verbal commands with pointing and hand gestures represents a significant step forward in creating more intuitive and effective human-robot interactions. This multimodal approach not only enhances the naturalness of interactions but also provides the robot with richer contextual information, enabling it to perform tasks more accurately and efficiently.

The paper follows this structure: the subsequent section provides a concise overview of prior research on gesture recognition techniques and Natural Language Understanding in Human-Robot Interaction (HRI) design. We then proceed to elaborate on the methodology of our work. Subsequent chapters incorporate our evaluation, including experimental results and observations. Finally, we summarize our findings in the concluding section of this paper.

## 2. Related Works

### 2.1. Natural Language Understanding in HRI

In [1], the author presented a hierarchical recurrent network coupled with a sampling-based planner to enable the comprehension of sequences of natural language commands within a continuous configuration space. Similarly, in [2], the author devised a system that interprets natural language directions for robots by extracting spatial description clauses, using a probabilistic graphical model that grounds landmark phrases, evaluates spatial relations, and models verb phrases. In [3], the author explored the application of statistical machine translation techniques to enable mobile robots to interpret unconstrained natural language directions, effectively mapping them onto environment maps, leveraging physical constraints to manage translation complexity and handle uncertainty. Additionally, in [4], the author demonstrated the robot's capability to learn action sequences' conditions from natural language, promptly updating its environment state knowledge and world model to generate consistent new plans, highlighting both specific operational success and the dialogue module's scalability and responsiveness to untrained user commands. In [5], the author explored spatial relationships to create a natural communication channel between humans and robots, showcasing in their study how a multimodal robotic interface integrating linguistic spatial descriptions and data from an evidence grid map enhances natural human-robot interaction. In addition, in [6], the author introduced Generalized Grounding Graphs, a dynamic probabilistic graphical model that interprets natural language commands for autonomous systems navigating and manipulating objects in semi-structured environments.

While prior research predominantly addressed navigational tasks, our approach extends this by employing deep learning techniques to extract specific parameters from single instructions pertinent to collaborative tasks.

### 2.2. Gesture Recognition In HRI

In [7], the author proposed a two-stage Hidden Markov Model (HMM) approach aimed at enhancing Human-Robot Interaction (HRI) by enabling intuitive robot control via hand gestures. The first stage identifies primary command-like gestures, while the second stage focuses on task recognition, leveraging Mixed Gaussian distributions within HMM to improve recognition accuracy. In [8], the author introduced a robust HRI system that continuously performs gesture recognition to facilitate natural human-robot interaction by employing online-trained ad-hoc Hidden Markov Models to accommodate intra-user variability, evaluated through studies on hand-formed letters and natural gesture recognition scenarios. In [9], the author introduced an HRI system using gesture recognition that incorporates multiple feature fusion, failure verification, and validation through real-world testing with a mobile manipulator. In [10], the author presented a gesture-based human-robot interaction framework, utilizing wearable sensors and an artificial neural network for gesture classification, and introducing a parameterization robotic task manager for intuitive robot task selection and validation in collaborative assembly operations. In [11], the author introduced a parallel convolutional neural network (CNN) method optimized for recognizing static hand gestures in complex environments, particularly suited for space human-robot interaction tasks, demonstrating superior accuracy over single-channel CNN approaches and other existing methods.

We have implemented two gesture recognition methods: a heuristic-based pointing gesture recognition and pointing direction estimation, and neural network based hand gesture recognition. In both methods, we leveraged the extracted landmarks from body skeleton and hands respectively.

## 3. Methodology

### 3.1. Extracting information from verbal commands

In a typical human collaboration, shared instructions often encompass specific details like the required action, the target object, navigation directions, and the particular location of interest within the scene. Additionally, we frequently use descriptive attributes such as size, relative position, shape, pattern, and color to specify objects, as seen in phrases like "bring the green box," "the book on the right," or "hold the blue box" [12]. These details outline various aspects of a task, as depicted in Figure 1, which showcases various task parameters linked to particular instructions.

In our research, we developed a dataset tailored for collaborative robotic commands, comprising verbal instructions that specify actions and include details on at least one of the following attributes:

object name, object color, object location, or object size. This dataset contains 60,769 samples, each annotated with five labels. We thoroughly assessed eight different model architectures for training, ultimately determining that the single-layer Bi-directional Long Short-Term Memory (Bi-LSTM) model delivered the best performance.

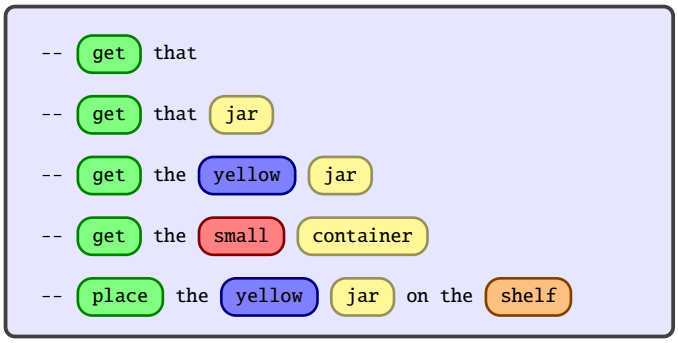



Figure 1: The task action is denoted by the green box, while the object's location in the scene is highlighted by the orange box. The red box indicates the size of the object, while the yellow and blue boxes respectively highlight the object of interest and its corresponding attributes.

Figure 2 depicts the model architecture, which comprises three neural layers. The model starts with an embedding layer, followed by a Bi-LSTM layer which is connected to a fully connected layer (FCN). The dataset vocabulary size, denoted as V, is used to one-hot encode each word, resulting in a vector size $W \in \mathbb{R}^{1\times V}$. The input sequences, consisting of $n$ words, are processed by an embedding layer represented as $\mathcal{E}$. The output from Bi-LSTM cells is concatenated and then passed through four layers. The resulting outputs from the FCN layer are subjected to softmax activation for the classification of five task parameters. Each classifier is evaluated using Cross Entropy loss $\mathcal{L}_c$. To update the model, we compute the mean of these losses as $\mathcal{L}_m = \frac{1}{5}\sum_{c=1}^{4}\mathcal{L}_c$.

## 3.2. Recognition of pointing gestures

We employed AlphaPose [13] to capture the skeletal joint positions for predicting pointing gestures and their overall direction. For simplicity, we assumed that the user uses one hand at a time for pointing. Following the categorization by [14], the authors distinguished pointing gestures into two types: extended (large) and bent arm (small) gestures. Furthermore, we generalized the forearm's orientation concerning the body into three categories: across, outward, and straight, as depicted in Figure3(b).

We analyze the forearm angle $\theta_a$ (Figure 3(a)), comparing it against a predefined threshold $\theta_t$ to distinguish between across and outward pointing gestures. When the user isn't pointing (Figure 3(b)), the forearm angle is smaller compared to when they are pointing. When pointing directly towards the camera (robot's vision) (Figure 3(c)), the angle approaches 0. To refine this analysis, we introduce the forearm length ratio $\rho_a$. If the user isn't pointing, both forearms show similar lengths (Figure 3(b)). Conversely, a noticeable difference suggests the user is pointing directly (or very close) towards the camera with that arm (Figure 3(b)). Additionally, we determine the pointing direction $d$ by analyzing the relative positions of the wrist and elbow of the pointing arm, enhancing navigational command interpretation.

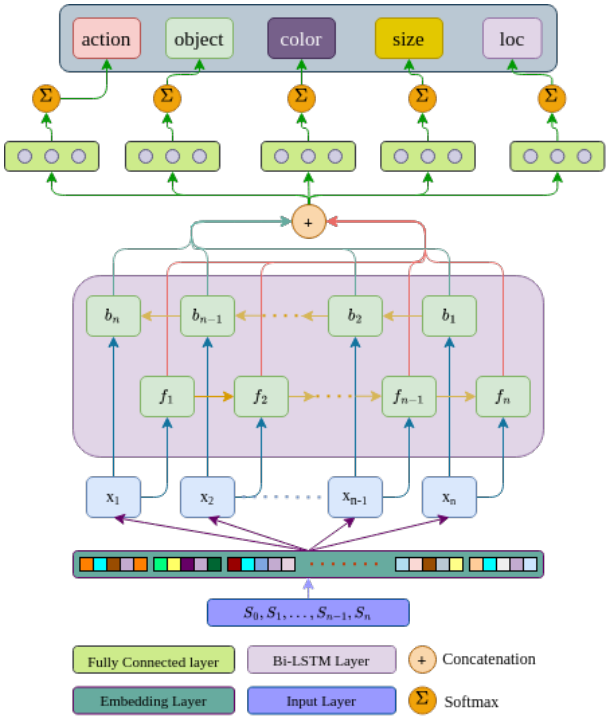



Figure 2: Neural Network (NN) model for parameter extraction from verbal commands.

### 3.2.1. Deriving $\theta_a$ from the positions of the wrist and elbow

We specifically need the locations of certain skeletal joints. These are locations of the left elbow, left wrist, right elbow, and the right wrist. This ensures our method remains effective even if some body parts are obscured, as long as the pointing hand's joints are detected. Let $(x_1, y_1)$ denote the coordinates of the elbow, and $(x_2, y_2)$ denote those of the wrist. By defining the 2D vector from the elbow to the wrist as $\vec{a} = (x_2 - x_1, y_2 - y_1)$ and using $\vec{v} = (0, 1)$ as the reference vertical vector, we can calculate the pointing angle $\theta_a$ using Equation 1:

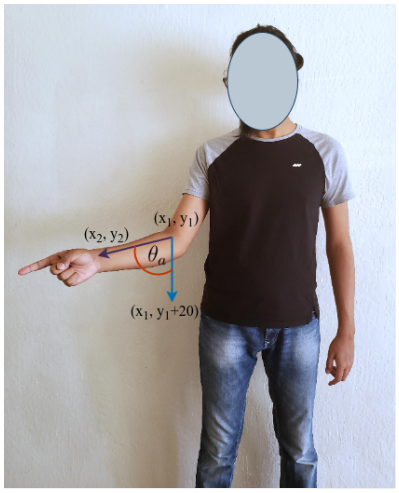
$$\theta_a = \cos^{-1}\frac{\vec{a}\cdot\vec{v}}{|\vec{a}||\vec{v}|} \tag{1}$$

If $\theta_a$ exceeds $\theta_t$, the corresponding forearm is identified as performing the pointing gesture. Next, we assess the $x$ coordinates of the wrist and elbow to determine the overall pointing direction within the scene—either left or right relative to the body. Additionally, we evaluate the forearm length ratio $\rho_a = \frac{\text{Length of the arm of interest}}{\text{Length of the other arm}}$ against a predefined ratio $\rho_t$ to determine if the user is pointing directly ahead. Specifically, $\rho_t$ is set to 0.8 and $\theta_t$ to 15°.

## 3.3. OOI Prediction

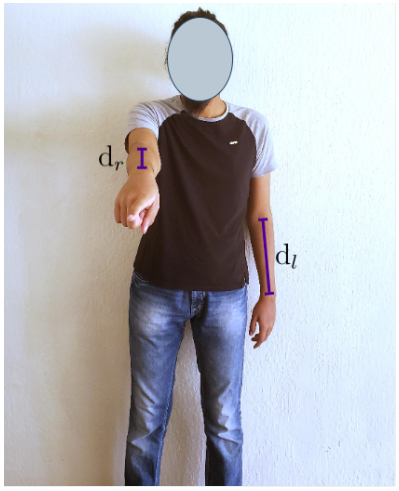
For every object identified, we establish its central point as a reference. Next, the perpendicular distance from the center of each object to the direction vector is computed.. The object found to have

the shortest distance is considered the Object of Interest (OOI) is shown in figure 4.




(a)


(b)




(c)

Figure 3: (a) Generated angle $\theta_a$ , (b) length of forearms $d_l, d_r$ when not pointing, and (c) pointing straight

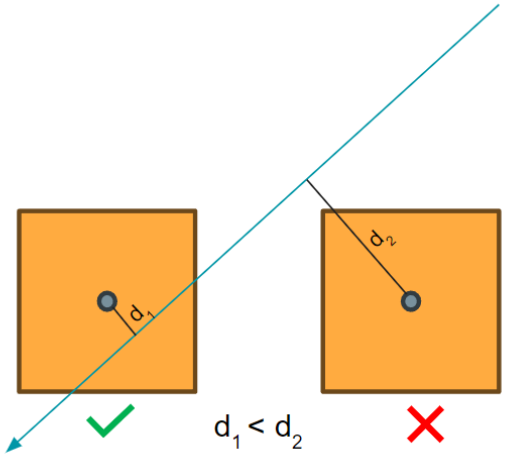



Figure 4: Determining Object of Interest (OOI) from pointing direction.

### 3.4. Gesture recognition

We have identified four common gestures for instructing robots: bring, hold, stop, and point gestures (see Figure 5). This capability enables the robot to navigate toward either an object or a designated location within the scene. Utilizing Google's Mediapipe library [15], we extracted hand landmarks, providing 21 landmark points for each hand (see Figure 6). These landmarks were captured for both hands during the aforementioned gestures to compile a dataset. Subsequently, the dataset underwent training using a 3-layer fully connected neural network model. Each fully connected layer's outputs were subjected to a dropout layer and then activated by ReLU (Rectified Linear Unit). The model's architecture is illustrated in Figure 7.

(a) Bring

(b) Hold

(c) Stop

(d) Point

Figure 5: Gesture categories

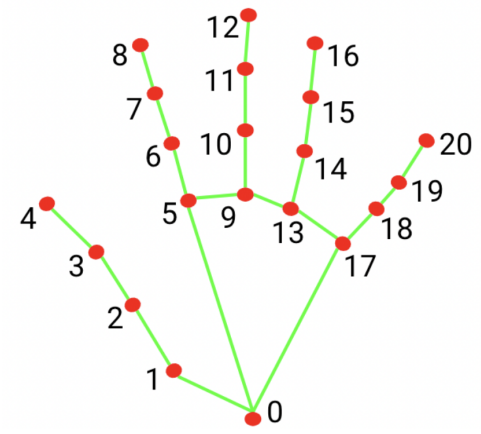



Figure 6: Extracted hand landmark [16]

## 4. Experimental Result

We have combined pointing gestures and hand gesture recognition systems with the task parameter extraction module and evaluated them separately.

### *4.1. Pointing Gesture With Verbal Command*

During our experiments, participants were assigned to perform precise pointing gestures in predefined scenarios. Each scenario depicted a scene with three distinct objects: two books and a Cheez-It box. Participants were directed to point to one object at a time. For instance, in a specific scenario, they were instructed to extend their right hand and point to the leftmost object. Thus, from this particular data sample, we could analyze that the participant executed a pointing gesture with their right hand, directing it towards their left, aiming at the object positioned farthest to the right (from their perspective). This dataset served as the foundation for our quantitative evaluations.

The experiments involved positioning the user at distances of 1.22, 2.44, 3.66, and 4.88 meters from the camera. Each system component underwent separate evaluation, encompassing tasks such as extracting parameters from verbal commands, detecting the active hand, estimating pointing direction and predicting the object of interest. The extracted task parameters were then presented in tabular format to illustrate the results.

We evaluated each frame's prediction against its label, assessing accuracy, precision, and recall. For instance, if a frame's label specifies "Right hand: pointing; Left hand: not pointing," a correct prediction of "Right hand: pointing" counts as a True Positive; otherwise, it registers as a False Negative. Conversely, predicting "Left hand: pointing" when the label indicates otherwise is a False Positive, while accurately predicting "Left hand: not pointing" is a True Negative. Table 1 details the accuracy, precision, and recall metrics across various distances.

Table 2 illustrates various sample scenarios and their corresponding task parameters extracted from data sources. The column labeled "Structured Information" showcases data derived from both the Pointing State and Verbal Command. Each row pertains to a specific scenario, beginning with an indication of whether pointing was involved, followed by the experiment number. Verbal commands, typically involving the fixed task action "get," are listed alongside extracted information from both verbal commands and simultaneous pointing states. Predicted Objects of Interest (OOI) that necessitate action are noted, along with the system's corresponding response. Instances of ambiguity are highlighted in bold within the cells.

Table 1: Pointing Gesture Recognition

| Distance (m) | Accuracy | Precision | Recall |
|---|---|---|---|
| 4.88 | 1 | 1 | 1 |
| 3.66 | 0.995 | 1 | 0.99 |
| 2.44 | 0.995 | 1 | 0.99 |
| 1.22 | 0.995 | 1 | 0.99 |

Ambiguity occurs when the object of interest (OOI) cannot be identified solely from the verbal command and pointing gesture provided. In these situations, the system informs the user with the message "Additional information needed to identify the object," and it waits for the user to provide more input, either by repeating the pointing gesture or by adjusting the command given.

In Table 2, observations indicate that ambiguity arises in different scenarios. When the system is in the "Not Pointing" state, ambiguity occurs due to insufficient object attributes (e.g., Exp# 1, 3), which hinder the unique identification of the OOI, leading the system to request more information. Conversely, in the "Pointing" state, ambiguity arises when the pointing direction does not intersect with any object boundaries. Verbal commands play a crucial role in reducing this ambiguity by providing additional information.

### *4.2. Hand Gesture Recognition With Verbal Command*

The system processes verbal commands by identifying and extracting up to five distinct task parameters, which are subsequently stored for sequential task execution. The transcription of the verbal commands and their corresponding extracted parameters is presented in Table 3.

If no matches are identified, the respective parameters are denoted as ***None***. Each command initiates a task, recorded in the order of execution. Furthermore, Figure 8 illustrates the performance comparison among different models. Subfigure a illustrates the overall accuracy, while subfigure b highlights the accuracy of Object of Interest (OOI) prediction tasks. Across both evaluations, the Bi-LSTM based model consistently outperforms all other models.

Table 4 showcases the accuracy, recall, and f1-score achieved in recognizing four specific gestures. The model consistently demonstrates high accuracy in interpreting user gestures. Figure 9 illustrates the confusion matrix for these gestures, highlighting occasional misclassifications where the 'Bring' gesture is mistakenly identified as 'Stop.' However, considering users receive feedback until the correct action is chosen, these rare errors hold minimal consequence.

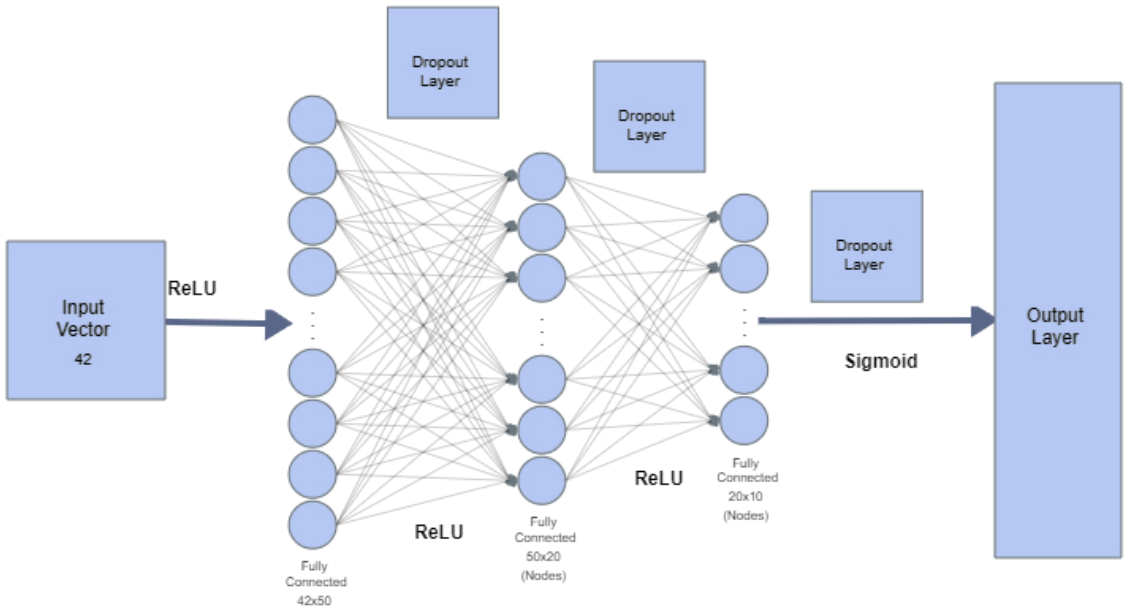



Figure 7: Gesture recognition model architecture.

Table 2: Generated Task Parameters With Pointing State

| Pointing State | Exp # | Verbal Command | Structured information | Identified Object | Feedback |
|---|---|---|---|---|---|
| *Pointing* | 1 | get that, get me that | {*action: "get", pointing_identifier: True, object: "book", object_identifiers:* {*attributes: null, position: null*}} | "book-1" | None |
| | 2 | get the red book | {*action: "get", pointing_identifier: True, object: "book", object_identifiers:* {*attributes: "red", position:* }} | "book-2" | None |
| | 3 | get that red thing | {*action: "get", pointing_identifier: True, object: null, object_identifiers:* {*attributes: "red", position:* }} | "cheez-it" | None |
| *Not Pointing* | 1 | get that, get me that | {*action: "get", pointing_identifier: False, object: null, object_identifiers:* {*attributes: null, position: null*}} | **None (ambiguous)** | **"Additional information is needed to identify object"** |
| | 2 | get the red book | {*action: "get", pointing_identifier: False, object: "book", object_identifiers:* {*attributes: "red", position: null*}} | "book-2" | None |
| | 3 | get that red thing | {*action: "get", pointing_identifier: False, object: null, object_identifiers:* {*attributes: "red", position: "right"*}} | **None (ambiguous)** | **"Additional information is needed to identify object"** |

Table 3: Extracted task parameters from various verbal commands

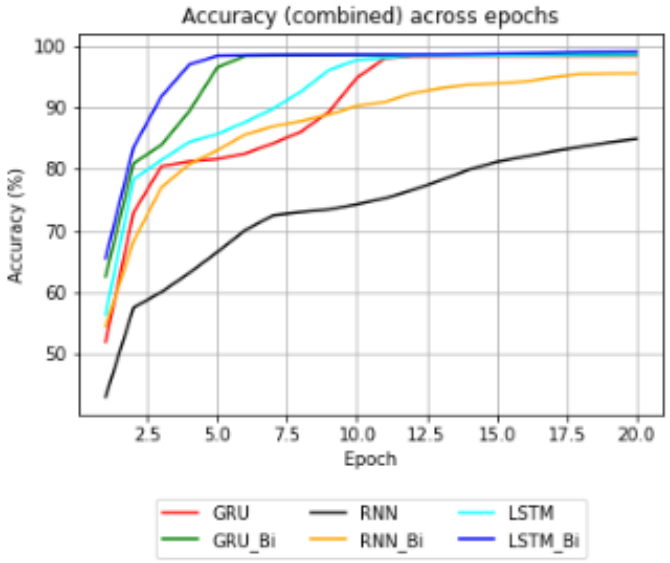
| Verbal command: "bring me the jar" |
|---|
| Object: jar — Action: bring — Attributes: None — Location: None |

| Verbal command: "give me that black box" |
|---|
| Object: box — Action: give — Attributes: [black] — Location: None |

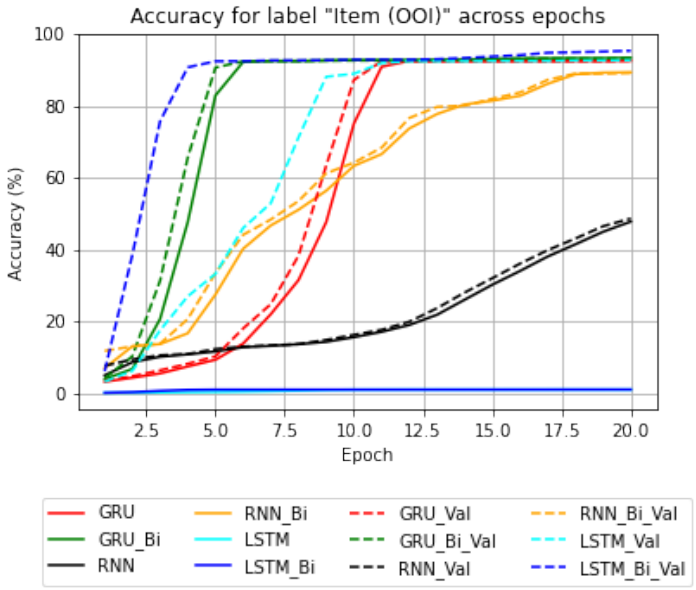
| Verbal command: "turn right" |
|---|
| Object: None — Action: turn — Attributes: None — Location: right |

| Verbal command: "hold the small white jar on your left" |
|---|
| Object: jar — Action: hold — Attributes: [white, small] — Location: left |

| Verbal command: "place the box on the shelf" |
|---|
| Object: box — Action: place — Attributes: None — Location: shelf |




(a)




(b)

Figure 8: The comparative performance of various models: (a) the aggregate accuracy of all five extracted task parameters over epochs, and (b) the accuracy for the parameter "Item (OOI)" over epochs.

Table 4: Performance metrics

| Gesture | precision | recall | f1-score |
|---|---|---|---|
| Bring | 0.93 | 1.00 | 0.97 |
| Hold | 1.00 | 0.99 | 0.99 |
| Point | 1.00 | 0.99 | 0.99 |
| Stop | 1.00 | 0.95 | 0.98 |

Subsequently, we explored scenarios where users performed gestures alongside predefined natural language instructions. Extracted information was utilized to establish task parameters, with follow-up responses issued in case of ambiguity. Table 5 delineates the sequential steps of a sample interaction, wherein gestures assist in identifying crucial task elements such as 'action' and 'object.' Notably, in step 2, the system requests additional information to identify the object of interest (OOI). Conversely, in step 4, although no verbal instructions are provided, the system maintains the previous OOI and executes the 'hold' action accordingly. This table underscores the utility of combining gesture and verbal cues for robust task configurations.

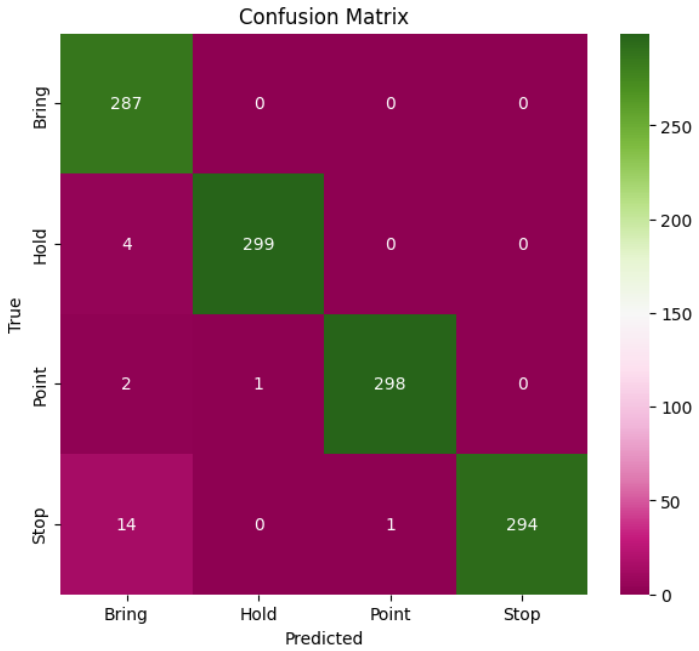



Figure 9: Confusion matrix for the four gestures

## 4.3. Qualitative Insights and Analysis

The results of our experiments provide significant insights into the effectiveness of integrating pointing gestures, hand gestures, and verbal commands for enhancing robotic task configurations. By evaluating each component separately, we were able to assess the accuracy, precision, and recall of the system in identifying the operating hand, estimating pointing direction, and predicting the object of interest (OOI).

Our findings indicate high accuracy in pointing gesture recognition across various distances, as demonstrated in Table 1. This suggests that the system can reliably interpret pointing gestures even from a distance of up to 4.88 meters, which is crucial for practical applications in diverse environments.

The integration of verbal commands with gestures significantly improves the system's ability to disambiguate and infer task parameters. As shown in Table 2, the combined use of pointing gestures and verbal instructions enhances the system's capability to identify objects and actions accurately. However, instances of ambiguity still arise, particularly when the OOI cannot be determined solely from the given inputs. In such cases, the system effectively prompts the user for additional information, demonstrating a robust error-handling mechanism.

Additionally, the hand gesture recognition component, when paired with verbal commands, consistently achieved high accuracy, recall, and f1-scores, as illustrated in Table 4 and Figure 9. This

Table 5: Extracted Task Parameters With Gesture Recognition

| Step# | Gesture Performed | Verbal Instruction | Structured Information | Feedback |
|---|---|---|---|---|
| 1 | Stop | - | action: stop, object: None, identifier: None, location: None | - |
| 2 | Bring | give me that | action: give, object: None, identifier: None, location: None | ”Additional information is needed to identify object” |
| 3 | Point | bring me that book | action: bring, object: book, identifier: pointed direction, location: None | - |
| 4 | Hold | - | action: hold, object: jar, identifier: None, location: None | - |
| 5 | Point | bring it here | action: bring, object: jar, identifier: None, location: pointed location | - |
| 6 | Bring | that red jar on the shelf | action: bring, object: jar, identifier: red, location: shelf | - |
| 7 | Point | put it here | action: put, object: None or , identifier: None, location: None | - |
| 8 | Point | go there | action: go, object: None, identifier: None, location: pointed location | - |

indicates the system's reliability in interpreting user gestures and extracting relevant task parameters, as further evidenced by the sequential task execution detailed in Table 5.

Overall, the implications of these results contribute to the overarching goals of the paper by showcasing the potential of multimodal interaction systems to facilitate natural and efficient human-robot interactions. The high accuracy and robustness of the system in various scenarios underline its practicality for real-world applications, where reliable task configurations are paramount for effective robotic assistance.

## 5. Conclusion

This paper presents a Human-Robot Interaction (HRI) framework tailored for extracting parameters essential for collaborative tasks between humans and robots. Operating in real-time, the framework concurrently manages multiple inputs. Verbal communication is leveraged to capture detailed task information, encompassing action commands and object attributes, complemented by gesture recognition. The amalgamation of these inputs yields named parameters, facilitating subsequent analysis for constructing well-structured commands. These commands seamlessly communicate task instructions to robotic entities and streamline the task execution processes.

To detect pointing gestures and infer their directions, we utilized a third-party library for skeleton landmark extraction. Additionally, we introduced a hand gesture recognition system capable of identifying four distinct hand gestures. This involved extracting hand landmarks and training a model to interpret these gestures. Furthermore, verbal commands captured by sensors are transcribed into text and processed through a pre-trained model to extract task-specific parameters. The amalgamation of this information culminates in the creation of the final task configuration. In instances where required parameters are lacking or ambiguities arise, the system offers appropriate feedback.

We evaluated the system's performance by subjecting it to various natural language instructions and gestures to generate task configurations. The extracted task parameters, corresponding to different verbal commands and gesture states, were arranged in a table to illustrate the effectiveness of our methodology.

It is important to highlight that our Human-Robot Interaction (HRI) framework showcases a robust capability to integrate verbal communication and gesture recognition in real-time, significantly enhancing the accuracy and efficiency of task parameter extraction. This integration is crucial for developing more intuitive and natural human-robot collaborative environments.

Furthermore, our experimental results validate the system's reliability in interpreting complex task instructions, which underscores its potential for practical applications in diverse settings. The high accuracy achieved in recognizing gestures and extracting task-specific parameters indicates that our approach can greatly improve the seamless execution of tasks by robotic entities.

Looking ahead, we see promising future research directions in exploring more intricate interaction scenarios. Investigating interactions involving multiple users, dynamic and continuous gestures, and complex dialogues will not only enhance the robustness of our system but also contribute to the broader evolution of HRI systems. By addressing these challenges, we aim to develop even more sophisticated and meaningful interaction frameworks that can further bridge the communication gap between humans and robots.

## References

[1] Y.-L. Kuo, B. Katz, A. Barbu, “Deep Compositional Robotic Planners That Follow Natural Language Commands,” in 2020 IEEE International Conference on Robotics and Automation (ICRA), 4906–4912, IEEE, 2020, doi:10.1109/ICRA40945.2020.9197464.

[2] T. Kollar, S. Tellex, D. Roy, N. Roy, “Toward Understanding Natural Language Directions,” in 2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 259–266, IEEE, 2010, doi:10.1109/HRI.2010.5453186.

[3] C. Matuszek, D. Fox, K. Koscher, “Following Directions Using Statistical Machine Translation,” in 2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 251–258, IEEE, 2010, doi:10.1109/HRI.2010.5453189.

[4] R. Cantrell, K. Talamadupula, P. Schermerhorn, J. Benton, S. Kambhampati, M. Scheutz, “Tell Me When and Why to Do It! Run-Time Planner Model Updates via Natural Language Instruction,” in Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction, 471–478, 2012, doi:10.1145/2157689.2157840.

[5] M. Skubic, D. Perzanowski, S. Blisard, A. Schultz, W. Adams, M. Bugajska, D. Brock, “Spatial Language for Human-Robot Dialogs,” IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), **34**(2), 154–167, 2004, doi:10.1109/TSMCC.2004.826273.

[6] S. Tellex, T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller, N. Roy, “Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation,” in Proceedings of the AAAI Conference on Artificial Intelligence, volume 25, 2011, doi:10.1609/aaai.v25i1.7979.

[7] N. Nguyen-Duc-Thanh, S. Lee, D. Kim, “Two-stage hidden markov model in gesture recognition for human robot interaction,” International Journal of Advanced Robotic Systems, **9**(2), 39, 2012, doi:10.5772/50204.

[8] S. Iengo, S. Rossi, M. Staffa, A. Finzi, "Continuous gesture recognition for flexible human-robot interaction," in 2014 IEEE International Conference on Robotics and Automation (ICRA), 4863–4868, IEEE, 2014, doi:10.1109/ICRA.2014.6907571.

[9] G. H. Lim, E. Pedrosa, F. Amaral, N. Lau, A. Pereira, P. Dias, J. L. Azevedo, B. Cunha, L. P. Reis, "Rich and robust human-robot interaction on gesture recognition for assembly tasks," in 2017 IEEE International conference on autonomous robot systems and competitions (ICARSC), 159–164, IEEE, 2017, doi:10.1109/ICARSC.2017.7964069.

[10] P. Neto, M. Simão, N. Mendes, M. Safeea, "Gesture-based human-robot interaction for human assistance in manufacturing," The International Journal of Advanced Manufacturing Technology, **101**, 119–135, 2019, doi:10.1007/s00170-018-2788-x.

[11] Q. Gao, J. Liu, Z. Ju, Y. Li, T. Zhang, L. Zhang, "Static hand gesture recognition with parallel CNNs for space human-robot interaction," in Intelligent Robotics and Applications: 10th International Conference, ICIRA 2017, Wuhan, China, August 16–18, 2017, Proceedings, Part I 10, 462–473, Springer, 2017, doi:10.1007/978-3-319-65289-4_44.

[12] F. H. Previc, "The Neuropsychology of 3-D Space." Psychological Bulletin, **124**(2), 123, 1998.

[13] H.-S. Fang, S. Xie, Y.-W. Tai, C. Lu, "RMPE: Regional Multi-person Pose Estimation," in ICCV, 2017.

[14] C.-B. Park, S.-W. Lee, "Real-Time 3D Pointing Gesture Recognition for Mobile Robots With Cascade HMM and Particle Filter," Image and Vision Computing, **29**(1), 51–63, 2011, doi:10.1016/j.imavis.2010.08.006.

[15] Google, "Google/mediapipe: Cross-platform, customizable ML solutions for live and streaming media." https://github.com/google/mediapipe, accessed: 2022-03-13.

[16] "Hand landmarks," https://developers.google.com/static/mediapipe/images/solutions/hand-landmarks.png, accessed: 2023-12-12.