# A Virtual NAO Robot System: Upper-Body Motion and Speech Recognition in Webots

Aizhan Kozhamutratova
*Computer Science*
Hof, Germany
akozhamuratova@hof-university.de

Zhaniya Zhaksylyk
*Computer Science*
Hof, Germany
zzhaksylyk@hof-university.de

Artur Teleubayev
*Computer Science*
Hof, Germany
ateleubayev@hof-university.de

Aslan Zholamanov
*Computer Science*
Hof, Germany
azholamanov@hof-university.de

***Abstract*—It is a well-known fact that the evolving era of technology paves the way for the unprecedented breakthroughs that simplify the life of human-being. In this regard, humanoid robots are intriguing fields of study for both engineers as well as the general public. In this paper, a NAO humanoid robot with speech recognition is leveraged to mimic user gestures in real time. Our system uses MediaPipe for pose tracking and utilizes Google's Speech API for multimodal interaction. Based on the Webots simulator, preliminary tests show successful integration of pose tracking, speech recognition, gesture imitation and command execution. During the project, it was noted that whole-body control of robots to keep the balance in its movement caused a significant challenge due to its high degrees of freedom. Therefore, our focus was on the upper-body movement mimic and independent actions of the robot with speech recognition commands. The paper serves as an assistant, offering insight and a holistic overview of the work with virtual humanoid robots. Taking everything into consideration, future works include detailed performance evaluation of the robot to quantify its accuracy and reliability, along with advancing the development of whole-body motion replication and refining speech recognition capabilities under diverse scenarios.**

***Keywords—Humanoid Robots, NAO, Gesture and Speech Recognition, Webots.***

## I. INTRODUCTION

Despite the current less advanced application, humanoid robots make a valuable contribution to a wide variety of areas including healthcare, education, and customer service [1]. They are differentiated by their physical appearance like humans and unique combination of three attributes: locomotion (movement around a human-oriented environment), dexterity (interaction in useful ways with their surroundings), and intelligence (independent perception and engagement with the world) [2].

As the birth of virtual reality (VR) systems, such as Oculus or HTC Vive, they are becoming viral to control virtual robots in simulation environments. To cite an example, a study published in MDPI Robotics Journal developed a VR teleoperation system using HTC Vive and Unity, that allows gesture-based robot control and smooth interaction in a virtual simulation environment [3].

Whereas external sensors and VR interfaces realm accurate motion capture, they often require additional hardware, leading to system complexity and cost. Analytical methods and deep learning approaches offer sensor-free solutions but may involve intensive computations or extensive training data.

We have designed an approach that aims for a balance between accuracy and system simplicity. Unlike other methods, our approach relies on the humanoid robot, called NAO and the robot simulator, Webots, where real-time gesture replication using the robot's built-in motions were developed.

By leveraging MediaPipe for pose tracking and OpenCV for image processing, our system enables the NAO robot to mimic user gestures in real-time. This approach differs from virtual simulation methods by focusing on direct human-robot interaction without intermediary virtual environments. During the project, it was found to add the speech recognition function where the robot perceives the voice through a microphone and executes the commands dynamically.

This work will be a contribution to human-robot interaction, with the scalability and portability of the solution, without any extra hardware for gesture-based control of robots, also providing a starting point to pore over future directions in this area. The next section is devoted to an overview of our approach and the methodology, followed by the third part which caters for the thorough exploration of the achieved results. Finally, the last section concludes with key insights and implications.

## II. APPROACH

Chief among recent advancements in hardware development is the increasing demand for simulation and its arising importance, particularly in robotics [4]. In this study, we have implemented a virtual humanoid NAO robot system using the Webots simulation platform. It combines human upper-body imitation in real-time interaction and speech recognition with the robot motions. This section provides a detailed review of the methodology which was used to achieve a simulation-based humanoid NAO robot capable of mimicking human gestures and responding to voice commands. Careful selection of tools, integration of suitable algorithms, and meticulous optimization in the process were prioritized to ensure an accurate interaction between the user and the robot. Key steps consist of platform selection, gesture recognition integration, speech command implementation, and multi-threaded motion programming.

### *A. Simulation Platform Selection*

The creation and evaluation of algorithms and methodologies in robotics are increasingly prone to rely on secure testing environments provided by simulation platforms such as Gazebo, V-Rep, and Webots [5]. Our project was not an exception. Selection an appropriate simulation platform to model and simulate the robot's movements effectively was the first challenge we have faced. According to the previous works, platforms such as PyBullet and socket-based server frameworks were evaluated for their suitability. However, PyBullet, while known for its efficiency in physics simulation, lacked the intuitive tools and pre-built robot models required for rapid prototyping. Whereas, socket-based platforms, despite their flexibility, required substantial technical expertise and caused significant challenges for the integration

of external libraries for the precise synchronization of motion coordination.

Through meticulous evaluation, we opt for Webots, renowned for its ability to simulate realistic robotic environments, offering extensive tools for designing, programming, and testing robots with optimized precision. Its comprehensive simulation environment, multi-platform support, and native integration with the NAO humanoid robot model played as main selection factors. Webots is an open-source desktop application designed for robotic simulation, providing features such as:

- A pre-built kinematic and dynamic NAO model.
- Support for setPosition functions, allowing precise control of joint angles.
- Extensive documentation and a user-friendly interface.

We configured the robot model within Webots to analyze its kinematic capabilities and joint constraints. Official documentation was reviewed to understand the kinematics and control mechanisms, including joint angle adjustments using the setPosition function. This ensured precise control over the robot's movements during subsequent development stages.

### *B. Gesture Recognition Using MediaPipe*

Gesture recognition was a key requirement to make effective and successful human-robot interaction. With an exploration of different pose estimation frameworks, such as OpenPose [6] and PoseNet [7], MediaPipe was used because it performs real-time capabilities, has low computational overhead, and boasts great performance in detecting human body and hand landmarks.

1. Integration and Configuration

MediaPipe's Pose and Hands modules were integrated to detect and track the user's gestures. These modules provided precise 3D coordinates of key landmarks like shoulders, elbows, wrists, and fingers.

2. Coordinate Mapping and Gesture Processing

The MediaPipe output, consisting of normalized landmark coordinates, was transformed to match the kinematic constraints of the NAO robot. The relative positions of landmarks (wrist with respect to shoulder and elbow) were converted into joint angle values. These angles were subsequently passed to the robot's joint controllers using Webots' APIs.

For example, the shoulder pitch and elbow roll are calculated from vertical and horizontal displacements of corresponding landmarks. The process was guided by robotic kinematics models. This would allow the robot to perform complex gestures, such as waving, pointing, and hand raises.

3. Testing and Debugging

The gesture recognition system was properly tested with diversified user movements. During the implementation, several challenges were encountered, including incorrect coordinate mappings and synchronization delays. These issues were systematically resolved through debugging and iterative testing. It was done according to the errors detected in landmarks and ensuring proper mapping to robot joints.

### *C. Library Implementation*

In order to endow the robot with the ability to perform the commands a human says, we have implemented the Google Speech Recognition API, which works to capture and process voice commands in real time. According to Stenman, the problems of speech recognition arise because of background noise interference, differences in accents, dialects and physiology affecting our speech, vocabulary size and content, detecting utterances such as coughs as non-words [8].

As for our investigation, there were 2 speech recognition platforms, Google Speech Recognition API and Pocketsphinx, where our project could be carried on. The selection of the Google Speech Recognition API over Pocketsphinx for our project was driven by its highlighted performance in key evaluation metrics, as demonstrated in the study [8]. The study illustrates that Google's Speech Recognition API significantly outweighs Pocketsphinx with regards to accuracy and achieves a markedly lower word error rate (WER), particularly in noisy environments or when processing diverse accents. Moreover, Google's Speech Recognition API offers a cloud-based architecture, leveraging powerful computational resources that allow for robust performance and able to adapt to various domains and contexts lacking extensive retraining or customization. In contrast, Pocketsphinx operates offline and is constrained by the processing capabilities of the host system, which can lead to latency or errors in speech recognition, particularly for extended or complex vocabularies. By taking into account findings from Stenman's evaluation, we concluded that the Google Speech Recognition API offers practical and effective solutions for getting real-time, accurate, and versatile speech recognition for both technical reliability and user experience excellence.

Commands such as "hello", "go forward/backward", "turn left/right", "dance", and "exhausted" were mapped to specific robot motions and integrated into the Webots environment. The recognized commands were processed and translated into corresponding motion actions for the robot.

The integration of the Google Speech Recognition API required careful consideration of the NAO robot's motion capabilities and the Webots framework. This API outputs recognized speech in real time and transforms it into text, which matches predefined commands stored in a dictionary within our system. To elaborate, the command "turn left" initiated a turnLeft60.motion function call, utilizing Webots' motion controller to execute the movement accurately. The motions were leveraged using Webots' Motion class, allowing accurate synchronization of recognized speech with robot actions.

### *D. Multi-threading for Real-Time Performance*

As the speech recognition was added, the delays were noticed. To achieve accurate recognition, smooth operation and reduce latency, multi-threading played a significant role. Its importance can be emphasized with the processes of speech recognition, command mapping, and motion execution that run simultaneously without delays. Such multi-threaded implementations are crucial in maintaining real-time interaction, as a principle supported by the study of Y. Wang, H. Gao, Y. Yang, and P. Xu [9]. This confirmed that the following modules operated in parallel:

- **Camera feed processing:** MediaPipe-based pose detection.

- **Speech recognition:** real-time voice command processing.
- **Motion execution:** robot movements with detected gestures and commands synchronization.

This multi-threaded architecture minimized delays and made sure that all subsystems operated concurrently without interfering with one another. Extensive testing was conducted to verify the system's responsiveness and to address synchronization challenges. However, it is important to notice that in order to achieve smoother experience, the program requires a significant demand on the strong Central Processing Unit (CPU) and memory performance. Insufficient system strength may lead to lag, delays, or reduced accuracy in real-time processing, affecting the overall reliability of the robotic system.

### *E. System Integration and Final Testing*

The final phase involved merging the motion replication and speech recognition modules into one system. The integrated framework was tested extensively to identify and resolve any residual bugs. Key performance metrics were taken into account to witness that the system meets the desired results.

## III. EXPERIMENT

Based on the mentioned approaches, we conducted NAO robot simulation and this section provides a comprehensive overview of the developed system, emphasizing the integration of gesture and speech recognition with multi-threading capabilities.

- *Device initialization and configuration:* The initial phase involved configuring all essential devices within the Webots simulation framework to support the NAO humanoid robot model. This step was crucial for seamless interaction with the robot's sensors and actuators, leading to real-time control.
- *Camera enablement:* The robot's top camera was activated using the findAndEnableDevices() function. This camera served as the primary input for capturing human gestures, processed later for motion replication.
- *Joint motor configuration:* Each motor, including those controlling the robot's shoulders, elbows, wrists, and phalanges, was activated and calibrated to allow precise joint manipulation. The setPosition API facilitated joint angle adjustments, critical for gesture replication.

### *A. Gesture Recognition System*

The gesture recognition system, was achieved using MediaPipe's Pose and Hand Landmark models, that enabled the translation of human movements into robotic actions. The system was modularized into sub-functions to facilitate scalability and future enhancements.

1. Pose Landmarks:
   - updateHeadPosition(): The normalized coordinates of the nose landmark were mapped to the robot's yaw and pitch angles, ensuring dynamic head movement.
   - updateArmJoints(): Joint angles for shoulder pitch, elbow roll, and wrist yaw were computed from the relative positions of shoulder, elbow, and wrist landmarks.
2. Hand Landmarks:
   - updateFingerPhalanxPositions(): Relative distances between landmarks (thumb and index) were converted into joint positions for robotic phalanges, enabling grasping and other finger-based motions.

### *B. Speech Recognition and Command Handling*

To enable voice interaction, Google Speech Recognition API was integrated into the system. This module processed real-time audio input and mapped recognized commands to corresponding robot actions.

1. Multi-threading design: Speech recognition and gesture processing ran concurrently, ensuring minimal delays.
2. Keyword spotting: All commands were designed with specific prefixed keywords; however, to enhance the accuracy of recognition and ensure reliable command execution, utilizing complete sentences is recommended. For example, articulating a phrase such as "robot, perform a forward action," which initiates linear motion, yields better results compared to issuing a single-word command.

### *C. Action Execution Framework*

Robot motions were categorized into basic actions and gesture-based responses:

1. Basic Motions: Included forward/backward movement, left/right turns, and side-stepping actions. These commands allowed the robot to navigate its environment effectively.
2. Gesture-Based Actions:
   - "Hello" Command: Triggered a hand-waving gesture.
   - "Exhausted" Command: Simulated a "wiping forehead" motion.

*Table I* outlines the implemented commands and their functionalities, while *Figure 1* demonstrates some of them.

TABLE I. SPEECH RECOGNITION COMMANDS

| ***Command*** | ***Implementation Detail*** |
|---|---|
| dancing | Triggers dance motion using TaiChi motion |
| stop | Stops current action and sets robot to neutral position |
| stand | Executes stand-up motion from a predefined position |
| turn left | Executes whole-body left rotation using turnLeft60 motion |
| turn right | Executes whole-body right rotation using turnRight60 motion |
| forward | Triggers linear forward movement using forwards motion |
| backward | Triggers linear backward movement using backwards motion |
| left | Executes stepping left using sideStepLeft motion |
| right | Executes stepping right using sideStepRight motion |

| *Command* | *Implementation Detail* |
|---|---|
| hello | Triggers a hand-waving gesture to simulate a greeting action. |
| exhausted | Simulates a "wiping forehead" motion to indicate fatigue or exhaustion. |

*Key Feature:* A "stop" command is required between actions to prevent unintended overlaps.

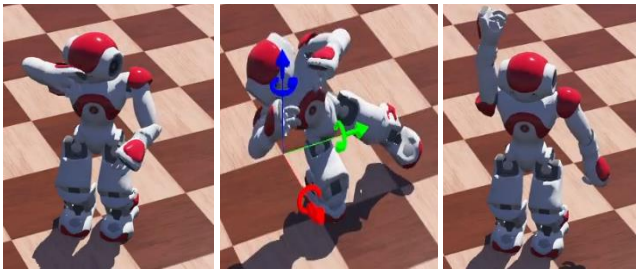
### *D. System Integration and Results*

The final system combined all modules into an integrated framework. Testing involved comprehensive validation of gesture replication and voice commands:

1. **Gesture imitation:** NAO replicates upper-body human gestures in real time wit.
2. **Motion accuracy**: The robot's actions with regards to voice commands lived up with the results we expected to achieve in execution of predefined motions.
3. **Multi-threading performance**: Concurrent execution of modules generally ensured smooth and responsive operation. Except the case, when the speech recognition module was actively processing, the camera feed occasionally became unstable. This behavior is attributed to the computational load that can be handled with higher-end systems with stronger CPUs and sufficient memory.

FIGURE 1. **ROBOT PERFORMING ACTIONS**

1.1. forward 1.2. left 1.3. turn right with camera hand gesture



1.4. exhausted 1.5. dancing 1.6. hello

## IV. CONCLUSION

This work stems from the authors' desire to develop a virtual robot, mimicking human gestures alongside with voice recognition abilities. Taking advantage of the open-source resources, our approach demonstrates the successful integration of upper-body movement recognition and speech command functionality into a virtual NAO humanoid robot within the Webots simulation environment. Implementation of Mediapipe for pose estimation and the Google Speech Recognition API for voice processing, laid a foundation for real-time human-robot interaction.

Despite its success, the project faced challenges, primarily in achieving full-body motion control due to the complexity of balancing the NAO robot's kinematic constraints. The vast focus on upper-body gestures and in-built motions allowed it to maintain robustness while emphasizing the potential for future improvements.

The system's multimodal approach—gesture recognition and speech processing—illustrates a step forward in humanoid robotics. By eliminating the need for additional hardware, it sheds light on a cost-effective and scalable solution for human-robot interaction. The work emphasizes the importance of combining advanced simulation tools with state-of-the-art algorithms to shape intelligent and interactive robotic systems. For future work, whole-body motion replication, differences in talking speed, elimination of background noise and exploring real-world applications of NAO robots in fields such as healthcare, education, and entertainment could be analyzed, constructed and documented to get a more complete set of results beside with better enhancement performance.

## REFERENCES


[1] Ergo, "How humanoid robots could change our lives," 2024, https://www.ergo.com/en/next-magazine/digitalisation-and-technology/2024/the-future-is-now-how-humanoid-robots-could-change-our-lives#:~:text=Current%20areas%20of%20application%20for,for%20patients%2C%20especially%20in%20geriatrics.

[2] U.S.-China Economic and Security Review Commission, "Humanoid Robots," 2024. [Online]. Available: https://www.uscc.gov/sites/default/files/2024-10/Humanoid_Robots.pdf.

[3] G. Zhang, L. Zhao, and M. Wang, "Innovative Approaches to Robotic Control," Robotics, 2024. [Online]. Available: https://www.mdpi.com/2218-6581/12/6/163/pdf?version=1701241857.

[4] S. Ivaldi, V. Padois, and F. Nori, "Tools for dynamics simulation of robots: a survey based on user feedback," 2014. [Online]. Available: http://arxiv.org/abs/1402.7050.

[5] A. Bonoa, K. Brameldb, L. D'Alfonsoa, and G. Fedele, "Open Access NAO (OAN): a ROS2-based software framework for HRI applications with the NAO robot," arXiv, 2024. [Online]. Available: https://arxiv.org/pdf/2403.13960.

[6] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019. [Online]. Available: https://arxiv.org/pdf/1812.08008.

[7] A. Kendall, M. Grimes, and R. Cipolla , "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization," in IEEE International Conference on Computer Vision (ICCV), 2017. [Online]. Available: https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Kendall_PoseNet_A_Convolutional_ICCV_2015_paper.pdf.

[8] M. Stenman, "Automatic speech recognition An evaluation of Google Speech," 2015. [Online]. Available: https://www.diva-portal.org/smash/get/diva2%3A852746/FULLTEXT01.pdf.

[9] Y. Wang, H. Gao, Y. Yang, P. Xu, "Multi-Thread Real-Time Control Based on Event-Triggered Mechanism,". Electronics, 2023. [Online]. Available: https://doi.org/10.3390/electronics12092129.