electronics

MDPI

Article

Human–Robot Interaction Using Dynamic Hand Gesture for Teleoperation of Quadruped Robots with a Robotic Arm

Jianan Xie *, Zhen Xu, Jiayu Zeng, Yuyang Gao and Kenji Hashimoto

Graduate School of Information, Production and Systems, Waseda University, 2-7 Hibikino, Wakamatsu-ku, Kitakyushu 808-0135, Japan; davidxu@asagi.waseda.jp (Z.X.); jiayuzeng@asagi.waseda.jp (J.Z.); yuyang_gao@moegi.waseda.jp (Y.G.); kenji.hashimoto@waseda.jp (K.H.)

* Correspondence: xiejn@akane.waseda.jp

Abstract: Human–Robot Interaction (HRI) using hand gesture recognition offers an effective and non-contact approach to enhancing operational intuitiveness and user convenience. However, most existing studies primarily focus on either static sign language recognition or the tracking of hand position and orientation in space. These approaches often prove inadequate for controlling complex robotic systems. This paper proposes an advanced HRI system leveraging dynamic hand gestures for controlling quadruped robots equipped with a robotic arm. The proposed system integrates both semantic and pose information from dynamic gestures to enable comprehensive control over the robot's diverse functionalities. First, a Depth–MediaPipe framework is introduced to facilitate the precise three-dimensional (3D) coordinate extraction of 21 hand bone keypoints. Subsequently, a Semantic-Pose to Motion (SPM) model is developed to analyze and interpret both the pose and semantic aspects of hand gestures. This model translates the extracted 3D coordinate data into corresponding mechanical actions in real-time, encompassing quadruped robot locomotion, robotic arm end-effector tracking, and semantic-based command switching. Extensive real-world experiments demonstrate the proposed system's effectiveness in achieving real-time interaction and precise control, underscoring its potential for enhancing the usability of complex robotic platforms.

Keywords: human–robot interaction; dynamic hand gesture recognition; quadruped robots; robotic arm

Academic Editor: Cecilio Angulo

Received: 5 February 2025
Revised: 17 February 2025
Accepted: 19 February 2025
Published: 21 February 2025

Citation: Xie, J.; Xu, Z.; Zeng, J.; Gao, Y.; Hashimoto, K. Human–Robot Interaction Using Dynamic Hand Gesture for Teleoperation of Quadruped Robots with a Robotic Arm. *Electronics* **2025**, *14*, 860. https://doi.org/10.3390/electronics14050860

1. Introduction

With the rapid advancement of robotics and artificial intelligence, robots are increasingly being deployed in real-world applications such as disaster response, industry, and healthcare [1–3]. For example, in the industrial sector, robotic technology has become integral to automated production lines, unmanned warehouse management, and hazardous environment operations. Robots allow operators to manage tasks remotely from control rooms, minimizing direct exposure to dangerous environments while enhancing production efficiency and safety. In the medical field, surgical robots enable doctors to perform remote procedures, significantly expanding the reach of medical services—particularly in remote areas and during emergencies—and providing critical treatment opportunities. In the aerospace sector, robots play a vital role in the operation of satellites, lunar rovers, and Mars probes, ensuring efficient performance and data collection in extreme environments. The most impressive applications are that robotic technology-based sanitation, delivery patrolling, and screening have been widely used in the epidemic period of COVID-19 [4].

In this context, Human–Robot Interaction (HRI) technology plays a crucial role in facilitating smooth communication between humans and robots, which is essential for enhancing user experience, increasing operational efficiency, and enabling robots to handle complex tasks with greater accuracy and adaptability. In recent years, large language models (LLMs) have made breakthroughs in the field of HRI, enabling robots to understand human instructions more naturally. However, the application of LLMs in HRI still faces many challenges, including security, ethical issues, and the model's ability to truly understand context [5]. In addition, its application is still limited by the level of robot automation, and manual intervention is still required in complex or dynamic environments to ensure reliability in tasks. Meanwhile, wearable haptic devices enhance the HRI experience by providing tactile feedback [6,7]. However, these devices may increase the burden of wearing and pose challenges in design and adaptation due to the need for customization for different operators. Among various HRI approaches, hand gesture-based interaction has become a key focus in robotics research. Visual-based hand gesture recognition, as a non-contact and natural form of interaction, offers significant advantages by enhancing the intuitiveness and accessibility of robot teleoperation [8,9]. This technology is especially valuable in scenarios that demand real-time control of robots to perform complex tasks remotely, allowing operators to interact intuitively and efficiently without physical contact.

Currently, although hand gesture-based HRI has made great progress, it still faces many technical and application challenges. On the one hand, existing hand gesture-based HRI methods mainly focus on either static hand gesture recognition or tracking the spatial position and orientation of the hand. While effective for simple command execution, these methods are limited in their ability to control complex robotic systems that require dynamic, multi-dimensional inputs. On the other hand, achieving human-level adaptability and navigation capabilities remains a major challenge for humanoid robots. In contrast, quadruped robots equipped with robotic arms have emerged as versatile platforms, capable of navigating complex terrains while performing precise manipulation tasks. By combining legged mobility with dexterous manipulation, this kind of robotic system has great potential for application in hazardous or inaccessible environments. However, in such robotic systems, effectively analyzing and utilizing both the semantic and motion information embedded in dynamic hand gestures, while establishing reliable mapping between gestures and various robot actions, remains a key challenge. To address these issues, this paper presents a novel dynamic hand gesture-based HRI system for teleoperation of quadruped robots equipped with mounted robotic arms, as shown in Figure 1. The proposed system leverages a Depth–MediaPipe framework to accurately extract three-dimensional (3D) coordinates of 21 hand keypoints, ensuring robust and reliable hand tracking in real-time.

In addition, we introduce the Semantic-Pose to Motion (SPM) model, which interprets both the semantic meaning and pose of the operator's hand gestures, translating them into mechanical reactions. This enables seamless robot locomotion, robotic arm manipulation, and command switching through intuitive and natural hand movements.

The rest of this paper is organized as follows: Section 2 introduces the related work, including sensor-based and computer vision-based HRI methods for teleoperation of different kinds of robots, such as wheeled robots, legged robots, humanoid robots, and robotic arms. Section 3 provides a detailed introduction to our dynamic hand gesture-based HRI method, including a Depth–MediaPipe framework and a Semantic-Pose to Motion model. Experiments and results are shown in Section 4. Finally, the conclusion and future work are introduced in Section 5.

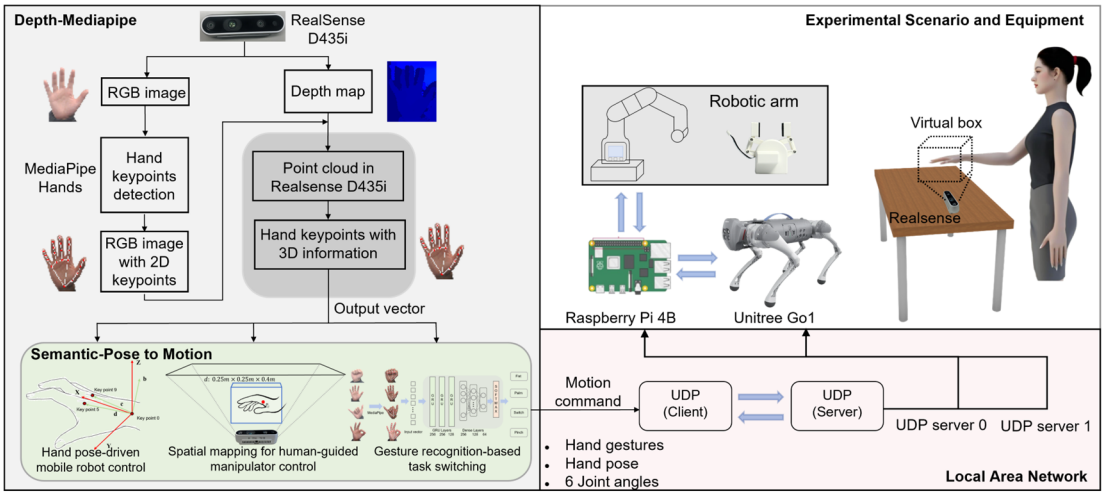


Figure 1. Overview of the proposed HRI system.

2. Related Work

As HRI advances, hand gesture-based methods have emerged as a promising approach for intuitive and efficient robot teleoperation, especially in environments where traditional control devices are impractical. Broadly, hand gesture-based HRI methods can be categorized into two primary types: sensor-based approaches, which rely on wearable devices to detect gestures, and computer vision-based approaches, which use cameras and software to track hand movements without physical contact.

2.1. Sensor-Based HRI Methods

Sensor-based HRI methods have been a focus of research due to their ability to provide high-accuracy and real-time gesture recognition, along with versatility and adaptability, making them widely applicable in various scenarios. For example: Nassour et al. [10] developed a soft sensory glove with 14 sensors using conductive liquid-filled silicone tubes to measure hand motions. They used machine learning-based methods to classify 15 gestures and regression algorithms to estimate joint angles. However, the glove requires recalibration when exposed to liquids of different temperatures, which may affect its practicality in varying environmental conditions. Antillon et al. [11] developed a smart dive glove capable of recognizing 13 static hand gestures for underwater diver communication. The glove uses five waterproof dielectric elastomer sensors and machine learning classifiers, including decision trees and support vector machines, for gesture recognition. Despite its effectiveness, gesture recognition accuracy decreases during scuba diving, affecting the overall reliability in real-world conditions. Kang et al. [12] developed a hand gesture recognition (HGR) system using a dry-type surface electromyography (sEMG) sensor and a binarized neural network (BNN). The system classifies nine dynamic gestures with high accuracy. However, the gesture recognition accuracy is limited to 95.4%, leaving room for improvement in high-precision system operations.

Recently, there has been a growing interest in Virtual Reality-based HRI methods, which provide an immersive and intuitive interactive interface, complementing or even enhancing traditional sensor-based approaches. George et al. [13], Iyer et al. [14], and Yim et al. [15] developed VR-based teleoperation systems for robot control. They utilized VR headsets to track the operator's body movements, which were then translated into control commands to enable the real-time manipulation of robots. These systems provide intuitive high-frequency feedback, especially environmental perception on the robot side, facilitating precise control in various tasks. However, their methods rely on the accuracy

of the VR headset's hand pose detection, which can be compromised by occlusions or misinterpretations, affecting precise teleoperation.

While sensor-based HRI methods offer real-time responsiveness and adaptability, they also increase the operator's physical burden, which may lead to fatigue and reduced operational efficiency.

2.2. Computer Vision-Based HRI Methods

Computer vision-based methods use visual sensors and algorithms to analyze and interpret human gestures for HRI applications. Li et al. [16] proposed a vision-based remote operating system that combines an end-to-end gesture regression network with a depth camera-based vision system to predict the joint angles of robots from depth images of human hands. This system enables accurate and stable control of dexterous robot hands. However, it depends on wrist markers for tracking hand poses, which may constrain natural hand movements and reduce flexibility in practical application. Sahoo et al. [17] introduced a compact CNN-based real-time hand gesture recognition method with a multiscale attention mechanism, to enhance finger feature representation. It achieves efficient gesture recognition in complex environments and enables real-time mobile robot control. However, they mapped specific gestures to the robot's movement direction; this discrete control command lacks continuity and flexibility, which may result in less smooth operation, making it difficult to adapt to complex tasks or dynamic environments. Chen et al. [18] proposed a mobile robot control system based on 3D hand motion. They utilized a LeapMotion sensor to capture the hand's pitch, roll, and yaw motion information while applying Gaussian filtering to reduce noise, successfully achieving teleoperation control of the WATER2 mobile robot. However, a 140 ms delay limits the system's real-time response, which may impact its usability in dynamic tasks requiring instant feedback.

While these studies demonstrate effective gesture-based control for robots, they still face challenges in real-time responsiveness, control flexibility, and adaptability to complex environments. Moreover, most of them primarily focus on single-robot systems, lacking consideration for multi-robot collaboration and scalability. Chen et al. [19] presented a LeapMotion-based control method for mobile manipulators, using a square workspace above the sensor to guide robot movement. However, the sensor's narrow detection range often causes the operator's hand to move out of view, which will limit the control precision. Xie et al. [20] proposed a one-handed gesture-based remote-control method for mobile manipulators, integrating MediaPipe (version 0.10.1) and a depth camera for hand tracking and recognition. Physical experiments validated its effectiveness. However, they only conducted experiments on mobile robotic arms based on wheeled robots and did not achieve the grasping function of the robotic arm, only controlling the position of the end effector to touch objects. Further experimental verification is needed to confirm its feasibility on different robot platforms.

To overcome the limitations of existing HRI methods, this paper presents a dynamic gesture-based HRI system capable of controlling wheeled robots, quadruped robots, and six degree-of-freedom robotic arms, ensuring natural operation, accuracy, and real-time performance.

3. Methodology

3.1. Depth–MediaPipe Framework

MediaPipe Hands [21] is a powerful open-source framework developed by Google, providing real-time, cross platform solutions for processing perceptual data. It excels in hand keypoints detection, and can provide high efficiency, fast processing, and easy

integration. MediaPipe Hands can extract 21 hand keypoints in real-time from RGB images, making it widely used in applications such as gesture recognition and virtual reality.

However, the depth coordinates provided by the MediaPipe Hands are inferred from the dataset. These coordinates are more appropriately referred to as 2.5D since the keypoints on the Z-axis (depth) are not truly captured, and this limitation makes precise 3D analysis challenging, as shown in Figure 2. Furthermore, relying on 2.5D information may lead to inaccuracies in real-world applications where spatial data are crucial.

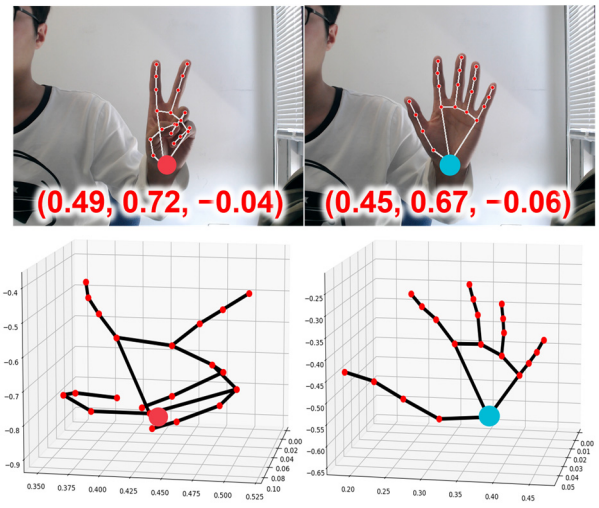


Figure 2. Example of 2.5D hand keypoints from MediaPipe Hands.

To address this issue, we propose a Depth–MediaPipe framework in Figure 3. This framework achieves 3D hand keypoint detection by combining RGB images with depth maps. Firstly, we utilize Intel RealSense D435i [22] to capture RGB images and depth maps at the same time. In the RGB channel, MediaPipe is used to extract 2D keypoints of the hand and annotate them on the image.

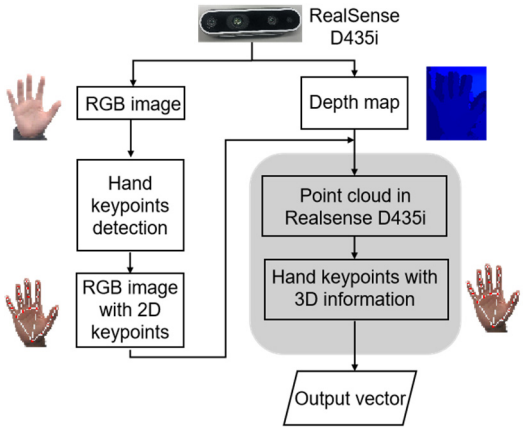


Figure 3. The Depth–MediaPipe framework.

To obtain 3D keypoints, the corresponding depth values are extracted for each detected 2D keypoint. Instead of processing the entire depth map, we apply a keypoint-based depth retrieval approach, ensuring computational efficiency. The workflow is as follows:

(a) 2D Keypoint Detection: We use MediaPipe Hands to extract 21 keypoints from the RGB image. These keypoints are normalized in the range [0,1].

(b) Mapping to Depth Image: The normalized keypoints are converted back to pixel coordinates (u, v) using the image resolution (640×480) in Equation (1):

$$u = x_{norm} \cdot W, \quad v = y_{norm} \cdot H \tag{1}$$

where W, H are the width and height of the depth image.

(c) Depth Value Extraction: The depth value D at each pixel (u, v) is retrieved from the depth map.

(d) 3D Point Cloud Computation: Using the camera's intrinsic parameters, we transform (u, v, D) from pixel coordinates into camera coordinates (X, Y, Z) as follows:

$$X = \frac{(u - c_x) \cdot Z}{f_x} \tag{2}$$

$$Y = \frac{(u - c_y) \cdot Z}{f_y} \tag{3}$$

$$Z = D \tag{4}$$

where (c_x, c_y) represent the camera principal point (optical center), (f_x, f_y) denote the camera focal length (in pixels), and Z is the depth value extracted from the depth map.

(e) Coordinate System Transformation: After obtaining the 3D coordinates in the camera coordinate system, we apply a Z-axis rotation matrix to align the coordinates with the world coordinate system as shown in Equation (5):

$$\begin{bmatrix} X_{world} \\ Y_{world} \\ Z_{world} \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{5}$$

where θ is the rotation angle around the Z-axis and equal to $-90°$. The world coordinate system is shown in Figure 4.

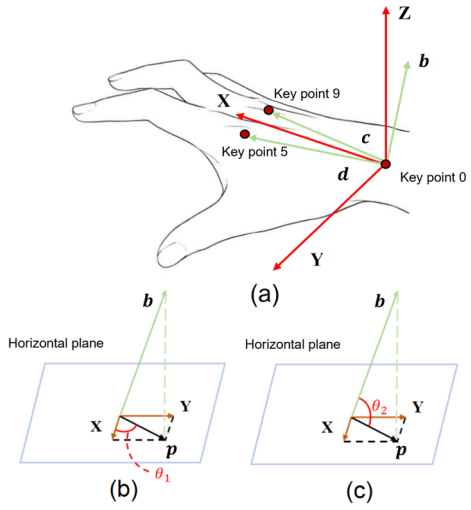


Figure 4. Hand Pose-Driven mobile robot control method. (**a**) shows our world coordinate system, as well as vector $\boldsymbol{c}$, $\boldsymbol{d}$ and normal vector $\boldsymbol{b}$. (**b**) shows the θ_1 used to control the direction of robot movement. (**c**) shows the θ_2 that controls the speed of the robot.

Finally, the framework outputs a set of 3D coordinates representing the 3D keypoints of the hand. These coordinates encapsulate the spatial structure of the hand and can be directly applied in various applications requiring 3D spatial data.

3.2. Semantic-Pose to Motion

In order to transform dynamic hand gestures into corresponding interactive commands in real time, we propose a Semantic-Pose to Motion model that consists of three key components: hand pose-driven mobile robot control, spatial mapping for human-guided manipulator control, and GRU-based gesture recognition.

3.2.1. Hand Pose-Driven Mobile Robot Control

Many existing studies [23–25] map static hand gestures (e.g., numeric gestures such as "1", "2", "3" etc.) directly to a robot's actions, resulting in relatively simple motion control. However, because these approaches rely on discrete gestures, they fail to exploit the full range of hand movements, thereby limiting flexibility and precision. To address these limitations, we propose a control system for mobile robots based on a normal hand vector. As illustrated in Figure 4, we determine this vector by connecting specific hand joints, forming vectors $\boldsymbol{d}$ (from point 0 to 5) and $\boldsymbol{c}$ (from point 0 to 9). We then obtain the normal vector which will describe the hand's orientation through the cross product, in Equation (6). This normal vector can be approximated as a virtual joystick.

$$\boldsymbol{b} = \boldsymbol{d} \times \boldsymbol{c} = \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix} \times \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} = \begin{bmatrix} d_y c_z - d_z c_y \\ d_z c_x - d_x c_z \\ d_x c_y - d_y c_x \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \tag{6}$$

This continuous tracking mechanism accommodates small rotations and tilts, which enable more refined robot control. When the hand rotates around its center, an angle θ_1 between 0 and 359 degree determines the robot's motion direction. Tilting the hand upward or downward will change the second angle θ_2, which governs speed: a shallow tilt yields a lower speed, while a steeper tilt increases it. Furthermore, when the normal vector has a negative X-component, a positive Y-component, and a negative Z-component, the robot turns right. If the normal vector is negative in all three components, the robot turns left. The calculation methods for θ_1 and θ_2 are shown as Equations (7) and (8):

$$\theta_1 = tan^{-1}\left(\frac{b_y}{b_x}\right) \tag{7}$$

$$\theta_2 = tan^{-1}\left(\frac{b_z}{\sqrt{b_x^2 + b_y^2}}\right) \tag{8}$$

We use only a few specific gestures as special action commands for the robot. For example, the numeric gesture "5" activates the robot, while "0" stops it. By integrating continuous rotational data with these discrete gestures, our approach offers a more intuitive, versatile, and precise method for operating various kinds of mobile robots.

3.2.2. Spatial Mapping for Human-Guided Manipulator Control

To enable intuitive and precise control of the robotic arm, we implement a spatial mapping approach that utilizes a virtual control space and maps it to the robotic workspace.

In the previous section, we used the Depth–MediaPipe to obtain 21 3D keypoints from the operator's hand. Among these, the palm center keypoint is selected as the control point, which represents the desired position of the robotic arm's end-effector. And then, we construct a virtual rectangular control space above the RealSense depth camera to capture the control point, shown in Figure 5.

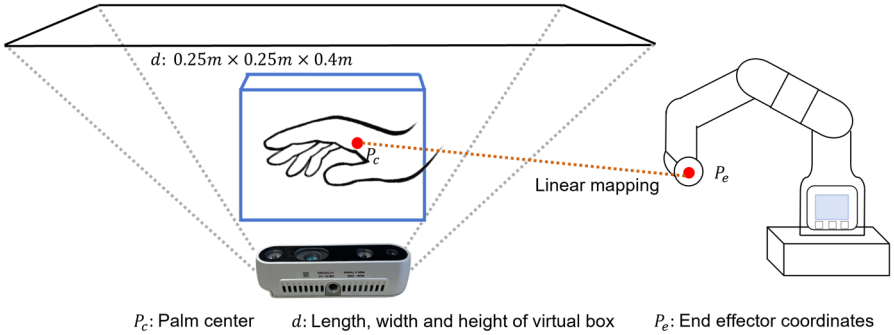


Figure 5. Robot arm space mapping based on virtual box.

This space also serves as an intermediary between the operator's hand gestures and the robotic workspace. The transformation process is defined as follows:

Coordinate definitions:

Virtual space bounds: $\left[x_{min}^{virtual}, x_{max}^{virtual}\right]$, $\left[y_{min}^{virtual}, y_{max}^{virtual}\right]$, $\left[z_{min}^{virtual}, z_{max}^{virtual}\right]$.

Robotic workspace bounds: $\left[x_{min}^{robot}, x_{max}^{robot}\right]$, $\left[y_{min}^{robot}, y_{max}^{robot}\right]$, $\left[z_{min}^{robot}, z_{max}^{robot}\right]$.

Linear transformation equation: The transformation from virtual space coordinates $P^{virtual} = \left[x^{virtual}, y^{virtual}, z^{virtual}\right]^T$ to robotic workspace coordinates $P^{robot} = \left[x^{robot}, y^{robot}, z^{robot}\right]^T$ is given by:

$$P^{robot} = T \cdot P^{virtual} + t \tag{9}$$

where the T is the scaling matrix, t is the translation vector:

$$T = diag\left(\frac{x_{max}^{robot} - x_{min}^{robot}}{x_{max}^{virtual} - x_{min}^{virtual}}, \frac{y_{max}^{robot} - y_{min}^{robot}}{y_{max}^{virtual} - y_{min}^{virtual}}, \frac{z_{max}^{robot} - z_{min}^{robot}}{z_{max}^{virtual} - z_{min}^{virtual}}\right) \tag{10}$$

$$t = \begin{bmatrix} x_{min}^{robot} \\ y_{min}^{robot} \\ z_{min}^{robot} \end{bmatrix} - T \cdot \begin{bmatrix} x_{min}^{virtual} \\ y_{min}^{virtual} \\ z_{min}^{virtual} \end{bmatrix} \tag{11}$$

In Figure 6, hand movements within the defined virtual space are translated directly into achievable positions for the robotic arm. This linear transformation ensures a one-to-one correspondence between the virtual space and the robotic workspace, enabling seamless and intuitive control.

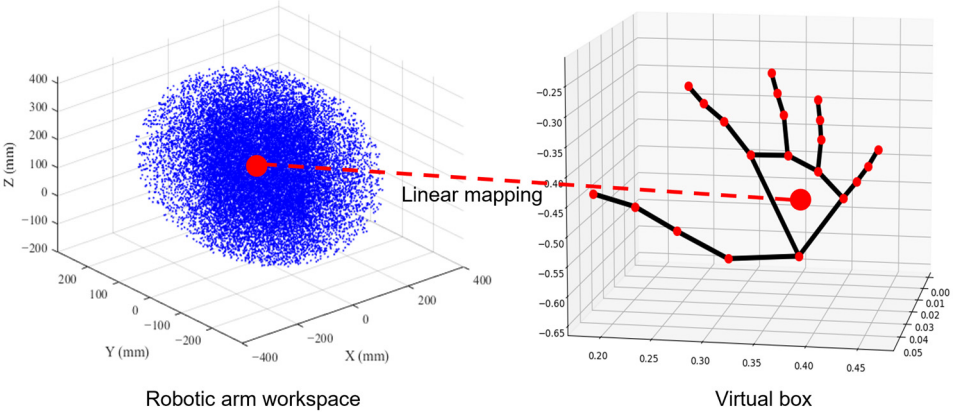


Figure 6. Coordinate mapping between two points.

After mapping the target position to the robotic workspace, the next step involves computing the joint angles required to move the robotic arm to the desired position. We

employ the PyBullet (version 3.2.6) inverse kinematics (IK) solver for this purpose. The implementation follows these steps:

a: Robot model loading: The manipulator's kinematic model will be defined using a URDF (Unified Robot Description Format) file and loaded into the PyBullet environment. This model specifies the manipulator's joint types, constraints, and degrees of freedom.

b: Target pose definition: The desired position and orientation of the end-effector, obtained from the spatial mapping process, are specified in Cartesian coordinates. These values serve as input to the IK solver.

c: Inverse kinematics computation: Using PyBullet's *calculateInverseKinematics*() function, the solver utilizes the Damped Least Squares (DLS) method, which iteratively searches for an optimal joint configuration through numerical optimization. The process continues until one of the following conditions is met [26]: (i) the distance between the target end-effector position and the computed end-effector position falls below a residual threshold (1×10^{-4}), or (ii) the solver reaches the maximum number of iterations, in which case it returns the best approximation obtained so far. Ultimately, the IK solver always strives to find a feasible solution that positions the end effector as close as possible to the target pose. The computed joint angles will be applied to the robotic arm's actuators, enabling real-time and precise control.

3.2.3. Gesture Recognition-Based Task Switching

In order to effectively control a robot system composed of mobile robots and robotic arms, we need a task switching mechanism. This mechanism allows operators to issue discrete commands to switch control between different components of the system. A gesture recognition-based method is intuitive and natural, enabling fast and accurate task switching.

The most popular CNN-based gesture recognition systems rely on convolutional neural networks (CNNs) for feature extraction and classification [27]. These methods require large, annotated datasets for training and involve high computational costs due to the processing of full image data. By contrast, we can directly utilize 21 skeletal hand coordinates extracted from MediaPipe Hands, significantly reducing computation and improving efficiency.

In addition, gesture switching is a dynamic process that can be explained as a continuous change of 21 key points. Predicting the final gesture based on the initial frames of these changes accelerates recognition and control.

Recurrent Neural Networks (RNNs) excel at handling sequential data by capturing contextual relationships. Among them, Long Short-Term Memory (LSTM) networks are widely used due to their ability to address the vanishing and exploding gradient problems. As a simplified variant of LSTM, Gated Recurrent Units (GRUs) [28] improve the structure of LSTM units by reducing the three gate units to two: the update gate and the reset gate. GRUs achieve comparable performance with fewer parameters and shorter training times, while efficiently handling long-term dependencies. The structure of a basic GRU cell is shown in Figure 7.

The general formulas of a basic GRU structure are as follows:

$$R_t = \delta(W_r \cdot X_t + U_r \cdot H_{t-1} + B_r) \tag{12}$$

$$Z_t = \delta(W_z \cdot X_t + U_z \cdot H_{t-1} + B_z) \tag{13}$$

$$N_t = tanh(W_h \cdot X_t + U_h \cdot (H_{t-1} \odot R_t) + B_h) \tag{14}$$

$$H_t = Z_z \odot N_t + (1 - Z_t) \odot H_{t-1} \tag{15}$$

$$Y_t = SoftMax((W_o \cdot H_t) + B_o) \tag{16}$$

$$\delta(x) = \frac{1}{1+e^{-x}} \tag{17}$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{18}$$

where R_t denotes the result of the reset gate at time t. It controls the influence of the previous time step's hidden state H_{t-1} in generating the current time step's candidate hidden state. Z_t in Equation (13) denotes the result of the update gate at time t. It determines how much of the previous time step's hidden state H_{t-1} should be retained for the current time step. The output of the update gate is a value between 0 and 1, a larger value retains more past information. N_t in Equation (14) represents a new candidate hidden state that is computed from a hyperbolic tangent activation function (*tanh*). H_t in Equation (15) represents the resulting hidden state of the GRU unit at time t. Y_t in Equation (16) represents the result of the GRU cell at time t.

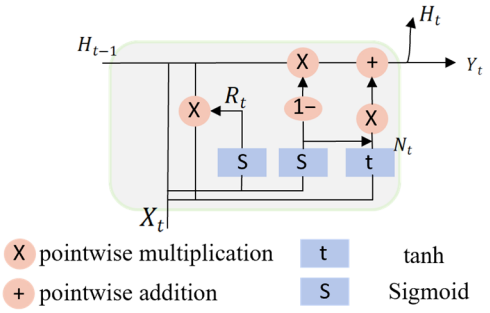


Figure 7. The structure of a GRU cell.

In addition, W_o, B_o indicate the weight and bias of H_t respectively. δ is the sigmoid activation function, W_r, W_z and W_h are the weight matrix, B_r, B_z and B_h are the bias term. And U_r, U_z and U_h are weights of the recurrent connections. ⊙ means the Hadamard product. The *tanh* means the hyperbolic tangent activation function.

This structure makes GRU an ideal choice for dynamic gesture recognition tasks, especially in scenarios that require high efficiency and real-time response. Therefore, we designed a lightweight GRU-based model for gesture recognition, as depicted in Figure 8.

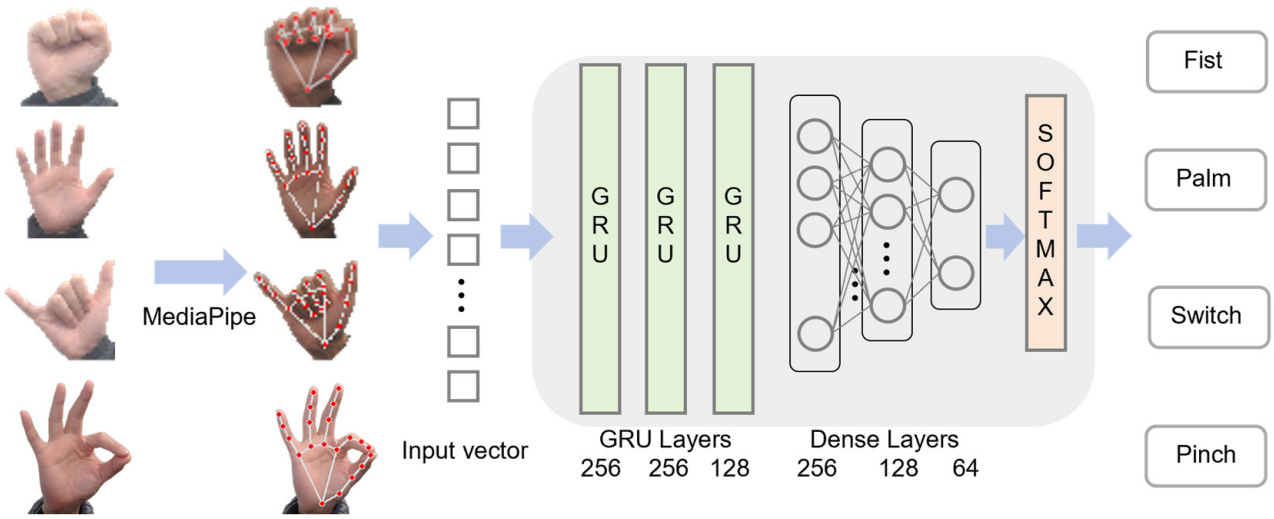


Figure 8. Four gesture classification models based on GRU.

Input Data: Several researchers [29–32] have developed similar gesture recognition models, with input vectors of shapes (F_{s1}, 1662) or (F_{s2}, 258), respectively. These vectors include coordinates of facial or other body key points. Unlike their studies, which primarily focus on sign language recognition, our research emphasizes capturing dynamic transitions of the right-hand gestures. To this end, we optimized the input vector to a shape of (30, 63), where 30, like F_{s1} and F_{s2}, represent the number of sequential frames, and 63 represents:

$$\text{keypoints in right hand} \times \text{three } \textit{dimensions} = 21 \times 3 = 63 \text{ keypoints} \tag{19}$$

This dimensionality reduction not only significantly decreases the computational complexity of the model but also enhances its focus on right-hand gesture transitions to improve recognition efficiency.

Model structure: Our model comprises three GRU layers with 256, 256, and 128 neurons, followed by a fully connected network with three dense layers, featuring 256, 128, and 64 neurons, each activated using ReLU. To enhance model robustness and reduce overfitting, a dropout layer with a rate of 0.2 is positioned before the final dense layer. We select cross-entropy loss as the loss function for our model, which is expressed by Equation (20):

$$H(y, \hat{y}) = -\sum_{i=1}^{n} y_i log(\hat{y}_i) \tag{20}$$

where $\hat{y}_i$ denotes the i-th element of the predicted vector, while y_i represents the corresponding i-th element of the true label vector. The cross-entropy loss, $H(y, \hat{y})$, measures the discrepancy between the true labels and the predicted values.

This model aims to effectively process sequence data with gesture information while maintaining simplicity and computational efficiency.

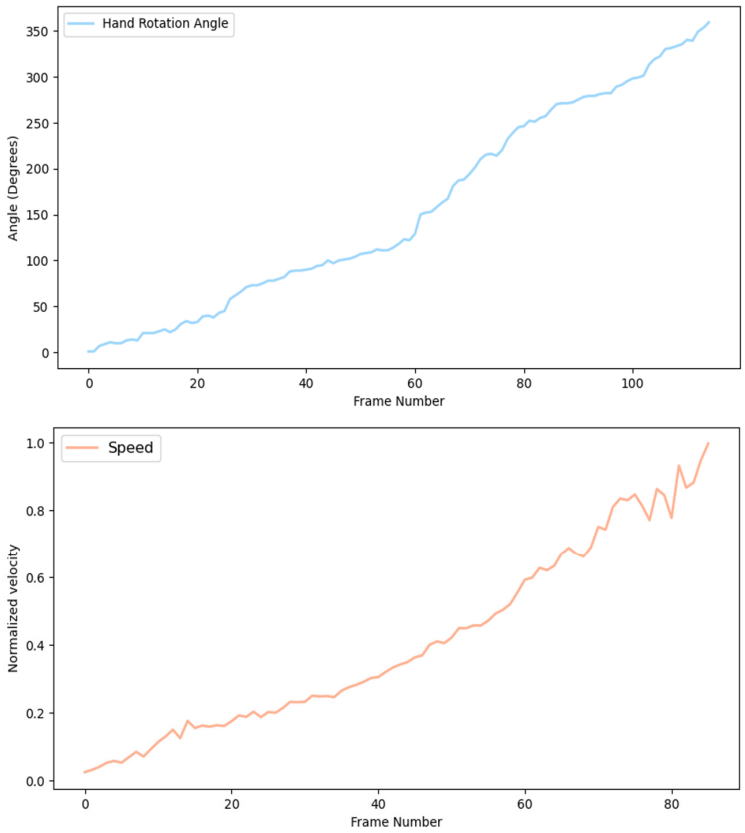
4. Experiment and Results

All the experiments in this research were conducted on a PC equipped with an Intel Core i7-14700F processor and 16.00 GB of memory, and a laptop with an Intel Core i7-11800H processor and 32.00 GB of memory.

4.1. Experiment on Mobile Robot Operation

We conducted experiments to verify the effectiveness of the proposed method for the motion control of mobile robots. By utilizing OpenCV (version 4.10.0) and the RealSense D435i to capture image frames, processing them with MediaPipe Hands, and transmitting motion commands to a mobile robot via the UDP protocol, we successfully achieved dynamic gesture-based control of a wheeled robot LIMO (Mecanum wheel mode). The relevant data are shown in Figure 9.

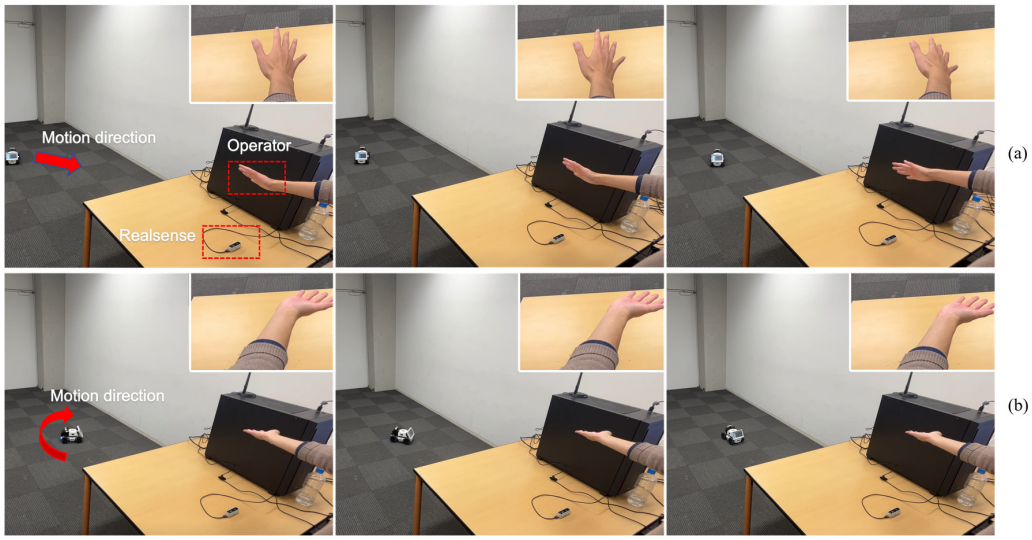
Figure 9a shows the test of direction control. It shows that we can almost achieve 0–359 degree control. Figure 9b shows the test of speed control, the normalized velocity related to the inclination of the palm from 0–1 can be achieved. By making use of such signals based on dynamic gestures, we can achieve motion control of the robot. Figure 10 presents our basic experiment. Figure 10a shows the gesture-based motion control of the wheeled robot moving towards the lower left direction. Figure 10b shows the rotation control of the wheeled robot.



(a)

(b)

Figure 9. The raw control signals for angle and speed of mobile robots. (**a**) shows the angle value obtained by rotating the wrist. (**b**) shows the velocity value obtained by tilting the hand.



(a)

(b)

Figure 10. Motion control experiment of wheeled robot. (**a**) shows the movement control of LIMO towards the bottom left. (**b**) shows clockwise rotation control of LIMO.

4.2. Experiment on Specific Gesture Classification

For training and evaluation of the proposed model, we generate our custom dataset, HG4-Dataset, for the dynamic hand gesture recognition experiment. In the robotic operation part of this research, we placed the depth camera horizontally, facing upward. As a result, all palms in our dataset faced the camera, with no occlusions other than self-occlusions (allowing for certain angles of tilt). This dataset contains 1420 gesture samples, each consisting of 30 consecutive frames, and they are proportionally divided into

a training dataset and a validation dataset. The four types of gestures shown in Table 1 are defined in the dataset: "fist, palm, pinch, and switch". The first label, "fist", specifically includes gesture transitions from "palm" to "fist", "pinch" to "fist", and "switch" to "fist", with the other labels constructed in a similar way. We also introduced an "empty label": this additional category takes into account unexpected or ambiguous gestures, allowing the model to distinguish between valid gestures and unrelated actions.

Table 1. Four types of hand gestures and their labels.

Label	fist	palm	pinch	switch	empty
Hand gesture					 etc.

We trained and evaluated our model using the HG4-Dataset and compared its performance with that of other models [33,34] trained on the same dataset.

To comprehensively evaluate the model, we used the following metrics, which are calculated using four values: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN):

Accuracy: Calculates the proportion of correctly classified samples to the total number of samples.

Precision: Calculates the proportion of correctly predicted positive samples out of all samples predicted as positive.

Recall: Calculates the proportion of actual positive samples that were correctly predicted as positive.

F1-score: Computes the harmonic mean of precision and recall.

The formulas are as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{21}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{22}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{23}$$

$$\text{F1 score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{24}$$

The results in Figure 11 and Table 2 indicate that our model achieved a good classification performance on the HG4 dataset, with high accuracy, precision, recall, and F1 score. These results validate the model's capability for real-world robot teleoperation, ensuring accurate and intuitive control across various robotic applications.

Table 2. Average results based on training and validation data.

Architecture	**Model**	**Data**	**Accuracy**	**Precision**	**Recall**	**F1score**
Abdulhamied et al. [30]	LSTM	Train Validation	98.90% 98.89%	98.92%	98.89%	98.89%
Abdallah et al. [29]	GRU	Train Validation	99.63% 99.26%	99.29%	99.26%	99.26%
Ours	GRU	Train Validation	100% 100%	100%	100%	100%

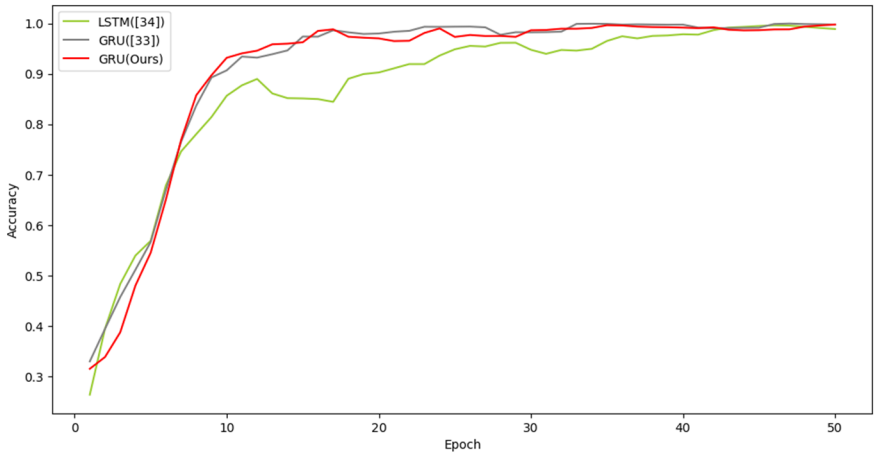


Figure 11. Model training performance comparison.

4.3. Experiment on Teleoperation of Quadruped Robots with a Robotic Arm

In this section, we utilized a quadruped robot, Unitree Go1, and a robotic arm, myCobot (Six degree-of-freedom collaborative robotic arm), to validate the proposed HRI system. A custom base was designed using Rhino 8 and a 3D printer to mount the robotic arm onto the quadruped robot, as shown in Figure 12.

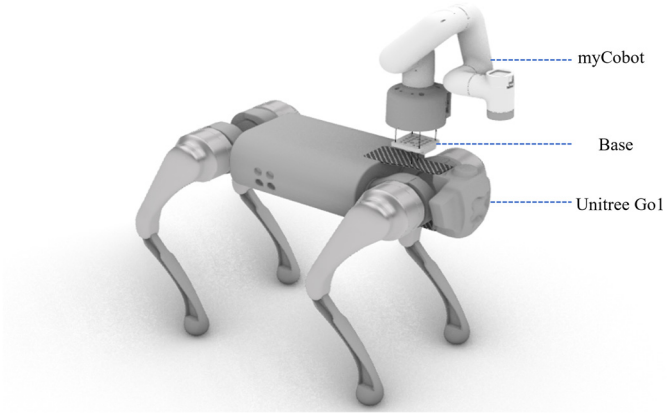


Figure 12. The model of the quadruped robot (Unitree Go1) equipped with a robotic arm (myCobot).

During the experiments, the robot's movement and stop commands were controlled using gestures "palm" and "fist", respectively, while switching between control targets was achieved through a "switch" gesture integrated with a state machine. The initial state was set to control the movement of the Go1, and upon detecting the "switch" gesture, the state transitioned to controlling the robotic arm. We use two UDP services to send instructions to Go1 and Raspberry Pi-myCobot respectively. The robotic arm's joint angle limitation is shown in Table 3.

Table 3. Robotic arm joint angle limitation.

	Joint 1	**Joint 2**	**Joint 3**	**Joint 4**	**Joint 5**	**Joint 6**
Lower limitation (rad)	−2.932	−2.356	−2.618	−2.531	−2.880	−3.14
Upper limitation (rad)	2.932	2.356	2.618	2.531	2.880	3.14

To evaluate the system's effectiveness, we designed a complete experimental task. The operator used hand gestures to control the robot to move from the starting point to target point 1, pick up an object, and then proceed to target point 2 to drop the object at a designated location.

Figure 13 qualitatively illustrates the experimental results of our method. Figure 13a–c illustrates the process of using the "palm" gesture to control the Go1 robot as it approaches an object. Figure 13d demonstrates the use of the "switch" gesture to transfer control from Go1 to myCobot. Figure 13e depicts the "pinch" gesture being utilized to operate the manipulator's gripper for object grasping. Finally, Figure 13f–h showcases the execution of the previously mentioned gestures to operate the robotic system to the target location and release the grasped object.

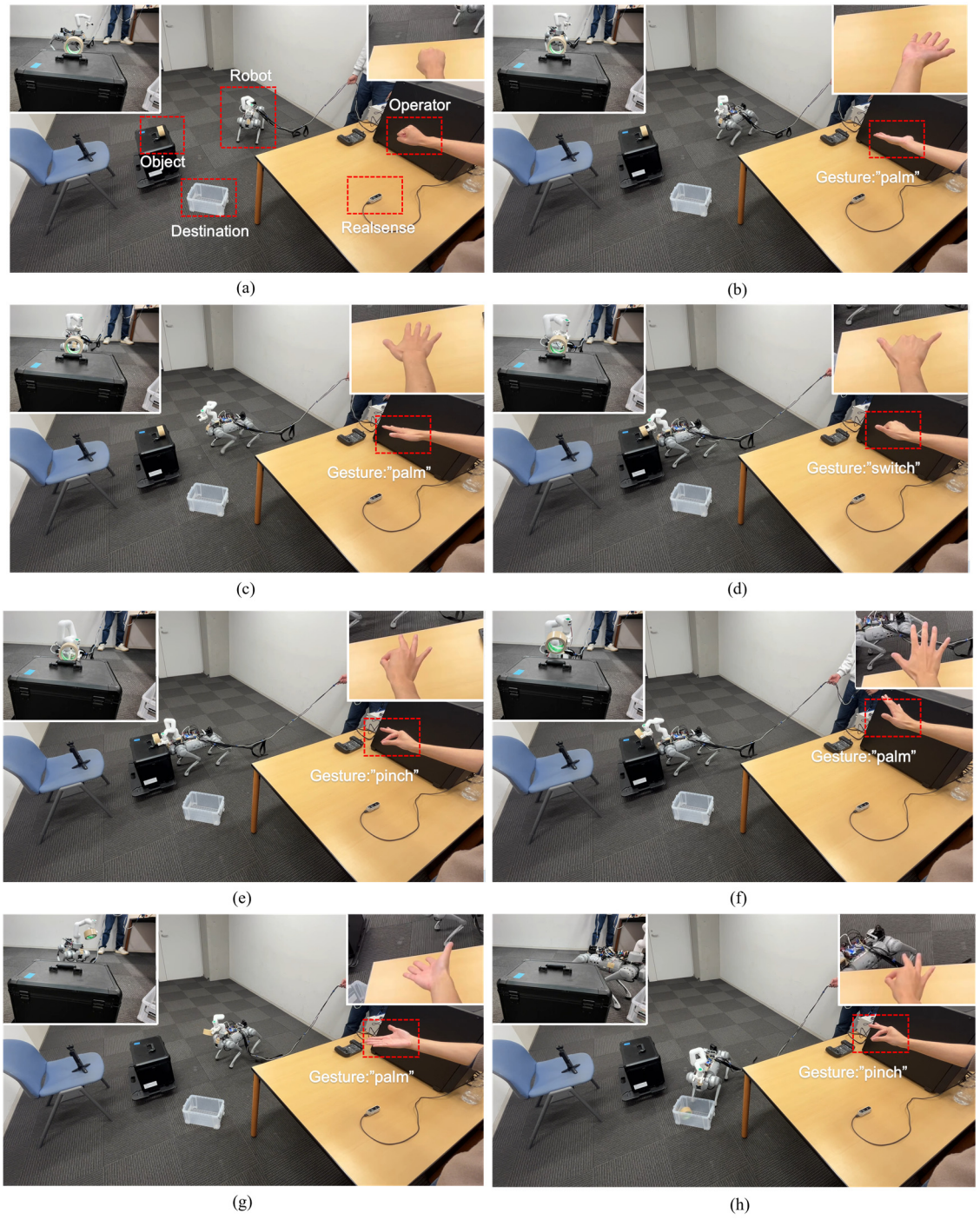


Figure 13. Hand gesture-based teleoperation process of a quadruped robot with robotic arm. (**a**–**c**) demonstrate gesture based quadruped robot movement control. (**d**,**e**) demonstrate gesture based robotic arm control. (**f**–**h**) shows our control of the robotic system to move to the target and place the object.

For the mobile robot part, as demonstrated by the experiments, the robot's movement direction is controlled by hand pose, allowing the operator to intuitively guide the robot's path based on their palm's orientation. In addition, the applicability of our control system to both wheeled and legged robots underscores its generality and robustness, which indicates its potential for real-world applications across various robotic platforms. For the robotic arm part, the position of the end effector is directly controlled by the position of the palm, enabling intuitive and precise control. To further evaluate system performance, we conducted 15 grasping experiments, where 13 attempts were successful and 2 failed. The main reason is that the occlusions between the object and the robot arm affected the operator's judgment of the grasping position. Table 4 numerically displays the time consumption, average time, and success rate of each experiment in numerical form. The basic pick-and-place experiments demonstrated the feasibility of teleoperation based on dynamic hand gestures.

Table 4. Time consumption and success rate of grasping experiments.

Time Consumption (seconds)	61	67	58	50	×	60	47	73
	×	59	73	51	45	65	41	
Average (seconds)	57.69							
Success rate	86.67%							

Figure 14 shows the delay of our gesture recognition system. The average delay is 29.77 ms.

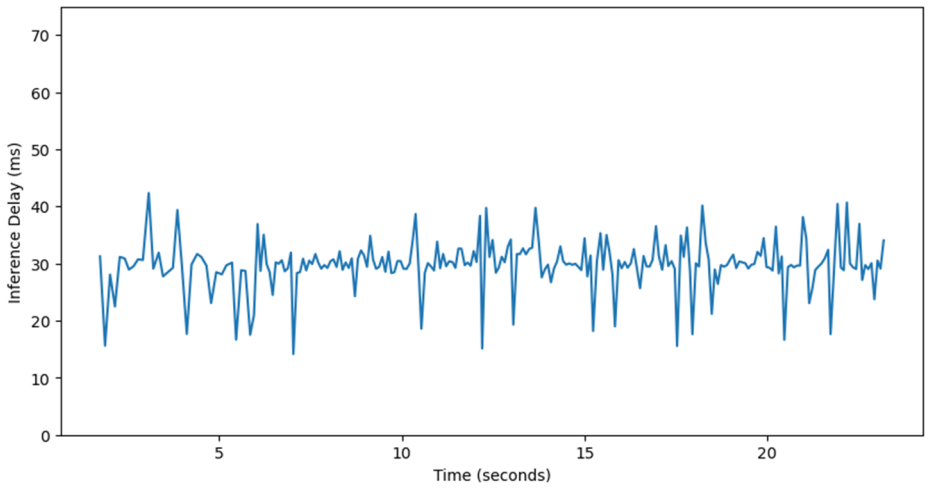


Figure 14. The inference delay over time.

Overall, this novel HRI system shows great potential for deployment in real-world applications, as its low inference delay ensures responsive and seamless interaction, which is crucial for intuitive robot teleoperation.

5. Conclusions

This paper presents a novel gesture-based HRI system for teleoperating quadruped robots equipped with robotic arms. The proposed Depth–MediaPipe framework and Semantic-Pose to Motion model effectively analyzed and utilized both semantic and motion information in dynamic hand gestures, and established a reliable mapping between gestures and various robot actions. This enables precise and intuitive real-time teleoperation of such robotic systems. Experimental validation on LIMO, Unitree Go1, and myCobot demonstrates the system's effectiveness in real-world robotic applications, while latency

analysis confirms its low delay and real-time responsiveness, making it suitable for practical deployment across various fields.

While our system shows strong potential, certain aspects still require improvement:

Hand posture constraints: The gesture recognition process currently imposes some constraints to ensure stable and reliable control. Although these constraints aligned with the normal hand movement range of operators in this study, they may limit flexibility in broader applications.

Challenges in real-time visual feedback: Our HRI framework relies on the operator's direct observation for the grasping position assessment. However, occlusions between the robotic arm and the object may limit visibility, reducing grasping accuracy.

Therefore, our future work will focus on relaxing hand posture constraints by integrating multi-view cameras and data fusion techniques, as well as data augmentation strategies to improve model generalization. Additionally, mounting a camera on the quadruped robot, combined with mixed reality (MR) technology, will provide operators with a clear view of the robot's surrounding environment, which can solve the problem of insufficient visual feedback. These enhancements will further improve the adaptability, flexibility, and usability of our approach, making it more applicable to real-world teleoperation scenarios.

Author Contributions: Conceptualization, J.X. and K.H.; methodology, J.X.; software, J.X. and J.Z.; validation, J.X., Z.X., Y.G. and K.H.; formal analysis, J.X. and Z.X.; investigation, J.X.; resources, K.H.; data curation, J.X.; writing—original draft preparation, J.X.; writing—review and editing, Z.X. and K.H.; visualization, J.X. and J.Z.; supervision, K.H.; project administration, K.H.; funding acquisition, K.H. All authors have read and agreed to the published version of the manuscript.

Funding: This study was conducted with the support of the Information, Production and Systems Research Center, Waseda University; Future Robotics Organization, Waseda University, and was a part of the humanoid project at the Humanoid Robotics Institute, Waseda University. This work was supported by JSPS KAKENHI Grant Number JP21H05055. This work was also supported by the Waseda University Grant for Special Research Projects (Project number: 2024C-188) and JST BOOST, Grant Number JPMJBS2429.

Institutional Review Board Statement: Not applicable. This study did not involve humans or animals and therefore did not require ethical approval.

Data Availability Statement: The original data supporting the conclusions of this study will be provided upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wilk-Jakubowski, G.; Harabin, R.; Ivanov, S. Robotics in crisis management: A review. *Technol. Soc.* **2022**, *68*, 101935. [CrossRef]
2. Dzedzickis, A.; Subačiūtė-Žemaitienė, J.; Šutinys, E.; Samukaitė-Bubnienė, U.; Bučinskas, V. Advanced applications of industrial robotics: New trends and possibilities. *Appl. Sci.* **2021**, *12*, 135. [CrossRef]
3. Chatterjee, S.; Das, S.; Ganguly, K.; Mandal, D. Advancements in robotic surgery: Innovations, challenges and future prospects. *J. Robot. Surg.* **2024**, *18*, 28. [CrossRef]
4. Zhao, Z.; Ma, Y.; Mushtaq, A.; Rajper, A.M.; Shehab, M.; Heybourne, A.; Song, W.; Ren, H.; Tse, Z.T. Applications of robotics, artificial intelligence, and digital technologies during COVID-19: A review. *Disaster Med. Public Health Prep.* **2022**, *16*, 1634–1644. [CrossRef]
5. Zhang, C.; Chen, J.; Li, J.; Peng, Y.; Mao, Z. Large language models for human-robot interaction: A review. *Biomim. Intell. Robot.* **2023**, *3*, 100131. [CrossRef]
6. Adilkhanov, A.; Rubagotti, M.; Kappassov, Z. Haptic devices: Wearability-based taxonomy and literature review. *IEEE Access* **2022**, *10*, 91923–91947. [CrossRef]
7. Peng, Y.; Sakai, Y.; Funabora, Y.; Yokoe, K.; Aoyama, T.; Doki, S. Funabot-Sleeve: A Wearable Device Employing McKibben Artificial Muscles for Haptic Sensation in the Forearm. *IEEE Robot. Autom. Lett.* **2025**, *10*, 1944–1951. [CrossRef]
8. Qi, J.; Ma, L.; Cui, Z.; Yu, Y. Computer vision-based hand gesture recognition for human-robot interaction: A review. *Complex Intell. Syst.* **2023**, *10*, 1581–1606. [CrossRef]

9. Ji, B.; Wang, X.; Liang, Z.; Zhang, H.; Xia, Q.; Xie, L.; Yan, H.; Sun, F.; Feng, H.; Tao, K.; et al. Flexible strain sensor-based data glove for gesture interaction in the metaverse: A review. *Int. J. Hum.–Comput. Interact.* **2024**, *40*, 6793–6812. [CrossRef]
10. Nassour, J.; Amirabadi, H.G.; Weheabby, S.; Al Ali, A.; Lang, H.; Hamker, F. A robust data-driven soft sensory glove for human hand motions identification and replication. *IEEE Sens. J.* **2020**, *20*, 12972–12979. [CrossRef]
11. Antillon, D.W.O.; Walker, C.R.; Rosset, S.; Anderson, I.A. Glove-based hand gesture recognition for diver communication. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *34*, 9874–9886. [CrossRef] [PubMed]
12. Kang, S.; Kim, H.; Park, C.; Sim, Y.; Lee, S.; Jung, Y. sEMG-based hand gesture recognition using binarized neural network. *Sensors* **2023**, *23*, 1436. [CrossRef]
13. George, A.; Bartsch, A.; Farimani, A.B. Openvr: Teleoperation for manipulation. *arXiv* **2023**, arXiv:2305.09765. [CrossRef]
14. Iyer, A.; Peng, Z.; Dai, Y.; Guzey, I.; Haldar, S.; Chintala, S.; Pinto, L. Open teach: A versatile teleoperation system for robotic manipulation. *arXiv* **2024**, arXiv:2403.07870.
15. Yim, L.S.; Vo, Q.T.; Huang, C.I.; Wang, C.R.; McQueary, W.; Wang, H.C.; Huang, H.; Yu, L.F. WFH-VR: Teleoperating a robot arm to set a dining table across the globe via virtual reality. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 4927–4934.
16. Li, S.; Hendrich, N.; Liang, H.; Ruppel, P.; Zhang, C.; Zhang, J. A dexterous hand-arm teleoperation system based on hand pose estimation and active vision. *IEEE Trans. Cybern.* **2022**, *54*, 1417–1428. [CrossRef] [PubMed]
17. Sahoo, J.P.; Sahoo, S.P.; Ari, S.; Patra, S.K. Hand gesture recognition using densely connected deep residual network and channel attention module for mobile robot control. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 1–11. [CrossRef]
18. Chen, L.; Li, C.; Fahmy, A.; Sienz, J. GestureMoRo: An algorithm for autonomous mobile robot teleoperation based on gesture recognition. *Sci. Rep.* **2024**, *14*, 6199. [CrossRef] [PubMed]
19. Chen, M.; Liu, C.; Du, G. A human–robot interface for mobile manipulator. *Intell. Serv. Robot.* **2018**, *11*, 269–278. [CrossRef]
20. Xie, J.; Zhang, Y.; Xu, Z.; Gao, Y.; Bai, J.; Zeng, J.; Hashimoto, K. One-Handed Wonders: A Remote Control Method Based on Hand Gesture for Mobile Manipulator. In Proceedings of the 2024 International Conference on Advanced Robotics and Mechatronics (ICARM), Tokyo, Japan, 8–10 July 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 686–691.
21. Zhang, F.; Bazarevsky, V.; Vakunov, A.; Tkachenka, A.; Sung, G.; Chang, C.L.; Grundmann, M. Mediapipe hands: On-device real-time hand tracking. *arXiv* **2020**, arXiv:2006.10214.
22. Intel RealSense. DepthCameraD435i. 2024. Available online: https://www.intelrealsense.com/depth-camera-d435i (accessed on 10 October 2024).
23. Ciudin, P.; Goia, H.Ș.; Popișter, F. Implementation of Human Gestures in the Control of Collaborative Robots. In Proceedings of the International Scientific-Technical Conference Manufacturing, Poznan, Poland, 14–16 May 2024; Springer Nature: Cham, Switzerland, 2024; pp. 27–42.
24. Li, J.; Liu, H.; Luo, X.; Chen, C.L.P.; Yang, C. Gesture-Based Human-Robot Interaction Framework for Teleoperation Control of Agricultural Robot. In Proceedings of the 2023 IEEE International Conference on Unmanned Systems (ICUS), Hefei, China, 13–15 October 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 1–6.
25. Carneiro, M.R.; Rosa, L.P.; De Almeida, A.T.; Tavakoli, M. Tailor-made smart glove for robot teleoperation, using printed stretchable sensors. In Proceedings of the 2022 IEEE 5th International Conference on Soft Robotics (RoboSoft), Edinburgh, UK, 4–8 April 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 722–728.
26. Coumans, E.; Bai, Y. PyBullet Quickstart Guide. Available online: https://docs.google.com/document/d/10sXEhzFRSnvFcl3XxNGhnD4N2SedqwdAvK3dsihxVUA/edit?pli=1&tab=t.0 (accessed on 16 January 2025).
27. Gadekallu, T.R.; Alazab, M.; Kaluri, R.; Maddikunta, P.K.; Bhattacharya, S.; Lakshmanna, K. Hand gesture classification using a novel CNN-crow search algorithm. *Complex Intell. Syst.* **2021**, *7*, 1855–1868. [CrossRef]
28. Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
29. Subramanian, B.; Olimov, B.; Naik, S.M.; Kim, S.; Park, K.H.; Kim, J. An integrated mediapipe-optimized GRU model for Indian sign language recognition. *Sci. Rep.* **2022**, *12*, 11964. [CrossRef] [PubMed]
30. Samaan, G.H.; Wadie, A.R.; Attia, A.K.; Asaad, A.M.; Kamel, A.E.; Slim, S.O.; Abdallah, M.S.; Cho, Y.-I. Mediapipe's landmarks with rnn for dynamic sign language recognition. *Electronics* **2022**, *11*, 3228. [CrossRef]
31. Amit, M.L.; Fajardo, A.C.; Medina, R.P. Recognition of real-time hand gestures using mediapipe holistic model and LSTM with MLP architecture. In Proceedings of the 2022 IEEE 10th Conference on Systems, Process & Control (ICSPC), Malacca, Malaysia, 17 December 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 292–295.
32. Maheswara, Y.D.; Al-Sulthon, M.A.; Wicaksono, P.A.; Afifah, K.; Prihatiningrum, N. Real-Time BISINDO Sign Language Recognition: A Dynamic Approach with GRU and LSTM Models Leveraging MediaPipe. In Proceedings of the 2023 6th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Batam, Indonesia, 11 December 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 226–232.

33. Abdallah, M.S.; Samaan, G.H.; Wadie, A.R.; Makhmudov, F.; Cho, Y.-I. Light-weight deep learning techniques with advanced processing for real-time hand gesture recognition. *Sensors* **2022**, *23*, 2. [CrossRef] [PubMed]
34. Abdulhamied, R.M.; Nasr, M.M.; Abdulkader, S.N. Real-time recognition of American sign language using long-short term memory neural network and hand detection. *Indones. J. Electr. Eng. Comput. Sci.* **2023**, *30*, 545556. [CrossRef]