Tsinghua Science and Technology
Doi: 10.26599/TST.2026.90100023

Research/Review Article

A Monocular Ranging Method in Floating Touch System

Fugang Liu, Songnan Duan (✉), Ruolin Zhou, and Yongtao Zhu

Abstract The field of human-computer interaction based on floating touch is an emerging research field. While existing floating touch systems rely on high-cost hardware (e.g., LiDAR, depth cameras), low-cost monocular camera-based solutions remain understudied. This paper presents a monocular ranging method in floating touch system. Firstly, a shallow neural network is used to fit an empirical function to establish a hand distance model, and a deep neural network proposes a distance estimation algorithm based on camera pose estimation. Then, a data fusion strategy is used to integrate the two to construct a multi-layer network that effectively improves the accuracy of monocular ranging for hands. Finally, a virtual hand-based pose compensation algorithm is introduced, significantly enhancing the overall method's precision and robustness in complex hand movements or angles. Experiments show that the ranging algorithm based on camera pose estimation can calculate the camera's offset components in all directions, addressing issues like pose angles confusion in monocular distance measurement. The pose compensation algorithm based on a virtual hand model can extract virtual coordinates and specific part parameter data, calculating compensation errors and improving the accuracy and robustness of hand distance measurement algorithms. The proposed monocular ranging method outperforms monocular metric depth estimation methods in outdoor and occlusion conditions, achieving stable millimeter level accuracy in the range of 20-100 cm. Compared with traditional distance measurement methods, the mean relative error is reduced by 15.45%.

Keywords Floating touch; Monocular vision; Monocular ranging; Image processing

• Fugang Liu, Songnan Duan and Yongtao Zhu are with School of Electronics and Information Engineering, Heilongjiang University of Science and Technology, Harbin, 150022, China. E-mail:liufugang@usth.edu.cn, dsn63447225@163.com and zyt078@163.com
• Ruolin Zhou is with the Department of Electrical and Computer Engineering, University of Massachusetts, Dartmouth, 02747, USA. E-mail:ruolin.zhou@umassd.edu
Manuscript received: 2025-05-04; revised: 2025-09-17; accepted: 2026-01-16

1 Introduction

With the rapid advancement of the digital era, floating touchless human-computer interaction technology has become increasingly integrated into daily life. Most existing touchless human-computer interaction approaches are predominantly dependent on high-cost hardware platforms, such as radar sensor systems[1,2], binocular vision cameras[3], wearable devices[4], depth stereo cameras[5], and multi-sensor fusion architectures[6]. For floating touch interaction, Nooruddin et al.[7] proposed an AR/VR wearable device-based interaction approach, which effectively enhances the accuracy of in-air text interaction.Subsequently, scholars gradually used monocular vision for human-computer interaction tasks, and Mishra et al.[8] used an end-to-end monocular vision architecture to achieve two-dimensional pose estimation of multiple hands. Qu et al.[9] proposed a Multi-Scale Channel Attention Network to realize an in-air handwriting system based on monocular vision, providing a new human-computer interaction method.

In an effort to reduce costs and promote the technology's accessibility, scholars have embarked on research utilizing monocular vision for floating touch. Current monocular vision-based systems for floating touch face four main challenges: (1) Monocular cameras struggle to effectively address issues such as lens distortion[10], lens-to-hand pose angle confusion, and other optical aberrations. (2) The anatomical complexity of hand structures poses a major challenge[11], especially when users exhibit rapid changes in hand pose during floating touch interactions, affecting both the precision and robustness of the distance measurement system. (3) Subtle hand movements can result in significant scale variations in captured images[12], requiring that models achieve millimeter-level monocular distance measurement accuracy to efficiently perform hand localization tasks. (4) Floating touch tasks require high system responsiveness and pose a much greater demand for real-time performance on mobile terminal devices compared to other platforms[13].

To address the aforementioned challenges, this paper proposes a floating touch method based on monocular vision that combines multiple algorithms. By extracting the coordinates of the hand in a two-dimensional space as a positional indicator, this method achieves floating functionality on the

device. Additionally, it extracts the three-dimensional distance from the hand to the camera lens as an operational executor, enabling touch interaction functionality on mobile devices. This strategy involves the processing of images captured by monocular vision and introduces a lens pose estimation distance measurement algorithm and a hand pose compensation algorithm. These algorithms map the two-dimensional hand image to three-dimensional space, extracting relative distance and coordinate orientation information between the hand and the camera lens, facilitating the development of a simple, cost-effective, and efficient floating touch application system.

The key innovations of this paper are:

1. Lens Pose Estimation-based Ranging Algorithm
   We propose a novel ranging method that converts the camera's intrinsic rotation into extrinsic rotation parameters. This approach unifies lens pose variations, computes multi-directional offset components, and estimates the current camera pose to enable image distortion correction. Our solution effectively addresses two critical challenges: (1) structured light distortion in camera lenses, and (2) inconsistent attitude angles between the lens and the measured hand. The method achieves precise hand-to-lens distance measurement with improved stability.
2. Virtual Hand Model-based Pose Compensation Algorithm
   We develop a compensation framework that constructs a frontal-position virtual hand model using detection results from the hand pose detector. By extracting virtual coordinates and parametric data of specific hand parts, the system calculates compensation errors dynamically. This innovation resolves measurement inaccuracies caused by extreme hand pose variations, significantly enhancing measurement precision and robustness, particularly for non-standard hand orientations or viewing angles.
3. Hybrid Neural Network Architecture for Monocular Floating Touch Interaction
   We integrate these algorithms into a unified floating touch framework by combining deep and shallow neural networks. The deep network leverages our lens pose estimation method to map 2D images to 3D coordinates, while the shallow network employs empirical regression to adaptively refine hand orientation and position. A confidence-weighted fusion strategy, enhanced by virtual hand compensation, optimizes the final distance estimation. Through model pruning and 8-bit quantization, we deploy the system as a 2MB ultra-lightweight model on Android devices, achieving real-time millimeter-level ranging within 20–100cm.

2 Related Work

The distance from the hand to the camera is a critical piece of information for the device to perceive their surroundings and make control decisions. However, this distance information is missing in the process of image formation and cannot be directly obtained from the images. Visual distance measurement algorithms fall primarily into two categories: depth estimation methods and geometric model-based methods. Depth estimation methods rely on feature point matching to acquire depth information, but they consume a significant amount of time, making it challenging to meet real-time requirements. Moreover, the deployment process of such methods can be complex and relatively expensive. In contrast, geometric model-based methods feature low cost, simple structures, and no need for data fusion, and can achieve real-time performance. However, distance measurement accuracy is highly dependent on camera parameters and target detection precision.

2.1 Ranging algorithm based on depth estimation

Depth information in scenes plays an extremely crucial role in various computer vision tasks, and it is typically obtained through active depth sensing. Active depth sensing involves the use of mechanical devices to acquire depth information from the scene, with the most common device being laser radar[14]. Traditional laser radar devices obtain depth information by reflecting laser beams off surfaces, enabling rapid surface model construction and scene depth estimation based on the acquired depth information[15]. However, these laser radar devices are expensive[16], and they consume significant energy to obtain dense and high-precision depth information, limiting their applicability in specialized scenarios. As industrial technology continues to advance, lower-cost and more user-friendly depth cameras have emerged, providing researchers with more options. Depth cameras, based on the principle of time-of-flight (TOF) using reflected near-infrared light, directly capture depth information from the scene. Additionally, these depth cameras often include RGB cameras, which allows for applications that require both depth and color information[17]. However, as research in this area deepens, applications based on depth cameras have revealed some challenges. Due to the limited range of depth measurement components and specific environmental requirements, they are not suitable for widespread outdoor use.

2.2 Ranging algorithm based on geometric Model

Real-time distance measurement of targets is a crucial component of floating touch. During the image formation process, the projection of the three-dimensional world onto a

two-dimensional image plane results in the loss of distance information. Therefore, it is essential to recover the original distance values from the images. Among these methods, geometric model-based distance measurement stands out because of its cost-effectiveness and simple model structure, making it more readily applicable in real-world production scenarios.

Wu et al.[18] initially locate the position of the target in the image, extract candidate regions, and enhance the accuracy of distance measurement by considering the position of the target's bottom shadow. However, the complex shape of hand shadows can still pose challenges in ranging accuracy. Liu et al.[19]combine images captured by PTZ cameras with data from millimeter-wave radar to perform detection. They also provide a detailed explanation of the coordinate transformation relationship between the sensors. Raza et al.[20] employ straightforward mathematical estimation methods to automatically measure the distance and dimensions (height and width) of targets located far from the camera. They achieve this by detecting L-shaped markers and storing their corner points. This method is suitable for stationary monocular camera environments. Stein et al.[21] introduce the foundational model of the similar triangle distance measurement algorithm and discuss the impact of pixel errors on distance measurement accuracy. However, this model does not consider the influence of camera pose angles on distance measurement. Subsequently, Liu et al.[22] extended this model by incorporating the impact of specific camera pose angles on distance measurement. However, their analysis primarily focuses on the precision issues caused by the camera's pitch angle and does not comprehensively analyze the impact of camera pose and lens structure on distance measurement.

In conclusion, it is evident that depth estimation-based methods struggle to effectively handle complex real-world scenarios, and their generalization capabilities are limited, often performing well only on the training dataset. However,geometric model-based methods not only need to account for factors like camera pose and environmental lighting but also need to extract relevant features from the images. This paper employs a geometric model-based distance measurement approach to achieve monocular floating touch tasks.

3 Proposed Approach

This paper presents a real-time 2D hand pose estimation model built upon Google's Mediapipe[23] framework, where machine learning (ML) is adopted for pose estimation. This method consists of two stages: (1) the input image is processed to segment the hand region from the background, followed by hand localization using detection anchor boxes; (2) the detected hand anchor boxes in the image are taken as input, and the regression model accurately localizes and dynamically updates the 2D coordinates of the 21 hand keypoints in real time.

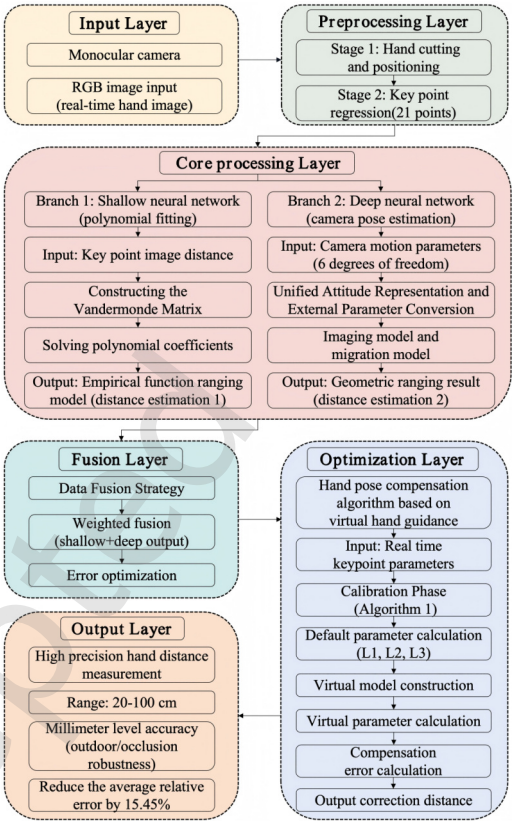

Fig. 1 System architecture of floating touch technology

Building upon this framework, the paper constructs a floating touch method based on monocular vision (as shown in Fig. 1) that integrates multiple algorithms. The system architecture comprises five interconnected layers and one output layer. The input layer captures real-time RGB images of the hand using a monocular vision camera. In the preprocessing layer, a two-stage 2D hand detection framework is implemented to extract and localize the 2D coordinates of the 21 hand keypoints in real time. The core processing layer employs a multi-stage neural network architecture to calculate the hand-to-camera distance through two parallel pathways: (1) regression-based ranging and (2) geometry-based ranging. A weighted data fusion mechanism integrates these two outputs at the data level. After the fusion layer, the optimization layer applies hand pose calibration and virtual model reconstruction to refine parameters and minimize measurement errors. The optimized results are transmitted to the output layer, which generates high-precision hand distance measurements.

3.1 Constructing a Shallow Neural Network based on function fitting

In this section, a shallow neural network (SNN) distance measurement model is constructed via polynomial curve fitting, where the least squares method is adopted to fit univariate polynomial curves. This involves constructing a Vandermonde matrix from the input features of the sample set, which transforms the nonlinear regression problem of a univariate *N*-th degree polynomial into a linear regression problem with *N*+1 elements (including the zeroth-order term).

Given a set of sample points $P(x, y)$, each point $P_i(x_i, y_i)$ $(i = 1, 2, \ldots, m)$ in $P(x, y)$ is obtained from multiple sampling runs of the polynomial defined in Eq. (1).

$$f(x_i) = \theta_0 + \theta_1 x_i + \theta_2 x_i^2 + \cdots + \theta_n x_i^n \tag{1}$$

where:

m: Number of sample points;

n: Order of the polynomial;

θ_j: Coefficients of the polynomial terms, where $j = 0, 1, 2, \ldots, n$ (with θ_0 denoting the zeroth-order term);

x_i: Image-plane distance calculated from the 2D keypoint coordinates of the hand in the image at time i;

y_i: Actual physical distance (in cm/mm) between the hand and the camera lens at time i.

The sum of squared errors for each data point within the sample dataset $P(x, y)$ is given by:

$$S = \sum_{i=1}^{m} [f(x_i) - y_i]^2 \tag{2}$$

Using Eq. (2), we can solve the coefficients θ_j $(j = 0, 1, 2, \ldots, n)$ of the optimal fitting function such that the sum of squared errors S is minimized.

In the algebraic approach, constructing matrices X and Y can be tedious and computationally expensive. To address this issue, this article reformulates the sum of squared errors S in matrix form. We first define the Vandermonde matrix X_v, coefficient vector θ, and the sample output vector Y_r as follows:

$$X_v = \begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^n \\ 1 & x_2 & x_2^2 & \ldots & x_2^n \\ 1 & x_3 & x_3^2 & \ldots & x_3^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \ldots & x_m^n \end{bmatrix}, \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}, \quad Y_r = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix} \tag{3}$$

Consequently, the sum of squared errors S can be rewritten in matrix form as:

$$S = (X_v\theta - Y_r)^T (X_v\theta - Y_r) \tag{4}$$

Here, X_v denotes the Vandermonde matrix, θ represents the coefficient vector consisting of polynomial coefficients, and Y_r is the output vector of the sample dataset. For the optimal fitting function, the first-order partial derivative of S with respect to θ must be zero, which yields:

$$\begin{aligned} \frac{\partial S}{\partial \theta} &= \frac{\partial \left[(X_v\theta - Y_r)^T (X_v\theta - Y_r) \right]}{\partial \theta} \\ &= 2X_v^T X_v \theta - 2X_v^T Y_r \\ &= 0 \end{aligned} \tag{5}$$

After simplifying the above equation (dividing both sides by 2), the polynomial coefficient vector θ of the optimal fitting function can be derived as:

$$\theta = \left(X_v^T X_v\right)^{-1} X_v^T Y_r \tag{6}$$

By substituting the obtained coefficient vector θ into the original polynomial (Eq. (1)), we derive an empirical function that approximates the ranging model, thus constructing a shallow ranging neural network.

3.2 Constructing a Deep Neural Network based on camera pose for distance Measurement

The motion of a camera can be regarded as rigid body movement in three-dimensional space. If a coordinate system is established based on the camera's initial pose, and the camera's pose at a later time is determined, the camera's pose after movement can be represented as a combination of rotation and translation relative to its initial pose. Therefore, the camera's pose change can be decomposed into six degrees of freedom, consisting of three rotational degrees of freedom and three translational degrees of freedom. The rotation component represents the camera's orientation in space and can be expressed in various forms, including Euler angles, rotation matrices, quaternions, etc. The translation component represents the camera's position in space, typically denoted as the x, y, and z coordinates of the translation vector.

During the experimental process, switching between different rotation representation modes can lead to deviations in dimension-related data, making it difficult to accurately describe the camera's rotational pose. To address this issue, this paper first uses Euler angles to describe the camera's pose. As shown in Fig. 2, Euler angles are converted into rotation matrices to unify the camera's pose into an extrinsic rotation representation. Finally, the Euler angles under the rotational pose are derived, and a deep neural network (DNN) is constructed by incorporating the camera's intrinsic and extrinsic parameters to sequentially calculate the x and y components of the camera's translation vector.

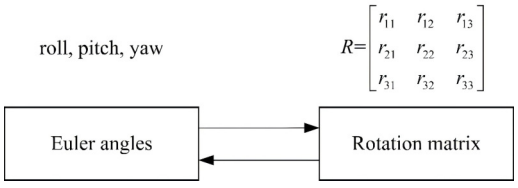

Fig. 2 Mutual conversion between Euler angles and rotation matrices

When describing rotational poses using Euler angles, three key elements are required: the rotation angles (α, β, γ), the rotation sequence, and the choice between intrinsic and extrinsic rotation. Two common methods for describing rotational poses using Euler angles are as follows:

(1) The rotation angles are specified as (α, β, γ), the rotation order is z-y-x, and the rotation mode is intrinsic. This convention is commonly referred to as the yaw-pitch-roll sequence.

(2) The rotation angles are specified as (α, β, γ), the rotation order is x-y-z, and the rotation mode is extrinsic. This convention is commonly referred to as the roll-pitch-yaw sequence.

Given the requirements of strict adherence to the rotation order and the choice of intrinsic/extrinsic rotation, Euler angles used for camera pose description must strictly comply with these conventions. This paper proposes a method to unify camera pose variations by converting intrinsic rotations to extrinsic rotations, enabling the extraction of camera Euler angles for subsequent distance measurement calculations.

In the context of intrinsic-extrinsic rotation transformation, Euler angles are well-suited for intuitively representing absolute camera poses in 3D space. However, they are unsuitable for scenarios that require relative pose calculations (e.g., pose interpolation or pose increment computation). To address this limitation, this paper adopts rotation matrices to normalize Euler angle representations.

$$
\begin{aligned}
R_{zyx}^{in} &= R_z(\gamma)R_y(\beta)R_x(\alpha) \\
&= \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \\
&\quad \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \\
&= \begin{bmatrix} \cos\gamma\cos\beta & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & \cos\gamma\sin\beta\cos\alpha + \sin\gamma\sin\alpha \\ \sin\gamma\cos\beta & \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha & \sin\gamma\sin\beta\cos\alpha - \cos\gamma\sin\alpha \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \\
&= \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}
\end{aligned} \tag{7}
$$

where R_{zyx}^{in} denotes the intrinsic rotation matrix following the z-y-x (yaw-pitch-roll) rotation order, and r_{ij} $(i, j = 1, 2, 3)$ represent the elements of the 3×3 rotation matrix.

$$
\begin{aligned}
R_{xyz}^{out} &= R_x(\alpha)R_y(\beta)R_z(\gamma) \\
&= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \\
&\quad \cdot \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} r_{11} & r_{21} & r_{31} \\ r_{12} & r_{22} & r_{32} \\ r_{13} & r_{23} & r_{33} \end{bmatrix}
\end{aligned} \tag{8}
$$

where R_{xyz}^{out} denotes the extrinsic rotation matrix following the x-y-z (roll-pitch-yaw) rotation order, which is the transpose of the intrinsic rotation matrix R_{zyx}^{in}.

According to Eqs. (7) and (8), the rotation matrix R_{zyx}^{in} (representing sequential intrinsic rotations about the z-y-x axes) describes the sensor's orientation in the camera coordinate system, while the output rotation matrix R_{xyz}^{out} represents transformations in the world coordinate system (extrinsic rotations in x-y-z order). The superscripts 'in' and 'out' denote the camera and world reference frames, respectively, and the subscripts indicate the Euler angle rotation sequence about the three coordinate axes. Successive extrinsic rotations by angles α, β, and γ in the x-y-z order are equivalent to sequential intrinsic rotations by angles γ, β, and α in the z-y-x order[24].

We thus unify the camera pose by converting it to a Euler angle-based rotational representation. By fixing the pitch angle (α), we sequentially calculate the offsets of the measured hand along the y-axis and x-axis induced by the camera's Euler angles. The imaging model for y-axis offset is illustrated in Fig. 3.

As shown in Fig. 3, point e denotes the position of the measured object C on the image plane. The camera's extrinsic parameters are known: α represents the pitch angle, and H is the camera height. The camera's intrinsic parameters include the focal length f (the distance between point o and point o_1), and d (the distance between point o_2 and point A) denotes the desired y-axis offset.

The angle ξ between line segment oA and o_2A represents the lens's angular offset along the y-axis, and the total angular offset is $\alpha + \varphi$ (where φ is the angular displacement from the image plane offset). The relationship between H and d is

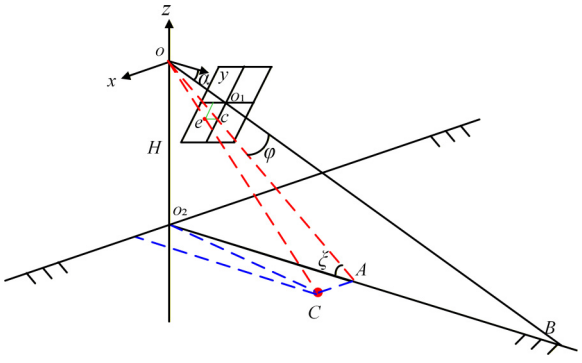

Fig. 3 Imaging model based on y-axis offset

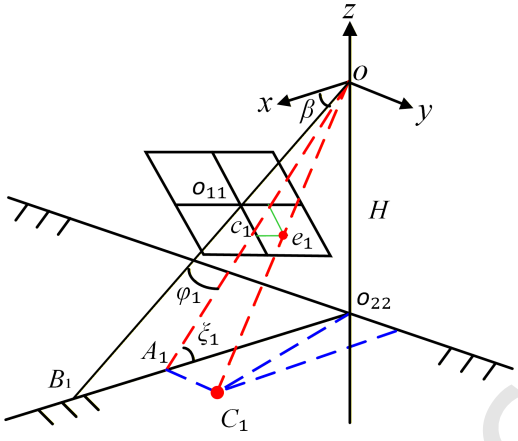

Fig. 4 Imaging model based on x-axis offset

derived as:

$$\tan(\alpha+\varphi)=\frac{H}{d}\Rightarrow d=\frac{H}{\tan(\alpha+\varphi)} \tag{9}$$

The distance h between point o_1 and point c is the offset distance of the object on the image plane, and the angular displacement φ corresponding to this offset is calculated as:

$$\tan\varphi=\frac{h}{f}\Rightarrow\varphi=\arctan\left(\frac{h}{f}\right) \tag{10}$$

Here, φ is the angle between oc and oo_1 at point o, corresponding to the image plane offset distance h.

In summary, for extrinsic rotations with angles (α,β,γ) in x-y-z (roll-pitch-yaw) order, the offset of the target along the camera's y-axis (denoted as d) is:

$$d=\frac{H}{\tan\left(\alpha+\arctan\left(\frac{h}{f}\right)\right)} \tag{11}$$

As shown in Fig. 4, point e_1 denotes the position of the measured object C_1 on the image plane. The camera's extrinsic parameters are known: β represents the yaw angle, and H is the camera height. The camera's intrinsic parameters include the focal length f (the distance between point o and point o_{11}), and d_1 (the distance between point o_{22} and point A_1) denotes the desired x-axis offset.

The angle ξ_1 between line segment oA_1 and $o_{22}A_1$ represents the lens's angular offset along the x-axis, and the total angular offset is $\beta+\varphi_1$ (where φ_1 is the angular displacement induced by the image plane offset). The relationship between H and d_1 is derived as:

$$\tan(\beta+\varphi_1)=\frac{H}{d_1}\Rightarrow d_1=\frac{H}{\tan(\beta+\varphi_1)} \tag{12}$$

The distance h_1 between point o_{11} and point c_1 is the offset distance of the object C_1 on the image plane, and the corresponding angular displacement φ_1 is calculated as:

$$\tan\varphi_1=\frac{h_1}{f}\Rightarrow\varphi_1=\arctan\left(\frac{h_1}{f}\right) \tag{13}$$

Here, φ_1 is the angle between line segment oc_1 and oo_{11} at point o, corresponding to the image plane offset distance h_1.

In summary, for extrinsic rotations with angles (α,β,γ) in the x-y-z (roll-pitch-yaw) order (extrinsic rotation convention), the offset of the target along the camera's x-axis (denoted as d_1) is given by:

$$d_1=\frac{H}{\tan\left(\beta+\arctan\left(\frac{h_1}{f}\right)\right)} \tag{14}$$

3.3 Hand Pose Compensation Algorithm based on virtual hand guidance

After constructing a multi-layer network by dynamically adjusting the contribution weight of shallow and deep networks based on hand detection confidence, this section focuses on compensating for pose-related design parameters. We first calibrate the hand pose to obtain default parameters for virtual hand model construction.

Algorithm 1 Hand pose calibration algorithm

Function: Pose calibration, extracting default parameters for users' hands
Input: RGB image
Output: Coordinate parameters of 21 hand key points
while Confidence level >0.5 **do**
  Extract target hand anchor box
  if The center point of the anchor frame coincides with the center point of the 2D coordinate system **then**
    Record the coordinate parameters of 21 hand key points
  else
    continue
  end if
end while

Using the pose calibration results, we obtain the 2D coordinates of 21 hand keypoints. As shown in Fig. 5, three

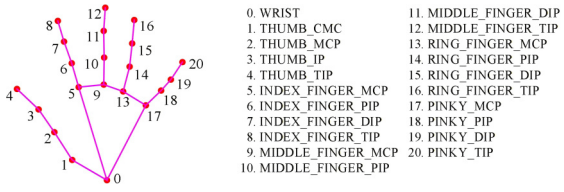

Fig. 5 Schematic diagram of hand keypoints (palm keypoints 0, 5, 17 highlighted)

palm-region keypoints (indexed as 0, 5, and 17) are selected. A distance measurement algorithm is used to calculate the lengths between keypoint 5 and 17 (σ), between keypoint 0 and 17 (μ), and between keypoint 0 and 5 (η), respectively.

These actual lengths and the angles between adjacent finger joints are defined as default parameters, which are then input into the compensation algorithm for the virtual hand model.

The compensation algorithm for the virtual hand model proceeds as follows:

(1) Input real-time 2D hand coordinate parameters (from the hand tracking network) and distance parameters (from the multi-layer ranging network).

(2) Calculate the lengths between real-time tracking keypoints (5-17, 0-17, 0-5), multiply each length by its corresponding scaling factor, and select the maximum length as the reference edge. Construct a virtual hand model in the canonical pose (neutral, fully extended palm) using this reference edge.

(3) Compare the virtual hand model parameters with real-time hand parameters to compute the required distance compensation. Weight this compensation and apply it to the distance measurement results from the multi-layer network to generate the final output.

Taking the edge between keypoint 17 and 0 in Fig. 5 as reference, Fig. 6 shows the visualization results of the virtual and real hand models. In this visualization, the blue solid line denotes the real-time detected real hand model, and the black solid line denotes the virtual hand model derived from the real hand model. Let A, B, and C represent the pixel lengths of the real-time detected hand model between keypoint 5 and 17, keypoint 0 and 17, and keypoint 0 and 5, respectively. Similarly, a, b, and c represent the pixel lengths of the virtual hand model (black solid line) between keypoint 5 and 17, keypoint 0 and 17, and keypoint 0 and 5, respectively.

First, each of A, B, and C is multiplied by its scaling factor (normalized by real physical lengths), and the maximum value among the resulting values is selected as the reference edge. The reference edge $\mathcal{E}$ is calculated as:

$$\mathcal{E} = \max\left(\frac{A\sigma}{L_{\max}}, \frac{B\mu}{L_{\max}}, \frac{C\eta}{L_{\max}}\right), \quad L_{\max} = \max(\sigma, \mu, \eta) \tag{15}$$

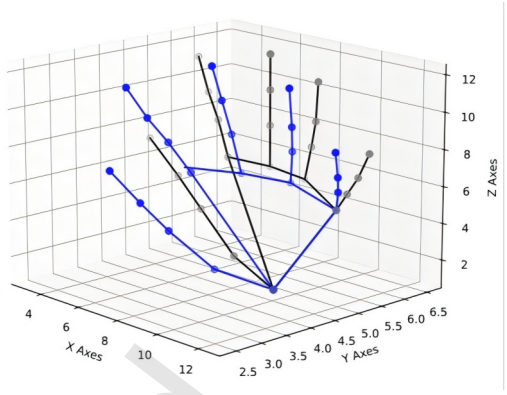

Fig. 6 Visual comparison between virtual and real hand shapes

where $L_{\max}$ denotes the maximum value of the real physical lengths σ, μ, and η.

Three distinct cases are considered based on the value of $\mathcal{E}$:

Case 1: $\mathcal{E} = \frac{A\sigma}{L_{\max}}$ (reference edge is keypoint 5-17)

$$\mathcal{E} = \frac{A\sigma}{L_{\max}} \tag{16}$$

In this case, the line segment between keypoints 5 and 17 is selected as the reference edge, so we set $a = A$. Using the calibrated default parameters, b and c are solved as:

$$\begin{cases} \frac{a}{\sigma} = \frac{b}{\mu} \Rightarrow b = \frac{a\mu}{\sigma} \\ \frac{a}{\sigma} = \frac{c}{\eta} \Rightarrow c = \frac{a\eta}{\sigma} \end{cases} \tag{17}$$

The hand ranging compensation distance ω is calculated by weighting the ratio of virtual-to-real lengths for the non-reference edges, multiplied by the real length of the reference edge (σ):

$$\omega = \sigma \cdot \frac{1}{2}\left(\frac{b}{B} + \frac{c}{C}\right) \tag{18}$$

Case 2: $\mathcal{E} = \frac{B\mu}{L_{\max}}$ (reference edge is keypoint 17-0)

$$\mathcal{E} = \frac{B\mu}{L_{\max}} \tag{19}$$

In this case, the line segment between keypoints 17 and 0 is selected as the reference edge, so we set $b = B$. Using the calibrated default parameters, a is solved as:

$$\begin{cases} \frac{b}{\mu} = \frac{a}{\sigma} \Rightarrow a = \frac{b\sigma}{\mu} \\ \frac{b}{\mu} = \frac{c}{\eta} \Rightarrow c = \frac{b\eta}{\mu} \end{cases} \tag{20}$$

The compensation distance ω is calculated as:

$$\omega = \mu \cdot \frac{1}{2}\left(\frac{a}{A} + \frac{c}{C}\right) \tag{21}$$

Case 3: $\mathcal{E} = \frac{C\eta}{L_{\max}}$ (reference edge is keypoint 0-5)

$$\mathcal{E} = \frac{C\eta}{L_{\max}} \tag{22}$$

In this case, the line segment between keypoints 0 and 5

is selected as the reference edge, so we set $c = C$. Using the calibrated default parameters, a and b are solved as:

$$\begin{cases} \frac{c}{\eta} = \frac{a}{\sigma} \Rightarrow a = \frac{c\sigma}{\eta} \\ \frac{c}{\eta} = \frac{b}{\mu} \Rightarrow b = \frac{c\mu}{\eta} \end{cases} \tag{23}$$

The compensation distance ω is calculated as:

$$\omega = \eta \cdot \frac{1}{2}\left(\frac{a}{A} + \frac{b}{B}\right) \tag{24}$$

4 Experiment and Results

4.1 Dataset and environment introduction

The hand training dataset consists of indoor, outdoor, and synthetic subsets. The outdoor dataset comprises 6,000 images of different hand types, covering various outdoor backgrounds, lighting conditions, and hand appearances. The indoor dataset includes 10,000 images with hand poses at various physical angles. The synthetic dataset is generated by rendering a high-quality hand model with 24 skeletal bones and textures for five different skin tones; video sequences of hand pose transformations are created from this model, and 100,000 images are extracted for network training.

Training and testing were conducted on a local platform equipped with an RTX 2080Ti 11G GPU, an i7 9700 CPU, and 32GB RAM, using the TensorFlow framework. For data collection, an Intel RealSense D435 depth camera and a laser rangefinder were utilized.

4.2 Performance evaluation and discussion of the hand tracking module

In the detection stage, this study adopts the hand tracking strategy based on MediaPipe[23]. By providing cropped hand images to the hand landmark module, the model reduces computational time and the workload of the graphics processing unit (GPU).

During training, incorporating a large synthetic dataset into the hand tracking module training reduces visual jitter between video frames, thereby mitigating the impact of vibrations during recording. Mixed training with real-world and synthetic datasets improves accuracy and effectively accomplishes the pre-detection task.

In the performance tuning phase, Table 1 presents the performance parameters for different model sizes. Increasing the model size of the hand tracking module yields only marginal performance improvements while significantly reducing inference speed; thus, the full-size hand tracking module is adopted in this paper.

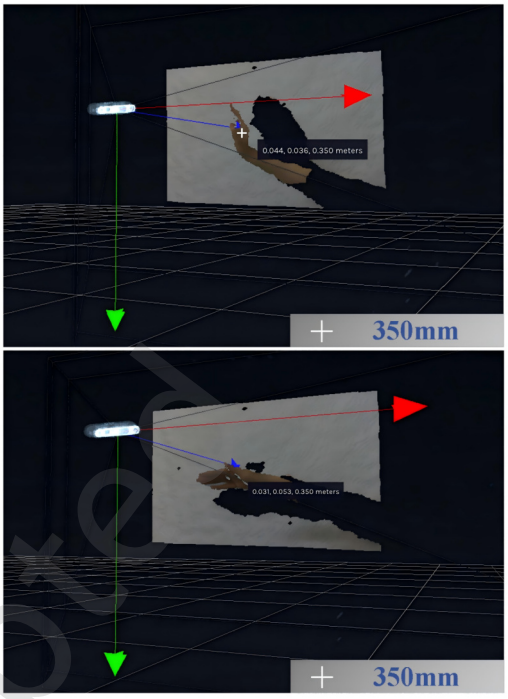

Fig. 7 3D visualization of actual distance

4.3 Experimental results and discussion of a floating touch method based on monocular vision estimation

After constructing the detection module for hand tracking, to evaluate the accuracy of the hand distance measurement algorithm, this experiment collected hand distance data in the range of 0.2 to 1.0 meters. The actual distance from the camera lens to the hand was obtained using the depth sensor of the Intel RealSense D435 stereo depth camera. Additionally, image data were collected via the camera's monocular RGB lens, and hand distance measurement data were acquired using the image processing methods proposed in this paper.

The Real-time data collected by the stereo depth camera is shown in Fig. 7. This paper achieves real-time fusion and rendering of 3D hand depth data with 2D hand image data, enabling visual 3D visualization of the actual hand distance and ensuring the reliability of real-time actual hand distance data collection.

During measurement data collection, instability of the detector's palm anchor box can cause fluctuations in the measured distance data. Thus, the Mean Relative Error (MRE) is calculated by averaging the error for each position in a video stream, as defined in Eq. (25):

$$\text{MRE} = \frac{1}{k}\sum_{f=1}^{k}\frac{|pd_f - rd_f|}{rd_f} \times 100\% \tag{25}$$

where k denotes the number of video frames, $f \in$

Table 1 Hand landmark model performance

Model	Parameters(M)	MSE	Time(ms) Pixel 3	Time(ms) Android 9	Time(ms) iOS 11
Light	1	11.83	6.6	5.6	1.1
Full	1.98	10.05	16.1	11.1	5.3
Heavy	4.02	9.817	36.9	25.8	7.5

$\{1, 2, \ldots, k\}$, pd_f is the predicted distance value, and rd_f is the actual distance value. The final MRE is obtained by accumulating per-frame errors for each position and computing the mean.

For comparison, we adopt the distance measurement models from reference [21] and [22] (with measurement methods similar to our experiment) as baselines. Table 2 presents the experimental results of different models: Model (1) uses the Traditional Triangular Ranging Algorithm, and Model (2) combines the empirical function fitting method with the Traditional Triangular Ranging Algorithm.

Comparing the results of Model (1) and Model (2), we observe that the mean relative error (MRE) of the traditional triangulation method proposed in reference [21] and [22] increases with the rise in actual distance. In contrast, the method proposed in this study achieves significant improvements: compared with the baseline method in reference [21] and [22], our distance measurement algorithm reduces the total MRE by 8.72 percentage points, with respective reductions by 9.53 and 10.48 percentage points in indoor and outdoor environments.

We analyze the limitations of shallow and deep networks individually for hand distance measurement: For shallow networks, fast variations in hand pose introduce errors in the output of the detection module, leading to data oscillations during empirical function fitting for interpolation. These oscillations cause unstable interpolation results, significantly degrading distance measurement accuracy. For deep networks, the accuracy of the Triangular Ranging Algorithm (TRA) is constrained by baseline length: a shorter baseline yields smaller measurement errors but a narrower measurement range, while a longer baseline expands the range at the cost of larger errors. Thus, the traditional TRA suffers from an inherent trade-off between baseline length and measurement accuracy, which contributes to distance measurement errors.

To address these issues, we propose a method for combining shallow and deep networks for hand distance measurement. By aggregating the inputs and outputs of multiple network layers and performing coefficient-weighted fusion on the outputs of shallow and deep networks, we obtain the final result of distance measurement. This approach not only eliminates abnormal output fluctuations caused by empirical function oscillations in shallow networks, but also substantially reduces long-baseline errors associated with extended measurement ranges.As shown in Table 2, Model (2) outperforms Model (1) in distance measurement precision as the actual distance increases, making it more suitable for performing hand distance measurement tasks across various distances.

4.4 Distance measurement results and discussion based on camera pose estimation algorithm

To validate the distance measurement algorithm based on camera pose estimation, we designed Model (3), which integrates the empirical function fitting method with the camera pose estimation-based ranging algorithm. The experimental results are presented in Table 3.

As shown in Table 2 and Table 3, the measurement performance of Model (2) and Model (3) is significantly superior to that of Model (1). By replacing the Traditional Triangular Ranging Algorithm (TRA) in Model (2) with our camera pose estimation ranging algorithm (Model (3)), the average mean relative error (MRE) is reduced by 4.96 percentage points. Notably, the error does not increase significantly with longer actual distances, primarily due to the mitigation of camera pose deviations and lens distortion effects.

Regarding camera pose deviations, the traditional TRA only measures the longitudinal distance of the hand. When the camera pose is deviated (e.g., the hand is not centered in the image), this method loses distance information in other directions. In contrast, our camera pose estimation-based ranging algorithm not only measures the longitudinal hand distance but also fuses multi-directional hand distance measurements based on camera pose. This approach effectively mitigates error cancellation and accumulation, alleviates the impact of camera pose deviations, and improves ranging accuracy.

In terms of lens optical distortion, drastic camera pose changes during dynamic hand image capture amplify lens optical distortion. If the detected hand is located at the edge of image, geometric distortion becomes more pronounced. The traditional TRA is suitable only for flat or near-flat surfaces and cannot handle varying magnification rates across the lens plane. Our camera pose estimation-based algorithm unifies the

Table 2 Comparison of experimental results between model (1) and model (2)

Model Name	Real Distance(cm)	Indoor Measured(cm)	Outdoor Measured(cm)	No Occlusion Measured(cm)	Occlusion Measured(cm)	MRE
Model (1) Traditional Triangular Ranging Algorithm [21,22]	20	21.55	21.80	21.47	22.95	9.71%
	30	32.39	31.72	32.84	35.19	10.12%
	40	43.59	44.39	44.12	44.69	10.49%
	50	55.94	56.87	56.41	58.64	13.93%
	60	68.11	69.93	67.53	72.01	15.66%
	70	80.74	82.61	80.23	86.34	17.83%
	80	92.94	95.13	93.45	99.14	18.96%
	90	107.78	109.60	106.27	115.08	21.87%
	100	121.62	124.37	123.10	130.32	24.85%
Model (1) Average MRE	-	13.66%	15.45%	13.90%	20.73%	15.94%
Model (2) Empirical Function Fitting + Traditional Triangular Ranging	20	21.19	20.91	20.85	21.85	6.00%
	30	32.20	31.83	31.96	33.89	8.23%
	40	38.91	41.86	42.35	44.29	5.99%
	50	48.67	52.54	52.64	57.24	6.88%
	60	61.68	56.12	62.21	69.51	7.20%
	70	72.73	67.78	68.64	81.54	6.38%
	80	82.58	76.32	81.94	93.54	6.79%
	90	93.92	85.89	93.28	108.68	8.33%
	100	95.79	94.43	104.92	122.02	9.18%
Model (2) Average MRE	-	4.13%	4.97%	4.28%	15.50%	7.22%

correction of camera pose variations by estimating the current camera pose from multi-directional lens offsets, enabling image distortion correction and more accurate hand distance calculation.

4.5 Data results and comparative analysis with hand pose compensation algorithm

Model (3) achieved improved overall performance via camera pose calibration, but complex hand poses still degraded its accuracy and robustness. To address this, we visualized real-time data to analyze the issue: Fig. 8 shows distance measurement results of Model (3) under two complex hand poses. In each subfigure, the left panel displays the actual distance (measured by a stereo depth camera from the palm center to the camera), while the right panel shows the distance measured by Model (3) using an RGB camera. Significant discrepancies between actual and measured distances are observed, leading to non-negligible measurement errors.

To address this issue, this paper employs a hand pose compensation algorithm to tackle the problem of distance measurement under complex hand poses. Firstly, the hand pose calibration algorithm is used to mitigate calculation errors caused by variations in hand size. Then, a virtual hand model with the correct pose is created based on the current hand position to calculate compensation and rectify the significant errors associated with complex hand poses.

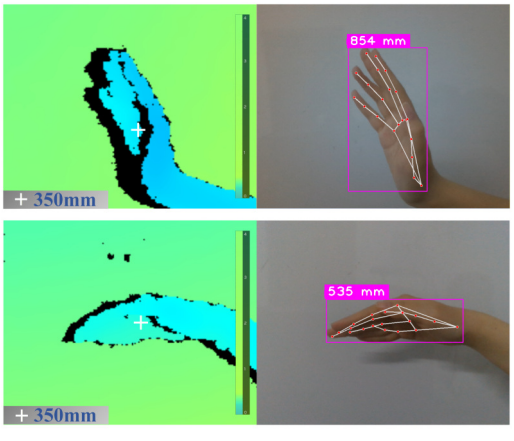

Fig. 8 Distance measurement results of model (3) under two complex hand poses

Table 3 Results of hand Ranging in model (3) and model (4)

Model Name	Real Distance(cm)	Indoor Measured(cm)	Outdoor Measured(cm)	No Occlusion Measured(cm)	Occlusion Measured(cm)	MRE
Model (3) Empirical Function Fitting + Camera Pose Estimation Ranging	20	19.69	19.72	20.34	22.12	2.26%
	30	30.39	30.46	30.29	27.67	3.81%
	40	40.61	39.24	39.73	42.51	2.89%
	50	49.27	49.16	50.64	47.50	2.59%
	60	59.20	58.78	59.12	62.31	2.36%
	70	70.13	69.40	70.21	72.50	2.17%
	80	79.52	81.26	79.69	78.17	1.23%
	90	91.46	88.57	91.53	88.52	1.21%
	100	102.39	98.65	98.22	104.32	1.64%
Model (3) Average MRE	-	1.33%	1.55%	1.14%	5.04%	2.26%
Model (4) Empirical Fitting + Camera Pose Estimation + Hand Pose Compensation Algorithm	20	20.07	19.78	20.21	20.02	0.65%
	30	30.11	30.49	29.82	30.13	0.75%
	40	40.23	40.13	40.12	39.81	0.41%
	50	49.71	50.42	50.54	49.77	0.74%
	60	59.82	59.58	60.21	60.46	0.52%
	70	69.73	70.52	70.10	70.34	0.43%
	80	80.40	79.63	79.97	80.49	0.40%
	90	90.23	90.34	90.12	90.28	0.26%
	100	99.89	100.33	100.27	100.13	0.21%
Model (4) Average MRE	-	0.38%	0.72%	0.44%	0.42%	0.49%

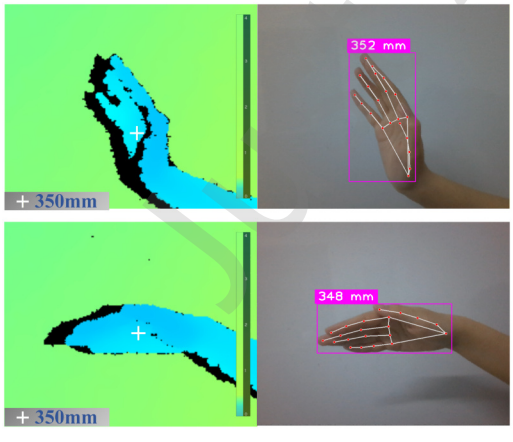

Fig. 9 Distance measurement results of Model (4) (with pose compensation) under complex hand poses

Fig. 9 illustrates the visualization results of Model (4) (incorporating the pose compensation algorithm), which outperforms Model (3) significantly. This method further enhances the accuracy and robustness of hand distance estimation, enabling reliable floating touch tasks. To validate the effectiveness of the pose compensation algorithm, we compare results before and after integration (Table 3).

As shown in Table 3, Model (4) reduces the average distance error by 1.77 percentage points compared to Model (3), and the occlusion error by 4.62 percentage points.Overall, Model (4) reduces the average MRE by 15.45 percentage points compared to Model (1), effectively improving ranging accuracy under occlusion.

Model (3) (shallow-deep network fusion) overlooks hand size variations, which accumulate errors under complex hand poses. In contrast, Model (4) introduces a hand pose compensation algorithm that calibrates virtual hand model parameters based on hand size and length, mitigating errors from neglected hand size variations.

When using only the deep and shallow network-based measurement method in Model (3), the system tends to overlook the issue of hand size, especially under complex hand poses, as hands naturally have varying sizes. With an increasing prevalence of complex hand poses during usage, this problem leads to accumulating errors, that significantly affect the overall precision and robustness of the method. In

Table 4 Comparison of the proposed method and monocular depth estimation

Model Name	Real Distance(cm)	Indoor Measured(cm)	Outdoor Measured(cm)	No Occlusion Measured(cm)	Occlusion Measured(cm)	MRE
Model (4) Empirical Fitting + Camera Pose Estimation + Hand Pose Compensation Algorithm	20	20.07	19.78	20.21	20.02	0.65%
	30	30.11	30.49	29.82	30.13	0.75%
	40	40.23	40.13	40.12	39.81	0.41%
	50	49.71	50.42	50.54	49.77	0.74%
	60	59.82	59.58	60.21	60.46	0.52%
	70	69.73	70.52	70.10	70.34	0.43%
	80	80.40	79.63	79.97	80.49	0.40%
	90	90.23	90.34	90.12	90.28	0.26%
	100	99.89	100.33	100.27	100.13	0.21%
Model (4) Average MRE	-	0.38%	0.72%	0.44%	0.42%	0.49%
BTS [25]	20	20.03	22.05	19.92	18.11	5.06%
	30	30.00	28.42	30.21	31.44	2.69%
	40	40.24	37.62	40.15	44.58	4.59%
	50	49.23	52.18	50.62	52.49	3.03%
	60	60.77	63.24	59.47	64.38	3.71%
	70	70.92	71.45	70.64	67.82	1.85%
	80	80.58	89.33	80.53	73.64	5.25%
	90	89.42	107.56	90.97	97.92	7.50%
	100	99.21	124.43	98.94	112.43	9.68%
BTS [25] Average MRE	-	0.78%	9.88%	0.91%	7.81%	4.82%

contrast, Model (4) introduces the hand pose compensation algorithm, which takes into account the length and size of the hand to determine the default parameters for the virtual hand model. This effectively mitigates errors arising from neglecting the hand's inherent size when measuring complex hand poses. We further prune and quantize Model (4) to achieve lightweight deployment on edge devices, ensuring stable detection results and evenly distributed errors across different positions.

4.6 Comparative analysis of the proposed method and monocular depth estimation

We conducted comparative experiments between our method and monocular depth estimation methods in a monocular camera setup. Depth estimation methods are broadly classified into relative depth estimation and metric depth estimation. Relative depth is represented by a grayscale image, which indicates the relative distance between pixels and is not the real depth value. Typically quantified between 0-255, the higher the depth value, the farther the distance, resulting in darker colors. In contrast, metric depth is represented by RGB color images, where pixel values directly correspond to the true distance from the camera, usually measured in meters.

Since floating touch tasks require absolute distance values, we selected BTS [25] (a representative monocular metric depth estimation algorithm) for comparison. The results are presented in Table 4.

As shown in Table 4, Model (4) outperforms BTS [25] by only 0.5 percentage points in indoor occlusion-free scenarios, but the performance gap reaches 9.16 and 7.39 percentage points in outdoor and occlusion scenarios, respectively. BTS exhibits significant errors (average MRE: 4.82%), while our Model (4) achieves an average MRE of only 0.49%.

We further compare the two methods in terms of deployment efficiency (Table 5). Our lightweight model shows significant advantages for mobile deployment, achieving real-time inference with low parameters, computational load, and memory usage; our high-precision model balances accuracy and resource consumption; while BTS incurs the highest resource costs, making it suitable only for offline desktop scenarios with sufficient computing power.

Table 5 Performance and resource consumption comparison

Metric	BTS	Paper	Paper (Android)
Parameters (M)	47.0	25.5	16.4
GFLOPs	20–25	10–15	0.6
Inference Time (ms)	100–200	80–150	33–67
Inference Memory (GB)	1.5–2.0	1.0–1.5	< 0.1

In outdoor environments, monocular depth estimation accuracy is degraded by complex hand poses, lighting, occlusion, algorithm design, and dataset quality (especially

cluttered backgrounds). Our method uses a high-precision hand keypoint extraction algorithm to mitigate environmental interference. In occlusion scenarios, BTS (global depth estimation) introduces redundant background information, while our method extracts hand feature regions and uses geometric-based distance compensation to reduce computational complexity and improve accuracy.

In summary, monocular depth estimation for hand ranging lacks theoretical underpinnings and relies heavily on the fitting capabilities of large-scale deep learning networks. While monocular depth estimation networks can achieve relatively accurate distance estimation in specific scenarios (e.g., indoor environments such as bedrooms and living rooms), they struggle to generalize to arbitrary scenarios. In contrast, our method—grounded in prior geometric knowledge and physical pose analysis—has a model size of only 2 MB and leverages lightweight image processing techniques compatible with low-power hardware, outperforming computationally intensive depth estimation algorithms in hand ranging accuracy.

Building on the aforementioned methods, we first propose a camera pose estimation-based hand ranging algorithm, followed by a hand pose compensation algorithm, and finally integrate a hand tracking algorithm into a multi-layer fusion network to construct a monocular vision-based floating touch system. Experimental results demonstrate that our method reduces the mean relative error (MRE) by 15.45 percentage points compared to traditional ranging methods, achieving an MRE of only 0.49% over distance range of 20–100 cm and thus allowing reliable hand ranging performance.

5 Conclusions

We propose a touchless control method based on a single camera and multiple algorithms to address key challenges in low-cost single-camera touchless control systems. Specifically, we develop a ranging algorithm based on camera pose estimation to construct a multi-layer fusion network, which resolves issues such as pose angle ambiguity during distance measurement; we also propose a pose compensation algorithm based on virtual hand modeling to mitigate numerical fluctuations in the ranging algorithm caused by drastic hand pose variations. Experimental results demonstrate that our proposed method significantly outperforms monocular metric depth estimation methods under outdoor and occlusion conditions, achieving sub-centimeter accuracy with a mean relative error (MRE) of only 0.49% over the 20–100 cm range, while maintaining an ultra-lightweight model architecture and high real-time inference performance. This work provides theoretical and practical guidance for the practical deployment and widespread adoption of touchless control technology.

Future work will focus on optimizing and extending the proposed method to broaden its applicability. This includes improving the algorithm's generalization ability across skin tones, lighting intensities, different levels of occlusion, and longer touch distances; enhancing dynamic hand tracking and gesture recognition to further advance the application of touchless control technology in augmented reality (AR), virtual reality (VR), and smart home systems. These optimizations will expand the application scenarios while improving the robustness and scalability of real-world deployments.

Acknowledgment

This work was supported in part by the Heilongjiang Provincial Natural Science Foundation (Research on Path Planning of Inspection Robots and Obstacle Avoidance of Manipulators in Unmarked Multi-track Scenarios), and the Basic Scientific Research Business Expenses Project of Provincial Universities in Heilongjiang Province (No. 2024-KYYWF-1102).

Reference

[1] S. Palipana, D. Salami, L.A. Leiva, et al. Pantomime: mid-air gesture recognition with sparse millimeter-wave radar point clouds, *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 5, no. 1, pp. 1–27, 2021.

[2] L. Wang, Z. Cui, et al. Low personality-sensitive feature learning for radar-based gesture recognition, *Neurocomputing*, vol. 493, pp. 373–384, 2022.

[3] D. Jiang, Z. Zheng, G. Li, et al. Gesture recognition based on binocular vision, *Cluster Computing*, vol. 22, no. 6, pp. 13261–13271, 2019.

[4] H. Zhou, D. Wang, Y. Yu, et al. Research progress of human-computer interaction technology based on gesture recognition, *Electronics*, vol. 12, no. 13, pp. 2805, 2023.

[5] X. Yao, Y. Pei. Study on human-machine interactive system based on binocular vision and depth camera, in *Proc. 5th Int. Conf. on Digital Signal Processing*, Chengdu, China, 2021, pp. 99–102.

[6] J. Zhu, H. Li, T. Zhang. Camera, LiDAR, and IMU based multi-sensor fusion SLAM: a survey, *Tsinghua Science and Technology*, vol. 29, no. 2, pp. 415–429, 2024.

[7] N. Nooruddin, R. Dembani, N. Maitlo. HGR: hand-gesture-recognition based text input method for AR/VR wearable devices, in *Proc. 2020 IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, Toronto, Canada, 2020, pp. 744–751.

[8] P. Mishra, K. Sarawadekar. Multiple-hand 2d pose estimation from a monocular rgb image, *IEEE Access*, 2024.

[9] X. Qu, M. Ye, W. Zhao. In-air handwriting system based on multi-scale channel attention network and monocular vision, *Applied Soft Computing*, vol. 162, pp. 111801, 2024.

[10] S. Wang, X. Li, Y. Zhang, et al. Error analysis of precision measurement with monocular vision, *Engineering Research Express*, vol. 5, no. 4, pp. 045066, 2023.
[11] S. Wu, Z. Li, W. Chen, et al. Dynamic modeling of robotic manipulator via an augmented deep Lagrangian network, *Tsinghua Science and Technology*, vol. 29, no. 5, pp. 1604–1614, 2024.
[12] C. Cao, N.S.N. Lam. Understanding the scale and resolution effects in remote sensing and GIS, in *Scale in Remote Sensing and GIS*, London: Routledge, 2023, pp. 57–72.
[13] A. Giatsintov, K. Mamrosenko, P. Bazhenov. Architecture of graphics system with 3D acceleration support for embedded operating systems, *Tsinghua Science and Technology*, vol. 29, no. 3, pp. 863–873, 2024.
[14] N. Lopac, I. Jurdana, A. Brnelić, et al. Application of laser systems for detection and ranging in the modern road transportation and maritime sector, *Sensors*, vol. 22, no. 16, pp. 5946, 2022.
[15] C. Qi, J. Yin, Z. Zhang, et al. Dynamic scene graph generation of point clouds with structural representation learning, *Tsinghua Science and Technology*, vol. 29, no. 1, pp. 232–243, 2024.
[16] K. Chen, B.T. Lopez, A. Agha-mohammadi, et al. Direct lidar odometry: fast localization with dense point clouds, *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2000–2007, 2022.
[17] R. Horaud, M. Hansard, G. Evangelidis, et al. An overview of depth cameras and range scanners based on time-of-flight technologies, *Machine Vision and Applications*, vol. 27, no. 7, pp. 1005–1020, 2016.
[18] J. Wu, W.J. Li, L. Geng, et al. Preceding vehicle detection and ranging based on monocular vision, (in Chinese), *Computer Engineering*, vol. 43, no. 2, pp. 26–32, 2017.
[19] Z.Q. Liu, Z.Y. Zhang, J. Ni, et al. Research on vehicle detection method of multi-sensor under dual dynamic condition, (in Chinese), *Machinery Design & Manufacture*, vol. 46, no. S2, pp. 6–10, 2018.
[20] M. Raza, C. Zonghai, S.U. Rehman, et al. Framework for estimating distance and dimension attributes of pedestrians in real-time environments using monocular camera, *Neurocomputing*, vol. 275, pp. 533–545, 2018.
[21] G.P. Stein, O. Mano, A. Shashua. Vision-based ACC with a single camera: bounds on range and range rate accuracy, in *Proc. Intelligent Vehicles Symposium*, Columbus, OH, USA, 2003, pp. 120–125.
[22] C.H. Liu, K. Shuai, W.R. Yang. Design of distance measurement system based on ARM embedded vision, (in Chinese), *Journal of Wuhan Institute of Technology*, vol. 37, no. 4, pp. 65–68, 2015.
[23] F. Zhang, V. Bazarevsky, A. Vakunov, et al. Mediapipe hands: on-device real-time hand tracking, arXiv preprint arXiv:2006.10214, 2020.
[24] M.K. Ozgoren. Comparative study of attitude control methods based on Euler angles, quaternions, angle-axis pairs and orientation matrices, *Transactions of the Institute of Measurement and Control*, vol. 41, no. 5, pp. 1189–1206, 2019.
[25] J.H. Lee, M.K. Han, D.W. Ko, et al. From big to small: multi-scale local planar guidance for monocular depth estimation, arXiv preprint arXiv:1907.10326, 2019.

Author biography

Liu Fugang received the PhD degree in communication and information system from Harbin Engineering University in 2013. He is a professor at Heilongjiang University of Science and Technology. His research interests include direction of arrival estimation of wide-band signals, wireless communication, and deep learning.

Duan Songnan received the MS degree from Heilongjiang University of Science and Technology, China, in 2024. He is currently pursuing the PhD degree at Nanjing University of Aeronautics and Astronautics, China. His research interests include 3D perception and computer graphics.

Zhou Ruolin received the PhD degree in electrical engineering from Wright State University, USA, in 2012. She is an associate professor at University of Massachusetts, Dartmouth, MA, USA. Her research interests include intelligent radio and RF systems, AI-enabled spectrum learning and management, and security of real-time embedded systems.

Zhu Yongtao received the ME degree from Heilongjiang University of Science and Technology, China. He is the chief engineer of Heilongjiang Longmei Shuangyashan Mining Co., Ltd. and also a PhD candidate at Heilongjiang University of Science and Technology. His research interests include direction of intelligent devices and security engineering.