Design and Implementation of Virtual Assembly System for Machine Tool Fixtures Based on Kinect and Unity3D

Zixin Jiao[1], Lianpeng Zhang[1], Yan Wu[1,*]
[1]School of Computer Science and Artificial Intelligence, Northeast Forestry University, Harbin 150040, China
Corresponding author: Yan Wu (e-mail: 19031818328@163.com).

ABSTRACT Traditional machine-tool fixture assembly training relies on heavy physical equipment, entails substantial purchase and maintenance costs, and limits repeated practice because of safety and space constraints. This study develops a low-cost virtual assembly system by integrating Kinect V2 somatosensory interaction with Unity3D. Kinect captures 25 human skeletal joints in real time, while Unity3D provides model rendering, physical simulation, and interactive scene management. A constrained dynamic time warping method is introduced to reduce the gesture-matching search region and reject invalid actions through a distortion threshold. Ray casting, collision detection, and multi-coordinate transformation are combined to support part selection, movement, rotation, alignment, assembly, and viewpoint control. Fixture models are preprocessed in SolidWorks and 3ds Max, assigned rigid-body and collider properties, and rendered with differentiated metallic materials. Experimental evaluation shows that the improved recognition method achieves 92.5% accuracy with a response delay below 78.2 ms, while increasing computational efficiency by more than 40% relative to conventional DTW. Compared with traditional physical training, the system increases assembly efficiency by 32.5% and reduces the operation-error rate by 46.8%. The results demonstrate the feasibility of using natural gestures for repeatable fixture-assembly teaching and provide a practical digital platform for engineering education, design verification, and pre-job training.

INDEX TERMS Virtual reality; Kinect somatosensory interaction; Unity3D; Machine tool fixture; Virtual assembly; Gesture recognition; Intelligent hardware

I. INTRODUCTION

The digital transformation of intelligent manufacturing and engineering education has increased the demand for practical training environments that are safe, repeatable, and economical. Virtual reality (VR) can reconstruct three-dimensional industrial scenes, present internal structures that are difficult to observe on physical equipment, and allow learners to repeat operations without consuming materials or damaging hardware [1]–[3]. These advantages are particularly relevant to machine-tool fixture assembly. Conventional training requires multiple metal components, dedicated workspaces, frequent maintenance, and close safety supervision. Heavy parts and restricted operating space also make it difficult for students to practice complete disassembly and reassembly procedures, while classroom demonstrations cannot always reveal fine positioning, interference, and constraint relationships. A virtual assembly platform can therefore supplement physical training by providing visual guidance, error feedback, and controlled practice at substantially lower deployment cost.

Somatosensory interaction provides a natural link between physical gestures and virtual operations. Kinect V2 offers real-time depth sensing and skeletal tracking at comparatively low cost, and Unity3D provides mature graphics rendering, scene construction, collision detection, and physical simulation. International research has applied similar technologies to industrial training, assembly verification, and process optimization, and domestic studies have explored virtual assembly, digital twins, and depth-sensing interaction [4]–[6]. Nevertheless, teaching-oriented fixture systems still face three practical limitations. First, gesture recognition may be unstable when operators execute the same action at different speeds. Second, interaction delay and false activation reduce operational continuity. Third, insufficient coordinate mapping and physical constraints can cause unrealistic penetration, misalignment, or visual feedback. Existing systems also tend to emphasize individual technical modules rather than an integrated workflow covering model preparation, gesture control, physical assembly, teaching guidance, and performance evaluation.

To address these limitations, this paper designs and implements a machine-tool fixture virtual assembly system

based on Kinect and Unity3D. The system uses a typical drilling fixture as the object, preprocesses its components for real-time rendering, and establishes a communication link between Kinect skeletal data and the Unity3D scene. A modified dynamic time warping algorithm restricts the search path and introduces a distortion threshold to improve recognition efficiency and suppress invalid gestures. Multi-coordinate transformation, ray casting, rigid-body simulation, and collision detection are then integrated to realize model picking, dragging, rotation, alignment assembly, and flexible viewpoint control. The principal contribution is an engineering-oriented, low-cost teaching solution in which the interaction algorithm, physical simulation, and instructional functions are evaluated as a complete system. Functional and comparative tests are used to examine recognition accuracy, response delay, assembly efficiency, error rate, and user experience. The resulting platform is intended not to replace physical practice entirely, but to provide a safe pre-training and repeated-practice environment that can support mechanical education, enterprise induction training, and preliminary fixture-design verification. It also supports transparent comparison between technical choices and teaching outcomes.

II. RELEVANT TECHNICAL BASIS

A. VIRTUAL REALITY TECHNOLOGY

Virtual reality constructs interactive three-dimensional environments by integrating computer graphics, sensing, physical simulation, and human-computer interaction. In industrial education, its main value is the ability to reproduce equipment structures and assembly procedures without the spatial, safety, and consumable constraints of physical training. Virtual assembly further allows parts to be observed, manipulated, constrained, and repeatedly assembled in a controlled environment, thereby supporting process demonstration, skill rehearsal, and preliminary design verification.

B. KINECT SOMATOSENSORY INTERACTION TECHNOLOGY

Kinect V2 integrates an RGB camera, an infrared projector and camera, and a microphone array. It can track the three-dimensional positions of 25 skeletal joints and transmit color, depth, and body data through Kinect for Windows SDK 2.0 [7], [8]. Figure 1 identifies the main sensing components, while Table 1 summarizes their functions. In this system, depth and skeletal streams provide the motion data used for gesture recognition and coordinate mapping, enabling contactless control without additional wearable devices.

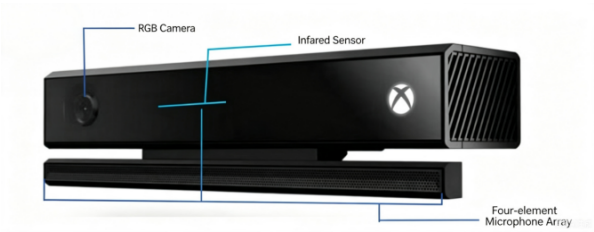

Fig. 1. Kinect Display.

TABLE 1. Function List of Various Parts of Kinect

Hardware name	Function
microphone array	Contains four small microphones for locating the location of sound sources and achieving automatic noise reduction.
Infrared projector	Projects an infrared spectrum outward, creating random spots that can be read by an infrared camera.
Infrared camera	Analyze and calculate the collected speckle data to establish a visual distance depth image.
USB cable	USB3.0 interface is used to transmit the data stream collected by Kinect. Due to Kincet's high power, an independent power supply is required.
RGB color camera	Used to capture color video images and collect RGB data streams.

C. UNITY3D DEVELOPMENT TECHNOLOGY

Unity3D is used as the core development platform because it combines 3D model import, C# scripting, scene management, rendering, and the PhysX physics engine. These capabilities support rigid-body motion, colliders, joints, material rendering, and external-device integration in one environment. Consequently, the same platform can process Kinect data, execute interaction logic, display teaching prompts, and simulate the physical behavior of fixture components [9].

D. DTW GESTURE RECOGNITION TECHNOLOGY

Dynamic time warping (DTW) compares temporal sequences that differ in duration by searching for a minimum-cost alignment path. This property is suitable for gestures because different users may perform the same action at different speeds while preserving a similar trajectory [10], [11]. Conventional DTW, however, searches a large state space and may match irrelevant motion. The present study therefore constrains the search region and introduces a rejection threshold to improve real-time performance and resistance to interference [12].

E. RAY TRACING AND COLLISION DETECTION TECHNOLOGY

Ray casting determines which virtual object lies along a line projected from the camera or interaction point, and is used here for part selection and interface triggering. Collision detection defines the spatial boundaries of fixture components and prevents penetration during movement and assembly. Combined with rigid-body and joint constraints, these mechanisms provide the positional feedback required for credible virtual assembly and ensure that user actions remain consistent with the geometry and connection rules of the physical fixture.

III. OVERALL SYSTEM DESIGN

A. SYSTEM REQUIREMENTS ANALYSIS

System requirements were derived from fixture-assembly teaching tasks and are organized into functional, performance, and user-experience categories, as shown in Figure 2. The design must support component management, natural gesture control, viewpoint adjustment, physical constraints, assembly guidance, and immediate feedback, while remaining deployable on ordinary teaching computers [13]–[16].

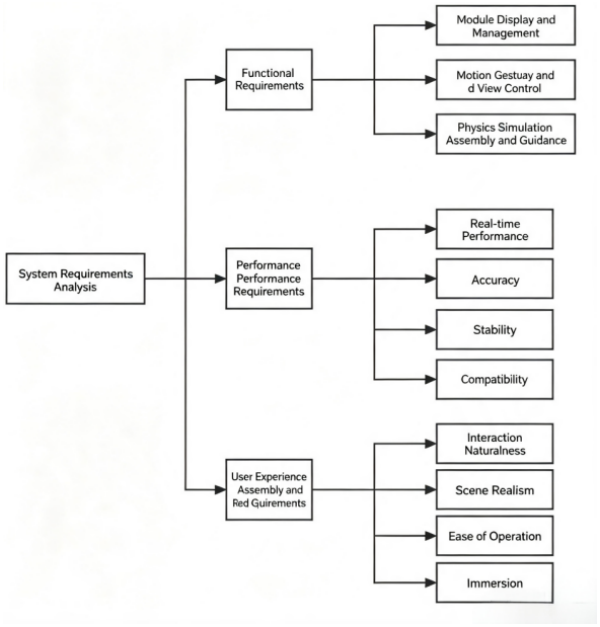

Fig. 2. System Requirements Analysis Diagram.

1) FUNCTIONAL REQUIREMENTS

The functional modules include model loading and reset, gesture-based picking, movement, rotation and alignment, camera rotation and zoom, collider-based interference prevention, and step-by-step assembly guidance with correctness feedback.

2) PERFORMANCE REQUIREMENTS

Performance targets are an interaction delay no greater than 100 ms, gesture-recognition accuracy of at least 90%, assembly-positioning error within 0.5 cm, continuous operation for more than four hours, and compatibility with Windows 10 or later on mainstream hardware.

3) USER EXPERIENCE REQUIREMENTS

The user-experience requirements emphasize contactless operation, clear visual instructions, realistic metal materials and lighting, and an interface that allows students to understand the assembly sequence without complex device training.

B. SYSTEM SOFTWARE AND HARDWARE ARCHITECTURE DESIGN

1) HARDWARE ARCHITECTURE DESIGN

The hardware platform consists of a computer, Kinect V2, and a display. The recommended configuration is an Intel Core i5-class processor, at least 8 GB of memory, and a GTX 1050-class graphics card. Kinect is connected through USB 3.0 and positioned approximately 0.9 m above the floor, with an effective user distance of 1.2-3.5 m. Tests showed that these settings provide stable body tracking while maintaining a simple and low-cost deployment.

2) SOFTWARE ARCHITECTURE DESIGN

The software adopts a four-layer architecture, illustrated in Figure 3. The data-acquisition layer obtains color, depth, and skeletal streams; the processing layer filters joint coordinates, recognizes gestures, and converts coordinate systems; the core-function layer performs model management, physical simulation, viewpoint control, and teaching guidance; and the application layer presents the assembly scene and user interface. This separation supports independent debugging and future module replacement.

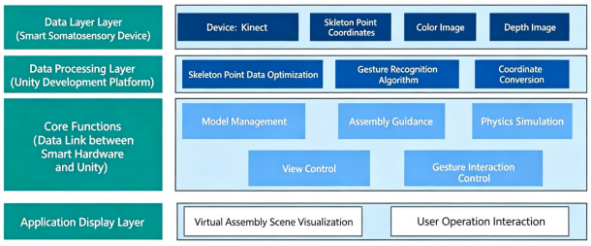

Fig. 3. System Software Architecture Diagram.

C. SYSTEM INTERACTION PROCESS DESIGN

Figure 4 summarizes the interaction sequence. After Kinect calibration and scene loading, skeletal coordinates are acquired and preprocessed, the improved DTW method identifies the gesture, and the corresponding command is mapped to a virtual operation. Unity3D then executes selection, movement, rotation, collision checking, and alignment. When all required components are assembled, the system records operation data and produces a training report.

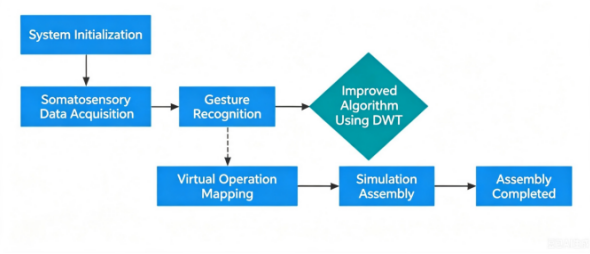

Fig. 4. System Core Interaction Process.

IV. SYSTEM DETAILED DESIGN AND IMPLEMENTATION

A. CONSTRUCTION AND PREPROCESSING OF 3D MODEL OF MACHINE TOOL FIXTURE

1) THREE-DIMENSIONAL MODEL CONSTRUCTION

A typical drilling fixture containing 12 components was selected as the test object. Each part was modeled in SolidWorks 2022 according to the dimensions of the physical fixture, so that the virtual geometry retained the relevant positioning and connection relationships. Figure 5 presents the resulting high-precision model used in the assembly scene.

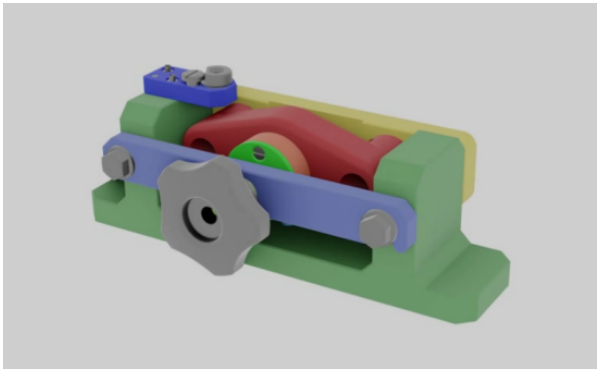

Fig. 5. High-Precision Model Example.

2) MODEL PREPROCESSING

The SolidWorks models were transferred to 3ds Max for cleaning, topology adjustment, polygon reduction, material assignment, and FBX export. The workflow shown in Figure 6 removes redundant geometry while retaining assembly-critical details and metallic appearance, thereby balancing visual fidelity with real-time rendering efficiency.

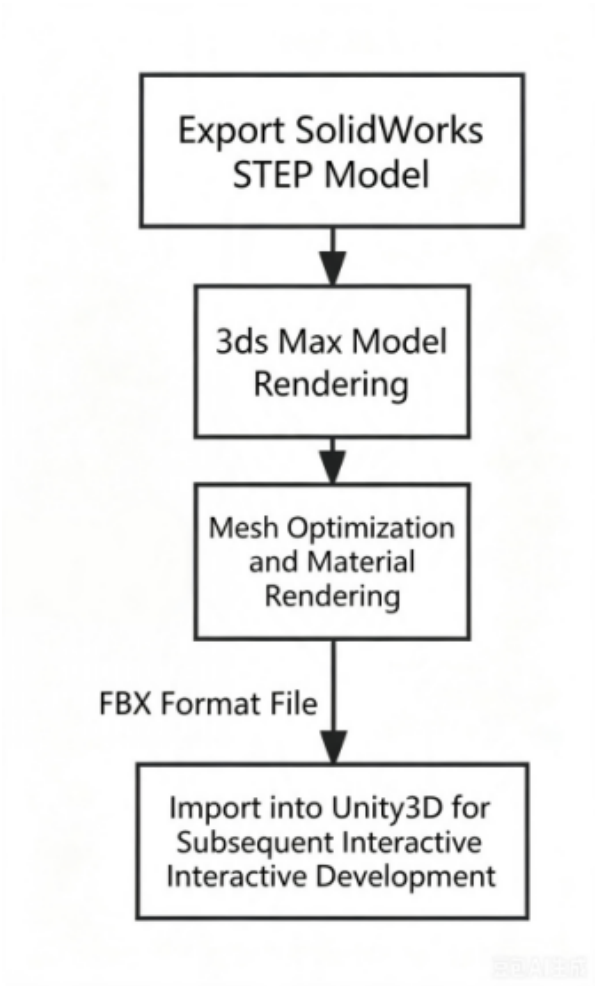

Fig. 6. Model Preprocessing Process.

B. SYSTEM SOFTWARE AND HARDWARE PLATFORM CONSTRUCTION

1) SOFTWARE DEVELOPMENT ENVIRONMENT CONFIGURATION

Development was completed under Windows 10 using Unity3D 2022.3.47f1c1, Visual Studio 2022, Kinect for Unity SDK 2.0, SolidWorks 2022, and 3ds Max 2022. Unity3D manages the scene and interaction scripts, Visual Studio supports C# development, the Kinect plug-in provides body-data access, and the modeling tools prepare fixture geometry.

2) KINECT AND UNITY3D DATA COMMUNICATION SETUP

KinectManager and BoneMapper were imported to establish the Kinect-Unity3D communication link. The device was initialized for skeletal and color streams, and data were updated at 30 frames per second. Joint-coordinate filtering and tracking thresholds were tuned experimentally to suppress jitter while retaining responsive motion. The resulting stream provides stable input for gesture recognition and coordinate conversion.

C. IMPLEMENTATION OF GESTURE RECOGNITION BASED ON IMPROVED DTW ALGORITHM

1) CORE INTERACTIVE GESTURE SCREENING

Combined with the operational requirements of the virtual assembly of machine tool fixtures, 9 core interactive gestures were screened out, covering operations such as model picking, movement, rotation, and perspective control. Each gesture corresponds to a unique interaction intention. The specific gesture definitions are shown in Table 2.

TABLE 2. Kinect Interactive Gestures

serial number	gesture	Semantics
1	Tpose	T position (arms extended and straight)
2	SwipeLeft	Swing right hand to left
3	SwipeRight	Swing left hand to right
4	SwipeUp	Swing upward with left or right hand
5	SwipeDown	Swing down with left or right hand
6	ZoomOut	With your elbows down, slowly separate your left and right palms
7	ZoomIn	With your elbows down, your palms slowly come together
8	Push	Push your left or right hand outward
9	Pull	Pull your left or right hand inwards

2) PRINCIPLE OF TRADITIONAL DTW ALGORITHM

DTW aligns a test gesture sequence with a reference sequence by minimizing cumulative pointwise distance. Let the two sequences have lengths m and n, as defined in Equations (1) and (2). The distance matrix and an admissible warping path are illustrated in Figure 7. Boundary, continuity, and monotonicity constraints ensure that the path begins and ends at the sequence limits and advances without reversing time.

Assume that there are 2 action sequences X and Y, with lengths m and n respectively, as shown in formula (1) and formula (2).

$$X = \{x_1, x_2, \ldots, x_m\} \tag{1}$$

$$Y = \{y_1, y_2, \ldots, y_m\} \tag{2}$$

The path cost is obtained recursively from the current point distance and the minimum predecessor cost. Equations (3)-(5) define the admissible path and final matching distance; a smaller cumulative distance indicates greater similarity between the observed and reference gestures.

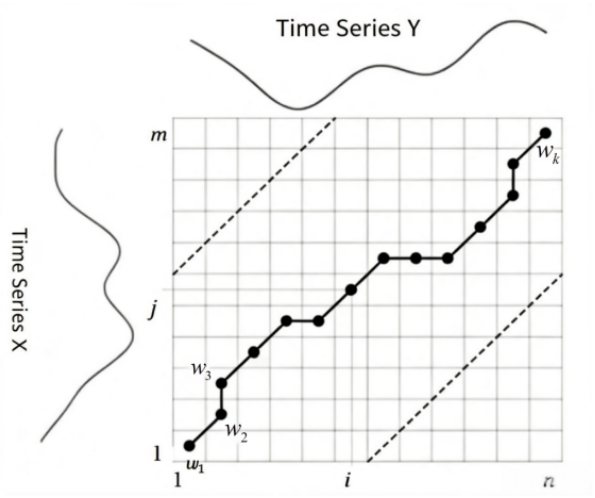

Fig. 7. Schematic Diagram of DTW Algorithm.

$$W = \{w_1, w_2, \ldots, w_T\},\ \max(m,n) \le T \le m+n-1 \tag{3}$$

$$Dis_{ij} = d_{ij} + \min\{Dis_{(i-1)}, Dis_{(i-1)(i-1)}, Dis_{(i-1)}\} \tag{4}$$

$$DTW(\alpha,\beta) = \min\left\{\frac{1}{T}\sqrt{\sum_{i=1}^{T} W_i}\right\} \tag{5}$$

3) IMPROVE DTW ALGORITHM DESIGN

The improved method reduces unnecessary computation in two ways. First, a slope-constrained search region restricts the warping path to a band around the diagonal, as shown in Figure 8. Second, a distortion threshold rejects motion sequences that are unlikely to match any stored gesture, preventing full template comparison for negative samples [17].

1. LIMIT THE SEARCH SCOPE

Equation (6) introduces weighting coefficients into the recursive distance calculation so that horizontal, vertical, and diagonal transitions remain within the constrained region. This reduces the number of evaluated states while preserving the temporal deformation required for different execution speeds.

$$Dis_{ij} = d_{ij} + \min\{\alpha \cdot Dis_{ij-ij}, \beta \cdot Dis_{(i-1)j-1)}, \gamma \cdot Dis_{ij-1)}\} \tag{6}$$

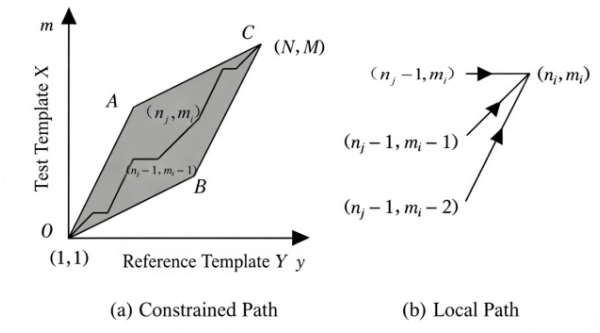

Fig. 8. Schematic Diagram of DTW Improved Algorithm.

2. DISTORTION THRESHOLD

Experimental distances for valid and invalid actions were mainly distributed between 40 and 120; therefore, the rejection threshold was set to T=100. Sequences with DTW distance at or below 100 proceed to template matching, whereas larger values are treated as invalid gestures and do not trigger an operation.

4) GESTURE RECOGNITION IMPLEMENTATION PROCESS

The recognition workflow is shown in Figure 9. Kinect samples upper-limb joints at 30 frames per second, the coordinates are filtered, and simple posture and motion conditions identify the beginning and end of an action. The resulting sequence is compared with stored templates using the improved DTW method. A match within the threshold activates the mapped command; otherwise, the action is ignored. Across the nine gestures in Table 2, the method achieved 92.5% recognition accuracy and a response delay no greater than 78.2 ms, with more than 40% higher computational efficiency than conventional DTW.

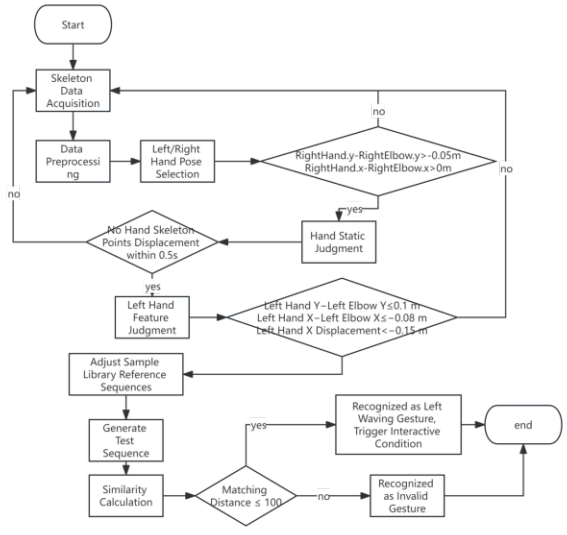

Fig. 9. Gesture Recognition Process.

D. IMPLEMENTATION OF CORE FUNCTIONS OF VIRTUAL ASSEMBLY

1) IMPLEMENTATION OF MULTI-COORDINATE SYSTEM TRANSFORMATION

Kinect reports joints in the camera-centered physical coordinate system, whereas Unity3D operations use screen and world coordinates. The conversion chain therefore maps camera coordinates to image-plane and pixel coordinates and then to the Unity3D world frame [18]. Figure 10 defines the Kinect camera axes, Figure 11 shows the pixel frame, and Figure 12 presents the projection relationship. Equations (7)-(14) describe the intrinsic, extrinsic, and homogeneous transformations, while Table 3 lists the API functions used for the corresponding conversions.

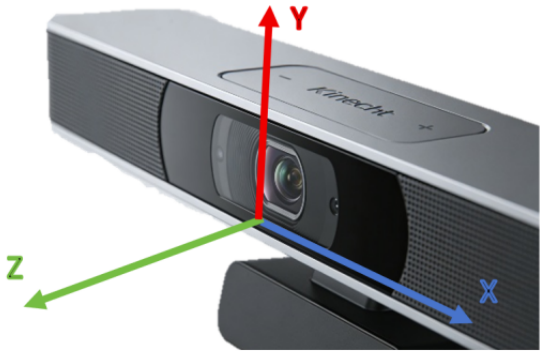

Fig. 10. Camera Coordinate System.

Pixel coordinate system. Taking the upper left corner of the image as the origin, the coordinate system used to describe the location of the pixel point is called the pixel coordinate system, in which the horizontal direction is the u-axis and the vertical direction is the v-axis, as shown in Figure 11.O_0

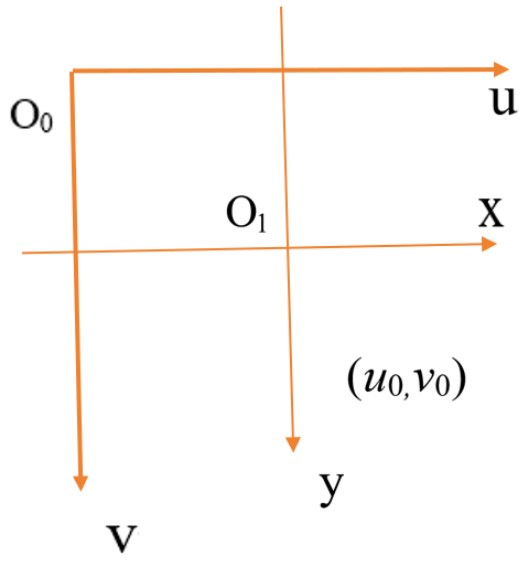

Fig. 11. Pixel Coordinate System.

$$u\& = \frac{x}{d_x} + u_0 \tag{7}$$

$$v = \frac{y}{d_y} + v_0 \tag{8}$$

The homogeneous expression is:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = L_W \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{10}$$

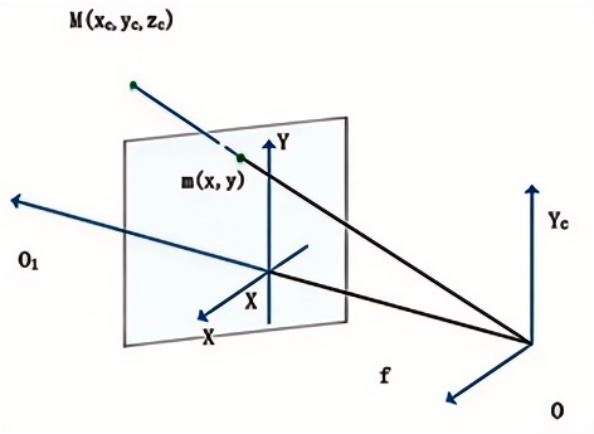

Fig. 12. The Relationship between Camera Coordinate System and Image Coordinate System.

$$\begin{cases} \&x = f\dfrac{x_C}{z_C} \\ \&y = f\dfrac{y_C}{z_C} \quad \&\& \end{cases} \tag{11}$$

$$Z_c \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} \tag{12}$$

$$\underbrace{Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{Projection Relationships}} \underbrace{\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{\text{The Camera and the World}}}_{\text{Pixels and the Image Plane}}} \tag{13}$$

$$\begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{14}$$

The external parameter matrix can be expressed as: consisting of the rotation matrix R and the translation vector. $\begin{bmatrix} R & t \\ O^T & 1 \end{bmatrix}$

During the Kinect interactive control assembly process, the movement of the model and the click of the interactive button require conversion between coordinate systems. The main coordinate conversion functions provided by the KinectManger API are shown in Table 3.

TABLE 3. Coordinate Conversion Program Instructions

serial number	name	program instructions
1	world coordinates → screen coordinates	`camera.main.WorldToScreenPoint(transform.position)`
2	Screen coordinates → camera coordinates	`camera.ScreenToViewportPoint(Input.GetTouch(0).position)`
3	Camera coordinates → screen coordinates	`camera.ViewportToScreenPoint()`
4	Camera coordinates → world coordinates	`camera.ViewportToWorldPoint()`

2) MODEL PHYSICAL SIMULATION AND COLLISION DETECTION IMPLEMENTATION

Unity3D PhysX supplies rigid bodies, colliders, gravity, friction, rebound, and joint constraints for the fixture components [19], [20]. Simple parts use a single collider, while complex parts use combined boxes, as illustrated in Figure 13. Mass and collision parameters are adjusted to match the intended metal-part behavior; Figure 14 shows a representative collider configuration. Collision events prevent penetration and provide immediate feedback when the assembly path or alignment is invalid.

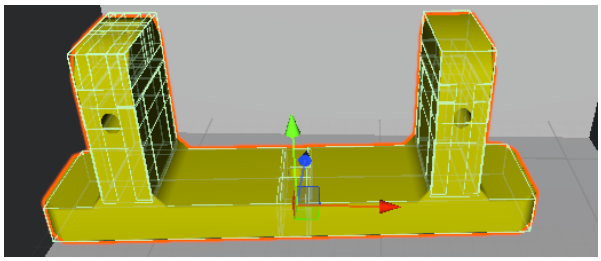

Fig. 13. Square Collider.

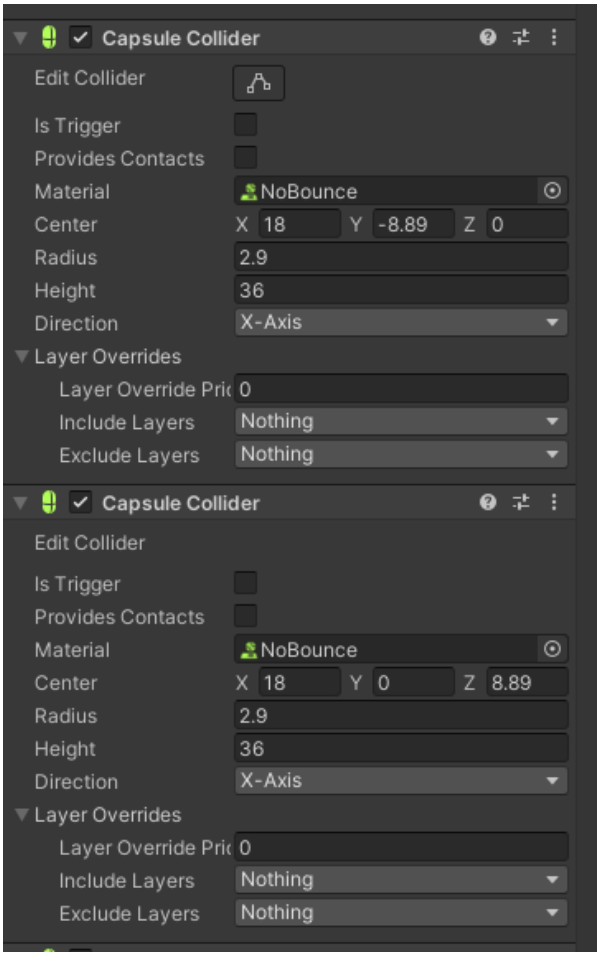

Fig. 14. Collider Parameter Settings.

3) MODEL PICKING AND MOVING IMPLEMENTATION

Part selection and movement combine the improved gesture recognizer with ray casting and coordinate mapping. A fist gesture projects a ray from the camera through the virtual hand position; an intersected collider is highlighted and assigned as the drag target. While the gesture is maintained, the part follows the converted hand coordinates with a stored depth offset. Releasing the fist fixes the part at its current position, after which collision and alignment rules determine whether the placement is valid.

4) IMPLEMENTATION OF VIEWING ANGLE CONTROL FUNCTION

Viewpoint control removes visual blind spots during assembly. Pull and push gestures translate the camera along its viewing axis, while SwipeLeft, SwipeRight, SwipeUp, and SwipeDown rotate the scene by fixed increments. The command mapping is summarized in Table 4. Continuous skeletal monitoring updates the camera only after a valid gesture is recognized, avoiding unintended motion while allowing users to inspect internal structure and fine alignment details.

TABLE 4. Camera Perspective Change Logic

serial number	gesture	Semantics
1	SwipeLeft	`camera.transform.Rotate (new Vector3(0, 5, 0))`
2	SwipeRight	`camera.transform.Rotate (new Vector3(0, -5, 0))`
3	SwipeUp	`camera.transform.Rotate (new Vector3(5, 0, 0))`
4	SwipeDown	`cameratransform.Rotate (newVector3(-5,0, 0))`
5	Pull	`camera.transform.Translate (newVector3(0,0,1))`
6	Push	`camera.transform.Translate (newVector3(0,0,-1))`

E. SYSTEM INTERFACE AND TEACHING SYSTEM CONSTRUCTION

1) SYSTEM OPERATION INTERFACE DESIGN

The Unity3D Canvas interface separates the assembly scene from a compact guidance panel. The scene occupies most of the display, while the panel presents the current step, required action, and correctness feedback. This arrangement keeps the operating area unobstructed and reduces the learning cost for first-time users.

2) CONSTRUCTION OF TEACHING-BASED VIRTUAL TEACHING SYSTEM

The teaching module divides the fixture procedure into six guided stages and records completion time, recognition errors, and assembly mistakes. After the final step, the system generates a report containing performance data, evaluation results, and targeted improvement suggestions. These functions connect virtual practice with teacher assessment and self-directed review [21].

V. CONCLUSION AND OUTLOOK

The study demonstrates that a Kinect-Unity3D platform can provide a technically and economically feasible supplement to machine-tool fixture training. The system uses commercially available hardware, ordinary computer configurations, and a modular software architecture, while the improved DTW method meets the preset real-time and accuracy requirements. Experimental results show 92.5% gesture-recognition accuracy, response delay below 78.2 ms, more than 40% higher matching efficiency than conventional DTW, a 32.5% increase in assembly efficiency, and a 46.8% reduction in operation errors. These results support the feasibility of natural-gesture interaction for repeated pre-training, process demonstration, and preliminary design verification. The main contribution is not a single algorithmic component, but the integration of gesture recognition, coordinate mapping, physical constraints, viewpoint control, instructional guidance, and performance reporting into a deployable teaching workflow. Nevertheless, the present evaluation is limited to one drilling-fixture model, hand-dominant gestures, a single-user desktop environment, and simulated physical feedback. The sample size and duration of the teaching evaluation also remain insufficient to establish long-term learning gains. Future work should therefore expand the fixture library to lathe and milling applications,

incorporate whole-body and multimodal interaction, and validate the system with larger student cohorts and longer controlled studies. Additional priorities include adaptive AI guidance, automatic error diagnosis, more realistic force or haptic feedback, networked multi-user collaboration, and augmented-reality registration with physical teaching aids. These extensions would improve generalizability and strengthen the connection between virtual practice and real manufacturing operations.

DECLARATION OF CONFLICTING INTERESTS

The author(s) declared no potential conflicts of interest with respect to the research, author-ship, and/or publication of this article.

DATA SHARING AGREEMENT

The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

FUNDING

The author(s) received no financial support for the research, authorship, and/or publication of this article.

REFERENCES

[1] D. Gu, J. Zhang, Z. Ma, et al., "Integration of virtual and real to reshape the classroom—Changchun University of Technology's innovative teaching model," CCTV, 2026-01-07.

[2] Ministry of Education, "Notice on the collection of typical application scenarios of "artificial intelligence + higher education"," (2025-12-10). [Online]. Available: http://www.moe.gov.cn.

[3] Southeast University. CiviX^3-LAB: "AI + Virtual +.

[4] F. Peters, "Christiani´s digital twin teaching system," Worlddidac, vol. 15, no. 2, pp. 34–38, 2024.

[5] X. N. Lai, T. Y. Zhang, L. Wang, et al., "Fingertip interactive tracking registration method for AR assembly system," Advances in Computational Intelligence, vol. 2, no. 2, pp. 1–12, 2022.

[6] Scientific Reports, "Application of virtual assembly for complex mechanical structures based on digital twin technology," Scientific Reports, vol. 15, p. 30306, 2025.

[7] B. Schmidt, L. Wang, and D. Zhang, "Toward safe human robot collaboration by using multiple Kinects based real-time human tracking," Assembly Automation, vol. 42, no. 3, pp. 289–298, 2022.

[8] A. Zillner, "Variations of 3D real-time avatar creation for virtual office collaboration using Azure Kinect and HoloLens 2," Munich: Technical University of Munich, 2026.

[9] M. Zhang, H. Li, and W. Wang, "Unity realizes linkage between physical and virtual workshops," Computer Simulation, vol. 41, no. 4, pp. 223–228, 2024.

[10] J. Zhang and B. Zhang, "Dynamic gesture recognition method for human-computer interaction considering optical flow matching," Modern Electronic Technology, vol. 48, no. 23, pp. 151–154, 2025.

[11] H. Zhang, T. Wang, and G. Li, "Research on virtual assembly gesture recognition based on Azure Kinect and SVM," Mechanical Design and Manufacturing, vol. 38, no. 2, pp. 112–118, 2024.

[12] L. Wang, H. Chen, and Y. Liu, "Gesture scoring based on Gaussian distance-improved DTW," Journal of Virtual Reality and Intelligent Hardware, vol. 6, no. 2, pp. 145–156, 2024.

[13] Guangdong Mechanical and Electrical Vocational and Technical College, "Construction practice of collaborative innovation center for intelligent production line fixtures," (2025-11-14). [Online]. Available: https://gdmec.edu.cn.

[14] Z. Huang, S. Luo, B. Zhou, et al., "Few-shot neural differentiable simulator: real-to-sim rigid-contact modeling," arXiv, 2603.06218.

[15] NetEase Yidun, "Description of coordinate system changes in spatial calculation positioning," (2024-11-15). [Online]. Available: https://dongjian-web.netease.com.

[16] G. Li and J. Wang, "Review of research on dynamic gesture recognition based on deep learning," Journal of Computer Science, vol. 47, no. 5, pp. 1023–1048, 2024.

[17] Y. Liu and W. Zhang, "Improvement and application of dynamic time warping algorithm," Pattern Recognition and Artificial Intelligence, vol. 38, no. 2, pp. 156–169, 2025.

[18] EasyAR, "EasyAR dense spatial map developer guide," (2025-11-14). [Online]. Available: https://help.easyar.com.

[19] M. Zach, "A volume-based penetration measure for 6DOF haptic rendering of streaming point clouds," Bremen: University of Bremen, 2017.

[20] J. Wang and L. Chen, "Research on the application of Unity3D physics engine in virtual assembly," Journal of System Simulation, vol. 37, no. 4, pp. 892–905, 2025.

[21] J. Tan and Z. Liu, "Intelligent manufacturing and digital twins: key technologies and development trends," Chinese Journal of Mechanical Engineering, vol. 61, no. 1, pp. 1–18, 2025.