electronics

MDPI

Article

Design of Dancing Robot Based on Machine Vision

Junmei Gong [1], Dan Lai [1,*], Chengnan Long [2], Yang Liu [2,*] and Dawei Gong [2]

[1] School of General Education, Chengdu Jincheng College, Chengdu 611731, China
[2] School of Mechanical and Electrical Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
* Correspondence: laidan@cdjcc.edu.cn (D.L.); ly2015@uestc.edu.cn (Y.L.)

Abstract

Aiming at the application requirements of artificial intelligence and robotics, this paper adopts a lightweight and low-cost scheme to construct a dancing-robot system integrating machine vision, mechanical structure and motion control so as to improve its accurate human-motion imitation capability. It provides an important reference for the engineering application of motion imitation of humanoid robots. A human-pose visual-recognition model is built based on Python and deep-learning techniques. A monocular camera captures two-dimensional images, from which skeletal key-point coordinates are extracted via 3D reconstruction. Joint angles are calculated by inverse kinematics to supply core input data for motion imitation. Mechanically, a humanoid joint structure composed of 16 servos is adopted to realize one-to-one mapping and execution of joint angles. In hardware, a collaborative control circuit is constructed with the main-control module and servo-drive module as the core. Software implements data-parsing, instruction generation and closed-loop control. After system integration and debugging, the robot can stably and accurately reproduce simple human dance movements, which verifies the feasibility and effectiveness of the proposed scheme.

Keywords: dancing robot; pose solution; deep learning; machine vision; steering-servo control

1. Introduction

With the rapid development of service robots and human–computer interaction technologies, humanoid robots with autonomous motion-imitation capability have become a research hotspot in the robotics field [1]. As a typical application of humanoid robots, dancing robots impose requirements not only on the bionic performance of mechanical structures, but also on strict technical indicators for the real-time performance of visual perception and the accuracy of motion control. Traditional dancing robots mostly adopt the control mode based on pre-set motion libraries. They lack the capability of autonomous recognition and self-adaptation for dynamic human motions, which restricts the flexibility of interactive scenarios and narrows their practical application scope.

The deep integration of machine-vision technology and deep learning provides core technical support for autonomous human-motion perception. Low-cost and non-contact human-pose capture can be realized with a monocular camera. Combined with 2D-to-3D reconstruction and kinematic analysis, human joint-motion parameters can be efficiently solved to lay a data foundation for robot motion imitation [2]. Against this background, this paper designs a dancing-robot system based on machine vision. Through the collaborative optimization of visual recognition, mechanical-structure design and motion-control strategies, autonomous imitation of simple human dance movements is realized, which

Academic Editor: Farhan Riaz

Received: 19 August 2026
Revised: 7 September 2026
Accepted: 9 September 2026
Published: 14 September 2026

provides a feasible technical path for improving the human–robot interaction capability of dancing robots.

2. Human-Pose Recognition Algorithm

Human-pose recognition based on traditional algorithms generally adopts various image-processing approaches, such as neural-network models, template-matching, feature-point detection and edge detection. These methods perform pose classification or extract joint-point information for targets in images, so as to realize pose recognition for single-person, two-person or multi-person scenarios [3,4]. Traditional human-pose-recognition algorithms have inherent limitations in algorithm architecture and model structure, which bring about many drawbacks in practical applications. They are only suitable for limited scenarios and exhibit poor performance under complex backgrounds. Meanwhile, the algorithms are highly sensitive to interferences including target occlusion and dynamic pose variation, resulting in a significant drop in recognition accuracy.

To overcome the above shortcomings, this paper adopts a deep-learning-based human-pose-recognition algorithm. Convolutional Neural Network (CNN) is utilized to extract human-pose information from images, and the model is trained on human-image datasets. Compared with traditional algorithms, the proposed method can capture human-pose information more accurately and extract richer semantic features, and achieves better performance when dealing with complex scenarios and diverse data [5–7].

In this work, BlazePose from MediaPipe0.10.1 is adopted as the off-the-shelf pose estimation backbone. Direct benchmark evaluation on public datasets is not the focus of this paper; instead, we target the practical imitation performance within our robot dance-imitation scenario. Comparative experiments against OpenPose 1.7.0 and template-matching methods are conducted under our real-world test scenarios (see Section 5.3.4), which quantitatively demonstrate the advantages of the adopted lightweight pipeline for monocular-vision-driven robot imitation tasks.

2.1. Deep-Learning-Based Human-Pose Recognition Algorithm

Representative deep-learning-based human-pose-recognition models include OpenPose and BlazePose. BlazePose is a lightweight convolutional neural-network model for human-pose estimation, which can detect 33 body key-points of a single person [8]. Its skeleton topological structure is shown in Figure 1. The model adopts a detector-tracker architecture for pose recognition. Its working mechanism is as follows: the tracker predicts key-point coordinates, the human-presence status in the current frame, and the Region of Interest (ROI), which refers to the image region covering all features of the target for key-point detection. The detector relies on the fast on-device face detector to predict the mid-point of human hips, the radius of the circumscribed circle enclosing the human body, and the tilt angle of the straight line connecting the mid-points of shoulders and hips. In the pose-detection procedure, the detector first locates the ROI in the initial frame. Then the tracker estimates all 33 key-point positions based on this ROI. For subsequent frames, the predicted key-points from the previous frame are adopted to update and locate the ROI [9].

The neural network of this model adopts a combined strategy of heatmap, offset and regression. During network training, heatmap loss and offset loss are combined for parameter optimization. Corresponding output layers are removed before inference, and heatmaps are utilized to supervise lightweight embedding. Finally, the encoder network performs regression for all joint points [10]. To predict occluded key-points, the model introduces a per-point visibility classifier, which belongs to the occlusion-aware scheme in machine vision. It judges the visibility status of each skeletal key-point to mitigate detection difficulties caused by target occlusion in images.

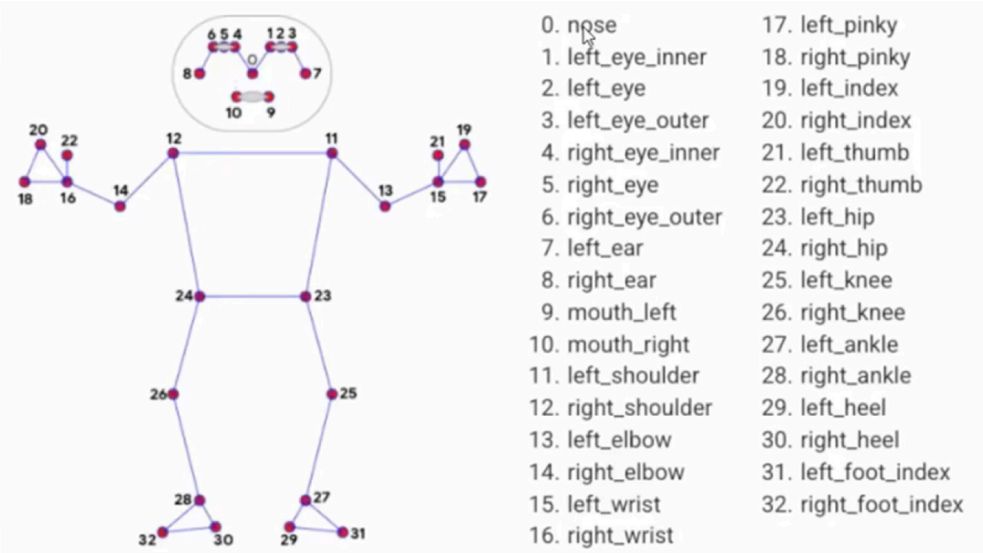


Figure 1. Skeleton and key-point topological structure of BlazePose.

In the experiment, the video file is processed frame-by-frame by the BlazePose model, and a dataset of 33 human body joint points is output. Each key-point contains the abscissa x, ordinate y, and the recognition visibility v of the corresponding joint. The obtained joint-point position information can completely preserve the pose features of the detected target. After preprocessing and kinematic analysis, such data can be directly applied to the motion control of the dancing robot.

2.2. Implementation of Control Algorithm Based on BlazePose Human-Pose Recognition

This paper designs the visual algorithm for a dancing robot based on the BlazePose model on the Python 3.7 platform. The main implementation steps of the algorithm include video extraction, invoking BlazePose from the MediaPipe library for human-pose recognition, acquiring key-point position information, processing position data to calculate bone-joint angles, serial-port communication configuration, and control-logic design.

2.2.1. Video Extraction and Pose Recognition

For video extraction, the VideoCapture function from Python's cv2 (OpenCV) library is adopted. It can obtain video streams from local video files at specified paths or camera devices. Among its member functions, isOpened() judges whether the video file or camera is opened successfully; the read() function returns a read-status flag together with a single video frame. It not only can verify the validity of video reading, but can also sequentially read each frame into the matrix variable image within a loop.

Pose recognition is realized by invoking the pose class in the MediaPipe library. MediaPipe Pose is a high-fidelity machine-learning solution for human-pose tracking built upon BlazePose research outcomes [11]. BlazePose is a lightweight convolutional neural-network architecture for single-person key-point detection. After each video frame is successfully read, the pose.process(image) function is called to perform pose estimation for each frame. This function returns the 3D-coordinate sequence of 33 skeletal key-points predicted by the BlazePose model. Running on a single mid-tier mobile-phone CPU, BlazePose is 25–75 times faster than OpenPose running on a 20-core desktop CPU, and can achieve real-time performance. The coordinate sequence is stored in the list lmlist for subsequent data parsing and processing. The workflow of video extraction and pose recognition is shown in Figure 2.

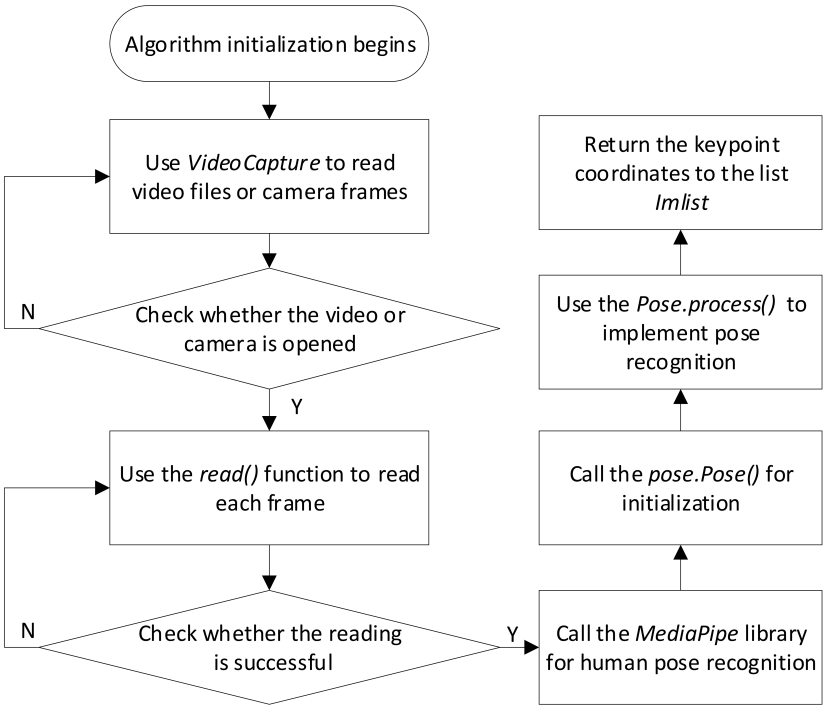


Figure 2. Flowchart of video extraction and pose recognition.

2.2.2. Key-Point Data Processing and Bone-Angle Calculation

The output of pose recognition consists of three-dimensional position information of 33 skeletal key-points. Since the dancing robot in this paper only imitates simple dance movements, only the skeletal key-points of the trunk, shoulders, hips and other parts (key-points numbered 11~16 and 23~28 in Figure 1) need to be extracted. To calculate bone-joint angles, this paper adopts the method of "constructing spatial vectors from key-points and solving bone-joint angles by vector-included-angle operation". Before that, the raw coordinates output by the model should be converted into 3D coordinates in real physical space. As introduced above, BlazePose outputs 33 sets of joint data containing abscissa x, ordinate y and joint-point visibility v. The direction and sequence of the original x, y, z coordinate axes need to be adjusted to obtain the practical coordinate system x', y', z', so as to acquire key-point coordinates under the real-world coordinate frame.

After obtaining the key-point coordinates in the practical coordinate system, the included angles of each bone joint can be solved. The specific calculation methods are described as follows:

(1) Angle calculation for Single-DOF (single-degree-of-freedom) joints

Single-DOF joints correspond to key-points numbered 13, 14, 25 and 26 in Figure 1. The connected bone segments at these joints can only rotate around one single direction. Taking key-point No. 13 as an example, the solving procedure of the angle for single-DOF joints is illustrated below.

Let practical-world coordinates of points 11, 13, 15 be (x_{11}, y_{11}, z_{11}), (x_{13}, y_{13}, z_{13}), (x_{15}, y_{15}, z_{15}). Define the vectors as follows:

$$\vec{v_1} = (x_{15} - x_{13}, y_{15} - y_{13}, z_{15} - z_{13}),$$

$$\vec{v_2} = (x_{13} - x_{11}, y_{13} - y_{11}, z_{13} - z_{11}),$$

The included angle θ at marker 13 is computed by

$$\theta = arccos(\vec{v_1} \cdot \vec{v_2} / (\left|\vec{v_1}\right| \cdot \left|\vec{v_2}\right|)) \quad (1)$$

(2) Angle calculation for Double-DOF (double-degree-of-freedom) joints

Double-degree-of-freedom joints correspond to the key-points numbered 11, 12, 23, 24, 26 and 27 in Figure 1. Two bone segments are connected at these joints, so their motion posture cannot be fully described by only one single angle parameter. In terms of robot motion control, each servo only provides a single-degree-of-freedom rotation. To reproduce the motion of such joints, at least two angle parameters are required, which means that two or more servos should be configured at the corresponding robot joint. Taking key-point No. 11 as an example, the method of characterizing the motion features of bone segment 11–13 with two angles is illustrated below:

A reference coordinate system is established for key-point 11, as shown in Figure 3. The y-axis is collinear with segment 12–11; the x-axis is perpendicular to the y-axis and coplanar with the plane formed by segments 11–12–24; the z-axis is perpendicular to the plane spanned by the x-axis and y-axis, forming a rectangular coordinate frame. When calculating the two angles at key-point 11, considering the anatomical feature that the human shoulder is perpendicular to the upper arm, the initial orientation of arm segment 11–13 is set along the x-axis. Two rotation angles, θ_1 and θ_2, are calculated, corresponding to rotating the initial x-axis around the z-axis first and then around the y-axis to the direction of the actual arm segment 11–13. The obtained angles are the target angles of the two servos corresponding to key-point 11. The angle calculation method for double-DOF joints is illustrated in Figure 4.

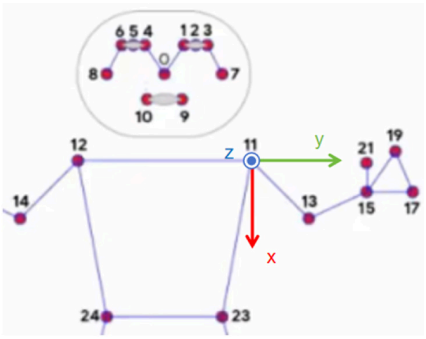


Figure 3. Schematic diagram of reference coordinate system for key-point 11.

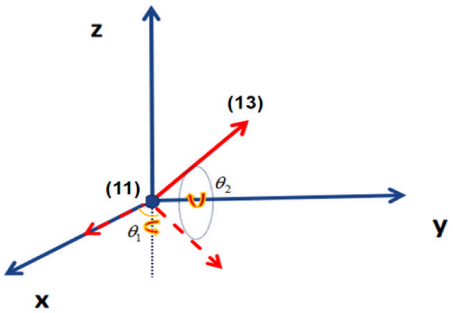


Figure 4. Schematic diagram of angle calculation for double-DOF joint.

In Figure 4, vector 11–13 ($\vec{v_h}$) is obtained by sequentially rotating the initial vector along the x-axis around the z-axis and the y-axis. Angle θ_1 is complementary to the included angle formed by vectors $\vec{v_h}$ and $\vec{y}$, namely,

$$\theta_1 = 90 - arccos(\vec{v_h} \cdot \vec{y} / (|\vec{v_h}| \cdot |\vec{y}|)) \quad (2)$$

For the calculation of angle θ_2 rotating around the y-axis, angle θ_2 is equal to the included angle between vector $\vec{v_h}'$ (the projection of vector $\vec{v_h}$ onto the xOz plane) and the $\vec{x}$, as shown in Figure 5.

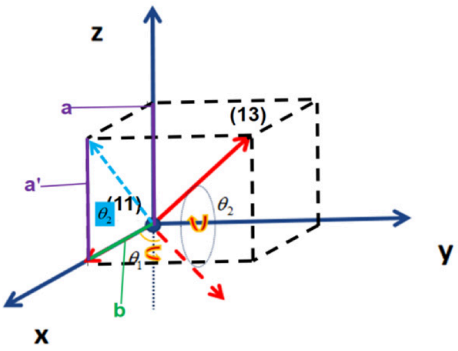


Figure 5. Coordinate system for solving the rotation angle around the y-axis.

Let the projections of vector $\vec{v_h}$ on the x-axis and z-axis be $\vec{b}$ and $\vec{a}$, respectively. Translating $\vec{a}$ to the position of $\vec{a'}$ in Figure 5, θ_2 can be expressed as

$$\theta_2 = \arctan(|\vec{a}|/|\vec{b'}|) \tag{3}$$

where $|\vec{a}| = \vec{v_h} \cdot \vec{z} / |\vec{z}|$, $|\vec{b}| = \vec{v_h} \cdot \vec{x} / |\vec{x}|$.

The two-point method can be adopted to calculate the angles of single-DOF and double-DOF joints. With one joint selected as the reference point, two other joint points are defined, and the angles between the x-axis and the line connecting the reference point to each target joint are calculated. This method reduces the angle error caused by relative jitter between key-points and realizes accurate angle measurement. Finally, all calculated angles of important joints are rounded and stored in an array to facilitate subsequent data transmission. The data processing workflow is shown in Figure 6.

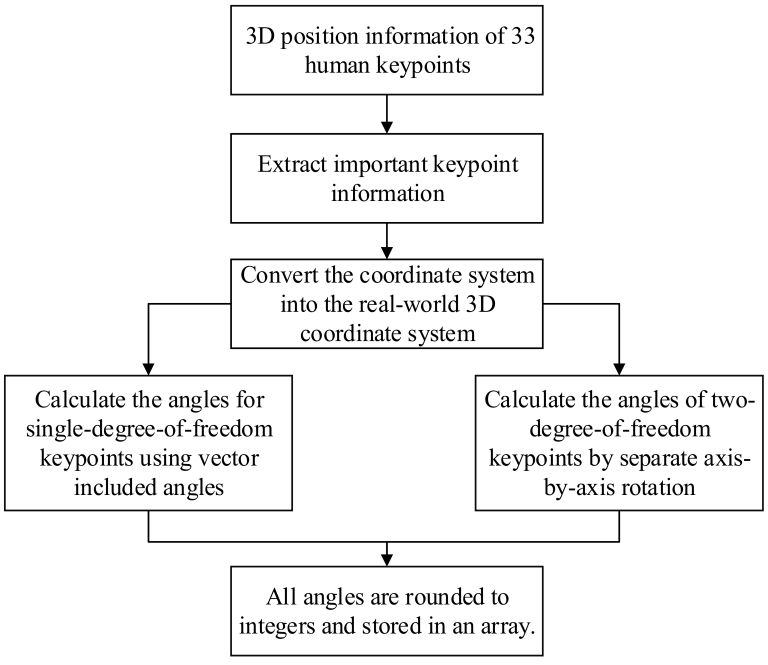


Figure 6. Flowchart of data processing and bone-joint angle calculation.

2.2.3. Algorithm Performance Analysis

Figure 7 shows the key-point recognition results of images captured by the computer front camera through the vision module. The human key-point recognition model adopted in this paper can rapidly estimate 26 major human key-points with high detection accuracy and favorable real-time performance. The model can recognize human joints even under partial occlusion. Conventional image recognition algorithms usually suffer from accuracy degradation caused by occlusion, and the excellent anti-occlusion capability is a prominent advantage of the adopted model.

Figure 7. Human key-point recognition effect of front camera images.

3. Robot Mechanical-Structure Design

3.1. Joint-Mechanism Design of the Dancing Robot

The BlazePose topology diagram (Figure 1) involves major human joints including shoulders, elbows, hips, knees and ankles. During robot structural design, the motion characteristics of these joints should be comprehensively considered to ensure that the designed mechanism can reproduce human joint movements.

Since the dancing robot only imitates simple human dance movements, the robot structure can be reasonably simplified:

(1) Elbow and Knee Joints:

When the robot performs dance movements, the elbow and knee joints mainly realize flexion-extension motion, and independent rotation of the forearm or crus rarely occurs. Therefore, the elbow and knee joints can be simplified as single-DOF joints.

(2) Shoulder and Hip Joints:

The shoulder and hip joints are typical multiaxial ball-and-socket joints capable of three-axis rotation. However, limited by the machine-vision pose recognition capability, the algorithm can only obtain the position information of joint key-points. For instance, the rotation of the upper arm about its longitudinal axis at the shoulder joint cannot be determined, and the same problem occurs at the hip joint. As a result, it is difficult to reproduce the axial rotation of the upper and lower limbs, and these joints are simplified to double-DOF configurations. Although this simplification may reduce the accuracy of motion imitation, a double-DOF structure allows the elbow or knee to reach any point on a sphere centered at the shoulder or hip joint with the connecting rod length as the radius. Therefore, simplifying the shoulder and hip joints into double-DOF joints has little impact on the overall function of the dance imitation robot.

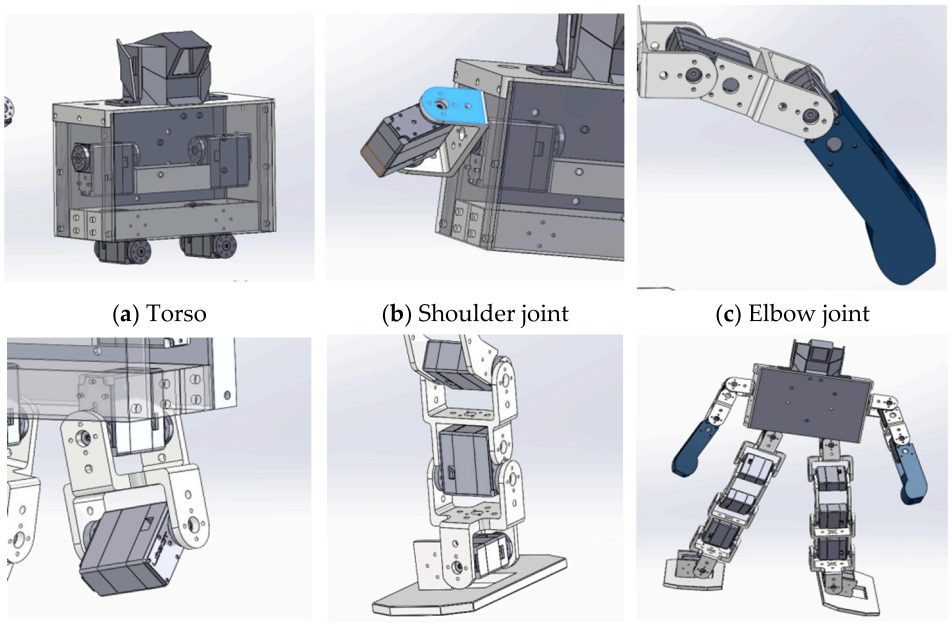
(3) Ankle Joint:

The ankle joint is a trochlear (hinge) joint with four primary motion modes: flexion, extension, inversion and eversion. Flexion denotes toe movement toward the body, whereas extension refers to toe movement away from the body; inversion represents foot rotation toward the medial side of the body, and eversion toward the lateral side. Therefore, the motion of the ankle joint can be regarded as double-DOF motion.

3.2. 3D Modeling of the Dancing Robot

SolidWorks2022 is adopted for the 3D modeling of the robot structure. Servos are used as joint actuators, and a servo-linkage mechanism is designed to endow the robot with better anthropomorphic characteristics.

The 3D robot model consists of four main parts: head, torso, upper limbs and lower limbs. The torso is assembled from five plates reinforced by connecting parts, where servos are mounted. The top plate fixes the robot head, and the internal space of the head accommodates the hardware of the vision system. All plates are fastened by bolts to form a box structure with reserved holes for circuit board installation and cable routing, as shown in Figure 8a. The structures of the shoulder, elbow, hip, knee and ankle joints are illustrated in Figure 8. Each servo realizes uniaxial rotation corresponding to one degree of freedom. The number of servos is configured according to the DOF requirement of each joint, and servos are connected by linkages to form the robot limbs. A drawing of the overall assembly of the robot is presented in Figure 8f. The robot is equipped with 16 servos in total, and the main body can be fabricated by assembly with 3D-printed connectors.



(**a**) Torso (**b**) Shoulder joint (**c**) Elbow joint

(**d**) Hip joints (**e**) Ankle joint (**f**) Overall assembly

Figure 8. Three-dimensional model of the dancing robot.

4. Robot Motion Solution

The motion trajectory of the dancing robot is planned according to the robot 3D model and machine vision model. The robot's movements are realized by the rotation of servos at each joint, so motion planning corresponds to the control planning of these 16 servos. Joint coordinate data are obtained using the BlazePose-based pose estimation algorithm. Developed by Google, BlazePose is a lightweight human pose-tracking model with an accuracy three times that of previous similar models. It supports on-device real-time human

pose-tracking, runs at over 30 FPS on Pixel 2 mobile phones, and detects 33 body landmarks. Virtual landmarks are adopted to improve stability for complex motions, enabling cross-platform deployment on mobile devices, Web and desktop terminals. Data processing and spatial geometric analysis are performed on the landmark coordinates to calculate joint angles. The joint angles extracted by visual recognition are used to control the rotation of corresponding servo motors on the dancing robot.

In short, if the joint angles acquired by the vision system are adopted to control the dancing robot, the motion trajectory planning problem can be divided into two parts: the control of the arm linkage from the shoulder joint to the end effector (hand), and the control of the leg linkage from the hip joint to the end effector (foot).

(1) Arm Motion Planning

In the arm motion model, the shoulder joint has two degrees of freedom controlled by two servos. Based on the coordinate information of the elbow joint, the pose estimation algorithm calculates two rotation angles θ_1 and θ_2 of the upper arm around the front vertical axis and the lateral vertical axis of the body, respectively. During robot control, the tiny error caused by the installation distance between the two shoulder servos can be ignored. Servo 2 rotates by angle θ_1 and Servo 1 rotates by angle θ_2, as shown in Figure 9. The elbow servo only needs to rotate to the joint angle obtained by machine vision recognition. With this planning method, the robot arm can effectively imitate human dance postures and realize bionic arm motion reproduction.

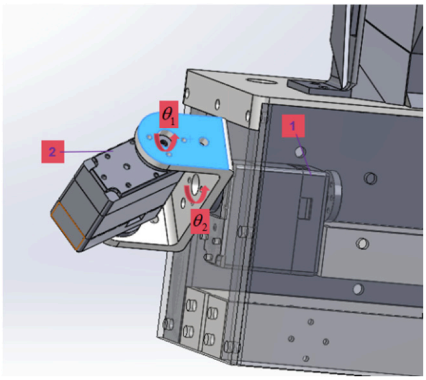


Figure 9. Motion planning of the shoulder joint.

(2) Leg Motion Planning

The hip joint in the leg model is designed with two degrees of freedom. Different from the linear arm linkage, the thigh structure is not a straight connecting rod, which increases the difficulty of leg motion planning. For the servo control of the hip joint, the pose recognition algorithm calculates two rotation angles of the thigh, namely angle θ_1 rotating around the vertical axis of the body side and angle θ_2 rotating around the vertical axis of the body front. To simplify the control process, a virtual line segment connecting the upper and lower servos of the hip joint is constructed to equivalently replace the physical thigh linkage. Accordingly, the hip joint servos are controlled by rotating Servo 2 by angle θ_1 and Servo 1 by angle θ_2. The motion planning principle is illustrated in Figure 10.

The motion planning of the knee joint is consistent with that of the elbow joint. The corresponding servo only needs to rotate to the knee joint angle obtained by machine vision recognition.

To realize ankle joint motion planning and prevent the robot from tipping over during movement, two motion constraints are defined: the foot remains in contact with the ground, and the torso stays perpendicular to the ground with the robot head pointing vertically

upward. With these constraints, ankle motion planning can be performed. Figure 11 shows a structural model of a single robot leg. The rotation axes of Servo 1 and Servo 5 are parallel, and the rotation axes of Servo 2, Servo 3, and Servo 4 are parallel. To satisfy the above constraints, the rotation of servos about the two axes should be controlled separately.

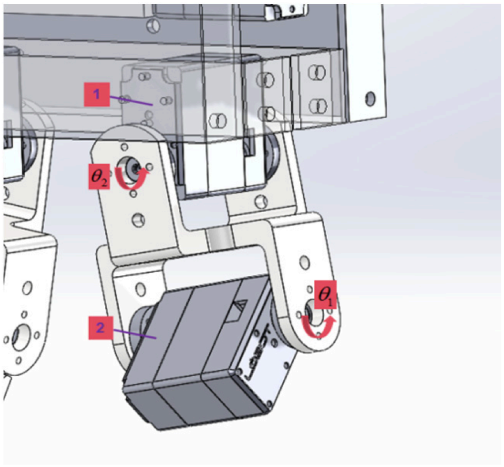


Figure 10. Motion planning of the hip joint.

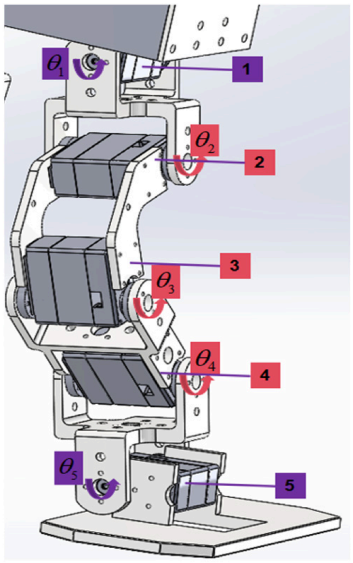


Figure 11. Leg model and angles of each servo.

For rotation about the axes of Servo 1 and Servo 5, qualitatively, the rotations of Servo 1 and Servo 5 must counteract each other to maintain the position of the torso and foot, i.e., $\theta_1 = -\theta_5$. Quantitatively, the leg model is simplified and projected onto the plane perpendicular to the rotation axes of Servo 1 and Servo 5 (Figure 11), and the following relation can be derived from geometric constraints: $\theta_1 = -\theta_5$.

For rotation about the axes of the remaining servos, the leg model is simplified and projected onto the plane parallel to the lateral side of the torso, as illustrated in Figure 12. The corresponding geometric relations are expressed as follows:

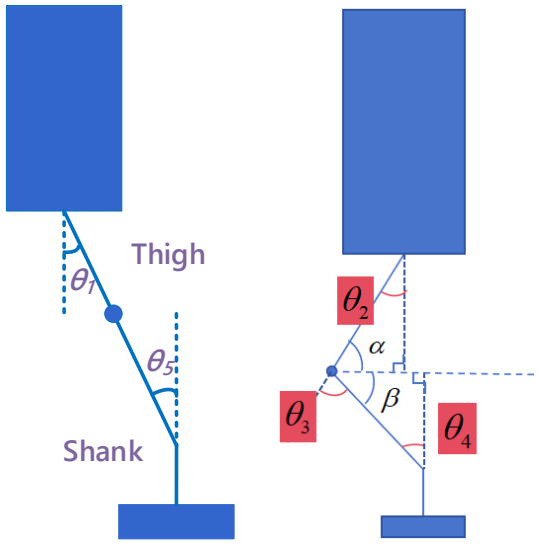
$$\begin{cases} \alpha = 90^\circ - \theta_2 \\ \beta = 180 - \alpha - \theta_3 \\ \theta_4 = 90 - \beta \end{cases} \quad (4)$$

The rotation angle of servo 4 is solved as

$$\theta_4 = \theta_3 - \theta_2 \tag{5}$$

Note that servo angles represent deflections relative to mechanical home positions, which are defined in lower-level firmware.

The above completes the motion planning for the arms and legs of the dancing robot. It should be noted that the servo rotation angle is defined as the deflection relative to its initial position. Hence, the initial position of each servo must be determined in advance for motion planning, and this parameter will be specified in the lower-level controller programming.



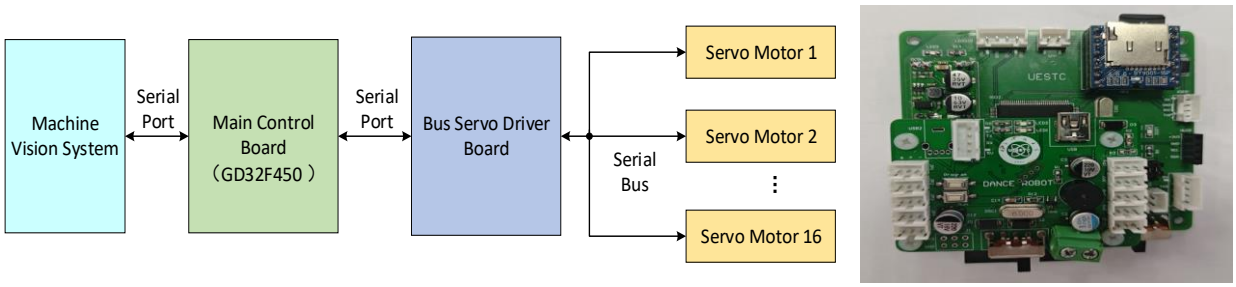
(**a**) Front projection; (**b**) lateral projection.

Figure 12. Projection of the leg model.

5. Control System Design and Experimental Verification

5.1. Control System Design

The control system of the dancing robot consists of a visual-recognition system, a main control board and a servo drive board [12–14]. The visual perception algorithm runs on a laptop PC rather than on-board embedded hardware. The monocular USB camera captures RGB image frames, and pose estimation is executed via MediaPipe-BlazePose on the PC. After pose-solving and joint-angle calculation, parsed control data are transmitted to the GD32F450 main-control board through serial communication. The main control board is built around the high-performance GD32F450 MCU (GigaDevice Semiconductor Inc., Beijing, China). Equipped with a 200 MHz Cortex-M4 core, this MCU supports a complete DSP instruction set, parallel computing capability and a dedicated floating-point unit. It achieves a computing performance of 250 DMIPS at the maximum clock frequency, and its code execution efficiency is 10–20% higher than that of other Cortex-M4 MCUs operating at the same frequency. After receiving the parsed human-pose data, the main MCU rapidly converts the data into servo control commands with its powerful computing capability, and sends these commands to the servo drive board via serial ports [15,16]. All joints of the dancing robot are driven by 16 serial bus servos. In this work, a serial servo drive board based on MM32F103 and CH559T is designed. Upon receiving serial instructions from the main MCU, the drive board actuates the corresponding servo groups to perform target motions. The hardware of the dancing robot control system is shown in Figure 13.



(**a**) Block diagram of the dancing robot control system. (**b**) Main control circuit board.

Figure 13. Hardware of the dancing robot control system.

5.2. Data Processing and Control Logic Design

During robot kinematic solving, the included angles of key joints obtained via pose recognition are stored in an array. A data encapsulation scheme is designed herein: each angular value is converted into a four-character string with a positive or negative sign. These strings are concatenated sequentially to form a string with a total length of 12×4. Characters *S* and *E* are added at the start and end of the combined string respectively to construct the serial data packet transmitted from the upper computer to the lower controller. An example format is 'S + 020 + 030. . . - 120 - 090E'. After receiving the complete string through serial interrupts, the lower-level program parses the packet to restore the target angles of each servo. A cyclic extraction method is adopted to obtain the rotation angle of each joint in sequence. Afterwards, these angles are converted into servo control commands, which are transmitted by the microcontroller to the drive board via serial ports.

In the practical working range of 1.2–2.2 m (human–camera distance for dance imitation), the pseudo-3D key-point depth output from monocular BlazePose exhibits an average absolute depth error of 85 mm-142 mm, and depth error increases with growing camera–subject distance. Such depth deviation propagates to joint-angle computation, introducing additional joint-angle error within $\pm 4.2^{\circ}$, which is included inside the total joint-angle error reported in Table 1.

Table 1. Statistics of joint angle errors under different dance movements.

Test Movement	Maximum Angle Error (°)	Average Angle Error (°)
Hands on Hips	4.1	2.2
One-arm Horizontal Lifting	4.5	2.6
T-pose Standing	3.8	1.9
Inward Arm Flexion	4.7	2.9

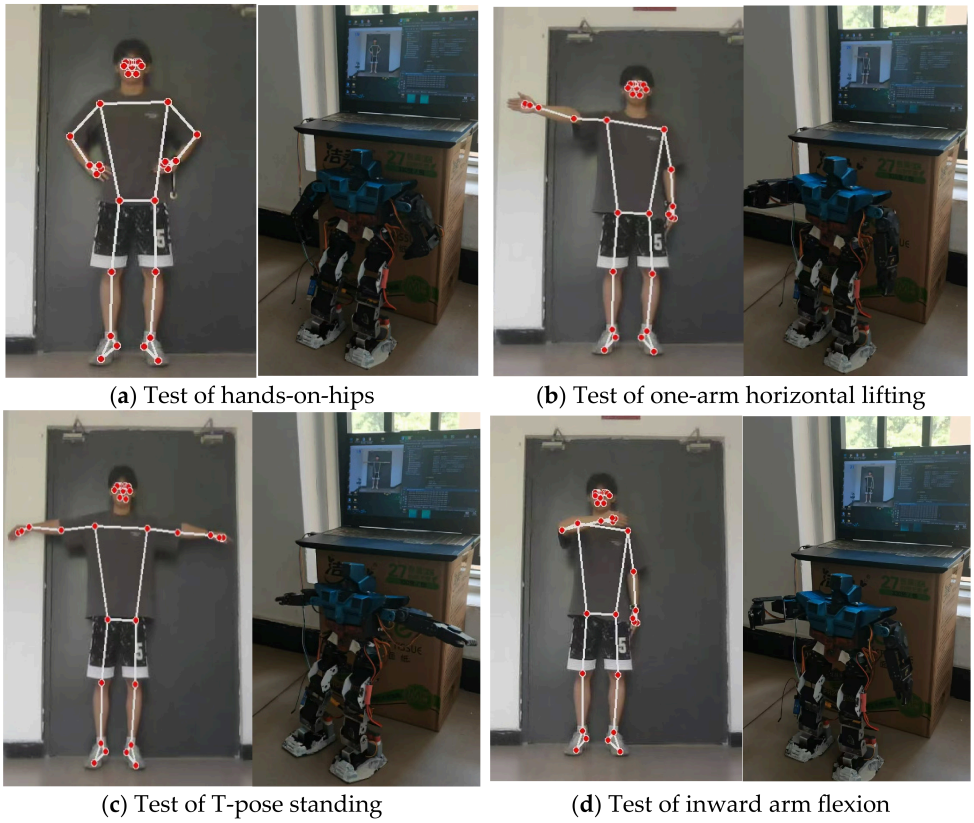
Note: Reference angles are derived from BlazePose outputs, not independent motion-capture ground truth.

A simple exponential moving average (EMA) filter is applied to raw key-point coordinates frame-by-frame before joint-angle calculation, to suppress high-frequency key-point jitter caused by image noise. The filter coefficient is set to 0.75. This filtering operation avoids servo chattering triggered by rapid frame-to-frame key-point fluctuation.

Each servo has predefined mechanical rotation limits. When the computed target joint angle exceeds the hardware motion range, a saturation-clipping strategy is adopted: the target angle will be clamped to the nearest hardware-allowable boundary value before sending serial commands to servos. Meanwhile, the system records clipping events for offline analysis.

5.3. Experimental Verification

To quantitatively evaluate the motion imitation accuracy, real-time performance and trajectory reproduction capability of the dancing robot system, four groups of typical simple dance movements, namely hands on hips, one arm raised horizontally, standing in a "T" pose and inward arm flexion, are selected as test samples. Each movement is repeated 10 times to reduce random test errors. The tester demonstrates standard movements within the field of view of the monocular camera. The system adopts the BlazePose model to capture 33 human skeletal key-points and calculate the target joint angles. The GD32F450 lower-level main control board receives serial commands and drives the 16 bus servos to reproduce the movements. In the experiment, a high-precision angle acquisition module is used to record the actual rotation angle of each robot servo synchronously. The end-to-end response latency of the whole system is calculated combined with timestamps from the upper computer, and the joint trajectory data of both human and robot are recorded simultaneously. The experimental setup is shown in Figure 14.



(**a**) Test of hands-on-hips (**b**) Test of one-arm horizontal lifting

(**c**) Test of T-pose standing (**d**) Test of inward arm flexion

Figure 14. Experimental test for visual recognition of dance postures.

5.3.1. Angle Error Test Results

Eight major moving joints of the robot, including the left and right shoulder joints, elbow joints, hip joints and knee joints, are selected for statistical analysis. Each of the four movements is tested ten times. The maximum angle error and average angle error of each joint under different movements are calculated, and the statistical experimental results are shown in Table 1.

As shown in Table 1, the maximum joint angle error of all test samples is 4.7°, and all joint angle errors are within ±5°. Large-range limb movements such as T-pose standing achieve a lower average error, while local small-bending movements like inward arm flexion produce relatively larger errors. This phenomenon is mainly caused by noise in monocular 3D reconstruction, servo mechanical clearance, and the structural difference be-

tween human flexible skeletons and rigid robot links. The overall error can meet the visual requirement for reproducing simple dance movements, which verifies the effectiveness of the proposed posture calculation and servo angle mapping strategy.

It should be clarified that the reference joint-angle values used for Table 1 are derived from BlazePose pseudo-3D outputs from the same input video stream, rather than from independent motion-capture ground truth. Therefore, the reported angle error mainly reflects the mechanical tracking fidelity of robot servos toward vision-computed target angles, i.e., how accurately the robot hardware executes the target angles given by the visual algorithm, instead of the pose estimation error relative to real human anatomical joint angles. This limitation is further discussed in Section 6.1.

5.3.2. Response Latency Test Results

The end-to-end response latency of all test samples for the four movements is counted, and the statistical results are shown in Table 2.

Table 2. Statistics of system response latency.

Test Movement	Minimum Latency/ms	Maximum Latency/ms	Average Latency/ms
Hands on Hips	58	86	70.4
One-Arm Horizontal Lifting	62	91	74.6
T-Pose Standing	54	82	67.2
Inward Arm Flexion	65	95	78.1

The maximum response latency among all samples is 95 ms, and the response latency of all test samples is less than 100 ms. The latency components are broken down as follows: BlazePose pose inference takes approximately 22–35 ms, angle calculation and data encapsulation take about 8–12 ms, serial communication transmission consumes roughly 4–6 ms, and servo mechanical actuation latency ranges from 20 to 42 ms. The inward arm flexion movement requires the simultaneous motion of more joints, which increases the servo load and results in slightly higher latency compared with other movements. The overall system achieves favorable real-time performance without obvious visual lag between human and robot motions.

5.3.3. Analysis of Motion Trajectory Consistency

The original human joint trajectories and reproduced robot joint trajectories are recorded by the upper computer to calculate the normalized trajectory coincidence degree. The trajectory coincidence degrees of the four tested movements are as follows: 93.1% for hands on hips, 91.7% for one-arm horizontal lifting, 94.5% for T-pose standing, and 90.2% for inward arm flexion. Large-range limb movements achieve higher trajectory coincidence. Affected by visual noise and servo mechanical clearance, small-bending movements show slightly increased trajectory deviation. Overall, the robot trajectories are highly consistent with the original human motions, and the robot can reproduce the posture characteristics and motion trends of dance movements.

The normalized trajectory coincidence degree C between human joint trajectory $H(t)$ and robot joint trajectory $R(t)$ is defined as

$$C = \left(1 - \frac{\frac{1}{N}\sum_{t=1}^{N}|H(t) - R(t)|}{H_{range}}\right) \times 100\% \quad (6)$$

where

N—total number of sampled frames within one test sequence;

$H(t)$—human joint angle at the t-th frame (°);

$R(t)$—measured actual joint angle of robot at the t-th frame (°);

H_{range}—full motion range of the corresponding human joint in this test sequence, $H_{range} = \max(H(t)) - \min(H(t))$.

The value of C is clipped to $0\% \leq C \leq 100\%$. When the robot trajectory is completely consistent with human trajectory, $C = 100\%$. Larger deviation between two trajectories yields smaller C.

5.3.4. Comparative Experiment of Different Pose Recognition Schemes

To further verify the comprehensive performance of the BlazePose monocular vision scheme adopted in this paper for dance imitation scenarios, comparative experiments with the traditional OpenPose pose estimation scheme and template-matching pose-recognition scheme are carried out under identical hardware prototypes and the same four groups of test movements. The comparison indicators include maximum angle error, average response latency and average trajectory coincidence degree. The comparison results are shown in Table 3.

Table 3. Performance comparison of different pose recognition schemes.

Scheme	Maximum Angle Error (°)	Average Latency/ms	Average Trajectory Coincidence Degree/%
Template-Matching Pose Recognition	8.8	147	76.3
OpenPose Pose Estimation	6.4	121	84.6
Proposed BlazePose Scheme	4.7	72.6	92.4

Note: OpenPose-body-25 model is adopted for comparison. It runs on PC CPU (Intel i7-10750H), input image resolution 640 × 480. Template-matching algorithm also runs on the same PC platform with identical input resolution. All three schemes share identical input video frames for fair comparison.

The test results show that the traditional template-matching method has poor adaptability to motion variation, large angle error and the worst real-time performance. OpenPose achieves acceptable recognition accuracy, yet it involves heavy computational load and yields an overall response latency over 100 ms, which cannot satisfy real-time imitation requirements. The lightweight BlazePose-based scheme proposed in this paper limits the maximum angle error within ±5° with an average response latency of only 72.6 ms, and obtains markedly higher trajectory coincidence degree compared with the other two schemes. Benefiting from the lightweight inference capability of BlazePose, combined with the vector angle calculation and servo angle mapping algorithm proposed in this paper, the scheme balances recognition accuracy and real-time performance on low-cost monocular vision hardware, making it more suitable for real-time motion imitation of dance robots in teaching scenarios.

6. Conclusions and Limitations

6.1. System Limitations

This system has several notable limitations.

(1) Monocular-Vision Anti-Interference Performance: Under strong lighting variation, heavy background clutter, or severe limb partial occlusion during dancing, BlazePose key-point visibility drops, which will increase joint-angle error by up to ±8–11°. Our current system works best under moderate-indoor-illumination and mild-occlusion scenarios.

(2) Axial Rotation Distortion: Since monocular BlazePose cannot capture limb axial rotation, for dance movements requiring upper-limb/lower-limb axial twisting, imitation

distortion can reach 12–18°, which means such torsional movements cannot be faithfully reproduced by our 2-DOF simplified shoulder/hip joints.

(3) Statistical Repeatability: Each test movement is repeated for 10 trials. The standard deviations of joint-angle error across repeated trials are within 0.4–0.9°, and the standard deviation of trajectory coincidence degree lies within 1.1–2.3%. Individual trial raw data and standard-deviation statistics for repeated experiments are provided in Supplementary Materials.

(4) The ground-truth reference for human joint angles lacks independent motion-capture equipment, which is a limitation of our current experimental setup.

6.2. Conclusions

This paper deeply integrates deep learning, machine vision, mechanical structure design and motion control technology, and completes the design and implementation of a full machine vision-based dance imitation robot system. Based on the Python development environment, the software algorithm framework is constructed. The lightweight BlazePose deep learning model is adopted to realize real-time detection of 33 human skeletal keypoints from monocular images. Combined with 2D-to-3D reconstruction and inverse kinematics algorithms, human joint angles are accurately solved. In terms of mechanical structure, 16 servos are deployed to construct a humanoid joint motion structure. On the hardware platform, a laptop running the visual perception algorithm serves as the vision module. It communicates with the GD32F450 main-control chip via serial communication to realize cooperative circuit control. Correspondingly, the software system completes visual data analysis, motion command generation and system closed-loop control, enabling the robot to reproduce human dance movements.

Experimental results on multiple typical dance movements demonstrate that the proposed robot can stably imitate basic actions including hands on hips, one-arm horizontal lifting, T-pose standing, and inward arm flexion. The joint angle error is controlled within $\pm 5^{\circ}$ and the end-to-end response latency is less than 100 ms. The reproduced trajectories are highly consistent with original human motions, achieving satisfactory recognition accuracy and real-time performance, which meets the real-time imitation requirements of small humanoid robots. Compared with the traditional template-matching method and OpenPose scheme, the proposed algorithm exhibits significant advantages in imitation precision, system real-time performance and trajectory reproduction stability. It balances low-cost embedded deployment and visual imitation quality, and can provide a reliable reference for related research and the engineering applications of visual motion-following and reproduction for small humanoid robots.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/electronics15184161/s1. We have provided the raw data of 10 repeated trials for joint-angle error and trajectory coincidence degree in Supplementary Material S1.

Author Contributions: Conceptualization, J.G., D.L., C.L., Y.L. and D.G.; methodology, J.G., D.L., C.L. and Y.L.; software, J.G., C.L. and D.L.; validation, J.G. and C.L.; formal analysis, J.G. and C.L.; data curation, J.G., D.L., C.L. and Y.L.; writing—original draft preparation, J.G. and D.L.; writing—review and editing, J.G., D.L., Y.L. and D.G.; visualization, D.L.; supervision, J.G., D.L. and D.G.; project administration, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Fundamental Research Funds for the Central Universities (Project Nos. 2682025CX080, 2682025XJ007).

Data Availability Statement: The original contributions presented in this study are included in the article/Supplementary Materials. Further inquiries can be directed to the corresponding author.

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

References

1. Zhu, Y. Current situation and enlightenment of humanoid robot industry development. *Sci. Technol. Ind.* **2023**, *23*, 136–141.
2. Zhang, J.; Liu, J.; Zong, H.; Ji, P.; Fang, L.; Li, Y.; Yang, H.; Xu, B. Bridging the Gap to Bionic Motion: Challenges in Legged Robot Limb Unit Design, Modeling, and Control. *Cyborg Bionic Syst.* **2025**, *6*, 0365. [CrossRef] [PubMed]
3. Chen, C.; Huang, X.F.; Zhou, X. Experiment on pose calculation and trajectory control of humanoid robot based on monocular vision. *Exp. Sci. Technol.* **2022**, *20*, 56–61.
4. Lin, J.; Chen, X.R.; Zhu, X.Q. Design of human pose recognition experimental system based on deep learning. *Res. Explor. Lab.* **2021**, *40*, 67–72.
5. Dong, H. Research on Weak Light Human Pose Estimation Based on Improved Feature Fusion Strategy. Master's Thesis, Xidian University, Xi'an, China, 2021.
6. Deng, Y.N.; Luo, J.X.; Jin, F.L. A review of human pose estimation methods based on deep learning. *Laser Optoelectron. Prog.* **2021**, *58*, 69–88.
7. Feng, Y. Research on Human Pose Recognition Algorithm Based on Deep Learning. Master's thesis, Harbin Engineering University, Harbin, China, 2024.
8. Lü, S.J.; Qi, Y.M.; Deng, S.P.; Xia, Y.H.; Liu, H. Research on human motion recognition algorithm based on 3D skeleton data. *Equip. Manuf. Technol.* **2022**, *9–11*, 30.
9. Bazarevsky, V.; Grishchenko, I.; Raveendran, K.; Zhu, T.; Zhang, F.; Grundmann, M. BlazePose: On-device Real-time Body Pose tracking. *arXiv* **2020**, arXiv:2006.10204.
10. Yang, L.F.; Zong, Z.W. Research on human joint recognition of tennis serve based on BlazePose. *Ind. Control Comput.* **2023**, *36*, 115–117+120.
11. Yang, X.; Sun, M.; Li, G. Visual experimental research on robot motion imitation based on MediaPipe. *Exp. Sci. Technol.* **2023**, *21*, 44–49.
12. Wei, H.L.; Shang, Y.T.; Sun, S.; Kong, X.Z.; Wang, J.; Liu, C.H. Design of industrial robot sorting experimental platform based on machine vision. *Exp. Sci. Technol.* **2021**, *19*, 132–137.
13. Luo, J.; Chen, J.H.; Peng, Z.X.; Li, J.; Wu, W.N.; Zhou, G.B. Robot grasping experimental system based on machine vision. *Exp. Technol. Manag.* **2022**, *39*, 45–50.
14. Wang, Q.; Li, L.J.; Liu, J. Design of robot visual positioning experimental teaching platform based on OpenCV. *Res. Explor. Lab.* **2020**, *39*, 112–116.
15. Zhao, L.; Wang, F.; Wu, J. Development of experimental platform for humanoid robot joint control driven by servo motors. *Exp. Technol. Manag.* **2020**, *37*, 89–93.
16. Zhou, M.; Liu, C.; Huang, W. Experimental design of servo closed-loop control for robot based on embedded MCU. *Res. Explor. Lab.* **2022**, *41*, 98–102.