

Enhancing the Naturalness of Game Character Movements Through the Fusion of Skeletal Rigging and Physics Engine Technology

Gaofeng Su

College of Arts, Beijing Union University, Beijing 100101, China

Corresponding author: Gaofeng Su (e-mail: guaguakanword@163.com)

ABSTRACT Traditional animation techniques often produce stiff character movements and insufficient physical feedback, especially when simulating virtual fabrics, clothing, and flexible structures. This limitation affects not only visual realism but also the digital representation of wearable antenna substrates, flexible electromagnetic components, and human-body interaction scenarios in electromagnetic wave applications. To address this problem, this paper proposes a character movement driving system based on the fusion of skeletal rigging and physics engine technology. Dynamic constraints are established by setting rotation, elasticity, and damping parameters for bone nodes and combining them with the Inverse Kinematics (IK) algorithm. The physics engine calculates external forces and inertial feedback in real time and transfers the results to the skeletal system for adaptive motion adjustment. A Proportional-Derivative (PD) controller is introduced to coordinate dynamic balance between the animation layer and the physics layer, while a lightweight neural network prediction module is used for posture optimization. In addition, a multi-threaded rendering synchronization mechanism is designed to integrate movement, physical response, and visual output. Experimental results show that the proposed method keeps the standard deviation of joint angular acceleration below 1000 rad/s², controls the horizontal centre-of-gravity offset within 5.2 cm in collision scenarios, reduces the average action response delay to 35 ms under thrust, and achieves a subjective naturalness score of 7.2–8.5. The method improves the smoothness, stability, and real-time expressiveness of character movement, and provides a simulation reference for flexible textiles and wearable electromagnetic structures.

INDEX TERMS skeletal rigging, physics engine, motion prediction, wearable antennas, flexible electromagnetic structures

I. INTRODUCTION

REALISM and naturalness in 3D character animation remain central challenges in real-time interactive applications, such as video games and virtual simulation. Achieving authentic motion requires not only smooth visual transitions but also physically plausible responses to environmental interactions, including gravity, collisions, and dynamic forces acting on the body. Traditional animation pipelines—relying on key-frame interpolation and motion capture—provide high-fidelity kinematic sequences but often lack consistent physical feedback, resulting in unnatural inertia, stiff posture shifts, and inadequate balance recovery under external disturbances [1,2]. The simulation of dynamic clothing and virtual textiles further amplifies this challenge, as it demands tight coupling between skeletal motion and physics-based deformation [3,4]. Consequently, characters may appear detached from their surroundings, reducing immersion and visual coherence.

When characters engage in complex actions such as falling, running, or being pushed, traditional systems struggle to simulate appropriate physical reactions, particularly in terms of limb coordination and balance adjustment [5,6]. As interactive applications strive for greater realism, the need for animation methods that are both visually smooth and physically accurate continues to drive research in real-time character animation [7,8].

Early studies on skeletal animation focused on interpolation methods and techniques for improving motioncapture fidelity. The most common method for driving skeletal animations is bone binding, in which bones and their weights remain static, producing keyframe animations that match motion-capture data. Tsai et al. [9] created a baseball throwing simulator using the Unity Engine and digital gloves, which allowed the use of real-time motion data to simulate motion. Gao [10] conducted a study that investigated how to create immersive motion sensing interactive games and

also how to enhance the human-computer interface in virtual environments.

This paper proposes an integrated motion-driving system that fuses skeletal rigging with a physics engine to enhance the naturalness and physical consistency of character motion, including the dynamic behavior of virtual apparel. The core contributions include: (1) a parameterized skeletal constraint layer with dynamic damping and elastic response; (2) a real-time physics-feedback pipeline that maps rigid-body dynamics to joint-level torques; (3) a PD-based fusion controller for balancing kinematic targets with physical perturbations; (4) a lightweight neural predictor for adaptive pose optimization; and (5) a multi-threaded rendering synchronization mechanism for visual-physical coherence. Experimental results demonstrate significant improvements in joint smoothness, force-response consistency, balance stability, and subjective naturalness, validating the effectiveness of the proposed approach for real-time character animation.

II. CONSTRUCTION OF THE SKELETAL PHYSICS FUSION DRIVING SYSTEM

A. MULTIDIMENSIONAL PARAMETER BINDING OF THE SKELETAL CONSTRAINT LAYER

1) Skeletal Node Parameter Definition and Dynamic Constraint Construction

The architecture of the character motion driving system based on bone binding and physics engine fusion is shown in Figure 1.

Figure 1 shows the overall architecture of the character motion driving system proposed in this article. The system achieves deep integration of skeletal animation and physical simulation through multi-layer modular design: the input layer receives animation data and external forces; the skeletal constraint layer parameterizes the modeling of joints and combines inverse kinematics to solve the pose. The physics engine layer calculates the force on the rigid body in real-time and feeds it back to the skeletal system. The fusion control layer utilizes a PD controller to achieve dynamic balance and centroid correction between the animation layer and the physical layer. The neural prediction module optimizes the next frame pose based on historical motion data. The multi-threaded rendering mechanism processes physical calculations, pose updates, and graphics rendering in parallel, and achieves state synchronization and visual output through GPU. The figure clearly indicates the data flow and interaction relationship between each module, reflecting the core mechanism of the system to improve motion naturalness and physical consistency while maintaining high real-time performance.

The core task of the skeletal constraint layer is to establish an accurate node control model as shown in Figure 2, enabling the character to maintain a stable posture under external forces. To achieve this goal, each joint node of the character skeleton is first parameterized, setting the rotation angle θ_i , elastic coefficient k_i , and damping coefficient c_i ,

and storing the constraint relationships between nodes in matrix form within the system.

The bone hierarchy adopts a parent-child structure for computational propagation [11]. The rotation matrix R_p of the upper node and the relative transformation matrix T_c of the lower node satisfy the constraint relationship, as shown in Formula 1:

$$R_c = R_p \times T_c \quad (1)$$

R_c represents the real-time rotation state matrix of the child node. In each frame calculation, the system dynamically adjusts θ_i according to the external force information and applies a viscous damping model in the elastic response function, as shown in Formula 2:

$$\tau_i = -k_i(\theta_i - \theta_{0i}) - c_i\dot{\theta}_i \quad (2)$$

In the formula, τ_i is the torque on the node; θ_{0i} is the static equilibrium angle; $\dot{\theta}_i$ is the angular velocity. This model establishes a direct mapping between rotation and force in the bone layer, so that the node motion is both constrained and has physical feedback characteristics.

In the calculation process, in order to prevent high-frequency oscillation from causing instability of posture, the system applies a limiting function for the joint angular velocity and sets an adaptive time step update mechanism to keep the parameter response convergent in the high dynamic scene. For joints with complex torsional structures (such as the shoulder and hip), quaternion interpolation is used for rotational accumulation to avoid gimbal lock-up problems under Euler angle representation. The rotation solutions for all nodes are completed in the local coordinate system and then uniformly transformed to the global coordinate space to achieve structural continuity between multi-layer skeletal motions [12,13]. Through this method, the skeletal constraint layer can output physically interpretable posture changes while maintaining computational stability.

Figure 3 illustrates the dynamic response of character joints under combined skeletal constraints and physics engine feedback. Sub-figure (a) plots joint angle variation over time, showing damped oscillation decay after an external impulse—indicating stable posture recovery. Sub-figure (b) displays corresponding joint-torque responses, where peaks and decay rates reflect parameter-dependent energy absorption. Notably, hip and knee joints exhibit higher torque sensitivity, confirming their role in maintaining balance under perturbation. Together, these curves validate the effectiveness of the skeletal-parameter tuning and physics-coupling mechanism in achieving stable, physically consistent motion.

2) Attitude Balance Solution Based on Inverse Kinematics

After establishing the parameterized constraints, in order to maintain the attitude balance of the skeletal system under complex force conditions, the system applies an inverse kinematics (IK) solution module. This module takes the

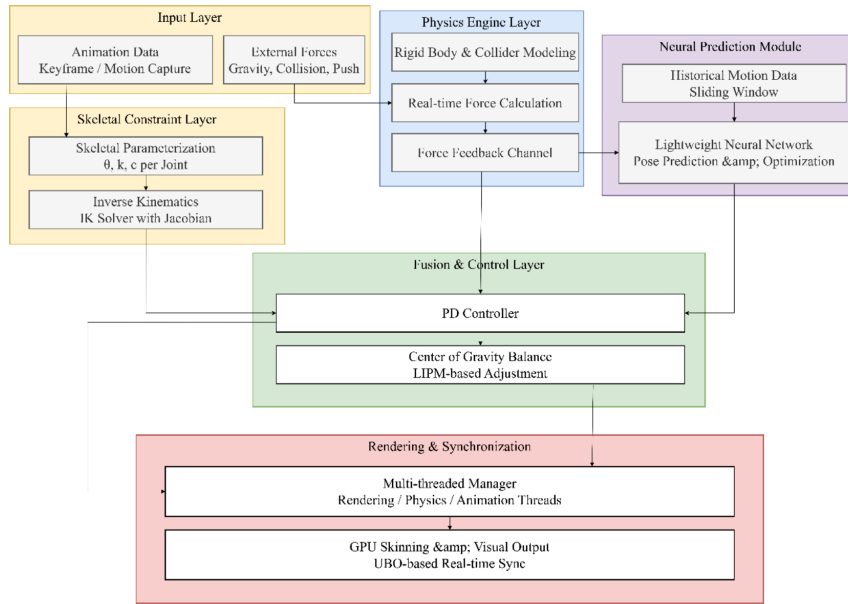


FIGURE 1. Architecture of Character Motion Driving System Based on Skeleton Binding and Physics Engine Fusion

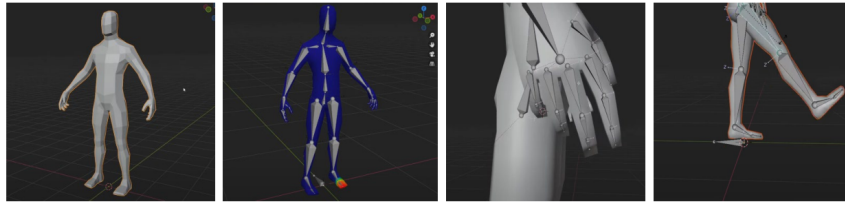


FIGURE 2. The process of constructing bones in character modeling

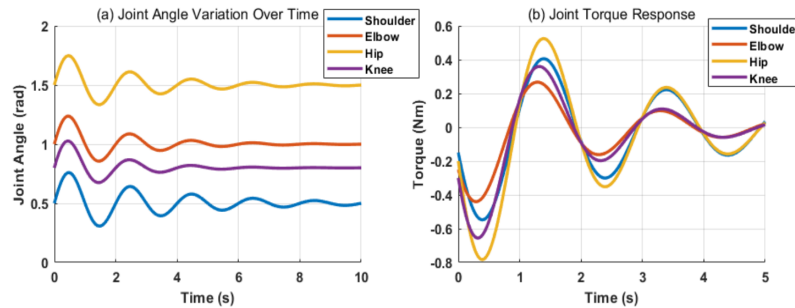


FIGURE 3. Analysis of joint response characteristics under skeletal constraint driven; Figure 3(a). Joint angle variation curve over time; Figure 3(b). Dynamic response curve of joint torque over time

deviation between the target position P_t of the character and the position P_c of the end effector as input, and calculates the joint angle adjustment $\Delta\theta$ through the Jacobian matrix J , as shown in Formula 3 [14]:

$$\Delta\theta = J^+(P_t - P_c) \quad (3)$$

J^+ is the generalized inverse matrix, which is used to minimize the position error under the constraint conditions. During the solution process, the IK module uses the physical constraint parameters of the skeletal nodes as boundary conditions and dynamically adjusts the angle of each joint to ensure that the final attitude satisfies both the kinematic

constraints and the physical response. In order to enhance the computational stability, the damped least squares (DLS) method is used to regularize the Jacobian matrix so that the solution can maintain smooth convergence in the high degree of freedom link.

B. REAL-TIME FORCE FEEDBACK CALCULATION OF THE DYNAMICS ENGINE

1) Modeling and Updating of Rigid Bodies and Colliders

The dynamics engine module undertakes the task of real-time calculation of force data during character movement. Its first step is to establish accurate models of rigid bodies and colliders. The main parts of the character's body are

decomposed into rigid-body units according to the skeletal topology, and each unit is approximated by a convex polyhedron. The mass m_i and the inertia tensor I_i are calculated based on the geometric volume and density distribution. The collision detection module performs spatial division based on the generalized bounding volume hierarchy (BVH) to reduce the number of invalid detections and updates the contact state between the rigid body and the scene in each frame operation. Each rigid body node independently calculates the force composition in the physics engine. The system records gravity $F_g = m_i g$, external force F_{ext} , collision reaction force F_c , and friction force F_f , and updates the linear velocity and angular velocity of the rigid body by integrating them according to the time step Δt [15], as shown in Formula 4:

$$v_i(t + \Delta t) = v_i(t) + \frac{F_{\text{total}}}{m_i} \Delta t \quad (4)$$

$F_{\text{total}} = F_g + F_{\text{ext}} + F_c + F_f$. The friction model is based on Coulomb friction, where the static (μ_s) and dynamic (μ_d) friction coefficient are adaptively adjusted according to different material contact conditions, ensuring physically consistent velocity variation during ground motion.

2) Force Feedback Transmission and Skeleton Adaptive Correction

After completing the rigid body force calculation, the system transmits the force vector and torque information output by the physics engine to the skeleton constraint layer through the force feedback channel. Each skeleton node corresponds to a rigid body instance, and the two are bound by a unique index during the initialization phase. During the feedback phase, the angular velocity and position correction of the bone nodes are updated according to the time-step synchronization strategy to ensure that the bone posture is consistent with the rigid body motion in the physics engine. The force feedback conversion module uses a local coordinate mapping method to project the rigid body force results F_i and torque τ_i onto the local coordinate system of the bone to eliminate the cumulative error caused by coordinate offset [16,17]. When the bone system is updated, the joint angles are corrected according to the feedback torque, as shown in Formula 5:

$$\theta_i(t + \Delta t) = \theta_i(t) + \frac{\tau_i}{I_i} \Delta t \quad (5)$$

I_i represents the moment of inertia of the corresponding node. When there is a sudden change in the external force that exceeds a certain threshold, the local buffering mechanism filters out high-frequency signals through low-pass filters to prevent jittering of the skeleton. The feedback is updated asynchronously, resulting in temporal decoupling between the calculations of the physics engine and the rendering of the animation. This reduces the frame delay that occurs due to feedback from the physics engine and increases the accuracy of the character's movement.

The rigid body angular velocity ω and linear acceleration a calculated by the physics engine need to be converted into equivalent torque in the local coordinate system of the skeleton node to drive the posture correction of the skeleton. This transformation is achieved through the following steps: Firstly, the angular velocity ω of the rigid body is mapped to the local coordinate system of the corresponding bone node based on the binding relationship, obtaining the local angular velocity. Subsequently, using the inertia tensor and angular acceleration rate of the node, calculate the required torque $\tau_i = I_i \cdot \dot{\omega}_{i(\text{local})}$.

C. DYNAMIC BALANCING AND FUSION BASED ON PD CONTROLLER

1) Fusion Layer Structure Construction and Control Signal Transmission

The motion fusion layer is designed to harmonize animation-driven motion signals with physical feedback. This enables the character to maintain a stable posture while responding to external forces. The motion fusion layer is positioned in between the module for calculating animations into keyframes to produce target postures for the character and the skeletal control layer that maintains the skeletal structure of the character through physical force influences acting upon it, as received from real-time dynamic simulation of the character. The motion fusion layer uses data from both types of signals to perform motion fusion control based on the Unit of Time in simulation. The system first reads the target joint angle θ_{target} and angular velocity $\dot{\theta}_{\text{target}}$ output from the animation layer in each frame calculation, and simultaneously obtains the current state θ_{current} , $\dot{\theta}_{\text{current}}$, and force vector F_{cxt} of the skeletal nodes from the physics engine. The fusion layer constructs control signals based on the error term to drive the PD controller to generate corrective torque input.

In the fusion system proposed in this article, the inverse kinematics module and the proportional derivative controller undertake different stages and cooperate with each other in attitude control responsibilities. The IK module is mainly responsible for calculating the ideal expected angles of each joint based on high-level animation instructions at the motion generation level, in order to set the desired static or kinematic optimal posture. Its calculation process is based on geometric constraints and joint motion range, and does not directly handle the dynamic effects of physical forces.

The PD controller is located at the level of physical response and dynamic fusion, and its core responsibility is to calculate the required correction torque based on real-time external force, inertia, and current joint state feedback from the physics engine in each frame, driving the skeletal system to track and stabilize at the desired posture given in the physical simulation environment.

2) Attitude Stability Calculation and Center of Gravity Correction Based on PD Control

The core control algorithm of the fusion layer uses a proportional-derivative control model (PD Controller) to correct the character's attitude in real time. Each controlled node calculates the output torque according to the following formula, as shown in Formula 6:

$$\tau_i = K_p(\theta_{\text{target},i} - \theta_{\text{current},i}) + K_d(\dot{\theta}_{\text{target},i} - \dot{\theta}_{\text{current},i}) \quad (6)$$

K_p is the proportional gain, which controls the response intensity of the attitude deviation; K_d is the derivative gain, which is used to suppress the oscillation caused by the change in angular velocity. The calculation of this controller is performed in parallel structure of Graphics Processing Unit (GPU) to ensure that multiple nodes respond simultaneously. The proportional term is used to quickly adjust the joint angle to the target range, and the derivative term is used to reduce attitude vibration under external force disturbance. The system dynamically adjusts the gain parameters according to the character's motion state in each frame. When the change in angular velocity exceeds the threshold, K_p is automatically reduced, and K_d is increased to prevent attitude correction overshoot [18].

After the attitude correction is completed, the system calculates the distance between the overall center of gravity position Cg of the character and the projection distance of the support plane. If the center of gravity is detected to be outside the range of the supporting polygon, the fusion layer immediately triggers the center of gravity balancing module. This module redistributes the lower limb bone angles based on the direction of the external force and the character's orientation, so that the support points form a new balance structure. The calculation uses a simplified center of gravity formula based on the Linear Inverted Pendulum Model (LIPM), as shown in Formula 7:

$$\ddot{x}_c = \frac{g}{z_c}(x_c - p) \quad (7)$$

x_c is the horizontal coordinate of the center of gravity; z_c is the height of the center of gravity; p is the position of the support point. The fusion layer corrects the value of p by adjusting the angle of the lower limb nodes, so that the system acceleration term tends to zero, thereby restoring stability.

The overall control of the character in dynamic balance is performed at the same time in time as the physics engine. Each time the engine is updated, four stages of processing occur simultaneously; they are control signal generation, posture control, center of mass adjustment, and synchronization via feedback [19]. The fusion layer contributes to maintaining the overall smoothness of the character's movements and conformity to the principles of mechanical balance control during periods of external force application or the character moving through space dynamically.

The critical elements of motion control in both adjusting dynamically and ensuring stability are depicted graphically in Figure 4, showing the basic mechanics of how a controller alters the motion of a game character while also stabilizing the controller. In Figure 4(a), the horizontal axis is time, while the vertical axis displays normalized responses. For different combinations of control gain using PD feedback, Figure 4(a) clearly shows the high gain providing an earlier rise time (time needed to reach full desired response), and the low gain providing longer stabilization time. High gain produces a larger overshoot than low gain. Therefore, the tradeoff between speed and stability is obvious. The x- and y-axes of Figure 4(b) represent the horizontal and vertical displacements of the center of gravity when viewed from the front-and-back/place, while its color reflected during the tracking of the trajectory shows the start (red) and end (green) locations of the center of gravity during the stabilization-disturbance-recovery phases of testing. The results confirm that external disturbances temporarily force the center of gravity to cross the boundary of the unstable region before it returns to a stable state. This demonstrates the high stability margin of the fusion control mechanism under repeated disturbances and its effectiveness in enabling dynamic balance adjustment.

To achieve stable and efficient response of PD controller in dynamic scenarios, this paper designs a parameter adaptive adjustment strategy based on real-time state feedback. The proportional gain K_p and differential gain K_d are not fixed and unchangeable, but are dynamically adjusted based on the current joint angular deviation $\Delta\theta$ and its time derivative $\Delta\dot{\theta}$, external force magnitude, and attitude recovery stage. Specifically, when detecting that $\Delta\dot{\theta}$ exceeds the preset threshold indicating sudden motion changes or significant force impact, the system will automatically reduce K_p to suppress overshoot, while increasing K_d to enhance damping effect, thereby quickly absorbing disturbance energy, ensuring that the notations for the angular deviation and its time derivative are presented in a consistent mathematical form with the dot operator applied directly to the variable in proper equation formatting.

The parameter tuning process is quantitatively evaluated based on time-domain response performance indicators, mainly including rise time, maximum overshoot, and steady-state error. By conducting batch simulation experiments in different typical motion scenarios, the optimal parameter configuration for this scenario is selected as the combination of K_p and K_d that ensures the system has an overshoot of no more than 15%, a rise time of less than 200 ms, and a steady-state error of less than 2°. The final controller adopts a combination of segmented parameter tables and linear interpolation to achieve smooth switching of control gains based on different motion states and external force conditions, thus achieving the optimal balance between dynamic balance and response speed.

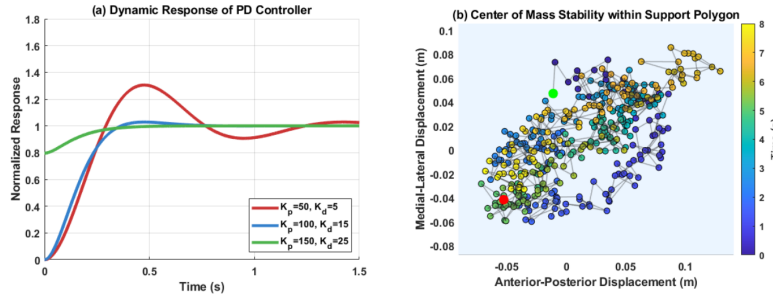


FIGURE 4. Dynamic response and balance stability analysis; Figure 4(a). Dynamic response characteristics of the PD controller; Figure 4(b). Visualization of center of gravity stability

D. INTELLIGENT POSE OPTIMIZATION OF THE NEURAL PREDICTION MODULE

1) Action Sequence Modeling and Network Input Design

The lightweight prediction network used in this system is based on a gated recurrent unit structure, which can effectively capture temporal dependencies in motion sequences. The input of the network is a sequence of historical states within a sliding time window, with a window size set to 10 frames (corresponding to approximately 166 milliseconds, based on a 60 FPS update rate). Each frame's input features include the local positions of all major joints, angular velocities, force-feedback vectors from the physics engine, and control signals from the fusion layer, forming 128-dimensional feature vectors. The network outputs the joint angle offset and angular velocity prediction value for the next frame, with dimensions corresponding to the number of joints.

The next frame's optimal pose is predicted by the neural prediction module based on historical action sequences and physical response data. The historical data used for input consists of joint position, joint angular velocity, force feedback information, and the control signals coming out of the fusion layer. The system maintains a sliding time-window for recording the previous frame's state sequence. The data are normalized and time-aligned using a sliding window, filtered with a low-pass filter to remove noise, and subjected to differential encoding of angular velocity changes, which enables to capture motion trends and inertial characteristics.

2) Attitude Prediction Output and Adaptive Correction

The network training adopts mean square error as the main loss function, combined with joint angular velocity smoothness constraints. The total loss function is defined as:

$$\begin{aligned} \mathcal{L} = & \lambda_1 \cdot MSE(\theta_{pred}, \theta_{gt}) \\ & + \lambda_2 \cdot MSE(\dot{\theta}_{pred}, \dot{\theta}_{gt}) \\ & + \lambda_3 \cdot \left\| \ddot{\theta}_{pred} \right\|^2 \end{aligned} \quad (8)$$

The hyperparameters are set to $\lambda_1 = 1.0$, $\lambda_2 = 0.5$, and $\lambda_3 = 0.1$. The network uses the Adam optimizer to train on a mixed motion dataset that includes walking, running,

jumping, and external disturbances, ultimately achieving a significant improvement in the naturalness and physical consistency of posture transitions while maintaining low inference overhead.

During runtime, the prediction module generates the target attitude parameters utilized to update each frame based upon the data gathered from the previous time window. The parameters outputted to the PD controller include angular position, angular velocity and the priority weights of the joint system. The environment's force from the physics engine is assessed against the predicted attitude. Corrective action is applied when the difference between the actual and predicted attitude exceeds the predetermined threshold. The input for the next frame is corrected via a recursive update method, allowing for adaptive convergence of the prediction process [20].

Figure 5 displays the correlation matrix for predicting multi-joint motions. The horizontal and vertical axes indicate the joint indices for the shoulder, elbow, wrist, hip, knee, and ankle. In this matrix, each cell represents the correlation coefficient (r) calculated from all joint angles recorded during motion generation (with corresponding physical feedback). The colors approaching red indicate a greater correlation than colors farther from red. The correlation is strongest between the knee and ankle joints (0.87), indicating strong coupling between adjacent joints during motion generation/physical feedback fusion. A rather weak correlation exists between the shoulder and ankle joints (0.41). This correlation matrix enables enhanced visibility into the key features of multi-joint motion coordination and naturalness when developing the skeletal rigging with physics engines, thus providing the data needed to refine model prediction/control strategies.

E. REAL-TIME RENDERING SYNCHRONIZATION MECHANISM

1) Multi-Threaded Rendering Structure and Parallel Data Scheduling

The real-time rendering synchronization mechanism provides a means for the parallel processing of skeletal animation computations alongside the processing of physical feedback while minimizing the delay between frames and enhancing responsiveness to user input. It does this by separating the rendering thread, physics thread, and animation

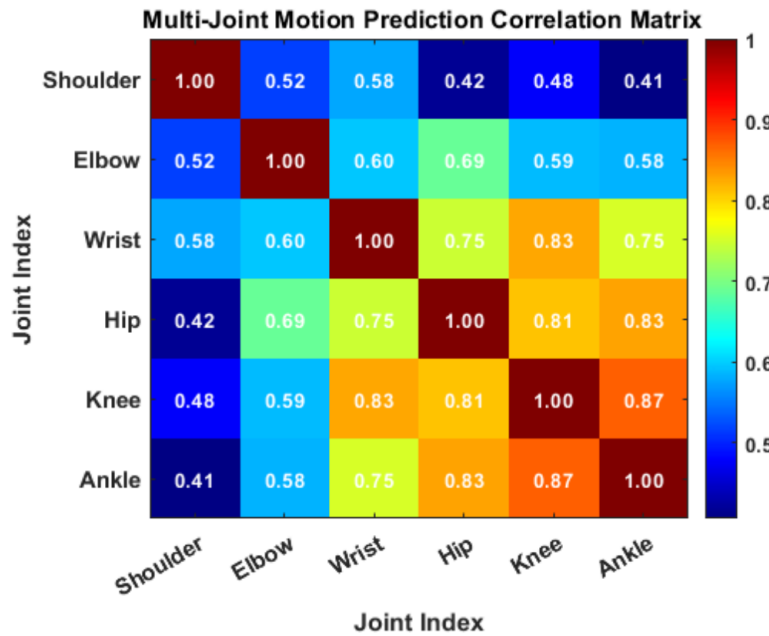


FIGURE 5. Correlation matrix of multi-joint motion prediction

calculation thread of the main loop into separate threads. The rendering thread only performs graphic rendering and scene lighting calculations; the physics thread performs its own collision detection and force feedback functions; the animation thread manages the updating of skeletal poses and motion interpolation. Each of these threads communicates with each other via a shared memory buffer containing skeletal matrices and physical states, with the goal being to limit any potential delays due to data transmission to within one rendering cycle.

Figure 6 shows the execution status and load distribution of different types of threads across consecutive frames. On a horizontal and vertical graph, the frame number (X-axis) and the different types of Threads (Yaxis) can be learned, which include rendering, physics, animation and AI threads. Each rectangle represents the time taken to execute a particular thread for that specific frame and each type of thread has been distinguished by color. The average utilization of each thread type is displayed on the right-hand side. The rendering thread shows the largest amount of utilization at 8.9%, with an intermittent, dense usage of these threads; the bulk of this thread's responsibility is rendering the graphics. The physics thread runs at an average frequency with a utilization of 8.2%, with this thread handling both calculating collision detection and mechanics. The Animation thread runs at a very high frequency with a very small amount of load (7.3% utilization), as this thread is used to update the animated skeletal. The AI thread uses 4.9% of the total utilization, as this thread processes the logic needed to make decisions.

2) Rendering State Synchronization and Dynamic Visualization Output

The real-time rendering phase updates vertex transformation information on a real-time basis with bone matrices generated by the animation thread and executes bone skinning operations utilizing the GPU. To achieve the timing correlation between physical feedback and rendering output, the rendering engine also applies an inter-frame interpolation algorithm to create estimates for poses that have not yet reached the synchronized point, resulting in a seamless, smooth visual experience with no jerks or delays. Each frame of the rendering engine extracts the physical state verified in the prior frame and merges it with the new pose data during rendering to guarantee that the movements of the character are visually synced to their force response.

Using a Uniform Buffer Object (UBO) at the GPU level provides access to up-to-date bone matrix and joint weight pose data via the rendering module. The rendering thread uses the physical contact feedback data of how clothing stretches, limbs shake, and how the ground acts during impact to update both deformation and contact point effects within the rendering process. This allows for the physical effects on the model and visual effects being shown on screen to be output together within milliseconds of each other, providing the scene with a higher level of dynamic consistency than previously possible.

III. MOTION EVALUATION AND PERFORMANCE VERIFICATION

The data used for this study comes from multiple sources, including motion capture systems and simulation software. Motion capture was performed using highly accurate equipment that tracked the movements of people performing typi-

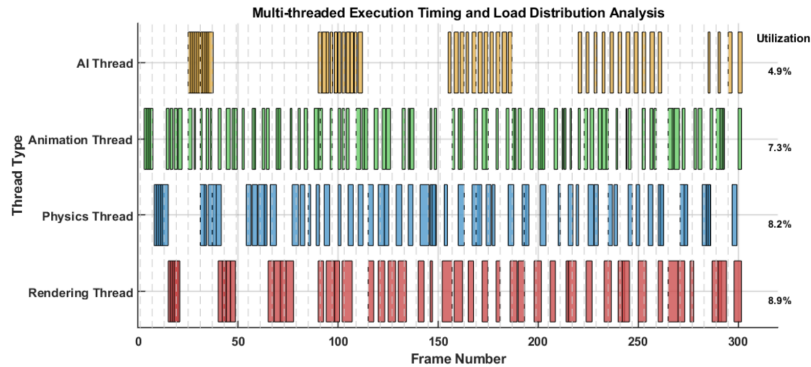


FIGURE 6. Multithreaded execution timing and load distribution analysis

cal day-to-day activities like walking, running, jumping, and many more. This instrumentation records information such as joint angles, angular velocities, and bone positions over every frame of video taken during the experiment. A physics simulator simultaneously reconstructs the motion of the character as a result of various forces acting upon it (i.e. contact with other objects, pushing, and rubbing against other surfaces), thus producing unpredictable movements. In the data collected, time series are aligned and filtered to reduce discrepancies introduced into the subsequent motion and/or pivot distance between the different actions performed under various force conditions. The data are sampled at a constant rate and synchronously exported in the original skeletal configuration and physical state of each subject, enabling subsequent quantitative evaluation of motion naturalness. In addition to motion performance based on smoothness of joints, uniformity of force responses, stability of the center of gravity, time delay in motion responses, and perceived naturalness, this paper provides a comprehensive evaluation of the system used for this investigation using a range of metrics.

A. JOINT SMOOTHNESS EVALUATION

The evaluation of joint smoothness involves recording the change curve of each joint's angle throughout time. The skeletal joint angle information is obtained from the animation generated by the user and is used to calculate the movement patterns of joint angular velocities and accelerations as they occur over time. The difference in joint angular velocity between consecutive frames are statistically analyzed to compute a smoothness score (i.e., lower scores indicate smoother motion transitions). Joint smoothness scores are calculated synchronously for each major joint, and the results of the calculations for each joint are processed independently. To calculate joint smoothness scores, a moving average filter removes instantaneous, high frequency shaking from the calculated score, and the resulting average score is compared to the original calculated score. The summary of joint smoothness scores is used to estimate the continuity of animated sequences and the quality of motion transitions.

Figure 7 illustrates the smoothness performance of the main joint movements of the human body under different

methods. Figure 7(a) plots time (s) on the horizontal axis and angular velocity (rad/s) on the vertical axis, presenting the joint angular velocity change curves for four methods: traditional keyframe, physics engine, the proposed method, and motion capture benchmark. The smoother the change in angular velocity over time, the smoother the motion. Compared to traditional keyframe and single physics engine, the proposed method exhibits a smoother angular velocity and more fluid motion, most closely resembling the ideal value of the motion capture benchmark (yellow). Figure 7(b) further reveals the distribution of acceleration standard deviation of the eight major human joints through the coordinate system of each joint's smoothness index. It can be seen that the motion capture benchmark has the lowest overall index and the best smoothness, followed by the proposed method. The standard deviation of angular acceleration is less than 1000 rad/s²; the traditional keyframe fluctuates greatly; the smoothness difference between the right and left knee joints is obvious, indicating that there are significant differences in lower limb joint control among different methods. The proposed method, while ensuring high smoothness, approaches the effect of real motion capture, demonstrating its effectiveness in motion generation.

B. FORCE RESPONSE CONSISTENCY

As part of the evaluation of the consistency of the character's physical response to applied forces the test subject has been exposed to a number of types of external force throughout the course of an experimental procedure, and the manner in which that subject's movement in relation to the application of those external forces has occurred has been documented. All joint angle variations as well as the magnitude and direction of the applied external forces are recorded. Based on these data, the consistency between the actual joint motion and the corresponding force application is quantitatively evaluated. A statistical analysis of the data collected has enabled the comparison of trends for all experimental sessions and to assess the degree to which consistency of force response exists across experimental sessions.

Figure 8 analyzes force-response consistency across different external-force types. Sub-figure (a) plots joint displacement over time under forward thrust, lateral impact,

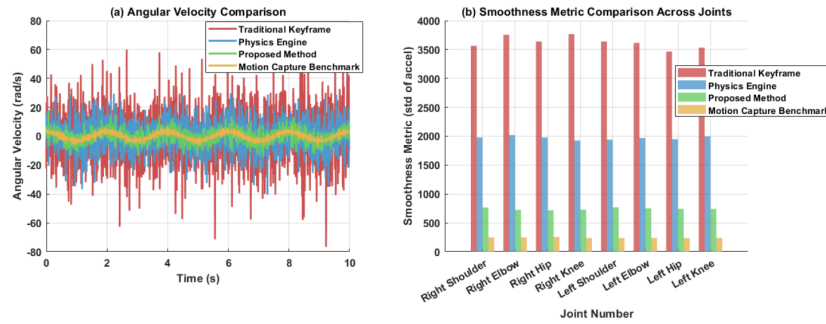


FIGURE 7. Joint motion smoothness analysis; Figure 7(a). Angular velocity comparison; Figure 7(b). Comparison of smoothness indicators for each joint

rotational torque, and continuous pressure, demonstrating the physics engine's ability to distinguish force profiles. Sub-figure (b) compares response-consistency error (vertical axis) versus force magnitude (horizontal axis) for four methods. The proposed approach maintains the lowest error across varying force intensities, closely tracking the motion-capture benchmark and outperforming traditional key-frame and physics-only baselines. These results confirm the system's robustness in preserving physical plausibility under diverse dynamic conditions.

C. CENTER OF GRAVITY STABILITY ASSESSMENT

The determination of centre of gravity stability involves monitoring the motion path of an individual across time with respect to their centre of gravity location from three-dimensional space. Stability is evaluated by calculating the center-of-mass coordinates at each frame under multiple motion conditions and measuring the distance between the center of mass and the support point (ground). A large deviation in the path of travel from the position of the centre of mass of the individual indicates that the centre of mass is likely unstable and does not allow for maintaining balanced postures. An average and a maximum deviation distance are calculated through statistical analysis for each of the various situations of action, and all of the information collected is scaled to account for the varying distances travelled from the centre of mass to the point of support. This test can determine the balance and naturalness of an individual's movements during dynamic interactions with a character.

To objectively evaluate the advantages of the proposed method in dynamic balance control, this study compared the center-of-gravity stability performance of the following methods under four typical actions and identical experimental conditions: traditional keyframe animation, physics-engine-only driving, and the proposed fusion method. The test indicators include average horizontal offset, maximum horizontal offset, average vertical offset, maximum vertical offset, and normalized average offset. As shown in Table 1, the method proposed in this paper is significantly superior to traditional keyframe and single physics engine methods in all indicators. Especially in collision scenarios, its maximum horizontal offset (5.2 cm) is reduced by about 60% compared to traditional keyframe methods (average 12.8

cm), and it is closer to the stable trend of real motion capture data, verifying the effectiveness of the proposed fusion control mechanism in maintaining attitude stability in complex dynamic environments.

TABLE 1. Center of gravity stability under different action types

Motion Type	Method	Avg. Horizontal Offset	Max Horizontal Offset	Avg. Vertical Offset	Max Vertical Offset	Normalized Avg. Offset
Walking	Traditional Keyframe	4.5	9.2	3.1	6.8	0.45
	Physics-Only Engine	2.8	5.6	1.9	4.1	0.28
	Proposed Method	1.2	2.5	0.8	1.5	0.12
Running	Traditional Keyframe	8.3	15.7	5.5	10.2	0.83
	Physics-Only Engine	4.1	7.9	3.2	6.5	0.41
	Proposed Method	2	3.8	1.5	2.9	0.2
Jumping	Traditional Keyframe	10.6	18.4	7.8	13.9	1.06
	Physics-Only Engine	5.2	9.7	4.1	7.8	0.52
	Proposed Method	2.5	4.5	2	3.8	0.25
Collision	Traditional Keyframe	12.8	22.3	9.5	16.7	1.28
	Physics-Only Engine	6.5	11.4	5.2	9.3	0.65
	Proposed Method	3	5.2	2.2	4	0.3

Table 1 shows the evaluation results of the character's center of gravity stability under different action types. The character's balance control ability is measured by five indicators: average horizontal offset, maximum horizontal offset, average vertical offset, maximum vertical offset, and normalized average offset. In walking, the average horizontal offset is only 1.2 cm; the maximum horizontal offset is 2.5 cm; the average vertical offset is 0.8 cm, indicating that the character's center of gravity is relatively stable. The normalized average offset is also only 0.12 cm, indicating that the movement amplitude is small, and the posture is stable. As the intensity of the action increases, the center of gravity offset of running and jumping increases significantly. The average horizontal offset of running is 2 cm, and the maximum horizontal offset is 3.8 cm. The average vertical offset of jumping reaches 2 cm, and the maximum vertical offset is 3.8 cm, indicating that the character's center of gravity is prone to large fluctuations under high-speed or jumping actions. The offset is greatest under collision actions, with an average horizontal offset of 3 cm, a maximum horizontal offset of 5.2 cm, an average vertical offset of 2.2 cm, a maximum vertical offset of 4 cm, and a normalized average offset of 0.3 cm, indicating that the difficulty of the character's balance control increases significantly under external disturbances. Overall, Table 1 provides an important basis for the balance stability of the character in dynamic interaction by quantifying the center of gravity position offset. It also reflects the impact of different action intensities on posture stability, providing a reference

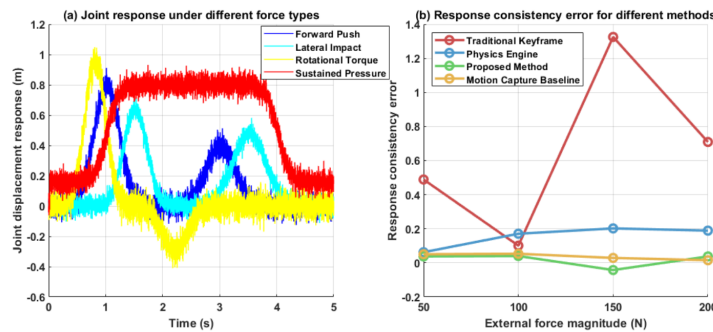


FIGURE 8. Joint force response and consistency error analysis; Figure 8(a). Joint displacement response under different force types; Figure 8(b). Comparison of response consistency errors of different methods

for the naturalness of actions and the optimization of system control algorithms.

D. MOTION RESPONSE DELAY

The motion response delay test is used to accurately measure the time difference between the system being subjected to a sudden external event (such as a collision) and the character's posture undergoing a visible correction. In the experimental procedure, the system records the moment of the external impact event and the starting frame of the posture change output by the skeletal system. The delay time from force perception to angle change for each joint is collected, and the average, maximum, and minimum delays are statistically analyzed across all joints in the body. Multiple sets of continuous impact events are used in the test to cover different force intensities and directions. To reduce measurement errors, the sampling frequency is kept consistent with the animation update frequency, and the average of multiple experimental results is taken. This metric is used to evaluate the system's motion adaptation speed to external disturbances in a real-time interactive environment.

To evaluate the response performance of different methods in real-time interactive environments, this study compared the joint average response delay of each method under the same external event. As shown in Table 2, under four typical external forces of thrust, ground impact, lateral impact, and sustained pressure, the fusion method proposed in this paper shows significantly lower average delay, maximum delay, and minimum delay than traditional keyframe and single physics engine methods, demonstrating better real-time adaptability and motion coherence.

TABLE 2. Motion response delay of the character's whole body joints

External Event Type	Method	Avg. Response Delay	Max Response Delay	Min Response Delay
Push	Traditional Keyframe	85	120	60
	Physics-Only Engine	50	75	35
	Proposed Method	35	50	25
Ground Impact	Traditional Keyframe	110	150	80
	Physics-Only Engine	65	95	45
	Proposed Method	40	60	30
Lateral Impact	Traditional Keyframe	95	130	70
	Physics-Only Engine	55	80	40
	Proposed Method	32	48	22
Sustained Pressure	Traditional Keyframe	100	140	75
	Physics-Only Engine	60	90	42
	Proposed Method	38	55	28

The response delay of the character's joints under external forces (push, ground impact, etc.) is reported in Table 2. The performance of the motion system for adapting to motion in real time via interaction with external sources can be evaluated by looking at three indicators (average response delay; maximum response delay; minimum response delay). In particular, for thrust, the average delay for all joints is 35 ms (the max delay is 50 ms; the minimum delay is 25 ms), which shows a relatively low overall level of delay in response to abrupt/excessive external forces. Delays are somewhat higher for ground impact (40 ms average; 60 ms maximum; 30 ms minimum); high-intensity ground impacts are likely to produce significant lag in many joints. The average delay for lateral impacting is 32 ms (minimum = 22 ms). This indicates a rapid adaption of the motion system to lateral disturbances. For continuous pressure (38 ms average; 55 ms maximum; 28 ms minimum), the motion system maintains a level of stability in terms of response rhythm under prolonged periods of external force. The overall trends in data in Table 2 show that all three types of external force have an effect on rate of response delays. The maximum delay does not exceed 60 ms, indicating that the system has a high real-time response capability under various dynamic disturbances, providing a quantitative basis for action control and interaction naturalness assessment.

E. SUBJECTIVE NATURALNESS RATING

The subjective naturalness assessment is conducted using a standardized scoring scale, inviting 20 judges with backgrounds in game development, animation production, or related visual media research to participate in the scoring. Each judge independently watches the animation sequence of the same character in different motion states (including walking, running, jumping, and collision), and each sequence is repeated 3 times. The scoring criteria are defined in four dimensions using a 0-10 scale: fluency (smooth and continuous motion), rationality of force application (consistency between the character's response and physical intuition), posture balance (maintenance of dynamic balance during movement), and overall naturalness (perceived realism of motion). The weight of each dimension's score is consistent, and the final score is the arithmetic mean of the

four dimensions. To reduce individual bias, calculate the average score of all videos for each judge as their final score, and then summarize all judge data to obtain the average score, highest score, lowest score, and standard deviation.

Using the details from observers' visual evaluations, subjective naturalness ratings measure how well an animated character performs in their movements. In this study, judges view several different versions of animation clips containing the same character but varying in speed and applied force; each judge then assigns a score for how smoothly those movements occur, how rationally the applied force appears to operate on that body, the balance of the character's posture, and the overall naturalness of the clip. Raters were asked to score at least 10 different clips using a standardized 0–10 scale for subsequent statistical analysis. To eliminate individual differences among judges, the system calculates the mean score for each judge and records the highest, lowest, and standard deviation scores.

Naturalness scores, based on how subjectively observers view the movement of a character regarding the following categories of smoothness, rationality of force application, posture balance, and general naturalness, are shown in Table 3. Among the actions evaluated it appears the most natural when walking, having an average score of 8.5 (maximum = 9/ minimum = 7), and the lowest standard deviation of 0.6, which suggests the character's performance on this aspect is very much the same and likely to be very high quality across a broad range of observers. Running achieves the second-highest average naturalness score, with a mean value of 8.0. Although the inter-observer agreement is acceptable, the standard deviation for running (0.9) is higher than that for walking. The jumping motion achieves a slightly lower average naturalness score than running, with a mean of 7.8, a minimum score of 6, and a standard deviation of 0.8, indicating reduced control of the aerial posture (i.e., maintaining an upright body during the jump). The collision motion yields the lowest average score among all actions (7.2), with a minimum of 5 and a standard deviation of 0.9, indicating that the character's movement under external impact is less natural and less consistent than during walking.

TABLE 3. Subjective naturalness rating

Motion Type	Average Score	Max Score	Min Score	Standard Deviation
Walking	8.5	9	7	0.6
Running	8	9	6	0.9
Jumping	7.8	9	6	0.8
Collision	7.2	8	5	0.9

IV. CONCLUSIONS

This paper proposes a method that integrates skeletal rigging with a physics engine for realistic character animation, enabling detailed and dynamic simulation of virtual textiles and apparel. The first approach, called skeletal rigging, defines the rotational, elastic, and damping parameters of the character rigging at the skeletal constraint layer. To help

maintain reasonable physical motion behavior of stressed joints, the physics engine calculates all forces generated by gravity, inertia, and collisions, feeding these forces back to the skeletal system for dynamic correction. In addition to providing real-time feedback to the skeleton, the physics engine also provides information to the skeleton via its PD controller, enabling a smooth transition from animation to physics and maintaining a stable center of gravity during movement. A neural predictor optimizes poses in previously captured motion sequences and eliminates motion/pose lag and other unpredicted pose anomalies. To ensure visual coherence between physical feedback and visual animation, a highly coordinated system comprised of a multi-threaded rendering synchronization mechanism is developed. Prototype testing shows that the new architecture significantly improves joint smoothness, consistency of joint force response, center of gravity stability, motion latency, subjective evaluation of character movement naturalness, and the effectiveness of character interactions.

AUTHOR CONTRIBUTIONS

Gaofeng Su designed, collected and analyzed the data, and drafted the manuscript. Gaofeng Su conducted the study, critically revised the manuscript for important intellectual content, and gave final approval of the version to be published. Gaofeng Su participated fully in the work, take public responsibility for appropriate portions of the content, and agreed to be accountable for all aspects of the work in ensuring that questions related to the accuracy or integrity of any part of the work are appropriately investigated and resolved.

CONFLICTS OF INTEREST

The author declares no conflict of interest.

FUNDING

This research was supported by Research on practical teaching mode of Digital Game Design course in Digital Media art major JJ2023Y040.

ACKNOWLEDGEMENTS

Not applicable.

REFERENCES

- [1] I. Jang, S. Choi, S. Hong, C. Kim, and J. Noh, "Geometry-aware re-targeting for two-skinned characters interaction," *ACM Transactions on Graphics (TOG)*, vol. 43, no. 6, pp. 1-17, 2024, doi: 10.1145/3687962.
- [2] N. T. Pickle, T. P. Zehnauer, V. J. Harrand, Zientara, GP, X. Zhou, and P. E. Roos, "Automated medical avatar animation for warfighter mission simulation," *Journal of Trauma and Acute Care Surgery*, vol. 91, no. 2S, pp. S107-S112, 2021, doi: 10.1097/TA.0000000000003278.
- [3] K. Cho, C. Kim, J. Park, J. Park, and J. Noh, "Motion recommendation for online character control," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 6, pp. 1-16, 2021, doi: 10.1145/3478513.3480512.
- [4] Y. Kang and J. Kim, "Animation character generation and optimization algorithm based on computer aided design and virtual reality," *Computer-Aided Design and Applications*, vol. 21, no. S14, pp. 46-62, 2024, doi: 10.14733/cadaps.2024.S14.46-62.

- [5] Y. X. Cai, H. Dong, W. Wang, and H. Song, "Realization of interactive animation creation based on artificial intelligence technology," *Computational Intelligence*, vol. 38, no. 1, pp. 51-69, 2022, doi: 10.1111/coin.12443.
- [6] J. A. Fernández-Fernández, R. Goldade, L. Kavan, J. Bender, and P. Herholz, "Interactive facial animation: Enhancing facial rigs with real-time shell and contact simulation," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 8, no. 4, pp. 1-21, 2025, doi: 10.1145/3747860.
- [7] J. P. Zhang, C. F. Pu, M. H. Guo, Y. P. Cao, and S. M. Hu, "One model to rig them all: Diverse skeleton rigging with unirig," *ACM Transactions on Graphics (TOG)*, vol. 44, no. 4, pp. 1-18, 2025, doi: 10.1145/3730930.
- [8] Y. Gao, G. Lang, C. Zhang, R. Wu, Y. Zhu, Y. Zhao, et al., "Rehabilitation exoskeleton system with bidirectional virtual reality feedback training strategy," *CAAI Transactions on Intelligence Technology*, vol. 10, no. 3, pp. 728-737, 2025, doi: 10.1049/cit2.12391.
- [9] Y. T. Tsai, W. Y. Jhu, C. C. Chen, C. H. Kao, and C. Y. Chen, "Unity game engine: Interactive software design using digital glove for virtual reality baseball pitch training," *Microsystem Technologies*, vol. 27, no. 4, pp. 1401-1417, 2021, doi: 10.1007/s00542-019-04302-9.
- [10] P. Gao, "Key technologies of human-computer interaction for immersive somatosensory interactive games using VR technology," *Soft Computing*, vol. 26, no. 20, pp. 10947-10956, 2022, doi: 10.1007/s00500-022-07240-3.
- [11] P. Li, K. Aberman, R. Hanocka, L. Liu, O. Sorkine-Hornung, and B. Chen, "Learning skeletal articulations with neural blend shapes," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1-15, 10.1145/3450626.3459852, 2021.
- [12] Y. Jiao, "Application of Virtual Reality in Biotechnological Processes: Enhancing Training and Simulation for Commercial Outcomes," *Journal of Commercial Biotechnology*, doi: 2024; 29(2):394-409.
- [13] C. Guo, X. Zuo, S. Wang, X. Liu, S. Zou, M. Gong, et al., "Action2video: Generating videos of human 3d actions," *International Journal of Computer Vision*, vol. 130, no. 2, pp. 285-315, 2022, doi: 10.1007/s11263-021-01550-z.
- [14] L. Li and T. Li, "Animation of virtual medical system under the background of virtual reality technology," *Computational Intelligence*, vol. 38, no. 1, pp. 88-105, 2022, doi: 10.1111/coin.12446.
- [15] C. Lamotte, "Discovering animation manuals: their place and role in the history of animation," *Animation*, vol. 17, no. 1, pp. 127-143, 2022, doi: 10.1177/17468477221080112.
- [16] T. Fukusato and A. Maejima, "View-dependent deformation for 2.5-d cartoon models," *IEEE Computer Graphics and Applications*, vol. 42, no. 5, pp. 66-75, 2022, doi: 10.1109/MCG.2022.3174202.
- [17] S. Dubey and M. Dixit, "A comprehensive survey on human pose estimation approaches," *Multimedia Systems*, vol. 29, no. 1, pp. 167-195, 2023, doi: 10.1007/s00530-022-00980-0.
- [18] E. Yu, F. Chevalier, K. Singh, and A. Bousseau, "3D-Layers: Bringing layer-based color editing to VR painting," *ACM Transactions on Graphics (TOG)*, vol. 43, no. 4, pp. 1-15, 2024, doi: 10.1145/3658183.
- [19] G. Lan, Y. Wu, F. Hu, and Q. Hao, "Vision-based human pose estimation via deep learning: A survey," *IEEE Transactions on Human-Machine Systems*, vol. 53, no. 1, pp. 253-268, 2022, doi: 10.1109/THMS.2022.3219242.
- [20] A. Vandewalle, S. Malliet, and K. Demoen, "Playing with character: a framework of characterization in video games," *Games and Culture*, vol. 19, no. 6, pp. 722-742, 2024, doi: 10.1177/15554120231179496.