DancingBox: A Lightweight MoCap System for Character Animation from Physical Proxies

Haocheng Yuan
University of Edinburgh
Edinburgh, United Kingdom
H.C.Yuan@ed.ac.uk

Adrien Bousseau
Inria, Université Côte d'Azur
Valbonne, France
adrien.bousseau@inria.fr

Hao Pan
Tsinghua University
Beijing, China
haopan@tsinghua.edu.cn

Lei Zhong
University of Edinburgh
Edinburgh, United Kingdom
zhongleilz@icloud.com

Changjian Li
University of Edinburgh
Edinburgh, United Kingdom
chjili2011@gmail.com

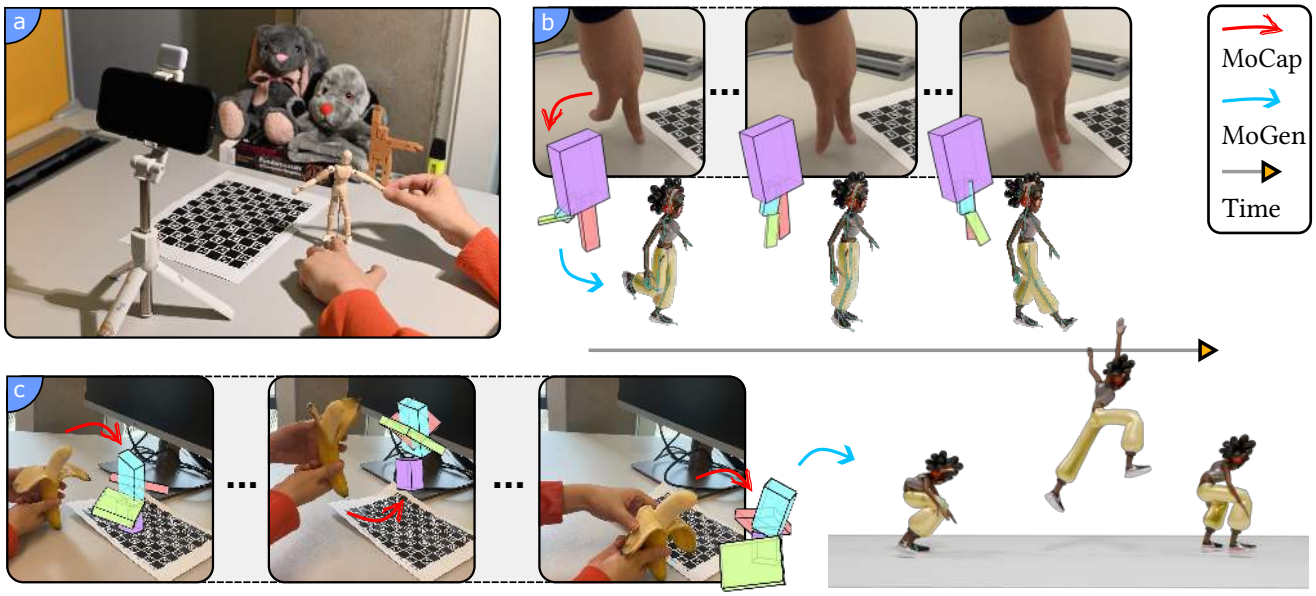

Figure 1: a) Overview of our *DancingBox* system. The interactive setup consists of a mounted web camera (*e.g.*, a smartphone) facing the scene, where users manipulate a physical proxy to convey their intended character animation, and a planar object (*e.g.*, a checkerboard) for calibrating the up direction of the ground. b) From the captured monocular video input of finger performance, *DancingBox* first estimates a sequence of bounding boxes using a lightweight vision-based motion capture module (*i.e.*, the red arrow, MoCap). It then transforms the coarse box motion into a natural skeletal motion through a conditional motion generation module (*i.e.*, the blue arrow, MoGen). c) Our system supports a wide range of physical proxies, such as plush toys, humanoid puppets, or even everyday objects, like a pen or a book. Notably, a banana with peeled skin, mimicking a jumping action with outstretched hands, is faithfully interpreted into the corresponding character motion. For improved visualization, skeletal motions are retargeted to the 'Michelle' character from Mixamo. Please see supplemental material for the animation clips of all results shown in the paper.

CHI '26, Barcelona, Spain

ACM ISBN 979-8-4007-2278-3/2026/04
https://doi.org/10.1145/3772318.3791202

Abstract

Creating compelling 3D character animations typically requires either expert use of professional software or expensive motion capture systems operated by skilled actors. We present *DancingBox*, a lightweight, vision-based system that makes motion capture accessible to novices by reimagining the process as digital puppetry. Instead of tracking precise human motions, *DancingBox* captures the approximate movements of everyday objects manipulated by users with a single webcam. These coarse proxy motions are then

refined into realistic character animations by conditioning a generative motion model on bounding-box representations, enriched with human motion priors learned from large-scale datasets. To overcome the lack of paired proxy–animation data, we synthesize training pairs by converting existing motion capture sequences into proxy representations. A user study demonstrates that *DancingBox* enables intuitive and creative character animation using diverse proxies, from plush toys to bananas, lowering the barrier to entry for novice animators.

CCS Concepts

• **Human-centered computing** → **Human computer interaction (HCI)**; • **Computing methodologies** → **Animation**.

Keywords

Character Animation, Motion Diffusion, Motion Capture, Conditional Motion Generation

ACM Reference Format:

Haocheng Yuan, Adrien Bousseau, Hao Pan, Lei Zhong, and Changjian Li. 2026. *DancingBox*: A Lightweight MoCap System for Character Animation from Physical Proxies. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26), April 13–17, 2026, Barcelona, Spain.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3772318.3791202

1 Introduction

Creating 3D character animations is at the core of computer graphics, serving widely across films, games, and mixed reality. A typical way of producing character animations is by manually specifying the position and orientation of body parts using professional software, which is known to have a steep learning curve and requires a great time and effort to achieve high-quality animation, even with expertise [62]. Alternatively, motion capture systems track the body parts of human actors to apply the same motion to virtual characters [7, 33]. But such systems often rely on specialized hardware to achieve high-quality tracking, and only skilled actors are able to perform diverse, compelling motions.

To make character animation more accessible, researchers in human-computer interaction have explored physical proxies as an interface for users to control character motion [11, 18, 19, 27, 57]. But these puppetry-based systems rely on custom hardware or are limited to specific motion types.

In this paper, we propose a lightweight puppetry-based motion capture system for novices. Instead of capturing the precise motion of professional actors, we capture the approximate motion of abstract physical proxies that users manipulate to *convey* the desired animation. We let users manipulate arbitrary everyday objects, covering rigid, articulated, and deformable items, e.g., pens, fruits, humanoid puppets, dolls. Then we capture their performance with a single web camera, from which we locate proxy parts and track their 3D motion using modern computer vision models. However, not only is such a low-cost capturing system imprecise, but manipulating everyday objects inherently provides only a simplified depiction of character animations. As a consequence, directly applying the captured motion to virtual characters would result in uncanny animations that lack many of the detailed, secondary motions that make character animations realistic. To address this challenge, we complement the user performance with realistic motion priors learned from a large dataset of high-quality human motion capture. Specifically, we treat the proxy motion as an input condition to a generative motion model, which produces a character animation that follows the approximate proxy animation while augmenting it with realistic secondary motion. Table 1 provides a comprehensive comparison of *DancingBox* with the existing motion authoring tools from physical input. Our method is the first capable of generating *high-quality motion* from *any object* with merely *one RGB camera*. Comparing against previous data-driven approaches, the generative model we build upon has been trained on a magnitude larger dataset and generalizes to unseen input without test-time training (please see Sec. F in the supplementary for details).

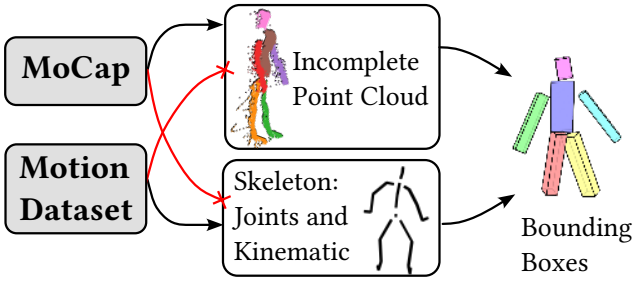

Figure 2: Our lightweight motion capture module produces noisy, partial point clouds, which we need to relate to the virtual skeletons used to represent character animation datasets. Extracting a clean skeleton from a point cloud, or synthesizing a defect-laden point cloud from a skeleton, are two difficult tasks. Our key observation is that abstract bounding boxes form a suitable middle-ground representation as they are easy to extract from both point cloud and skeleton data.

While conditional generative models have been successfully employed for related problems, such as sketch-based image synthesis [59], sketch-based 3D modeling [30, 61], and sketch-based character animation [62], applying this technology in our context raises specific challenges. Training a conditional generative motion model requires a large dataset of paired conditions and corresponding animations, which in our case would be very difficult to acquire since creating a proxy animation and its corresponding realistic animation implies significant human labor. Our goal of supporting arbitrary objects as proxies further complicates this task, as the number of possible combinations of animations and proxies is virtually infinite. Our solution to this challenge is to synthesize training data by converting a dataset of realistic character animations into a dataset of proxy animations, and to bridge the gap between our synthetic data and the captured proxy motion by expressing both with the same, abstract representation. Specifically, we represent proxy motions with *bounding boxes* of object parts (see Fig. 2 for a visual illustration). We describe how to extract such boxes from the input video and from the training motion capture data, and how to condition a motion generative model on proxies represented by a varying number of boxes.

We evaluate *DancingBox* through a user study consisting of two tasks: a replication task with provided puppets and a creative

Table 1: Comparison against existing physical motion authoring systems.

Method	Capture device	Physical Input	Motion quality	Virtual Outpu	Data-driven
KinEtre [4]	Depth camera	Human body	low	Objects	No
3D puppetry [16]	Kinect	Rigid objects	low	Objects	No
Puppeteer [18]	RGB camera	Human body	low	Human	Yes
Motionmontage [15]	Kinect	Rigid objects	low	Objects	No
Fingerpuppet [19]	RGB camera	Hand	low	Human	Yes
AniCraft [27]	RGB cameras	Markers + skeleton	low	Animals	No
Numaguchi et at. [34]	Specific tangible hardware	Specific tangible hardware	high	Human	Yes
Tangible Avatar [41]	Specific tangible hardware	Specific tangible hardware	low	Human	No
ARAnimator [57]	Phone	Phone	low	Human	Yes
Coolmoves [1]	VR Controller	VR Controller	high	Human	Yes
Ours	RGB camera	Any object	high	Human	Yes

task using unexpected objects proposed by participants. The results show that, with simple and intuitive user interaction (*i.e.*, manipulating everyday objects), our system can effectively generate realistic character motions that align with user intentions. Furthermore, the study also reveals users' preferences toward different puppets, providing design implications for future input devices. We also demonstrate how our motion capture system can be extended to keyframe-based animation for finer-control on character poses.

To conclude, this paper contributes to puppetry-guided animation creation in several aspects:

- We introduce a lightweight, vision-based system for puppetry digitization, allowing the motion capture of arbitrary objects with a single web camera.
- We describe how to complement approximate proxy animations with data-driven human motion priors to produce realistic character animations through puppetry.
- Our user study provides insights into system effectiveness, puppet preferences, and further features in demand for puppetry-based animation creation.

2 Related Work

We focus our discussion on prior research in using physical proxies to control character animations. We also introduce recent work in computer vision and generative models that enables our lightweight motion capture system.

2.1 Character Animation with Physical Proxies

The difficulty of using professional animation software is in part due to the challenge of manipulating individual body parts in a 2D interface to specify the 3D pose of human characters. This difficulty has motivated research about using various forms of proxies to define character poses directly in the physical world.

Dedicated tangible devices with physical sensors. Inspired by the ubiquity and playfulness of puppets, several works proposed custom tangible devices composed of limbs, joints and sensors that assemble to represent various skeletons, offering a direct mapping to the virtual skeleton to animate [1, 11, 20, 25, 26, 34, 41, 58]. Our work follows a similar motivation, but we aim at letting users manipulate everyday objects rather than a specific device. In particular, we leverage computer vision algorithms to track object parts instead of relying on numerous sensors placed over the object being manipulated. Moreover, while many previous systems assume that the input device has the same topology as the skeleton to be animated, we designed our system to support proxies with a varying number of moving parts, and we rely on data-driven priors to turn such partial input into realistic human motions.

Human body parts. Human bodies can perform humanoid character motion naturally. Motion capture systems [7, 33, 42, 48] have been widely studied to extract humanoid skeleton motion from actor performances. These systems typically involve rather complex environment setups, e.g., calibrated cameras, markers, and green screens. Another branch of work [4, 18, 19, 21, 31, 40] treats human body parts as proxies to control non-humanoid virtual characters. For example, Chen et al. [4] reconstruct the human body and link it to various digital characters, controlling character motions through body movements. Rhodin et al. [40] track body, facial, and hand motions of a human to control non-humanoid characters. Jiang et al. [21] optimize hand-to-avatar joint-to-joint mappings based on user-defined calibration for real-time control of characters. FingerWalking [31] and FingerPuppet [19] similarly show that low-DoF hand performances can act as intuitive proxies for producing full-body locomotion. Though the input of human body actions is accessible, mapping human body parts to non-humanoid characters inherently produces unnatural motions, since the topology and kinematics between humans and those characters are different. While we demonstrate our approach on human characters, we face a similar challenge as this family of work as we seek to map the abstract motion of simple proxies to the realistic motion of complex skeletons. While optimization-based techniques have been studied to synthesize physically-valid motion [9], *DancingBox* instead leverages data-driven motion priors to generate realistic and expressive animations.

Character-like daily objects. Several studies [15, 16, 27, 43, 57] offer solutions to produce virtual animations by manipulating common objects. 3D Puppetry [16] reconstructs rigid objects with a depth camera and maps manipulation of the object to the rigid motion of its digital twin. Similarly, P. T. Sin et al. [43] track a non-rigid stuffed toy to control a virtual model of the same toy in XR games. ARanimator [57] uses a cell phone as a 6 DoF sensor

and builds a regression model based on a user study to predict motion types from cellphone movements. AniCraft [27] proposes an affordable proxy for prototyping 3D character animation in MR. The process involves handcrafting the desired skeleton with metal wires and markers. Then, the manipulation of the proxy is mapped to skeleton motion automatically. While lightweight and accessible, these methods share two drawbacks. First, they rely on specific objects or devices to capture the user performance. Second, they only support a limited space of motion, such as rigid [15, 16] and cage-based [43] transformations, pre-defined motion templates [57], and prototyping-level unnatural motion [27]. In contrast, *DancingBox* allows users to produce realistic animations simply by manipulating common objects in front of a web camera.

2.2 Vision Foundation Models

Vision foundation models [2, 22, 24, 51] refer to a class of deep neural networks trained on massive datasets to address general vision problems, such as image segmentation, video tracking, and visual geometry reconstruction. Segment Anything [24] segments an image into meaningful semantic regions based on a text prompt or user clicks, while CoTracker [22] takes videos as input and tracks pixel correspondences across all frames. More recently, VGGT [49] and π^3 [51] have been proposed to predict 3D point clouds and camera parameters directly from images. By combining these tools, we build a motion capture system with minimal hardware requirements—only a webcam, without the need for calibration.

2.3 Motion Generative Models

Generative models learn to estimate real-world data distributions, and generate novel realistic data by sampling from the estimated distributions. Popular architectures include Generative Adversarial Networks [12] and Variational Auto Encoders [23], with diffusion [5, 8] and flow-based models [29] showing dominating performance recently.

Motion generative models have been trained on large motion capture datasets to generate different types of skeleton motions from different controlling signals. A major topic is text-conditioned human motion generation [3, 13, 14, 46, 60]. For example, MDM [46] is a diffusion model based on HumanML3D [14] and the Kit dataset [37], and is guided with text conditions through classifier-free guidance [17]. While human motion is the dominant problem of interest due to data availability, some recent research [10, 44, 56] focus on generating arbitrary skeleton motion. Also, different control signals like sketch [36, 62], audio [54], video [28], and joint positions [53] have been explored. Moreover, since motion generative models posses motion priors learned from motion capturing data, recent research [48] shows that such diffusion priors can be used backward to refine the motion capturing system's result.

We demonstrate our method with human motion generation models, enabling intuitive puppetry for humanoid characters' motion. Our motion capture system is agnostic to motion type, and our box-conditioned generative model could be generalized to arbitrary skeletons and character types if applied to the corresponding base generative models.

3 Method

Fig. 1(b) displays a high-level illustration of our method with two core modules - vision-based motion capture (MoCap) and conditional motion generation (MoGen). Given the recorded puppetry performance video, our goal is to recover an approximate 3D animation from the video and translate it into a realistic character animation. Specifically, we analyze the video frames to obtain the puppet parts and track the part movement to obtain the approximate motion, which, subsequently, serves as the spatial condition in the motion generation module with realistic motion priors prelearned from large-scale motion datasets.

Due to the lack of a direct puppetry-motion dataset, there exists a mismatch between the approximate motion from the video input (*i.e.*, partial point clouds) and the realistic motion from motion datasets (*i.e.*, clean SMPL human motion [14]). We propose to use 3D bounding boxes as an intermediate and abstract representation to bridge the two modalities. The choice of bounding boxes over other primitives, such as cylinders or line segments, is motivated by their ability to represent rotations around the object's symmetry axes. Ellipsoids could also be used, as they have the same degrees of freedom as cuboids, but we opt for bounding boxes since they are the de facto standard in computer vision tasks such as 3D object detection [32]. In the following, we elaborate on technical details.

3.1 System Setup and User Input

Fig. 1(a) shows our system setup, where a web camera (*e.g.*, a mobile phone) is mounted on a desk (as the ground plane), facing the performing space. A checkerboard marker is placed on the ground plane for calibrating the up direction of the ground. Users manipulate a physical proxy, such as a humanoid puppet, a toy, or even their fingers, to perform the intended motion. After the performance, the recorded video serves as the input to our algorithm, along with an optional text description of the desired motion. Note that we tested our system in an indoor environment with artificial lights or natural daylight.

In the current implementation, an extra user input is necessary. Given the first frame of the recorded video, users are asked to click a point on the ground object, and a few points on the different parts of the proxy they wish to articulate. These points are fed to a video segmentation model, as explained in the next section. Note that users do not need to segment the proxy into many small parts: 1-6 parts are often enough to convey the motion of the legs, torso and arms of a character. In supplemental materials, we illustrate a typical interactive segmentation session and visualize the user clicks for all results shown in the paper.

3.2 Motion Capturing with Vision Foundation Models

The goal of our motion capture module is to segment proxy parts, reconstruct their 3D pose in each video frame, and track the movement of the parts across frames. Fig. 3 shows an example video, illustrating the main technical steps of our approach.

3D reconstruction with π^3 [51]. To reconstruct the 3D information of the dynamic scene, we exploit π^3, running on each video frame to produce a 3D point cloud of each frame (Fig. 3, row one). Particularly, the point cloud takes the form of a depth map,

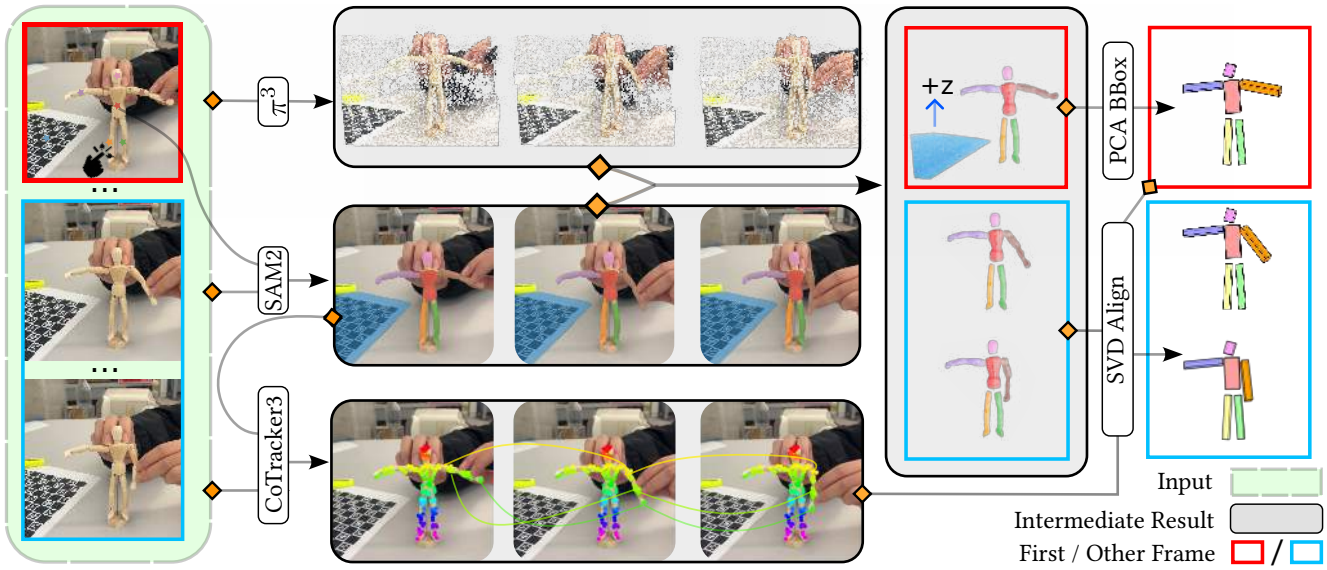

Figure 3: Overview of MoCap. From left to right: given the recorded video and user clicks in the first frame, we exploit SAM2 [39] to segment the parts of the puppet in all frames. The user clicks indicate the desired part segmentation, with different colors denoting different parts. The order of the clicks does not influence the result. We also run π^3 [51] to estimate the point cloud of each frame, and CoTracker3 [22] to produce dense pixel-wise correspondences between frames. Combining semantic segments, point clouds, and motion tracks allows us to recover 3D bounding boxes of proxy parts and their motion across the video clip.

providing point-pixel alignment, which facilitates the following segmentation and tracking propagation.

Segmentation with SAM2 [39]. We exploit SAM2 [39] to segment proxy parts over the entire video. This segmentation model takes as input the user-provided points in the first frame to extract the corresponding segments over all frames of the video (Fig. 3, second row). The model is robust to occlusions, producing a consistent segmentation even when parts of the proxy are not visible in some of the frames. Additionally, we leverage the pixel-point correspondence from π^3 to transfer the part segments onto the point cloud of each frame, effectively producing segmented point clouds. We rely on the segmented ground object (*i.e.*, the checkerboard) to rotate the global z-axis from π^3 so that it aligns with the upward ground direction.

Tracking with CoTracker3 [22]. In our formulation, we assume the motion to be rigid, since most everyday objects can be reasonably approximated as articulated rigid bodies. With the segmented parts in the per-frame point clouds, there are several potential solutions for inferring the motion information. For example, point cloud registration techniques can be used to estimate the per-part rigid transformation. However, there are two major difficulties. Firstly, the point cloud of each part is very sparse, as each part occupies only a small region, which poses challenges for registration algorithms. Secondly, these algorithms are highly sensitive to point noise introduced by imperfect segmentation and reconstruction near part boundaries.

Our solution is to resort to tracking pixels on video frames for a robust motion estimation. Specifically, we run CoTracker3 [22] on the input video. This tracking model takes as input so-called *query pixels* in a given frame of the video and tracks them across all frames. In our system, we automatically sample query pixels within each segment to produce tracks throughout the video (see Fig. 3, third row). In most cases, sampling these pixels in the first frame suffices to track object parts over the entire video. But the tracks might get lost in the presence of occlusions. Fortunately, the segments extracted by SAM2 are robust to occlusions, which allows us to resample the segments in subsequent frames where they appear to handle these more complex cases. Specifically, the resampling is automatically triggered repetitively when 70% of tracking pixels becomes invisible (see Fig. 4 in the supplementary for an example). Thanks to the pixel-point correspondence from π^3, we transfer the tracks from 2D pixels to 3D points. This process gives us, for each segmented part, a one-to-one point mapping between successive frames over a subset of the part points.

Motion with bounding boxes. Having the segmented point clouds with robust tracks in each of them, we next estimate the motion with bounding boxes. For the initial frame, we first filter out the low-confidence points and outliers based on their confidence scores predicted by π^3. We then compute an oriented bounding box through PCA estimation (see top-right of Fig. 3). Starting from these boxes, we rely on established point tracks to estimate the transformation through SVD alignment (Kabsch-Umeyama algorithm [47]), and transform the initial boxes to later frames, forming the motion sequence represented by bounding boxes (see the last column of Fig. 3). Note however that the system cannot estimate the transformations of the bounding boxes in the frames where the corresponding segments are fully occluded. We handle these cases by randomly removing some of the boxes during training of the conditional motion generation model, such that the model learns to synthesize plausible motion for the missing parts. Please see the

supplementary for a dedicated analysis of the robustness of our capture system under occlusion and in-place rotation.

3.3 Box-guided Motion Generation

Conditioned on the coarse motion extracted from the performance video, we aim to generate the corresponding realistic character motion exploiting the motion priors in learned models. Given the segmented 3D point cloud, a straightforward solution would be to convert every of its part into skeletal joints. However, this solution is only feasible when the number of parts corresponds to the number joints of a standard skeleton, which is rarely the case in our setting where everyday objects contain only a few moving parts. Furthermore, directly reconstructing a skeleton animation from the captured point clouds would yield uncanny results, as it would strictly replicate the user's imperfect physical manipulation and lack the realistic motion effects provided by generative priors. We thus introduce bounding boxes as an intermediate representation to bridge the input 3D points with realistic output motion. In the current implementation, we only target human-like motion due to the wide availability of existing human motion datasets.

Preliminary. Motion diffusion models typically represent human motions with a fixed number of body part joints. In addition to text conditions, spatial guidance such as 3D joint trajectories or keyposes can be injected into the motion diffusion process via the ControlNet [59] mechanism to further control the behavior of the generated motion [53, 62]. Additionally, the accuracy of the generated motion with respect to the spatial conditions can be enhanced by a so-called inference guidance loss [53, 62], inspired by classifier guidance [8] and loss-guided diffusion [45, 52].

Method overview. Built upon a pre-trained human motion diffusion model - MDM [46], we design our method as a conditional motion generation, as shown in Fig. 4. Briefly, the generator takes as input the extracted motion guidance and an *optional* text description, to denoise a Gaussian noise into a realistic motion with an inference guidance loss. However, in our setting, there are two unique challenges due to the bounding box-based intermediate motion representation. Specifically, different physical proxies express different levels of abstractions of character parts. For example, the banana in Fig. 1 has four boxes representing the upper body, lower body, and two arms, while the humanoid puppet in Fig. 3 has six boxes representing the head, body, two upper limbs, and two lower limbs. On the one hand, the number and the order of boxes vary depending on the proxy and the number of parts segmented by the user. On the other hand, the mapping from boxes to body part joints is infeasible to build explicitly. This prevents a one-to-one guidance between boxes and joints and also exposes difficulty in formulating the inference guidance loss term. To overcome these challenges, we have designed a novel permutation-invariant box motion encoder and an innovative box-joint guidance.

Box motion encoder. A key idea behind our system is to summarize the input bounding boxes – which can vary in number depending on the number of parts segmented by the user – into a compact fixed-size motion code. We design this encoding process at both the box level and the character shape level to be invariant to vertex and box ordering. Permutation invariance is essential to extract a unique feature representation for a bounding box, as the order of vertices does not alter the underlying geometry. We employ Mean/Max pooling, which are proven effective for general point cloud geometry learning (e.g., PointNet [38]). The network architecture is shown in Fig. 4 (right). Firstly, for each 3D bounding box within a frame, we employ a shared MLP to encode the 3D coordinates of each vertex into a latent feature. To make the box embedding invariant to vertex order, we aggregate the features by summing up the mean and maximum feature values of the eight box vertices. We repeat the box encoding for each box of the proxy. Secondly, for proxies made of several boxes, we exploit the self-attention mechanism to model the inter-box relationship within the frame, and then employ the same aggregation strategy to produce the shape-level feature for that frame.

Box-based spatial guidance. Users manipulate the physical proxy to drive character motion, even when represented by a single bounding box (*e.g.*, a pen or a book). In this sense, the boxes are expected to span all the joints, despite the absence of a direct box-joint mapping. Thus, the core idea of defining the discrepancy measurement is to make sure that *every bounding box contains at least one joint* (see Fig. 5, for an illustration). Specifically, given joints from one frame of the produced motion and corresponding conditional boxes, we measure the distance between every joint to the center of every box. Instead of setting a distance threshold to determine the coverage relationship, we use a *soft* association between each box and all the joints, and the weight decays exponentially with distance. Finally, we sum up all the weighted distances linearly and minimize the total value as the loss guidance at inference.

Network training and inference. The MDM [46] was pre-trained on the HumanML3D [14] dataset. As shown in Fig. 4, we only train the box motion encoder and the ControlNet on our constructed dataset, where each data item consists of the paired abstracted box motion and realistic skeletal motion. For robust network training, we adopt several data augmentation techniques. Specifically, we vary the number of boxes to achieve different levels of details, we randomly apply rigid transformations to a few boxes in a random frame of a motion sequence to mimic tracking failure, and we randomly drop either the text input or a few boxes to simulate different levels of abstraction in the condition. At inference time, given the detected conditional box motion sequence, the initial noise, and the optional text, our motion generator produces the corresponding realistic motion, conforming to the spatial conditions. *Please refer to the supplementary for our dataset construction process and implementation details of our method.*

4 User Experience Study

To evaluate how *DancingBox* facilitates character motion creation and assess user preferences across different interaction proxies, we conducted a comprehensive user study with diverse participants and task designs, and a questionnaire-based feedback collection.

4.1 Study Methodology

Study participants. We recruited 9 university students aged 20 to 30 years to participate in our study. The participants have mixed backgrounds and fluency with animation creation tools, with 4 having no prior experience, 3 possessing entry-level experience,

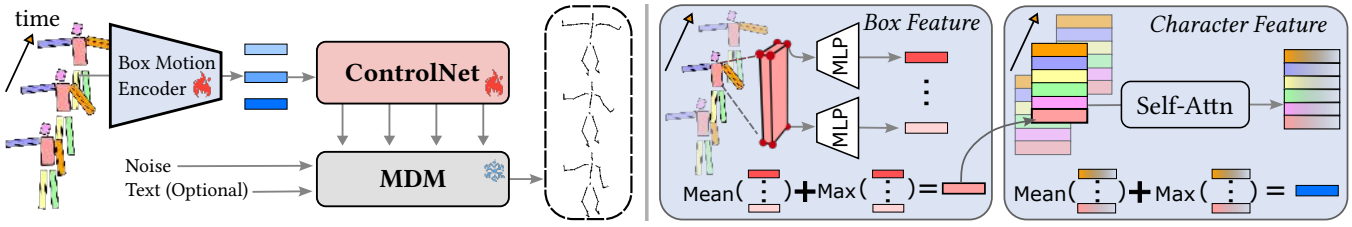

Figure 4: Left: overview of our conditional motion generator. We train a custom box motion encoder alongside a ControlNet module to condition a pre-trained Motion Diffusion Model. Right: for a given frame, our box motion encoder first encodes each vertex of a box (*e.g.*, the pink one) using an MLP, and aggregates all vertices of the box into a single code using mean and max operations to obtain a latent code that is invariant to vertex ordering. A self-attention layer then exchanges information between the latent codes of all boxes of the character proxy. Finally, the resulting latent codes are aggregated into a single code for the entire character, again using mean and max to be invariant to box ordering.

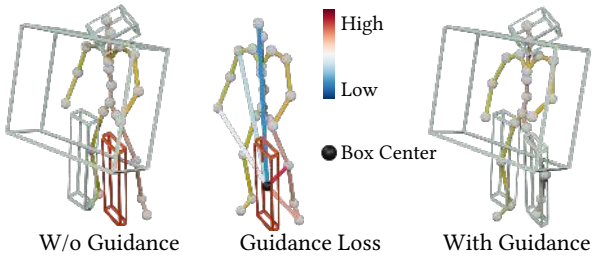

Figure 5: Impact of spatial guidance and its design illustration. Left: without spatial guidance, the bounding boxes are able to provide rough motion control, but do not guarantee precise alignment between boxes and joints (*e.g.*, the joints on the right leg vs. the red box). Middle: for an exemplar bounding box (in red), the guidance measurement loss is computed against every joint, then accumulated using distance-based combination weights. Five such distances are visualized, with colors indicating the corresponding weights. Right: With the designed spatial guidance, the generated motion aligns closely with bounding boxes, ensuring each box contains at least one joint.

and 2 familiar with a few software (*i.e.*, Blender, Cinema4D, 3ds Max and Maya).

Experimental setup. The study was conducted in a controlled laboratory environment featuring a desk-mounted setup with a smartphone serving as the capture device. Participants had access to various physical puppets for motion manipulation. All captured motion video was transmitted to a remote server equipped with an RTX 4090 GPU for *offline* processing and animation generation.

Study protocol. The study followed a structured four-phase protocol before a post-study evaluation.

1) **Pre-study assessment**: Participants completed an initial questionnaire documenting their background and experience with 3D animation tools. They are also asked to estimate how hard the replication task could be without our tool.
2) **System introduction**: We provided a 10-minute tutorial covering fundamental animation concepts and the *DancingBox* system workflow, ensuring all participants had equivalent baseline knowledge.
3) **Replication tasks**: Participants performed three motion replication exercises, where they recreated target animations using the provided tools and puppets. These tasks assessed the system's usability for reproducing specific motions. A default short text description is applied in this task.
4) **Creative design tasks**: Following the replication tasks, participants engaged in open-ended creative animation design, selecting from available puppets or incorporating their own objects to create original character motions. This phase evaluated the system's potential for creative expression. An optional text description is provided by the user.
5) **Post-study evaluation**: After reviewing their generated animations, participants completed a comprehensive questionnaire and participated in a semi-structured interview to gather qualitative feedback about their experience and system performances.

All the raw data from the user study can be found in the supplemental material.

4.2 System Usability and Result Motion Quality

Figs. 6 and 7 present representative results from participants for both replication and creative tasks. Overall, all participants successfully reproduced the given motions. Notably, when asked *before* the study to estimate how long it would take to create motions similar to the three examples with existing tools (Fig. 6), even the two most experienced participants anticipated needing at least *one full day*. In contrast, with our system, the operating time to reproduce a single motion is roughly 3 minutes for puppet manipulation, 2–4 minutes for interactive part clicking, and 4–5 minutes for the algorithm running (see Sec. 5 for a detailed breakdown). In the creative design stage, participants explored a variety of puppets to produce diverse motions, including unexpected outcomes such as a hand puppet crying, a soft-bodied doll dancing, and a banana performing a backflip (the first three examples in Fig. 7, respectively). We summarize the findings below.

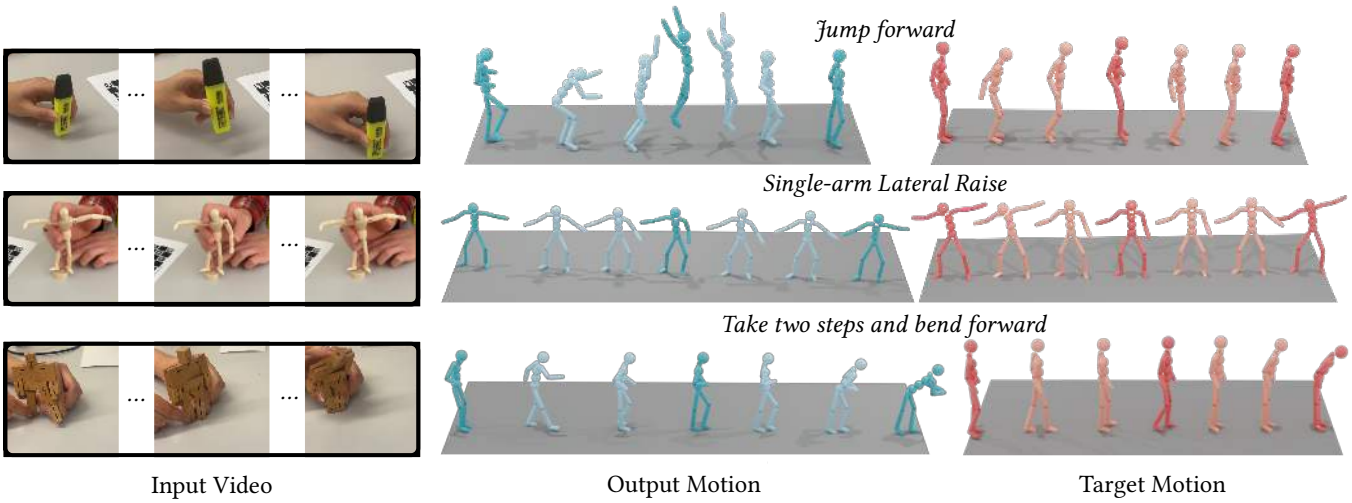

Figure 6: Representative replication results from our user study. Given a target motion (right), participants were asked to reproduce it (middle) by manipulating a designated physical proxy (left).

Users found the system extremely easy to use. Question responses indicate that participants found the system intuitive to learn (Q1, Fig. 8) and easy to operate using physical proxies (Q2, Fig. 8). As P3 put it, "*I used to play with dolls as a kid—this felt just as easy.*" All participants reported that they would use the tool for animation creation if available (Q3, Fig. 8). P7, who was working on a personal animation project, remarked: "*I wish your system were ready right now; it's much easier than my usual software (Blender).*" After completing both replication and creative design tasks, all participants believed they could create additional motions with the system (Q4, Fig. 8).

Users reported natural, realistic motions. The fifth question (Q5, Fig. 8) shows that participants perceived the generated motions as realistic and natural. Leveraging a generative prior learned from rich motion data, *DancingBox* can produce realistic motion even when proxy movements are somewhat unnatural. P8 commented: "*I thought the rabbit's head (second example in Fig. 7) wasn't moving the right way, but the final motion turned out perfect.*"

The system respects user intent. A common issue with generative methods is unintended drift from the user's intent. We evaluated how well the system preserves user intent across both tasks. In the replication task, participants rated the similarity between their manipulations and the generated motion with an average agreement score *4.44* (Q6, Fig. 8), indicating strong faithfulness to the proxy. In the creative task, participants produced various motions with their chosen tools, and most reported that the results reflected their intentions (Q7, Fig. 8). P3, P5, P4, and P8 each expressed a similar sentiment: "*This is exactly what I had in mind.*"

Our system facilitates motion replication. Imitating existing motion from video or image references (whether designed or recorded) is a common practice for creating skeleton motion in both motion capture pipelines and 3D software. We quantified how well our system fits this replication workflow in the user study. During the replication task, participants rated pairwise similarity among the target motion, their manipulation, and the generated motion. To evaluate effectiveness, we define a *replication failure* as any case where the generated motion is *less* similar to the target than the original manipulation:

$$\text{Sim}(\text{generated}, \text{target}) < \text{Sim}(\text{manipulation}, \text{target}).$$

Across 27 trials (9 participants × 3 replication tasks), we observed only 2 failures (7.4%), indicating that in over 92% of cases the system preserved or improved similarity to the target motion. These results suggest that, in replication scenarios, when proxies approximate the target motion reasonably well, our system reliably translates them into detailed motion sequences while maintaining similarity.

4.3 Implications for Physical Proxy

Our system supports diverse puppets—from simple rigid objects to soft dolls with virtually unlimited degrees of freedom. We further examined how participants' perceptions of different puppet types, providing insights for the design of future puppets and tangible interfaces.

Users prefer detailed puppets and want fine control. In the questionnaire, when asked to choose between abstract puppets (e.g., a pen, a banana, or a hand pose) and a detailed articulated human model, eight out of nine participants chose the articulated model. But the reasons varied. First, detailed puppets directly resemble humans, reducing cognitive overhead; as P2 noted, "*They're similar to real characters and give immediate feedback about the motion.*" Second, human-like puppets afford fine-grained control. P8 said, "*I can't really show hand movements with a pen, so I'd rather use a doll here.*" P7 added, "*Abstract puppets can work for stage play, but I want to control every single detail with the human model.*"

Controlling detailed puppets can be hard. P9—the only participant who preferred abstract puppets—explained, "*The pen and the banana were the easiest, mainly because it's hard to manipulate the*

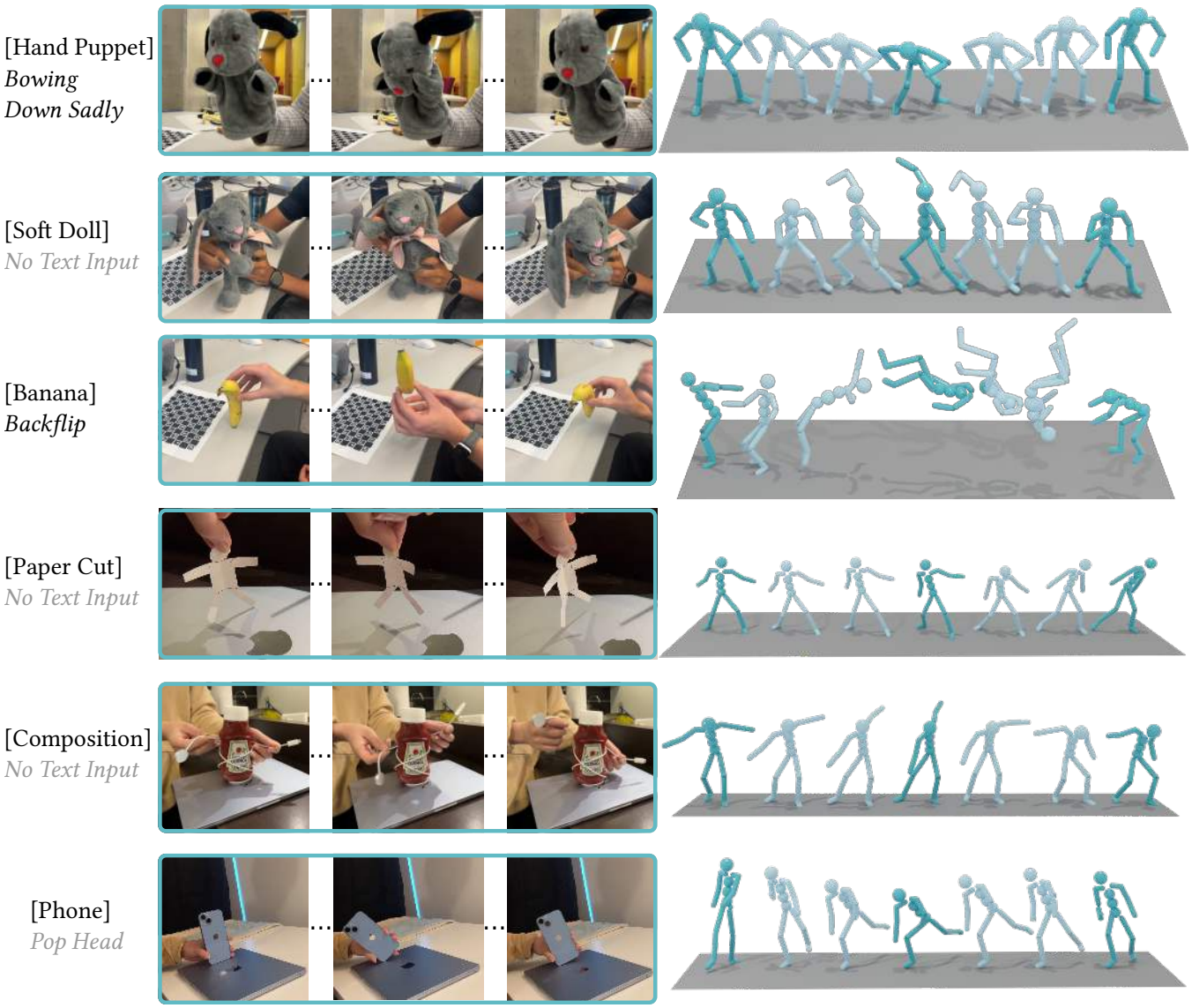

Figure 7: Creative results from participants manipulating diverse puppets (first three rows), and from an experienced user performing the action using paper cut (fourth row), compositing daily objects (fifth row), and a phone (last row), demonstrate the effectiveness of our system in producing high-quality motion from varying physical proxy. The left column shows the puppets and input texts (if any), the middle column shows the input video frames, while the right column presents the corresponding generated motions. Note that the text input is provided only in the first and third examples. For better visualization, we display the motion frames with spatial displacements on the ground, although these displacements are not part of the motion itself (*e.g.*, the motion in the second last example is basically moving in place).

humanoid puppets accurately in a short time." Other participants, while favoring detailed control, acknowledged that detailed puppets can be challenging. As P7 put it, "*That wooden tool is harder to use. ⋯ I'll need a third hand.*" When asked about the sources of difficulty, participants mentioned "*the joints are lagging*" (P5), "*the puppet easily falls over, so I have to keep it upright while moving*" (P3), and "*having to manage occlusion during manipulation*" (P2).

To conclude, participants generally prefer detailed, human-like puppets for expressive control, yet recognize they can be challenging to operate; abstract puppets remain useful for scenarios like stage-play animation. Because our *DancingBox* system is agnostic to puppet type, it can support future exploration of input devices (puppets and other physical proxies as tangible controls). Our findings suggest that a balance should be struck between finer detail

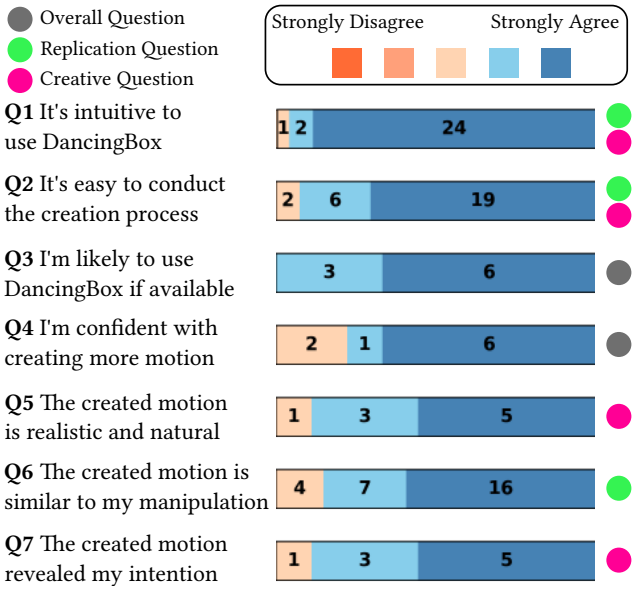

Figure 8: Questionnaire statistics (subset). A selected subset of questions with corresponding responses is shown. Gray, green, and magenta colors indicate question categories asked at different stages: *overall* questions were asked once after the study; *replication* questions were asked after each replication task, yielding $3 \times 9 = 27$ responses; *creative* questions were asked after the creative task with $1 \times 9 = 9$ responses. Some questions were asked after both the replication and creative tasks, with $27 + 9 = 36$ responses. Each question has five response options (*i.e.*, from Strongly Disagree to Strongly Agree, scored 1 to 5). Numbers on the bars indicate the count for each response (colored accordingly). Notably, across all selected questions, no participant chose *Disagree* and *Strongly Disagree*. See the supplementary for the complete set of questions and responses.

and controllability: adding too many flexible joints may reduce practical usability if users cannot physically control all of them.

5 Results and Discussions

Beyond the user study, we also demonstrate more appealing motions results created by an experienced user, as shown in Figs. 1 and 7. In this mode, our system effectively functions as a markerless motion capture pipeline: it takes an actor's performance video as input and outputs a skeleton motion sequence. The system is calibrated by placing a planar object to denote the ground, followed by a few clicks during segmentation.. Compared with established motion capture systems, our system offers two advantages: 1) it requires only monocular video from a consumer webcam, with no markers or suits; and 2) it can be used immediately with a smartphone and a user's own performance, with virtually no setup overhead. Compared with previous physical animation authoring tools, *DancingBox* can reproduce their results without any parameter tuning (Fig. 1(b) vs. FingerPuppet [19] and Fig. 7 last row vs. ARAnimator [57]). Moreover, *DancingBox* can accommodate a much wider range of input objects, giving users substantially greater freedom for exploration. However, the main limitations are reduced motion accuracy compared to multi-camera systems and non-real-time processing.

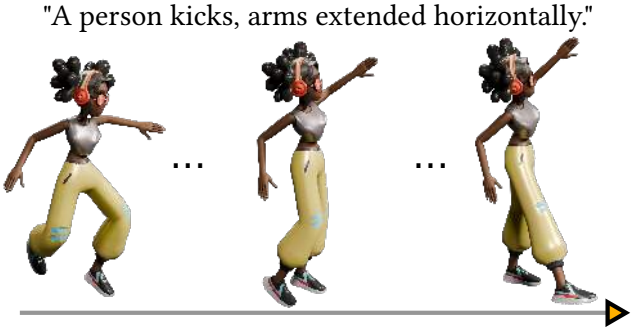

Figure 9: Generation result from the same video input of Fig. 1(b), but adding extra text description to control the arms.

To further validate the effectiveness of our method, we have conducted an ablation study and discussions of key design choices (Sec. 5.1), and demonstrated an application by extending our system to support keyframe-based motion capture (Sec. 5.2). Please refer to the *supplemental video* for better dynamic motion visualization.

Runtime efficiency. Excluding user interaction, the current implementation requires a non-trivial execution time on a single NVIDIA 4090 GPU. The complete pipeline runs in approximately 2min 40s, with the following breakdown: π^3 – 30s, SAM2 – 30s, CoTracker – 10s, bounding box estimation – 30s, and MoGen with spatial guidance – 60s. In addition, hardware constraints (*e.g.*, GPU memory) introduce further overhead (about 2 minutes), like loading and unloading vision models (see the discussions in Sec. 5.1).

5.1 Ablation Study and Discussions

Impact of text input. In our MoGen, the impact of the optional input text varies depending on the abstraction level of the bounding box:

- High abstraction level: as shown in the third example in Fig. 7, a single box covering the whole banana. In this case, the text input is *indispensable*. It helps constrain the generation space; otherwise, from a single rotating box, it is impossible to reason out the motion of a "Backflip".
- Medium abstraction level: in Fig. 1(b), the character's lower limbs are explained via several boxes, but the whole upper body, including two arms, is described with a single box, without the motion control. In this case, the input text is *complementary* to the box control by specifying the action of two arms. See the generated motion in Fig. 9 with two arms lifting horizontally.
- Low abstraction level: the humanoid puppet in the second example of Fig. 6 has all the body parts covered by a separate box. In this case, the text input is *impactless* as the boxes explain the motion well.

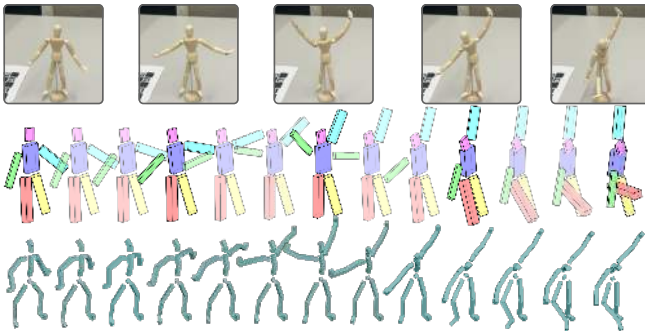

Figure 10: Top row: users specify five keyframes, indicating the motions of both arms, the torso, and one leg. Middle row: bounding boxes are estimated for the five keyframes using the same motion capture pipeline. Additional boxes are interpolated between keyframes and shown with transparent coloring. Bottom row: the complete bounding-box sequence serves as guidance to generate the final realistic character motion.

Occlusion. Because the input is monocular, our MoCap module is sensitive to occlusion. Occluded regions may be masked out by SAM, yielding incomplete point clouds. Moreover, CoTracker can lose track when large occlusions occur during puppetry. Since puppetry necessarily involves hands, complete avoidance of occlusion is difficult. A natural mitigation is multi-view capture, adding cameras at different angles. Generative completion is another direction: remove (or inpaint through) the occluding hand while reconstructing the hidden puppet geometry and motion. Participants also suggested practical aids such as "*automatically warn the user when important regions are occluded*" and "*use string puppets so hands are not in frame*". We leave these avenues to future work.

Toward real-time use. Although the current system runs non-real-time, several components can be substantially accelerated:

(a) *MoCap.* At present, three vision foundation models run sequentially to fit within a single NVIDIA 4090 GPU, and CoTracker ingests point samples taken from SAM masks to save memory. A significant fraction of runtime is spent on (i) *model swapping*—repeatedly loading/unloading large model weights between CPU and GPU, and (ii) *staging*— saving intermediate outputs (masks, point tracks, point clouds) to disk, and disk-CPU-GPU transfers to hand results from one module to the next. With sufficient GPU memory (∼100 GiB), the three models can run at the same time without switching and can share data directly on the GPU. In that case, CoTracker can track a uniformly sampled set of points directly, followed by trajectory filtering using SAM masks.

(b) *MoGen.* We currently use vanilla MDM with 1000 diffusion steps for sampling. Following established step-distillation approaches [6], this can be reduced to ≈ 20 steps, potentially yielding ∼50× speedups without materially affecting quality.

As the system is already complete in its current form, we defer these engineering optimizations to future work.

5.2 Application

Our system can be extended to support keyframe-based character animation. As illustrated in Fig. 10 (top row), the user first defines a sequence of key poses for the puppet and captures them in temporal order. These captured keyframes are treated as consecutive frames of a video sequence, which are then processed by the MoCap system to extract bounding boxes (middle row of Fig. 10). To synthesize intermediate frames, we interpolate the bounding boxes of adjacent keyframes along geodesics in SE(3), following Park et al. [35]. The playback speed can be controlled linearly by specifying the number of frames to be interpolated. Finally, the complete sequence of bounding boxes is provided to MoGen, which generates the final motion (bottom row of Fig. 10).

6 Conclusion and Future Work

We introduced *DancingBox*, a lightweight puppetry-based motion capture system that enables novices to animate characters using everyday objects and a single webcam. By conditioning generative motion models on bounding box representation, our approach transforms coarse object manipulations into realistic character motions. A user study validates that the system is effective and intuitive, supports diverse proxies, and offers design insights for tangible interfaces.

Future Work. There are inspiring directions to explore in the future.

1) Interaction between characters. The current system does not yet support direct character–character interaction. Nevertheless, such interactions can be composed by applying *DancingBox* to each puppet independently and then merging the results using 3D spatial correlations estimated from point clouds. For example, trajectories can be temporally aligned and spatially arranged based on pairwise distances, and contact cues can be derived from the reconstructed proximity. Another complementary solution is to upgrade the motion-generation module to a multi-character model [55], enabling multiple character interaction with constraints.
2) Generalization to non-human characters. Our MoGen is built upon a pre-trained motion diffusion model on human motion datasets, which limits our system to human-like motions. While it already covers diverse human actions, it could be further extended to non-human motions by leveraging a general-purpose motion generator [50].
3) Feature requests from participants. Participants proposed several practical features to enhance the system:
   - Connect the system to game engines to control in-game characters,
   - Provide explicit speed control over the generated motion,
   - Enable editing of specific joints at specific timestamps,
   - Add post-processing tools to compose and blend multiple actions.

   They also suggested a two-stage, coarse-to-fine workflow: first use an abstract proxy to craft the global trend, and then refine details with a more articulated model. We consider these feature suggestions highly inspiring and encouraging, and plan to explore them in future iterations.

References

[1] Karan Ahuja, Eyal Ofek, Mar Gonzalez-Franco, Christian Holz, and Andrew D Wilson. 2021. Coolmoves: User motion accentuation in virtual reality. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 2 (2021), 1–23.

[2] Muhammad Awais, Muzammal Naseer, Salman Khan, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, and Fahad Shahbaz Khan. 2025. Foundation models defining a new era in vision: a survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025).

[3] Samaneh Azadi, Akbar Shah, Thomas Hayes, Devi Parikh, and Sonal Gupta. 2023. Make-an-animation: Large-scale text-conditional 3d human motion generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 15039–15048.

[4] Jiawen Chen, Shahram Izadi, and Andrew Fitzgibbon. 2012. KinÊtre: animating the world with the human body. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. 435–444.

[5] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. 2023. Diffusion models in vision: A survey. *IEEE transactions on pattern analysis and machine intelligence* 45, 9 (2023), 10850–10869.

[6] Wenxun Dai, Ling-Hao Chen, Jingbo Wang, Jinpeng Liu, Bo Dai, and Yansong Tang. 2024. Motionlcm: Real-time controllable motion generation via latent consistency model. In *European Conference on Computer Vision*. Springer, 390–408.

[7] Yann Desmarais, Denis Mottet, Pierre Slangen, and Philippe Montesinos. 2021. A review of 3D human pose estimation algorithms for markerless motion capture. *Computer Vision and Image Understanding* 212 (2021), 103275.

[8] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems* 34 (2021), 8780–8794.

[9] Anthony C Fang and Nancy S Pollard. 2003. Efficient synthesis of physically valid human motion. *ACM Transactions on Graphics (TOG)* 22, 3 (2003), 417–426.

[10] Inbar Gat, Sigal Raab, Guy Tevet, Yuval Reshef, Amit Haim Bermano, and Daniel Cohen-Or. 2025. Anytop: Character animation diffusion with any topology. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 1–10.

[11] Oliver Glauser, Wan-Chun Ma, Daniele Panozzo, Alec Jacobson, Otmar Hilliges, and Olga Sorkine-Hornung. 2016. Rig animation with a tangible and modular input device. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–11.

[12] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).

[13] Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. 2024. Momask: Generative masked modeling of 3d human motions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1900–1910.

[14] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. 2022. Generating Diverse and Natural 3D Human Motions From Text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 5152–5161.

[15] Ankit Gupta, Maneesh Agrawala, Brian Curless, and Michael Cohen. 2014. Motionmontage: A system to annotate and combine motion takes for 3d animations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2017–2026.

[16] Robert Held, Ankit Gupta, Brian Curless, and Maneesh Agrawala. 2012. 3D puppetry: a kinect-based interface for 3D animation.. In *UIST*, Vol. 12. 423–434.

[17] Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598* (2022).

[18] Ching-Wen Hung, Ruei-Che Chang, Hong-Sheng Chen, Chung Han Liang, Liwei Chan, and Bing-Yu Chen. 2022. Puppeteer: Exploring intuitive hand gestures and upper-body postures for manipulating human avatar actions. In *proceedings of the 28th ACM symposium on virtual reality software and technology*. 1–11.

[19] Ching-Wen Hung, Chung-Han Liang, and Bing-Yu Chen. 2024. Fingerpuppet: finger-walking performance-based puppetry for human avatar. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. 1–6.

[20] Alec Jacobson, Daniele Panozzo, Oliver Glauser, Cédric Pradalier, Otmar Hilliges, and Olga Sorkine-Hornung. 2014. Tangible and modular input device for character articulation. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–12.

[21] Yu Jiang, Zhipeng Li, Mufei He, David Lindlbauer, and Yukang Yan. 2023. Handavatar: Embodying non-humanoid virtual avatars through hands. In *Proceedings of the 2023 CHI conference on human factors in computing systems*. 1–17.

[22] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. 2024. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831* (2024).

[23] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[24] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4015–4026.

[25] Brian Knep, Craig Hayes, Rick Sayre, and Tom Williams. 1995. Dinosaur input device. In *ACM Conference on Human Factors in Computing Systems (CHI)*.

[26] Fabrizio Lamberti, Gianluca Paravati, Valentina Gatteschi, Alberto Cannavo, and Paolo Montuschi. 2017. Virtual character animation based on affordable motion capture and reconfigurable tangible interfaces. *IEEE transactions on visualization and computer graphics* 24, 5 (2017), 1742–1755.

[27] Boyu Li, Linping Yuan, Zhe Yan, Qianxi Liu, Yulin Shen, and Zeyu Wang. 2024. Anicraft: Crafting everyday objects as physical proxies for prototyping 3d character animation in mixed reality. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. 1–14.

[28] Jiefeng Li, Jinkun Cao, Haotian Zhang, Davis Rempe, Jan Kautz, Umar Iqbal, and Ye Yuan. 2025. GENMO: A GENeralist Model for Human MOtion. *arXiv preprint arXiv:2505.01425* (2025).

[29] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. 2022. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747* (2022).

[30] Feng-Lin Liu, Hongbo Fu, Yu-Kun Lai, and Lin Gao. 2024. SketchDream: Sketch-based Text-To-3D Generation and Editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 43, 4 (2024). doi:10.1145/3658120

[31] Noah Lockwood and Karan Singh. 2012. Fingerwalking: motion editing with contact-based hand performance. In *Proceedings of the 11th ACM SIGGRAPH/Eurographics conference on Computer Animation*. 43–52.

[32] Jiageng Mao, Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. 2023. 3D object detection for autonomous driving: A comprehensive survey. *International Journal of Computer Vision* 131, 8 (2023), 1909–1963.

[33] Thomas B Moeslund, Adrian Hilton, and Volker Krüger. 2006. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding* 104, 2-3 (2006), 90–126.

[34] Naoki Numaguchi, Atsushi Nakazawa, Takaaki Shiratori, and Jessica K. Hodgins. 2011. A puppet interface for retrieval of motion capture data. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.

[35] Frank C Park and Bahram Ravani. 1997. Smooth invariant interpolation of rotations. *ACM Transactions on Graphics (TOG)* 16, 3 (1997), 277–295.

[36] Yichen Peng, Chunqi Zhao, Haoran Xie, Tsukasa Fukusato, Kazunori Miyata, and Takeo Igarashi. 2023. Dualmotion: Global-to-local casual motion design for character animations. *IEICE TRANSACTIONS on Information and Systems* 106, 4 (2023), 459–468.

[37] Matthias Plappert, Christian Mandery, and Tamim Asfour. 2016. The KIT Motion-Language Dataset. *Big Data* 4, 4 (dec 2016), 236–252. doi:10.1089/big.2016.0028

[38] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

[39] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. 2024. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714* (2024).

[40] Helge Rhodin, James Tompkin, Kwang In Kim, Edilson De Aguiar, Hanspeter Pfister, Hans-Peter Seidel, and Christian Theobalt. 2015. Generalizing wave gestures from sparse examples for real-time character control. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 1–12.

[41] Justine Saint-Aubert, Ferran Argelaguet, and Anatole Lécuyer. 2023. Tangible Avatar: Enhancing Presence and Embodiment During Seated Virtual Experiences with a Prop-Based Controller. In *2023 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 572–577.

[42] Soshi Shimada, Vladislav Golyanik, Weipeng Xu, and Christian Theobalt. 2020. Physcap: Physically plausible monocular 3d motion capture in real time. *ACM Transactions on Graphics (ToG)* 39, 6 (2020), 1–16.

[43] Zackary PT Sin, Peter Q Chen, Peter HF Ng, and Hong Va Leong. 2022. Tracking stuffed toy for naturally mapped interactive play via a soft-pose estimator. *Proceedings of the ACM on Human-Computer Interaction* 6, CHI PLAY (2022), 1–25.

[44] Chaoyue Song, Xiu Li, Fan Yang, Zhongcong Xu, Jiacheng Wei, Fayao Liu, Jiashi Feng, Guosheng Lin, and Jianfeng Zhang. 2025. Puppeteer: Rig and Animate Your 3D Models. *arXiv preprint arXiv:2508.10898* (2025).

[45] Jiaming Song, Qinsheng Zhang, Hongxu Yin, Morteza Mardani, Ming-Yu Liu, Jan Kautz, Yongxin Chen, and Arash Vahdat. 2023. Loss-guided diffusion models for plug-and-play controllable generation. In *International Conference on Machine Learning*. PMLR, 32483–32498.

[46] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. 2022. Human motion diffusion model. *arXiv preprint arXiv:2209.14916* (2022).

[47] Shinji Umeyama. 2002. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on pattern analysis and machine intelligence* 13, 4 (2002), 376–380.

[48] Jian Wang, Zhe Cao, Diogo Luvizon, Lingjie Liu, Kripasindhu Sarkar, Danhang Tang, Thabo Beeler, and Christian Theobalt. 2024. Egocentric whole-body motion capture with fisheyevit and diffusion-based motion refinement. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 777–787.

[49] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. 2025. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 5294–5306.

[50] Xuan Wang, Kai Ruan, Liyang Qian, Zhizhi Guo, Chang Su, and Gaoang Wang. 2025. X-MoGen: Unified Motion Generation across Humans and Animals. *arXiv preprint arXiv:2508.05162* (2025).

[51] Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. 2025. π^3: Scalable Permutation-Equivariant Visual Geometry Learning. *arXiv preprint arXiv:2507.13347* (2025).

[52] Zhenzhi Wang, Jingbo Wang, Dahua Lin, and Bo Dai. 2023. Intercontrol: Generate human motion interactions by controlling every joint. *arXiv preprint arXiv:2311.15864* 3 (2023).

[53] Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. 2023. Omnicontrol: Control any joint at any time for human motion generation. *arXiv preprint arXiv:2310.08580* (2023).

[54] Shuyang Xu, Zhiyang Dou, Mingyi Shi, Liang Pan, Leo Ho, Jingbo Wang, Yuan Liu, Cheng Lin, Yuexin Ma, Wenping Wang, et al. 2025. MOSPA: Human Motion Generation Driven by Spatial Audio. *arXiv preprint arXiv:2507.11949* (2025).

[55] Wenning Xu, Shiyu Fan, Paul Henderson, and Edmond S. L. Ho. 2025. Multi-Person Interaction Generation from Two-Person Motion Priors. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '25)*. Association for Computing Machinery, New York, NY, USA, Article 113, 11 pages.

[56] Zhangsihao Yang, Mingyuan Zhou, Mengyi Shan, Bingbing Wen, Ziwei Xuan, Mitch Hill, Junjie Bai, Guo-Jun Qi, and Yalin Wang. 2024. Omnimotiongpt: Animal motion generation with limited data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1249–1259.

[57] Hui Ye, Kin Chung Kwan, Wanchao Su, and Hongbo Fu. 2020. ARAnimator: In-situ character animation in mobile AR with user-defined motion gestures. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 83–1.

[58] Wataru Yoshizaki, Yuta Sugiura, Albert C Chiou, Sunao Hashimoto, Masahiko Inami, Takeo Igarashi, Yoshiaki Akazawa, Katsuaki Kawachi, Satoshi Kagami, and Masaaki Mochimaru. 2011. An actuated physical puppet as an input device for controlling a digital manikin. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 637–646.

[59] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*. 3836–3847.

[60] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. 2024. Motiondiffuse: Text-driven human motion generation with diffusion model. *IEEE transactions on pattern analysis and machine intelligence* 46, 6 (2024), 4115–4128.

[61] Xin-Yang Zheng, Hao Pan, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. 2023. Locally attentional sdf diffusion for controllable 3d shape generation. *ACM Transactions on Graphics (ToG)* 42, 4 (2023), 1–13.

[62] Lei Zhong, Chuan Guo, Yiming Xie, Jiawei Wang, and Changjian Li. 2025. Sketch2anim: Towards transferring sketch storyboards into 3d animation. *ACM Transactions on Graphics (TOG)* 44, 4 (2025), 1–15.