AnimoFlow

Guy Tevet

AnimoFlow is an open platform that makes motion-generation models accessible to creators within the tools they already use. Despite advances in motion quality and control, research outputs remain difficult to use on custom characters in established animation workflows. AnimoFlow bridges this gap with a shared pipeline that converts and retargets model outputs into editable character animation, exposed through a Blender add-on, ComfyUI nodes, a web interface, and an API. Researchers can extend the pipeline with new models and processing components. We hope this infrastructure will ease adoption of research advances and bring feedback from creative practice.

Website: animoflow.ai
Correspondence: guy@animoflow.ai



1 Introduction

Human motion generation models can synthesize realistic motion across a wide range of actions and styles (Tevet et al., 2023; Guo et al., 2024; Peng et al., 2018, 2021, 2022). The emergence of text-to-motion diffusion models brought a leap in motion fidelity and diversity, with natural language providing an intuitive way to specify the desired behavior (Tevet et al., 2023). Subsequent work built on these models to offer control over trajectories and waypoints (Karunratanakul et al., 2023; Xie et al., 2024), keyframes (Cohan et al., 2024; Goel et al., 2025), and style (Sawdayee et al., 2026; Zhong et al., 2024), as well as conditioning on music (Li et al., 2021; Siyao et al., 2022; Tseng et al., 2023) and interactions with scenes and objects (Hassan et al., 2021; Xu et al., 2023; Li et al., 2023; Tevet et al., 2025; Ron et al., 2025). Much of this progress comes from open research, with code and pretrained models publicly available for others to use and extend.

Despite this progress, advances in motion quality and control remain difficult for animators to put to use. Character animation has yet to see the broad adoption that Stable Diffusion (Rombach et al., 2022) and

ComfyUI (comfyanonymous and Comfy Org, 2023) helped bring to open image generation. Research models typically generate motion for standardized human skeletons or body models, whereas animators work with characters whose proportions and rigs are tailored to their designs, often deliberately departing from realistic human anatomy. They need editable motion on these characters within tools such as Blender (Blender Online Community, 2026) or Maya (Autodesk, Inc., 2026). Turning research outputs into usable animation therefore requires retargeting, software packaging, and interface development, work that falls outside the scope of most individual research projects and demands expertise beyond the animator's usual workflow. Without bridging this gap, the quality and control demonstrated in research remain largely out of reach in everyday animation practice.

We argue that this bridge is crucial for both communities. Animators gain access to research models within familiar tools, while researchers gain feedback grounded in creative practice, from control granularity and editability to motion artifacts and compatibility with existing workflows. This exchange exposes limitations that standardized evaluations on motion-capture datasets (Guo et al., 2022) may miss and helps steer research toward problems that matter to content creators. Adoption also provides a complementary measure of progress, testing whether advances in motion generation translate into tools that practitioners find useful.

We present **AnimoFlow**, an open platform that bridges motion generation research and animation practice through a shared pipeline for turning model outputs into editable character animation. By handling the conversion, retargeting, and integration work between a research model and an animator's character, AnimoFlow makes these models available for use and evaluation within creative workflows. Researchers can bring their models into this setting without developing a separate animation tool around each method.

AnimoFlow exposes a common backend through complementary interfaces that accommodate the overlapping workflows of animators, makers, researchers, and LLM-based agents. Models such as MDM (Tevet et al., 2023) and Kimodo (Rempe et al., 2026) can be explored through the web interface, used within animation scenes through the Blender add-on (Blender Online Community, 2026), or combined with other generative methods through ComfyUI nodes (comfyanonymous and Comfy Org, 2023). A REST API supports programmatic access for experiments, custom applications, and agent-driven workflows.

These interfaces share an open animation pipeline that wraps the generators with common preprocessing and post-processing stages, adapting user prompts and controls to each model and converting its output into editable motion on the target character. Users can extend the pipeline with new models, processing methods, and workflows. Contributed models gain access to the platform's existing animation tools and interfaces, while improvements to shared processing can benefit multiple models. Researchers and practitioners can therefore build on one another's work without repeatedly developing the infrastructure that connects generation to usable animation.

Our aim is for AnimoFlow to support a continuing exchange between motion generation research and creative practice. As new models and workflows emerge, shared infrastructure can shorten the path from research release to practical use, while experience from that use can inform the next generation of methods. We see this exchange as a foundation for animation tools whose development is shaped jointly by advances in research and the needs of their users.

2 Capabilities

Motion generation. AnimoFlow generates human character animation from natural-language descriptions of actions and styles. A language preprocessing stage translates multilingual descriptions into English motion descriptions using captions from HumanML3D (Guo et al., 2022) as a reference. Users select a generation model and a supported target character, then obtain motion applied to that character's rig. The same workflow accommodates different models, allowing users to explore their outputs without handling each model's native motion representation.

Spatial and temporal control. Beyond describing an action, users can specify where the character moves and how actions unfold over time. Trajectory control guides motion along a ground-plane path, while timed waypoints specify target positions at selected moments. Timeline composition combines successive action

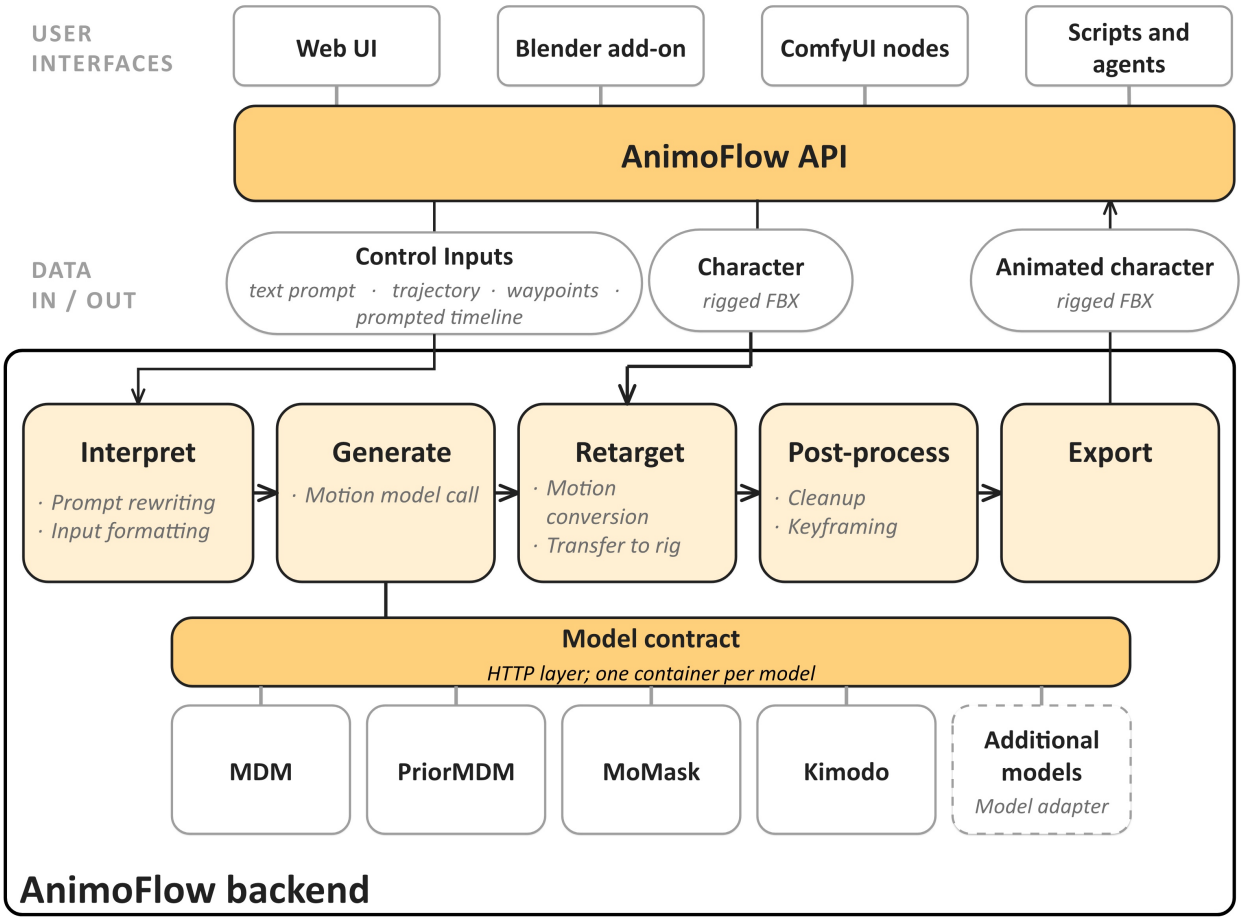

Figure 1 System overview. The AnimoFlow API denotes the access layer connecting client interfaces to the animation pipeline. The backend prepares the control inputs, invokes a wrapped motion model, and converts its output into editable animation on a target character. Model-specific adapters connect research implementations to the shared animation pipeline.

descriptions, each with its own duration, into a continuous motion sequence. The available controls depend on the selected model. Figures 2 and 4 illustrate trajectory control within a Blender scene and an agent-driven data-generation workflow, respectively.

Editable outputs. Generated motion is returned as animation on a rigged character in standard FBX or GLB (binary glTF) formats. These files can be imported into compatible animation tools and game engines, allowing users to revise the motion and incorporate it into existing scenes beyond Blender. Optional processing, including ground alignment and keyframe reduction, helps prepare the motion for subsequent editing.

3 System Design

AnimoFlow turns motion-generation models into tools for creating editable animation on user-selected characters. Client interfaces provide access from animation applications, node-based workflows, and code, while a shared backend runs the animation pipeline (Figure 1). Its *Interpret* stage prepares language and spatial controls (Section 3.1); *Generate* runs the selected motion model (Section 3.2); and *Retarget* converts its output and transfers motion to the target rig (Section 3.3). *Post-process* aligns the character with the ground and optionally reduces keyframes (Section 3.4); *Export* restores scene placement and writes animation files (Section 3.5). The access layer supports self-hosted and cloud execution (Section 3.6), with task requests and client workflows described in Section 3.7.

3.1 Interpret: control preprocessing

The interpretation stage prepares inputs for motion generation through language translation and rephrasing, and spatial-control coordinate transformations.

Language preprocessing. User instructions may differ from motion-training captions in both language and phrasing. We address this with retrieval-assisted rewriting into short English motion descriptions, using HumanML3D captions (Guo et al., 2022) as a reference. Given a text prompt, the multilingual sentence encoder `paraphrase-multilingual-MiniLM-L12-v2` (Reimers and Gurevych, 2020) retrieves the three most similar captions by cosine similarity from a precomputed corpus of approximately 53,000 unique HumanML3D captions. These captions provide contextual examples to Qwen2.5-1.5B-Instruct (Yang et al., 2024a), which translates non-English prompts and rephrases the requested motion as a concise English caption matching the style of the training descriptions, without additional training.

Our instruction prompt to Qwen specifies a single present-tense motion description with an explicit human subject. It instructs the model to remove conversational framing, preserve the requested action, and avoid introducing unspecified direction, speed, or style. Qwen rewriting uses greedy decoding with a maximum of 48 new tokens; the instruction requests a caption of at most 30 tokens. The same rewriting procedure serves all motion models, and timeline requests are processed separately for each segment. We chose the 1.5B model for its low inference cost and modest memory footprint, with both CPU and GPU execution supported. Despite its small size, Qwen2.5-1.5B-Instruct performs comparably to larger Qwen variants on this constrained translation and rephrasing task.

To avoid unnecessary changes to already suitable descriptions, the default mode bypasses rewriting when the input contains only ASCII characters, begins with a recognized person-subject phrase, contains a verb from a predefined motion vocabulary, and has at most 30 words. Callers can also explicitly force or skip rewriting. This heuristic provides a lightweight approximation of caption-style input, while the original and processed prompts are retained in the job record.

Spatial controls preprocessing. User-drawn paths and waypoints are expressed in scene coordinates, whereas generators may assume a canonical starting position and heading. We transform these controls into the model's coordinate convention and retain the inverse transformation for the output stage. This allows generation to follow the model's conventions while placing the resulting animation back into the user's scene.

3.2 Generate: motion models and wrappers

The generation stage takes the preprocessed controls and produces motion on the selected model's skeleton, using wrappers that translate the common inputs into each method's inference format. We reuse existing pretrained models and inference procedures, with AnimoFlow providing the surrounding adaptation and execution infrastructure. The available models offer different controls and computational requirements:

MDM. MDM (Tevet et al., 2023) is a lightweight text-conditioned diffusion transformer; our integration uses its released 50-step checkpoint on CPU or GPU, making it suitable for local text-to-motion generation without dedicated GPU hardware.

PriorMDM. PriorMDM (Shafir et al., 2024) extends MDM with trajectory conditioning and DoubleTake temporal composition. It supports CPU and GPU inference; our default service configuration uses CPU. Its trajectory conditioning and timeline composition provide spatial and temporal control even on machines without a dedicated GPU.

MoMask. MoMask (Guo et al., 2024) generates text-conditioned motion through masked modeling of residual-quantized motion tokens. It supports CPU and GPU inference and is suited to rapid iteration on text prompts, with lower generation latency than our diffusion integrations.

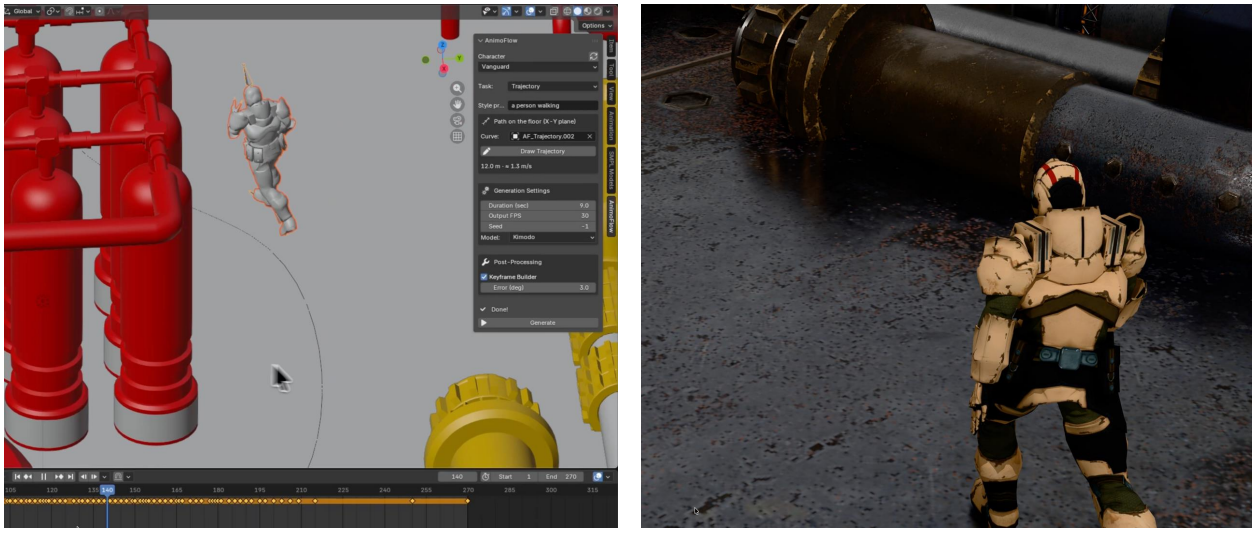

Figure 2 Motion generation in Blender. Left: the AnimoFlow add-on generates animation for the Vanguard character using Kimodo (Rempe et al., 2026), conditioned on a trajectory drawn in the scene. The panel provides generation and post-processing settings, and the resulting keyframes are available for editing in Blender. Right: a render of the animated character. The demo video is available at animoflow.ai.

Kimodo. Kimodo (Rempe et al., 2026) is a larger diffusion model with separate root and body denoising stages. Our supported configuration is GPU-only, running both text encoding and motion denoising on GPU to accommodate their computational cost. It exposes text and sparse waypoint controls, making it suitable when timed spatial constraints are needed and GPU resources are available.

The wrappers expose each model's supported tasks, parameters, and output representation. They preserve differences between methods, such as support for temporal composition or spatial constraints, rather than assuming that every model accepts the same controls. Model-specific dependencies are isolated from the rest of the system, allowing research implementations with different software requirements to coexist.

The pipeline also accommodates differences in generated motion representations. Models that return joint positions require inverse kinematics to reconstruct skeletal rotations (Section 3.3), whereas models that already predict rotations can retain that information. Both paths produce a common skeletal animation intermediate for character retargeting. Consequently, integrating a model requires adapting its inputs and outputs without reproducing the downstream character-processing stages.

3.3 Retarget: motion conversion and rig transfer

The retargeting stage converts generated motion into skeletal animation and transfers it to the target character, separating the skeleton used by the motion model from the rig to be animated. We use BVH as the skeletal intermediate because it represents a hierarchy, local joint rotations, and root translation independently of the character mesh. This provides a common boundary between model-specific motion representations and target-specific rig mappings.

Reconstructing rotations from positions. For positional outputs, we build on the inverse-kinematics converter released with MoMask (Guo et al., 2024). We first resample joint trajectories to the requested output frame rate, then recover rotations on the intermediate skeleton. Independent framewise fitting can select inconsistent rotational solutions, producing abrupt changes even when the joint trajectories are smooth. We therefore initialize each frame from the preceding frame's solution and refine it using the upstream IK updates. The current configuration uses ten iterations for the first frame and one for each subsequent frame.

We additionally adapt the intermediate skeleton's lower-body bone lengths to the generated clip using median bone-length estimates, averaged across corresponding left and right limbs. This reduces discrepancies between

the generated joint geometry and the conversion template, which can otherwise distort ankle and foot orientations. The upper-body template is retained because extending the same adjustment to the upper body degraded poses in our development comparisons.

Preserving predicted rotations. Kimodo follows a separate conversion path because its SOMA representation already provides joint rotations. We map these rotations to the intermediate skeleton using fixed rest-frame alignments and calibrated per-joint corrections. This retains rotational information, including bone twist, that would be lost by reducing the output to joint positions and reconstructing it through IK. The corrections were calibrated against our earlier joint-regressor-based conversion; the resulting constants are applied directly at runtime.

Transferring motion to a target rig. For character animation, we adapt the bone rotation and position transfer in Nick Keeline's KeeMap addon[1] to run without interactive Blender operations. Character mappings establish the correspondence between the intermediate skeleton and the target rig, including the corrections needed for different rest poses and bone conventions. AnimoFlow supplies the orchestration, rig-convention handling, and import/export integration around this existing retargeting method.

Humanoid robots use a separate target adaptation based on General Motion Retargeting (GMR) (Araújo et al., 2026). We adapt its configurations to the common skeleton and integrate the resulting root motion and joint angles into the output pipeline. Target-specific adjustments include trajectory scaling and direct elbow-angle transfer for the Unitree G1, where the position-based fit can otherwise produce excessive elbow flexion. These adaptations retain GMR as the underlying retargeting solver.

3.4 Post-process: ground alignment and keyframe reduction

The post-processing stage operates on the retargeted character to correct ground placement and optionally simplify the animation for editing. Retargeting changes the relationship between skeletal motion and the visible character surface. Differences in character proportions and rig conventions can therefore leave the animated mesh vertically displaced from the ground. We correct this displacement after retargeting, using the target mesh rather than assuming that a particular skeletal joint coincides with the contact surface.

We implement ground alignment using Blender's evaluated mesh geometry (Blender Online Community, 2026). The estimator identifies the lowest five percent of mesh vertices in the rest pose. At sampled animation frames (every second frame by default), it measures the fifth percentile of their vertical positions, then takes the fifth percentile of these measurements across the clip. The resulting estimate determines a single vertical translation of the character. These percentile choices reduce sensitivity to isolated skinning artifacts and unusually low frames, while the constant translation preserves the motion's vertical variation. This operation aligns the character with the ground; it does not enforce contact constraints or remove foot sliding.

An optional keyframe-reduction stage prepares dense generated animation for further editing. Our headless implementation follows the principle of error-based curve decimation, greedily removing intermediate keys according to interpolation error while coordinating retained times across the skeleton. It uses Blender's animation-curve API and automatic clamped Bézier handles (Blender Online Community, 2026).

3.5 Export: animation files and scene placement

The export stage packages the processed animation for use in client applications and downstream animation tools. It restores the scene transformation retained during control preprocessing, placing the animation in the coordinate frame of the original request, and exports the animated character as FBX or GLB. For reduced-keyframe animation, GLB export preserves the sparse curves rather than resampling them into dense animation.

[1] https://github.com/nkeeline/Keemap-Blender-Rig-ReTargeting-Addon

Client request POST /v1/jobs

```
{
  "input": {
    "type": "waypoints",
    "prompt": "a person walks",
    "waypoints": [
      {"x": 0, "y": 0, "z": 0, "t": 0},
      {"x": 0, "y": 0, "z": 1, "t": 45},
      {"x": 1, "y": 0, "z": 2, "t": 90}
    ]
  },
  "model": "kimodo",
  "character": "Y_bot",
  "duration": 4.0,
  "seed": 42
}
```

Server response Job accepted

```
{
  "job_id": "<id>"
}
```

The client polls GET /v1/jobs/<id> for completion.

Server response Animation ready

```
{
  "job_id": "<id>",
  "status": "done",
  "download_url":
    "/v1/files/<id>.glb",
  "output_fps": 30
}
```

Listing 1 A typical interaction with the AnimoFlow REST API. The client submits a waypoint-controlled animation request, specifying a motion model and target character. The server returns a job identifier for asynchronous processing. The client then polls the job status until the response includes a download URL for the generated animation. Selected response fields are shown for clarity.

3.6 API access and execution

The access layer connects clients to the animation pipeline, with different execution paths for self-hosted installations and AnimoFlow's cloud service. Both expose the AnimoFlow REST API, which accepts animation tasks through their control inputs, motion model, and target character (Listing 1). A self-hosted ComfyUI installation additionally accepts workflow graphs through ComfyUI's own backend API. The cloud service accepts task requests, not arbitrary ComfyUI graphs. In Figure 1, *AnimoFlow API* collectively denotes these access paths.

For REST requests to a self-hosted installation, the API preprocesses the controls and constructs a pipeline plan from shared stage definitions. It compiles this plan into a ComfyUI workflow graph and submits it to the local ComfyUI backend, whose nodes invoke the processing operations and model containers. Users working directly in ComfyUI can instead submit their own graphs containing AnimoFlow nodes. Those graphs bypass the task-level REST API; preprocessing is performed by the corresponding nodes included in the graph.

AnimoFlow's cloud service uses the same task-level REST interface and shared pipeline plan, but executes the stages directly through a Python runner and processing subprocesses, without constructing or submitting a ComfyUI graph. Shared stage definitions and processing code allow both execution paths to use the same animation pipeline. A locally running API can also be configured as a proxy to the cloud service; in that mode it forwards REST requests rather than executing them locally.

3.7 Requests and client workflows

The REST interface lets clients specify an animation task without constructing its processing graph. A request contains the control inputs, model, and target character, with parameters such as duration and random seed. Its input type distinguishes text descriptions, spatial trajectories, sparse waypoints, and timelines of successive actions. Listing 1 illustrates a waypoint request combining a walking prompt, three timed positions, a target character, and Kimodo.

Requests are processed asynchronously: submission returns a job identifier, which the client uses to poll for progress and retrieve the completed animation. The API reference[2] documents the request types, available

[2] https://animoflow-alpha.pages.dev/redoc

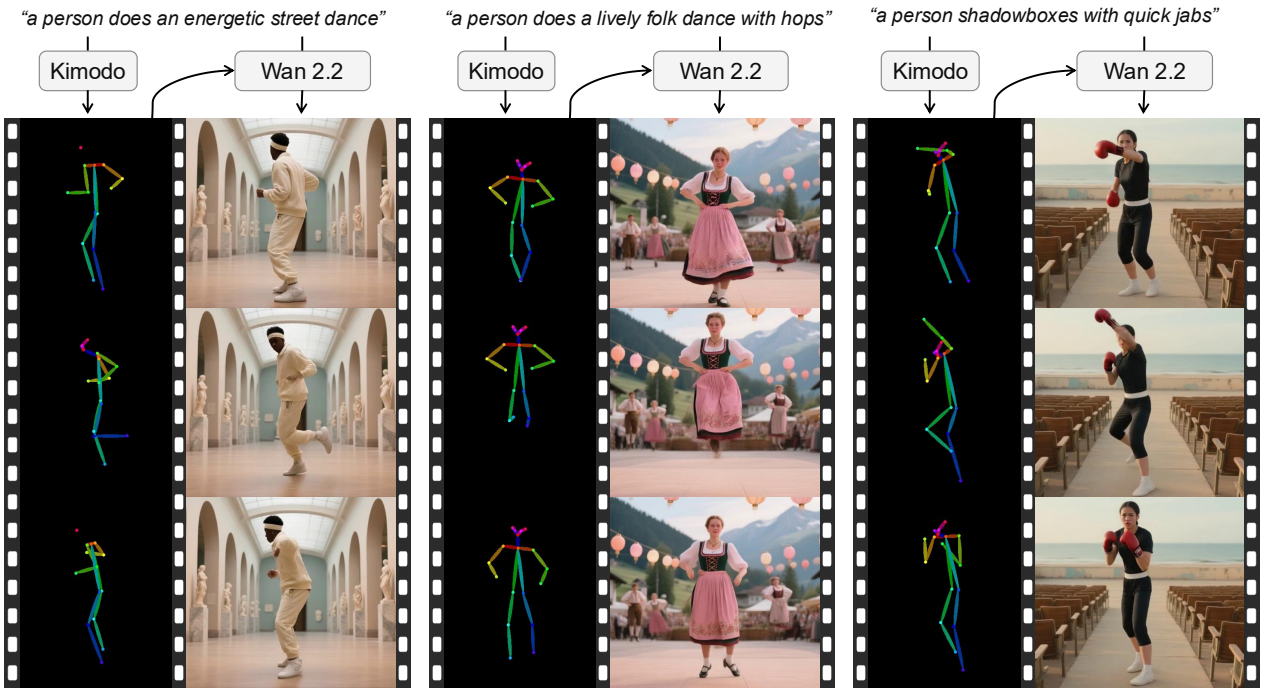

Figure 3 Motion-conditioned video generation in ComfyUI. AnimoFlow nodes connect motion generation to image and video models in a single ComfyUI graph (comfyanonymous and Comfy Org, 2023). Motion from Kimodo (Rempe et al., 2026) provides the pose sequence; Qwen-Image (Wu et al., 2025a) provides a reference image for Wan 2.2 Fun-Control (Wan et al., 2025). Each pair shows the pose control on the left and the generated video on the right, with time progressing downward. The workflow is available at https://github.com/AnimoFlow/comfyui-animoflow/blob/main/workflows/text_kimodo_video.json.

controls, and output fields.

The web application uses the cloud service for generation and inspection without installing the processing stack. The Blender add-on (Blender Online Community, 2026) connects to either the cloud service or a self-hosted REST endpoint, letting users specify controls within a scene and continue editing the resulting animation there (Figure 2). Scripts and agents use the same REST requests with either deployment. ComfyUI users work with a self-hosted ComfyUI backend, composing AnimoFlow nodes with other operations in its graph editor (comfyanonymous and Comfy Org, 2023).

4 Applications

AnimoFlow's interfaces allow generated motion to serve as an input to other workflows. We illustrate this with pose-conditioned video generation in ComfyUI and an agent-operated pipeline for generating robot locomotion data through the REST API.

4.1 Driving video generation

ComfyUI integration connects AnimoFlow to the broader ecosystem of generative models and processing nodes available in ComfyUI. Users can combine its motion-generation and animation-processing nodes with other components in the same graph. We demonstrate this through a workflow that uses generated motion to guide image and video synthesis (Figure 3).

Video-generation models can produce visually detailed scenes while exhibiting discontinuities in human motion (Ling et al., 2025) and anatomical or biomechanical inconsistencies (Fang et al., 2026). Controlling a character's path and the timing of its actions also requires more precise inputs than a text description alone.

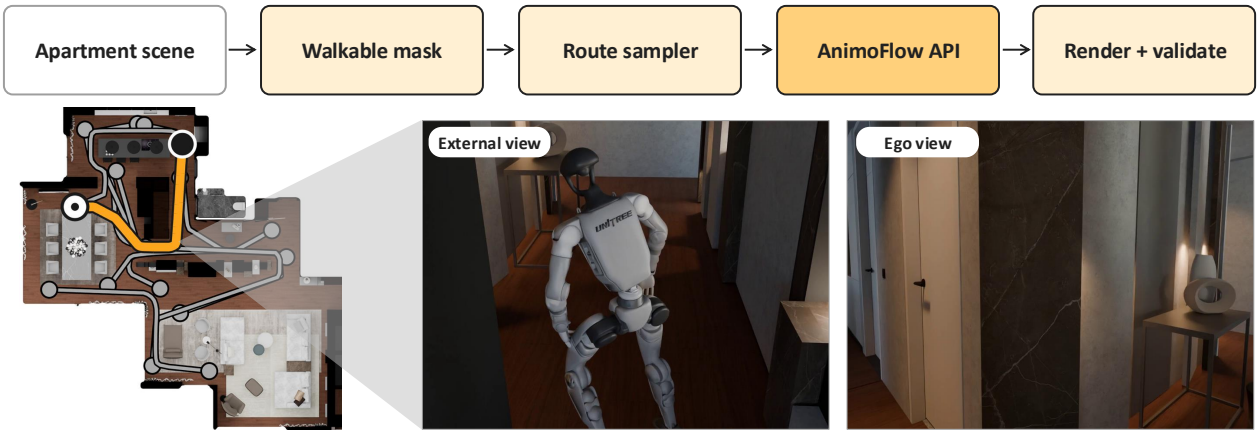

Figure 4 Generating locomotion data through the API. A language-model agent uses a CAD apartment and the AnimoFlow API to generate motions for Unitree G1. The left panel shows a selection of generated trajectories in the apartment, with one highlighted in orange. The right panels show one frame of the motion corresponding to that trajectory, from an external camera and the robot's egocentric view.

Spatial and temporal controls for video generation remain an active research area (Yin et al., 2023; Wu et al., 2025b; Li et al., 2025). Motion-generation models offer complementary controls over trajectories, waypoints, and action sequences (Karunratanakul et al., 2023; Xie et al., 2024; Shafir et al., 2024), which AnimoFlow exposes through its generation interfaces (Section 2).

Pose-conditioned video models use a sequence of skeletal poses to guide the depicted motion (Hu et al., 2023; Zhang et al., 2025). Prior work has used generated motion as this conditioning signal (Nam et al., 2025). Our example combines motion generation, pose rendering, and video synthesis using this approach; individual models can be replaced by others with compatible conditioning inputs.

Figure 3 shows a workflow in which Kimodo generates motion from a text prompt. We render the motion as an OpenPose control video from a chosen camera, then use its first pose to condition Qwen-Image (Wu et al., 2025a) through a ControlNet (Zhang et al., 2023). The resulting image specifies the character's appearance and initial pose. Wan 2.2 Fun-Control (Wan et al., 2025) uses this image and the control video to synthesize the final sequence. Deriving both conditioning inputs from the same motion encourages agreement between the initial image and the subsequent pose sequence.

4.2 Generating training data

Motion generation offers a way to produce reference trajectories for robot learning without recording each clip through motion capture or teleoperation. AnimoFlow can generate motion from text and spatial controls, then retarget it to a robot rig (Section 3.3). Its REST API also allows an agent to organize generation and validation around a scene.

We provided a language model with a code interpreter, the API reference, and an unlabeled CAD apartment containing 719 meshes, and asked it to generate locomotion data for the Unitree G1. The agent implemented and ran a workflow that extracts traversable floor space, samples trajectories, requests motion, and checks the results against the scene geometry (Figure 4). The scene provides context for both validation and rendering, including external and robot-mounted camera views.

The workflow identifies traversable floor space from scene geometry using selected floor and rug meshes. It rasterizes the apartment into a $c = 5\,\text{cm}$ grid, marking a cell free when a downward ray from standing height reaches a floor or rug surface with an upward normal and the 1.5 m column above it is clear. This produces a free-space mask M_{free}. To provide clearance for the robot, the workflow forms $M_{\text{safe}} = M_{\text{free}} \ominus (r + \delta)$, with robot radius $r = 0.2\,\text{m}$ and margin $\delta = 0.15\,\text{m}$, and a stricter planning mask $M_{\text{plan}} = M_{\text{safe}} \ominus 1.5c$.

For each trajectory $\tau = (p_1, \ldots, p_K)$, the start point A is sampled uniformly from M_{plan}. The endpoint B is sampled from cells at geodesic distance $d_g(A, B) \in [7.5, 11]\,\text{m}$, corresponding to approximately 6–9 s of

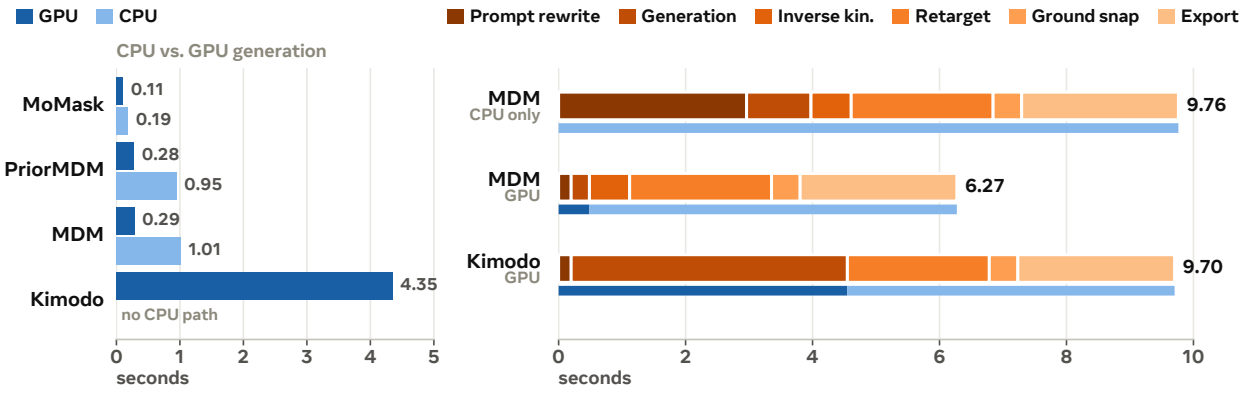

Figure 5 Component runtime profiling. Measurements on an AMD Ryzen 9 5950X and NVIDIA RTX A4000 (16 GB), excluding loading and warm-up. Left: generation timings on CPU and GPU. Right: component breakdowns for three configurations, including prompt rewriting. Totals sum separately measured components rather than reporting end-to-end API latency.

walking at $1.25\,\mathrm{m\,s^{-1}}$. An $A^\star$ search on the 8-connected grid disallows corner cutting (Hart et al., 1968). The resulting path is shortened by string pulling and smoothed with constrained Chaikin subdivision (Chaikin, 1974); each corner cut must remain within M_{plan} and preserve line of sight along the new chord. Paths are retained only if samples spaced at 2.5 cm remain inside M_{safe}. A second candidate passes through a random detour waypoint and is retained when its symmetric Chamfer distance from the first path exceeds 1 m. This procedure produced 185 candidate trajectories for the apartment.

We submit each path through the REST API using Kimodo's trajectory-conditioning mode and the text prompt "a person walks." The requested duration is the path length divided by a nominal walking speed of $1.25\,\mathrm{m\,s^{-1}}$, clamped to 6–9 s. This sets the time available to traverse the path rather than enforcing a fixed walking speed. AnimoFlow generates the motion and retargets it to the G1 rig; different random seeds provide alternative motions for the same path.

The workflow checks each result against the scene. It tests the lowest mesh point for floor penetration and airborne intervals longer than 0.4 s, checks that the root remains inside M_{safe}, and detects body–environment intersections using triangle tests accelerated by a bounding-volume hierarchy over nearby meshes. Endpoint offset and mean lateral deviation measure adherence to the requested trajectory. Rejected motions are retried with another seed. Of 170 generation attempts, 90 passed these checks and 80 were rejected.

These checks identify geometric failures such as collisions and prolonged loss of ground contact, but do not establish gait quality or physical trackability on the robot. The example demonstrates how an agent can use the API to assemble a scene-conditioned generation workflow; transfer to other environments and benefits for robot learning remain to be evaluated. The 90 accepted motions are rendered on the project website.

5 Runtime Profiling

We profile the computational cost of motion generation and the processing required to produce editable character animation. Figure 5 compares CPU and GPU generation and shows the surrounding pipeline costs for representative configurations. GPU acceleration reduces generation time, while conversion, retargeting, and export remain substantial contributors to the total. These measurements characterize computational cost rather than user-perceived latency, which also depends on deployment and communication overhead.

6 Contributing to AnimoFlow

AnimoFlow is intended as a shared bridge between motion research and creative practice, which researchers and creators can extend with their own models and processing tools. The open-source node pack and publicly

available model interface provide two complementary ways to contribute: researchers can bring new generators into existing animation workflows, while creators and developers can contribute nodes that expand how generated motion is controlled, processed, and used.

Model contributors retain their implementation and weights in their own repository and package inference behind the model-container interface described in Section 3.2. We provide a container template and a smoke test for validating that interface; contributors can then submit an entry to the community registry. Models with additional controls or output representations also require corresponding input or conversion adapters. The model contribution guide documents the interface, template, testing procedure, and registration process.

Processing contributions can add or improve ComfyUI nodes and the shared operations they invoke. Integrating an operation into the shared pipeline makes it available beyond manually constructed node graphs, through the standard client interfaces. Development setup and pull-request instructions are provided in the node-pack contribution guide. Licensing and contributor terms are described in Section 7.

7 Licensing and Governance

AnimoFlow's components have separate licenses. The ComfyUI node pack and Blender add-on are open source under AGPL-3.0 and GPL-3.0, respectively. The API server and deployment wrapper are source-available under the AnimoFlow Community License, which permits reading, modification, and self-hosting. Use, including commercial production, is free for individuals, non-profits, academia, and organizations below the size threshold specified in the license; larger organizations require a commercial license. This grant is perpetual for releases distributed under those terms. The web application is closed source, and the hosted demo is free to use.

Integrated models remain subject to their own code, weight, and dataset licenses, documented separately for each model. The container interface described in Section 3.2 allows contributed models to retain their existing licenses, including research-only terms. Integrating a model does not require relicensing it or signing a contributor license agreement (CLA). Contributions to AnimoFlow's own repositories require a one-time CLA.

The per-repository license map, full license texts, and FAQ are available at animoflow.ai/guide/licensing and github.com/AnimoFlow/legal.

8 Limitations and Future Directions

AnimoFlow currently focuses on human motion and supported humanoid rigs. Its available controls depend on the integrated generators, and extending support to new models and characters requires appropriate adapters and rig mappings. As motion-generation research expands beyond humans, we aim to extend the pipeline to animals and other non-human characters. Research on motion synthesis for animals and characters with diverse skeletal structures offers promising foundations for this extension (Li et al., 2022; Raab et al., 2024; Yang et al., 2024b; Gat et al., 2025; Liu et al., 2025). Supporting these models will require accommodating their representations and controls while preserving the character-specific properties that make their motion distinctive.

The quality of the final animation remains dependent on both the underlying generator and the subsequent processing stages. Conversion and retargeting can introduce artifacts when transferring motion between skeletons with different proportions or rig conventions. Ground alignment corrects overall placement but does not ensure consistent contacts, prevent foot sliding, or enforce collision constraints within a scene. Stronger character- and scene-aware processing is therefore an important direction, particularly when it can improve physical consistency while retaining the motion's intended style and editability.

The system's practical value also depends on how well it fits creators' working environments. We aim to extend access to applications such as Maya, Unreal Engine, and Unity, allowing users to generate and revise motion within the tools they already use. Runtime profiling characterizes computational cost, but does not establish how the system affects creative workflows. Working with creators on actual projects will help us understand where AnimoFlow saves effort, where it introduces friction, and which controls and integrations

are most useful. We intend this feedback to guide both interface development and improvements to the underlying pipeline.

9 Conclusion

We presented AnimoFlow, a system that connects motion-generation research to editable character animation through a shared processing pipeline and interfaces for creators, developers, and agents. By adapting existing generators to common animation workflows, AnimoFlow makes their capabilities accessible beyond their original research implementations. We envision this bridge developing through contributions from both researchers, who bring new models and controls, and creators, whose tools, applications, and feedback shape how those advances are used. In this way, progress in motion generation can reach creative practice more directly, while practical needs help inform the next research questions.

Acknowledgments

We thank Liron Tevet for advice on user experience.

References

João Pedro Araújo, Yanjie Ze, Pei Xu, Jiajun Wu, and C. Karen Liu. Retargeting matters: General motion retargeting for humanoid motion tracking. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, page 6097–6104. IEEE, June 2026. doi: 10.1109/icra57385.2026.11697263. URL http://dx.doi.org/10.1109/icra57385.2026.11697263.

Autodesk, Inc. Autodesk Maya, 2026. URL https://www.autodesk.com/products/maya.

Blender Online Community. Blender – a 3D modelling and rendering package, 2026. URL https://www.blender.org.

George Merrill Chaikin. An algorithm for high-speed curve generation. *Computer Graphics and Image Processing*, 3(4):346–349, December 1974. ISSN 0146-664X. doi: 10.1016/0146-664x(74)90028-8. URL http://dx.doi.org/10.1016/0146-664X(74)90028-8.

Setareh Cohan, Guy Tevet, Daniele Reda, Xue Bin Peng, and Michiel van de Panne. Flexible motion in-betweening with diffusion models. In *ACM SIGGRAPH 2024 conference papers*, pages 1–9, 2024.

comfyanonymous and Comfy Org. ComfyUI, 2023. URL https://github.com/comfyanonymous/ComfyUI.

Yusu Fang, Tiange Xiang, Tian Tan, Narayan Schuetz, Scott Delp, Li Fei-Fei, and Ehsan Adeli. Humanscore: Benchmarking human motions in generated videos, 2026. URL https://arxiv.org/abs/2604.20157.

Inbar Gat, Sigal Raab, Guy Tevet, Yuval Reshef, Amit Haim Bermano, and Daniel Cohen-Or. Anytop: Character animation diffusion with any topology. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, SIGGRAPH Conference Papers '25, page 1–10. ACM, July 2025. doi: 10.1145/3721238.3730621. URL http://dx.doi.org/10.1145/3721238.3730621.

Purvi Goel, Guy Tevet, C. Karen Liu, and Kayvon Fatahalian. Generating detailed character motion from blocking poses. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, SA Conference Papers '25, page 1–7. ACM, December 2025. doi: 10.1145/3757377.3763874. URL http://dx.doi.org/10.1145/3757377.3763874.

Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5152–5161, June 2022.

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

Peter Hart, Nils Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. ISSN 0536-1567. doi: 10.1109/tssc.1968.300136. URL http://dx.doi.org/10.1109/TSSC.1968.300136.

Mohamed Hassan, Duygu Ceylan, Ruben Villegas, Jun Saito, Jimei Yang, Yi Zhou, and Michael J. Black. Stochastic scene-aware motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11374–11384, October 2021.

Li Hu, Xin Gao, Peng Zhang, Ke Sun, Bang Zhang, and Liefeng Bo. Animate anyone: Consistent and controllable image-to-video synthesis for character animation. *arXiv preprint arXiv:2311.17117*, 2023.

Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided motion diffusion for controllable human motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2151–2162, October 2023.

Jiaman Li, Jiajun Wu, and C Karen Liu. Object motion guided human motion synthesis. *ACM Trans. Graph.*, 42(6), 2023.

Peizhuo Li, Kfir Aberman, Zihan Zhang, Rana Hanocka, and Olga Sorkine-Hornung. Ganimator: Neural motion synthesis from a single sequence. *ACM Transactions on Graphics (TOG)*, 41(4):138, 2022.

Quanhao Li, Zhen Xing, Rui Wang, Hui Zhang, Qi Dai, and Zuxuan Wu. Magicmotion: Controllable video generation with dense-to-sparse trajectory guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12112–12123, October 2025.

Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13401–13412, October 2021.

Xinran Ling, Chen Zhu, Meiqi Wu, Hangyu Li, Xiaokun Feng, Cundian Yang, Aiming Hao, Jiashu Zhu, Jiahong Wu, and Xiangxiang Chu. Vmbench: A benchmark for perception-aligned video motion generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13087–13098, October 2025.

Qingyuan Liu, Ke Lv, Kun Dong, Jian Xue, Zehai Niu, and Jinbao Wang. Text-to-any-skeleton motion generation without retargeting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12926–12936, October 2025.

Hyelin Nam, Hyojun Go, Byeongjun Park, Byung-Hoon Kim, and Hyungjin Chung. Generating human motion videos using a cascaded text-to-video framework, 2025. URL https://arxiv.org/abs/2510.03909.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph.*, 37(4):143:1–143:14, July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201311. URL http://doi.acm.org/10.1145/3197517.3201311.

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph.*, 40(4), July 2021. doi: 10.1145/3450626.3459670. URL http://doi.acm.org/10.1145/3450626.3459670.

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Trans. Graph.*, 41(4), July 2022.

Sigal Raab, Inbal Leibovitch, Guy Tevet, Moab Arar, Amit Bermano, and Daniel Cohen-Or. Single motion diffusion. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun, editors, *International Conference on Learning Representations*, volume 2024, pages 30549–30569, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/821d5c03ff3bbb574c876bb6beef98b2-Paper-Conference.pdf.

Nils Reimers and Iryna Gurevych. Making monolingual sentence embeddings multilingual using knowledge distillation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.365. URL https://aclanthology.org/2020.emnlp-main.365/.

Davis Rempe, Mathis Petrovich, Ye Yuan, Haotian Zhang, Xue Bin Peng, Yifeng Jiang, Tingwu Wang, Umar Iqbal, David Minor, Michael de Ruyter, Jiefeng Li, Chen Tessler, Edy Lim, Eugene Jeong, Sam Wu, Ehsan Hassani, Michael Huang, Jin-Bey Yu, Chaeyeon Chung, Lina Song, Olivier Dionne, Jan Kautz, Simon Yuen, and Sanja Fidler. Kimodo: Scaling controllable human motion generation. *arXiv*, 2026.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

Roey Ron, Guy Tevet, Haim Sawdayee, and Amit H Bermano. Hoidini: Human-object interaction through diffusion noise optimization. *arXiv preprint arXiv:2506.15625*, 2025.

Haim Sawdayee, Chuan Guo, Guy Tevet, Bing Zhou, Jian Wang, and Amit H. Bermano. Dance like a chicken: Low-rank stylization for human motion diffusion. *Computer Graphics Forum*, March 2026. ISSN 1467-8659. doi: 10.1111/cgf.70365. URL http://dx.doi.org/10.1111/cgf.70365.

Yoni Shafir, Guy Tevet, Roy Kapon, and Amit Haim Bermano. Human motion diffusion as a generative prior. In *The Twelfth International Conference on Learning Representations*, 2024.

Li Siyao, Weijiang Yu, Tianpei Gu, Chunze Lin, Quan Wang, Chen Qian, Chen Change Loy, and Ziwei Liu. Bailando: 3d dance generation by actor-critic gpt with choreographic memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11050–11059, June 2022.

Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-or, and Amit Haim Bermano. Human motion diffusion model. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=SJ1kSyO2jwu.

Guy Tevet, Sigal Raab, Setareh Cohan, Daniele Reda, Zhengyi Luo, Xue Bin Peng, Amit Haim Bermano, and Michiel van de Panne. CLoSD: Closing the loop between simulation and diffusion for multi-task character control. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=pZISppZSTv.

Jonathan Tseng, Rodrigo Castellon, and Karen Liu. Edge: Editable dance generation from music. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 448–458, June 2023.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025a. URL https://arxiv.org/abs/2508.02324.

Ziyi Wu, Aliaksandr Siarohin, Willi Menapace, Ivan Skorokhodov, Yuwei Fang, Varnith Chordia, Igor Gilitschenski, and Sergey Tulyakov. Mind the time: Temporally-controlled multi-event video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23989–24000, June 2025b.

Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. Omnicontrol: Control any joint at any time for human motion generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gd0lAEtWso.

Sirui Xu, Zhengyuan Li, Yu-Xiong Wang, and Liang-Yan Gui. Interdiff: Generating 3d human-object interactions with physics-informed diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14928–14940, October 2023.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

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

Shengming Yin, Chenfei Wu, Jian Liang, Jie Shi, Houqiang Li, Gong Ming, and Nan Duan. Dragnuwa: Fine-grained control in video generation by integrating text, image, and trajectory, 2023. URL https://arxiv.org/abs/2308.08089.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3836–3847, October 2023.

Yuang Zhang, Jiaxi Gu, Li-Wen Wang, Han Wang, Junqi Cheng, Yuefeng Zhu, and Fangyuan Zou. Mimicmotion: High-quality human motion video generation with confidence-aware pose guidance. In *International Conference on Machine Learning*, 2025.

Lei Zhong, Yiming Xie, Varun Jampani, Deqing Sun, and Huaizu Jiang. Smoodi: Stylized motion diffusion model. In *ECCV*, 2024.