POLITECNICO DI TORINO

Master's Degree in Cinema and Media Engineering

Master's Degree Thesis

AI-Driven Workflow for Optimizing the Crowd Generation process in the VFX industry

Supervisors

Prof. Andrea BOTTINO

Prof. Mattia MELONI

Candidate

Gaia LECIS

April 2025

Summary

The integration of artificial intelligence (AI) and machine learning (ML) into the visual effects (VFX) industry has brought about a significant transformation in the filmmaking context. Traditional VFX workflows, particularly in crowd simulation and animation, have historically depended on labor-intensive manual methods. This study explores the potential of AI-driven tools to enhance both time efficiency and realism in digital crowd generation, with a focus on developing and evaluating an AI-powered workflow tailored to meet the needs of a professional VFX company.

The primary objective of this research is to design and implement an AI-powered workflow for generating realistic human digital crowds, addressing key challenges in traditional crowd animation techniques. The study was conducted in collaboration with *EDI - Effetti Digitali Italiani*, with a focus on optimizing the animation and 3D asset creation processes within the company's existing production pipeline. The present research aims to enhance efficiency by reducing the time required for generating and animating large-scale digital crowds while maintaining or improving visual fidelity. It also seeks to improve realism by leveraging AI to create highly detailed, lifelike character animations and motion patterns. Another key goal is to streamline workflow integration by developing a practical, AI-assisted process that integrates seamlessly with industry-standard tools such as *SideFX Houdini*. Finally, the research evaluates AI-based animation tools, assessing the capabilities of AI-driven motion capture, including *Meshcapade*, *MoveAI* and *Rokoko Vision*, in the context of digital crowd animation. Furthermore, to address the necessity to create different variations of the same animation, another AI-powered tool was tested: *GenMM*. The tool seeks to create as many diverse motions from a single animation sequence and promised to synthesize high-quality motion in just a fraction of a second. To validate the effectiveness of the proposed workflow, a case study was conducted on the creation of a full-CG stadium crowd. This scenario was chosen due to the complexity involved in animating large-scale, diverse crowds while ensuring natural movement and variability.

After a comprehensive analysis of AI applications in the VFX industry, the focus shifts to AI-driven motion capture methods, which present a compelling alternative to traditional, resource-intensive animation techniques. To harness the potential of these technologies, a structured pipeline is proposed for integrating AI into the crowd generation process. This workflow consists of several key stages: generating animations using AI Markerless Motion Capture, generating variations of the animations captured, refining the raw motion data through post-processing, and seamlessly transferring the captured motion to the target agents' skeletons to prepare them for crowd simulation. The effectiveness of this workflow is then demonstrated through its application in the stadium crowd case study, showcasing its ability to enhance realism and streamline production.

A comparative analysis was then conducted to quantify the impact of AI-driven animation techniques against traditional methods within the crowd generation context. The evaluation focused first on the main problem introduced by AI-driven animation tools: foot-locking. The comparison has been done between an *Idle* animation and a *Cheering* one, recorded on both the tools evaluated: *Rokoko Vision* and *Meshcapade*. Results show that *Rokoko Vision* gives much better results in terms of stability of the feet joints throughout the movement. For this reason, *Rokoko Vision* was chosen by *EDI* for integrating it into their current animation pipeline. Another quantitative analysis focused on time efficiency. Similarly to the previous analysis, two types of animations were confronted: an *Idle* and a *Cheering* motion. In this analysis the time effort to produce both animations was taken into count, by confronting the timing that would take the animations to be produced with traditional methods and the time necessary to produce the animations with AI-powered methods. The results show that the AI-assisted approach reduced production time by up to 80%.

Afterwards, a qualitative analysis was conducted, and the key factors that played a big role were animation realism and integration success. Findings demonstrate that AI Markerless Motion Capture tools produce highly fluid and natural movements, requiring only minor post-processing to refine results. Among the tested tools, *Rokoko Vision* demonstrated the highest accuracy in human motion replication. Integration success was also a major focus, with the proposed AI-enhanced workflow effectively incorporated into *Houdini*'s procedural crowd setup, demonstrating seamless integration into *EDI*'s existing production pipeline. Additionally, the research assessed cost-effectiveness, concluding that although AI tools require paid subscriptions, their ability to accelerate the animation process translates into significant cost savings. Overall, the study confirmed that AI-driven workflows can enhance efficiency and quality in digital crowd animation, making them a valuable asset in modern VFX production.

The findings of this research underscore the transformative potential of AI in the VFX industry. By integrating AI into traditional crowd generation workflows,

production timelines can be significantly optimized while improving the final visual output. However, AI-driven methods should be viewed as complementary to, rather than replacements for, traditional animation methods, as human intervention remains essential for fine-tuning and creative direction. Future research could explore further refinements in AI-generated motion data, including motion in-betweening and motion interpolation. Additionally, continued advancements in AI-powered motion capture technologies may further bridge the gap between manual animation and automated realism, shaping the next generation of VFX workflows.

Table of Contents

List of Tables

List of Figures

Acronyms

AI

Artificial Intelligence

ML

Machine Learning

DL

Deep Learning

GAN

Generative Adversarial Network

CNN

Convolutional Neural Network

VAE

Variational Autoencoder

VFX

Visual Effects

Mocap

Motion Capture

CG

Computer Graphics

SOP

Surface Operator

DOP	Dynamic Operator
POP	Particle Operator

Chapter 1

Introduction

1.1 Context and reason of the research

In recent years, the rapid evolution of artificial intelligence (AI) and machine learning (ML) technologies has begun to redefine the landscape of various industries, with the visual effects (VFX) sector standing at the forefront of this transformation. Traditionally reliant on meticulous manual processes, the VFX industry has increasingly embraced Artificial Intelligence and Machine Learning tools to address both creative and logistical challenges. These cutting-edge technologies have enabled companies to push the boundaries of visual storytelling, producing stunning hyper-realistic effects while simultaneously streamlining production workflows and increasing overall productivity.

The adoption of AI and ML in VFX workflows is multifaceted. On the creative side, machine learning algorithms are being utilized to generate lifelike animations, automate the creation of complex assets, and enhance simulations for natural phenomena such as fire, water, and smoke. Neural networks, including generative adversarial networks (GANs), have facilitated the creation of hyper-realistic textures and facial reconstructions, enabling unprecedented levels of detail and immersion. For instance, AI-powered tools have played a pivotal role in de-aging actors in blockbuster films and crafting seamless virtual environments in popular series such as *The Mandalorian* (2019). On the productivity front, AI-driven tools have revolutionized traditionally time-consuming processes. Tasks such as rotoscoping, color correction, and compositing, which once required extensive manual effort, can now be executed with remarkable speed and precision using AI algorithms. This automation not only reduces production timelines but also allows artists to focus on more creative aspects of their work. Furthermore, AI-enabled predictive analytics helps studios make data-driven decisions, optimize resource allocation, and anticipate challenges before they arise. [1]

While AI and ML technologies have proven to be valuable assets in accelerating VFX production, they are not a substitute for human expertise. Instead, these tools should be viewed as instruments that enable artists to work more efficiently and creatively. The final results, particularly those tailored to meet specific client needs, still rely heavily on human input and decision-making. VFX companies must therefore embrace these technological advancements, integrating them thoughtfully into their workflows to enhance their capabilities while preserving the irreplaceable value of human creativity and judgment.

1.2 Objectives of the Thesis

The present thesis presents a comprehensive study on the implementation of a AI-powered workflow designed to efficiently gather the necessary data for creating realistic human crowds in the context of a VFX production pipeline.

The primary objective is to propose new solutions and innovative methodologies that aim to enhance both the existing processes and the final visual output. By offering fresh perspectives on crowd generation, this work seeks to address and rectify several weaknesses identified by the company where the research was conducted, *EDI - Effetti Digitali Italiani*, particularly in the areas of animation and 3D asset creation. Throughout the process, careful attention is paid to the constraints and requirements imposed by the company and usually by their clients, ensuring that the proposed workflow is not only effective but also aligned with industry standards. The core of the proposed workflow is centered on utilizing artificial intelligence, machine learning, and deep learning algorithms to accelerate traditional VFX techniques. By seamlessly integrating AI tools with established processes, this approach not only addresses the limitations of current practices but also opens up new avenues for achieving high-quality results with greater efficiency.

A key element of the present research is the application of this workflow to a case study: a full-CG stadium crowd, developed from scratch. This serves as a tangible example of how AI-driven tools can be employed to create complex, realistic crowds in a controlled, practical environment.

One of the main contributions of the present thesis is to showcase the considerable advantages of using AI-driven solutions in combination with traditional methods. The combination of machine learning techniques and conventional VFX tools enables a significant reduction in time required for animation and asset creation, while simultaneously improving the overall quality of the final product. By evaluating both the timing and the quality of the output, the present research highlights how such an AI-enhanced workflow can lead to more efficient production timelines, ultimately resulting in a more polished and refined visual experience. The analysis emphasizes how AI technologies, when properly integrated into existing workflows, can revolutionize the VFX industry, offering improvements in both speed and visual fidelity. The findings underscore the importance of balancing innovation with tradition, automation with human intervention, illustrating how AI can complement and elevate the creative process rather than replace the foundational techniques that have long defined high-quality VFX work.

1.3 Constraints given by the company

The research underpinning this thesis was conducted as part of an internship at *EDI - Effetti Digitali Italiani*, within their *Research and Development Department*. During this time, my primary responsibility was to explore and implement innovative solutions utilizing AI tools to gather the necessary data for the creation of realistic digital human crowds. This work aimed to address the growing need for more efficient methods of crowd simulation in visual effects, ultimately contributing to the development of a cutting-edge workflow that could improve both the creative and technical aspects of the production process. To ensure alignment with the company's established pipeline, I was provided with specific restrictions that guided my research and development process. These constraints were crucial in shaping the final outcome and ensuring the feasibility of integrating the proposed solutions into the company's ongoing and future projects.

One key requirement was the use of *SideFX Houdini* as the primary software to develop the procedural workflow. *Houdini*'s versatility and great capabilities made it an ideal choice for this project, allowing for the creation of scalable and procedural solutions that could be seamlessly integrated into the company's existing production pipeline. Furthermore, for the case study, I was tasked with working exclusively with 3D models of a stadium, a generic man and woman, which had already been produced by the company and that were typically used in projects involving crowds of this kind.

After an initial phase of research and testing, I carefully evaluated a variety of AI animation tools to determine the most suitable options for the project. The decision-making process was guided by several key factors, including the availability of a cost-effective subscription model that would allow *EDI* to continue using the selected tool for future projects. The selected AI tools needed to strike a balance between quality, functionality, ease of integration into the pipeline, and long-term usability, ensuring that the company would be able to derive maximum value from the investment while maintaining flexibility for future advancements.

Chapter 2

State of the art

2.1 AI definition and overview

Artificial Intelligence (AI) is a branch of computer science dedicated to creating machines that simulate human learning, reasoning, problem-solving, decision-making, and creativity. AI systems process vast amounts of data, identifying patterns and relationships that enable them to make predictions. These predictions continue to improve every time they are exposed to new information, with the goal to match or even surpass human cognitive capabilities in automating tasks and solving problems more efficiently.

Machine learning (ML) can be defined as a key subfield of AI, and equips computers with the ability to learn and adapt without explicit programming. By using mathematical models and trial-and-error methods, ML systems are able to analyze large datasets in order to identify patterns and relationships between them. These systems can then make predictions, classifications, or decisions based on new data. Over time, they refine their performance and adapt to new tasks by themselves. ML is generally categorized into four primary training methods:

- Supervised Learning: Relies on labeled datasets containing both inputs and corresponding outputs, enabling machines to learn patterns and make accurate predictions for new data.

- Unsupervised Learning: Processes unlabeled data to uncover hidden patterns, structures, and anomalies, often revealing insights that may elude human analysis.

- Semi-Supervised Learning: Combines small amounts of labeled data with larger quantities of unlabeled data, leveraging both to improve learning accuracy.

- Reinforcement Learning: Teaches machines through trial-and-error, rewarding desired behaviors and penalizing undesired ones, thereby enabling them to optimize tasks over time.

Deep learning (DL) is a specialized subfield of ML, which focuses on processing large volumes of unstructured data through architectures that replicate the human brain. DL systems thanks to multi-layered neural networks, identify complex patterns in text, images, audio, and other data formats, enabling them to generate highly accurate predictions and insights. An important advantage of DL is its ability to autonomously determine which features of data are important. For instance, in image recognition tasks, DL models are able to detect key characteristics, such as wheels or windows, when recognizing cars, without the need of explicit programming. As these systems process more data, they refine their understanding and become gradually more accurate.

Neural networks are computational systems designed to simulate the way the human brain processes information. They are characterized by interconnected layers of nodes called "neurons" organized as shown in Figure 2.1.

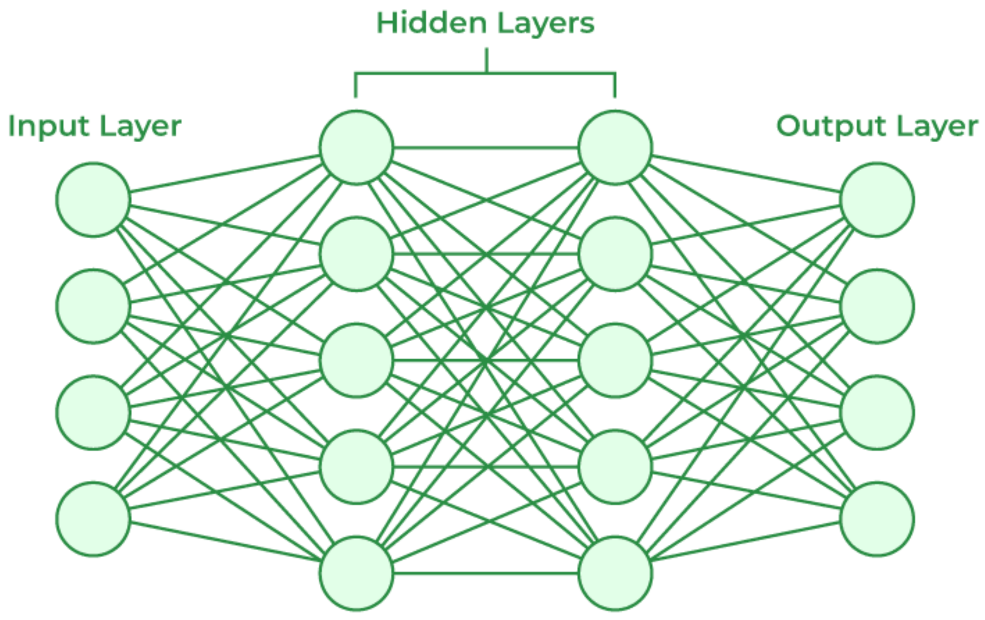


Figure 2.1: Neural Network Architecture

Source: https://tinyurl.com/greeksforgreeks

In the figure the following items can be recognized:

- Input Layer: the layer that receives raw data for processing.
- Hidden Layers: layers responsible to extract features and identify patterns through iterative computation.
- Output Layer: layer that produces the final result or prediction for the next iteration.

Each neuron connects to the next layer with associated weights and thresholds. If a neuron's output exceeds its threshold, it activates, transmitting data forward. This iterative process allows the network to improve its accuracy with continued training.

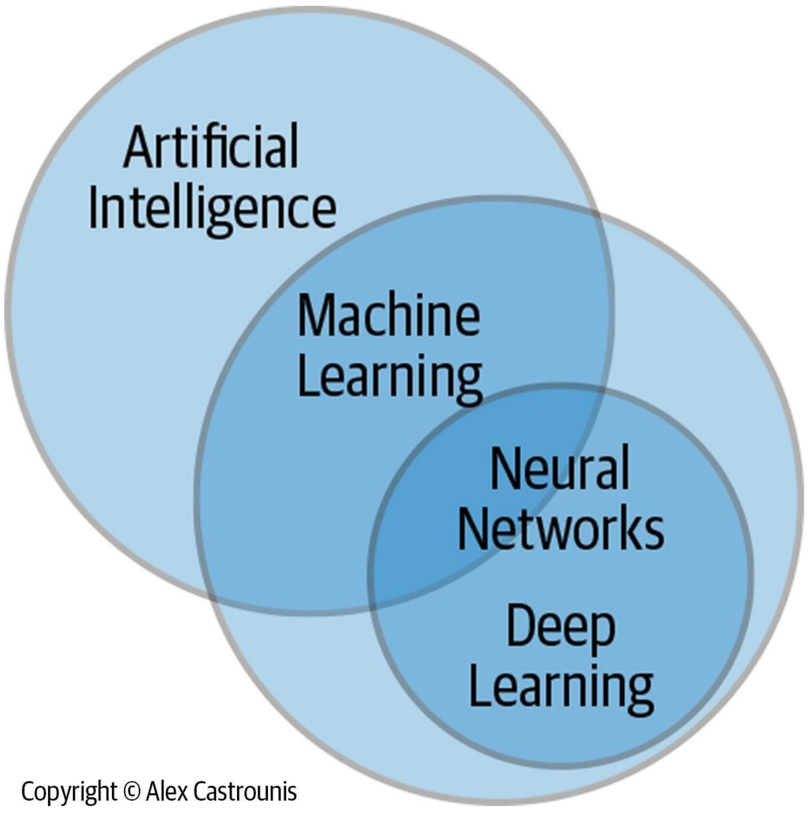


Figure 2.2: AI venn diagram

Source: https://tinyurl.com/ai-ml-venn-diagram

AI models are trained algorithms that perform tasks by learning patterns from data, without explicit programming. For example, translation models analyze thousands of sentence pairs in different languages to learn vocabulary, grammar, and structure for generating accurate translations. Foundation models on the other hand, serve as flexible bases for a wide range of AI applications. These models, due to a training based on large and diverse datasets, can then be easily adapted for specific case scenarios. The most common examples can be recognized in OpenAI's *GPT-4*, Anthropic's *Claude 3*, and Stability AI's *Stable Diffusion*. [2]

Recent SIGGRAPH conferences have highlighted AI's growing role in Visual Effects. Panel discussions showcased advancements in different models such as Convolutional Neural Networks (CNNs), Generative Adversarial Networks (GANs), and Autoencoders. These techniques enable applications like facial simulation, fluid dynamics, image denoising, character animation, and texture creation.

GANs (Figure 2.3) consist of two competing neural networks: the generator and the discriminator. The generator produces synthetic outputs (e.g., images or audio), while the discriminator evaluates their authenticity. This adversarial process improves both networks, enabling GANs to create realistic outputs such as high-resolution images and deepfake media.

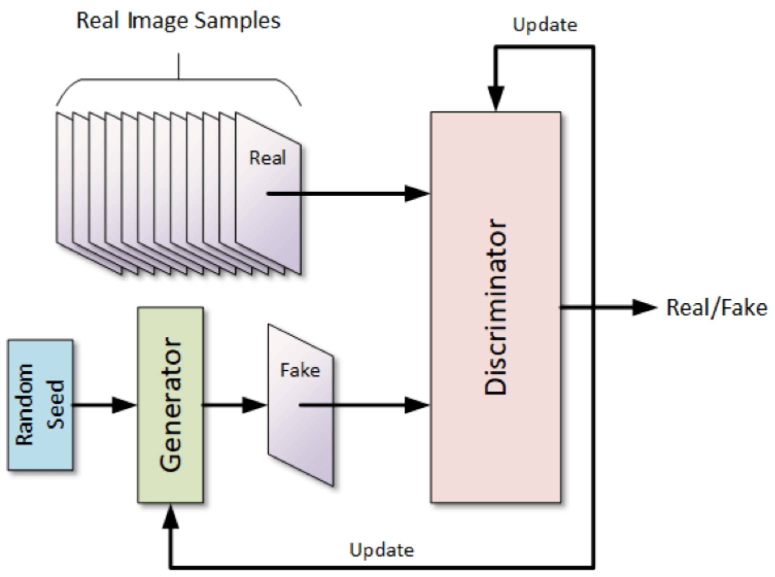


Figure 2.3: GANs architecture

Source: https://tinyurl.com/gans-diagram

CNNs (Figure 2.4) specialize in image recognition and dominate computer vision applications today. They were initially developed in the 1980s and then revived in

2012. CNNs are able to identify spatial hierarchies in visual data, and they excel in tasks such as image segmentation and object detection.

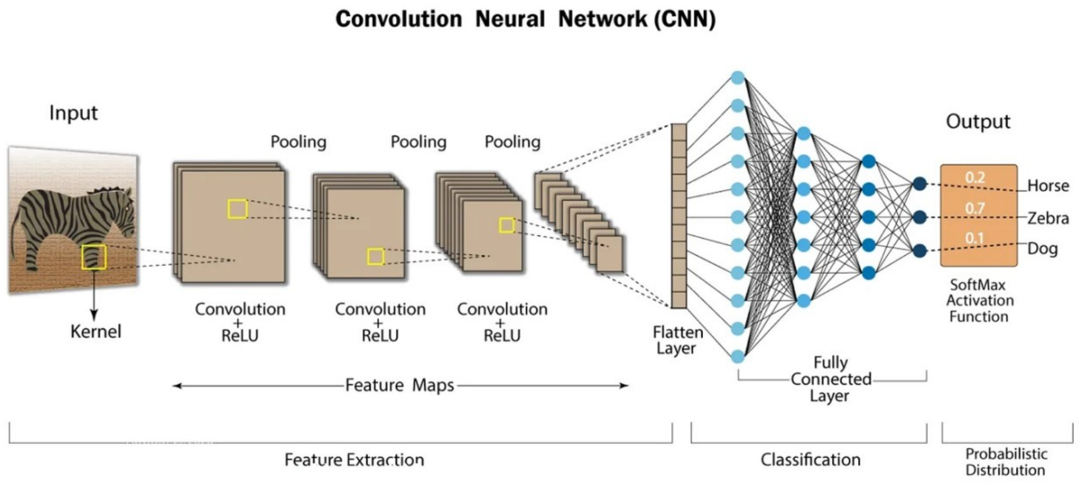


Figure 2.4: CNNs architecture

Source: https://tinyurl.com/cnns-architecture

Autoencoders (Figure 2.5) are unsupervised neural networks designed for data compression and reconstruction. Autoencoders consist of an encoder, a bottleneck, and a decoder. These kind of neural networks learn to represent data efficiently and to rebuild outputs that closely resemble the original inputs. Applications include image denoising and anomaly detection. Among different types of autoencoders, worth nominating is the Variational Autoencoder (VAE), which is one of the most important architectures used in the specific field of generative models.

2.2 AI applications in the VFX industry

AI's impact on film and cinema production extends across various workflows and processes. AI has been used in the VFX industry in many fields such as: rotoscoping, 3D animation, creation of 3D assets, physics simulations and more. As a result, many of the most famous VFX companies, are approaching AI as a new tool for speeding up the workflows drastically and lowering the costs of VFX. Also the creation and development of AI-powered workflows allow the artists to put more time into creativity, and improve the quality of the final result.

The use of ML in particular, can be observed in many of the recent works of the top VFX companies, like the digital creation of the character Thanos' face in *Avengers: Infinity War* (2018) by Digital Domain, that utilized the "Masquerade"

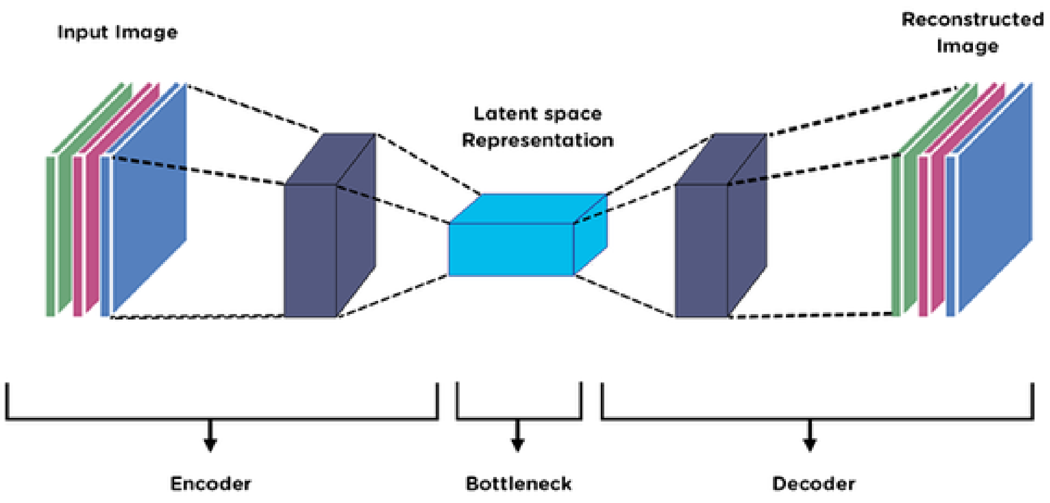


Figure 2.5: Autoencoders architecture

Source: https://tinyurl.com/autoencorders-architecture

facial-capture system for mapping the actor's face to the villain character; or the realistic facial animations made by Wētā for *Avatar 2* (2022). Thanks to a neural network approach the company managed to leverage more of what the actor was doing and hide away some of the complexity from the artists while giving them more control. [3]

Another advantage of AI technologies is that as they continue to evolve, they become more and more accessible to *prosumers* making the VFX process more and more democratized. In fact, while many AI tools are still under development or not yet widely commercialized, many others are accessible to users, usually through freemium offerings. These tools typically include a free version with limited capabilities, requiring a subscription or credits to access the complete set of features. Consequently, cost considerations play a crucial role in determining which tools are adopted for professional VFX productions.

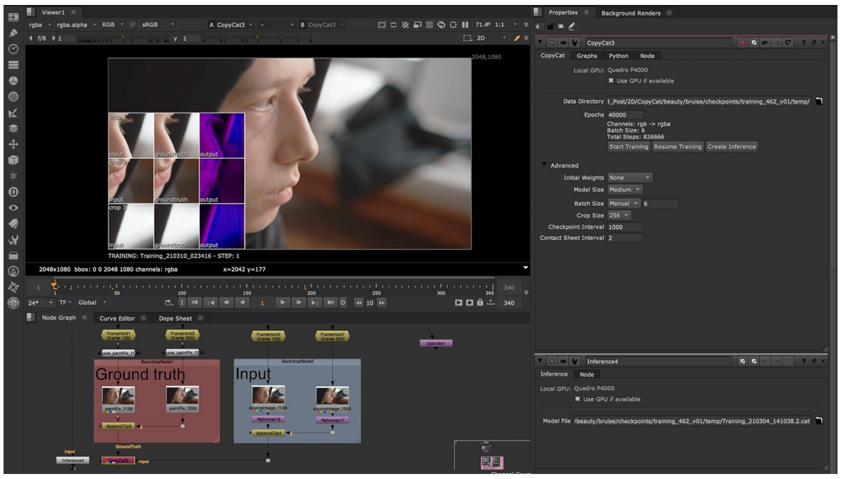
This chapter will provide a general overview of some key areas where the use of Artificial Intelligence and Machine Learning has revolutionized the workflows of both technicians and artists. Particular emphasis will be placed on the 3D domain, specifically focusing on 3D asset creation, the development of 3D digital characters, and their subsequent animation. These aspects are crucial for the generation of digital crowds, which represent the core focus of the present research.

2.2.1 Autonomous rotoscoping

Rotoscoping is a crucial part of VFX. It involves creating realistic animation by tracing objects frame by frame over motion picture footage. Notwithstanding its essential role in cinema post-production, this process is deemed tedious and time-consuming by many VFX artists. AI rotoscoping relies on ML and advanced neural networks in order to automate the process of generating matte images or masks that isolate individual objects in video footage. AI algorithms are capable to analyze large datasets of annotated video footage to recognize and segment elements based on their characteristics (i.e. motion, color, and texture).

The team responsible for the VFX of *The Mandalorian* (2019) used AI-powered rotoscoping throughout the film. The implemented solution could handle complex characters and scenes with multiple objects. As as result, the team reduced the time needed for rotoscoping by 90 percent, so that each one of them could focus on more creative and enjoyable tasks. [4]

An example of how ML has been integrated into compositing software is *CopyCat*. *CopyCat* forms part of the machine learning toolset that has been integrated into *Nuke*. This has been done in order to facilitate artists to perform specific tasks (e.g., garbage matting, beauty repairs, de-blurring) in a smarter and faster way. *CopyCat* makes the training of neural networks possible, in order to reach a specific goal, just by using a small amount of reference frames. By following this approach, it is then possible to replicate the effect "learned" by the neural network and implement it across similar sequences. [5]



TRAINING: Training_210310_023416 - STEP: 1

2048x1080 bbox: 0 0 2048 1080 channels: rgba x=2042 y=177

Local GPU: Quadro P4000
✖ Use GPU if available
Data Directory 3_Post/2D/CopyCat/beauty/bruise/checkpoints/training_462_v01/temp/
Epochs 40000
Channels: rgb -> rgba
Batch Size: 6
Total Steps: 826666

Advanced
Initial Weights None
Model Size Medium
Batch Size Manual 6
Crop Size 256
Checkpoint Interval 1000
Contact Sheet Interval 2

Local GPU: Quadro P4000
✖ Use GPU if available
Model File /beauty/bruise/checkpoints/training_462_v01/temp/Training_210304_141038.2.cat

Figure 2.6: CopyCat Nodes inside Nuke

Source: https://tinyurl.com/copycat-nuke

2.2.2 3D asset generation - Modeling and Texturing

In the world of VFX, 3D assets play a vital role in bringing imaginary worlds and characters to life on the big screen. From creating realistic environments to designing intricate creatures, 3D asset development is an essential part of VFX production. 3D assets are digital objects or characters created using 3D software such as *Maya*, *3DS Max*, *Blender* or similar. These assets can be manipulated and placed to form a 3D environment in order to create complex scenes, realistic characters and intricate objects. For example, in a movie like *Avatar* (2009), the entire world of *Pandora* was created using CG: from the floating mountains to the exotic flora and fauna. Similarly, in movies like *Avengers* (2012) and *Jurassic World* (2015), 3D assets were used to respectively create the superheroes and dinosaurs.

The process of creating a 3D asset involves several stages with its own set of challenges and requirements, starting from the concept art creation, to the modeling process in which polygons are combined in order to build "meshes" to the final texturing process that involves adding colors, patterns and materials to the 3D mesh in order to reach a realistic look. [6]

Among all AI technologies, Generative-AI is considered the most innovative, especially in this field. While it is best known for generating text, its impressive results have made it a key tool in creating 3D assets. When looking specifically at the modeling process, there are two main processes worth exploring:

- Text-to-3D Mesh
- Image/Images-to-3D Mesh

Text-to-3D models generate a 3D output, such as a mesh, from a text input.

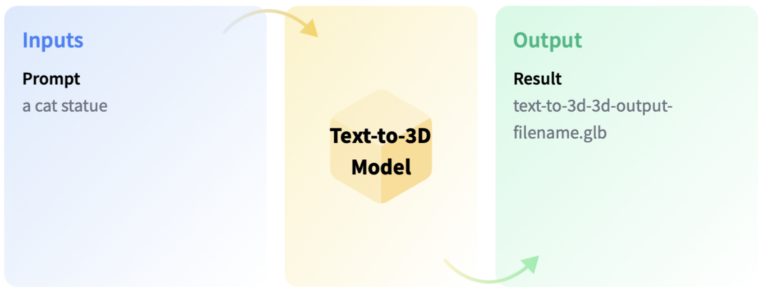


Figure 2.7: Text-to-3D scheme

Source: https://huggingface.co/tasks/text-to-3d

Similarly, Image-to-3D models create a 3D representation based on an image input. The input for these models can vary depending on the architecture, ranging from a single-view image to multiple-view images, with the latter providing a more precise 3D representation.

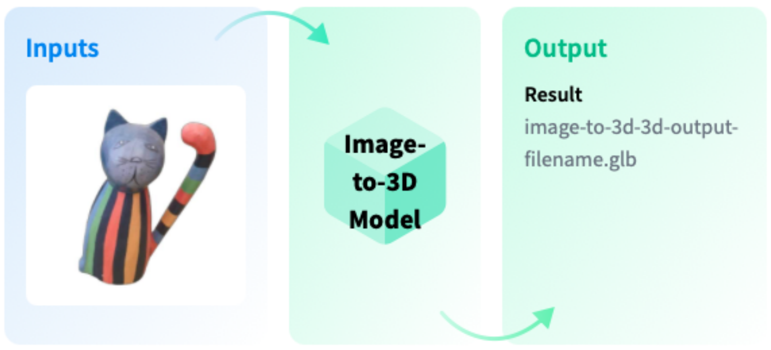


Figure 2.8: Image-to-3D scheme

Source: https://huggingface.co/tasks/image-to-3d

There are many AI models and ML algorithms that make Text-to-3D and Image-to-3D possible, and during this last year many of these have started to become commercialized, like *TripoAI*, *Meshy*, *Genie* and *CSM*. *TripoAI*, *Meshy*, and *CSM* offer web-based services for users, as well as REST APIs for developers to integrate these services into external tools. These are usually credit-based platforms, where each operation consumes a certain quantity of them. Credits are usually included with the initial free subscription but can be also be purchased singularly later on, or can be obtained through premium plans. In contrast, *Genie*, developed by Luma Labs, was launched in an alpha version and is currently free to use. These AI models enable the generation of 3D assets from text or images, producing both geometry and textures. However, the quality and fidelity of the results vary between models. Additionally, some of these models offer extra features, such as generating PBR textures for the assets or rigging and animating the 3D models. Worth naming also *Trellis 3D* by Microsoft and *Rodin* by Hyper3d. [7] *Trellis3D* (Figure 2.9) is designed to generate high-quality assets from image prompts. It utilizes a powerful Structured LATent (SLAT) representation to decode into various output formats, including Radiance Fields, 3D Gaussians, and traditional meshes. [8]

Rodin (Figure 2.10), instead, is a model capable to generate high-quality quad-meshes (meshes composed of quadrilaterals rather than triangles) through proprietary diffusion models. These quad meshes are more topologically consistent, allowing for cleaner texture mappings and edge flow. Rodin excels at creating organic character shapes, which can be easily edited and animated. [9] Rodin as well, is a credit based platform. Rodin offers different ways to generate a 3D mesh: it's possible to generate it via prompt, via single or multiple images but

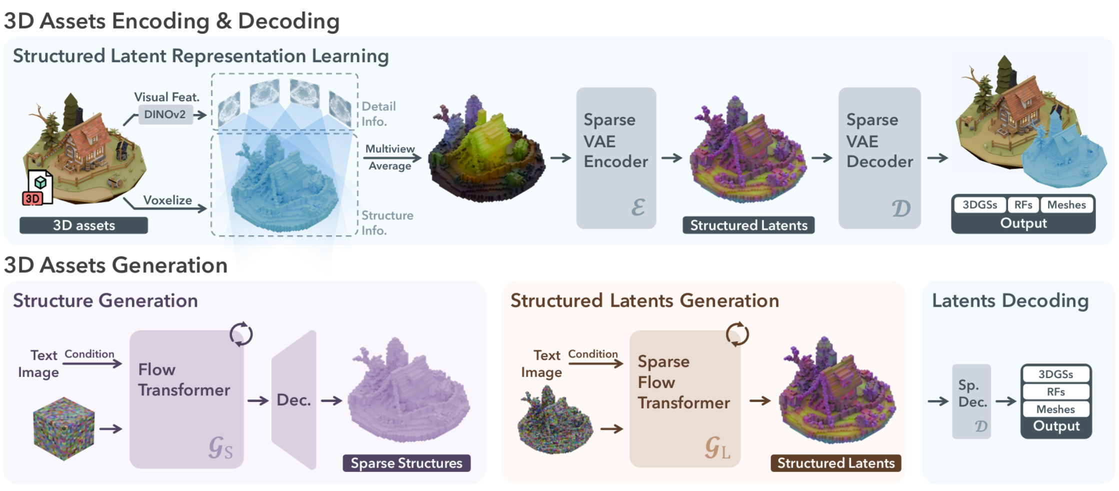


Figure 2.9: Trellis methodology scheme

Source: https://tinyurl.com/trellis-method

also by using bounding boxes, point clouds or voxels ControlNets. Hyperhuman, the company that developed *Rodin*, recently developed also *ChatAvatar* that is currently an Open Beta. This tool allows a progressive generation of animatable 3D faces under text or image guidance.

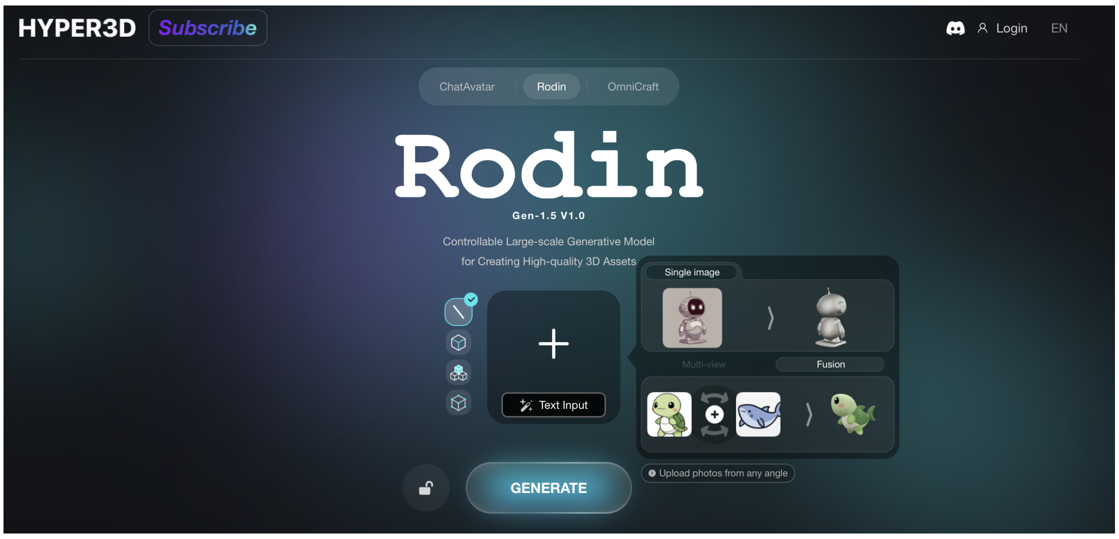


Figure 2.10: Rodin interface

Source: https://hyper3d.ai

Among these different tools, there are also many open-source AI models that can be run on local machines. Some of the most notable are *TripoSR*, *CRM*, *InstantMesh*, and *SF3D*. These models are particularly valuable because they allow users to experiment, refine, enhance, and integrate them into personal tools without relying on external services. In the following paragraphs all these models will be briefly described.

TripoSR The design of *TripoSR* follows the Large Reconstruction Model (LRM) network architecture, created by Adobe Research (Figure 2.11). It predicts, in just 5 seconds, the 3D model of an object from a single input image. *TripoSR* enhanced its data handling, architectural design, and training. It comprises three primary elements:

- the Image encoder,
- the Image-to-triplane decoder,
- the Triplane-based neural radiance field

The Image encoder uses a vision-transformer-based architecture to convert an RGB image into a series of latent vectors. They encapsulate the global and local properties of the image. The Image-to-triplane decoder converts the latent vectors into a triplane representation. The triplane is a new way to represent the 3D space using three 2D feature planes. In the end the Triplane-based neural radiance field (NeRF), uses the triplane representation to predict color and density at various points in the 3D space. These predictions are then used for volumetric rendering, which ultimately produces the 3D mesh based on the initial 2D image input. A triplane representation is an image projected onto a 3D space. This projection is typically conditioned on camera parameters, which describe the position and orientation of the camera relative to the scene, as well as the intrinsic properties of the camera, such as its focal length and lens distortion. In particular, the model is not given the camera parameters explicitly but instead, they are “guessed” during the training and inference phase. The model is trained to learn the relationship between the image and the triplane projection, and to infer the camera parameters based on that relationship. This approach allows the model to be more flexible and capable of handling a wide range of real-world scenarios without the need for precise camera information. *TripoSR* main improvements over the LRM include fine-tuning the number of channels for better information processing, the use of mask supervision to provide extra training guidance, and an upgraded crop rendering approach. This latter improvement enables the model to interpret more effectively and to learn from incomplete views of objects, a common occurrence in real-world images where the full object isn’t always captured. [10]

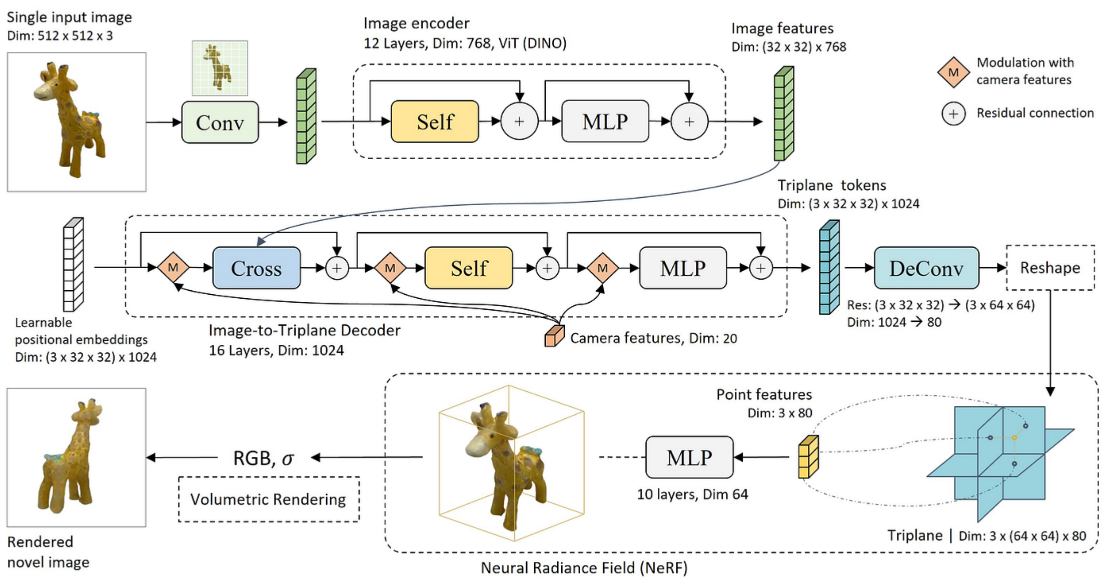


Figure 2.11: LRM architecture

Source: https://tinyurl.com/LRM-architecture

InstantMesh *InstantMesh* consists of two main components: a multi-view diffusion model and a sparse-view large reconstruction model. Given an input image, the multi-view diffusion model generates 3D consistent multi-view images, which are then sent to the sparse-view reconstruction model to reconstruct a high-quality 3D mesh. The multi-view diffusion model in *InstantMesh* is based on Zero123++, chosen for its reliable multi-view consistency and tailored viewpoint distribution that covers both the upper and lower parts of a 3D object. To address the issue of inconsistent gray backgrounds in generated images, the authors fine-tuned Zero123++ to synthesize consistent white-background images, ensuring the stability of the subsequent sparse-view reconstruction procedure. The sparse-view large reconstruction model in *InstantMesh* is built upon the LRM architecture, similarly to *TripoSR*, and is modified to accept multi-view images as input. [11] The model is trained using a two-stage strategy (Figure 2.12):

- In the first stage, the model is trained on a Neural Radiance Field (NeRF) representation. To speed up the training process, the model is initialized with pre-trained weights from OpenLRM, an open-source implementation of the Large Reconstruction Model (LRM). The model's image encoder is also modified by adding special layers (in particular, AdaLN camera pose modulation layers) to the Vision Transformer (ViT) architecture, allowing the model to understand the camera pose information associated with each input image. Following that, a transformer-based triplane decoder produces 64×64 triplanes, which are used to represent the implicit 3D geometry and appearance of the object. The model is then trained using a loss function that considers both the differences between the generated and target images (image loss) and the differences between their corresponding masks (mask loss).

- In the second stage, the model is trained on a mesh representation. A differentiable module called FlexiCubes is added to the reconstruction model, enabling it to extract mesh surfaces from the implicit fields represented by triplanes. The density MLP (multilayer perceptron), which was originally used to predict the density of points in the NeRF representation, is now repurposed to predict the signed distance function (SDF) values for the mesh. Two additional MLPs are introduced to predict the deformation and weights required by FlexiCubes. The loss function in this stage includes the losses from Stage 1 (image and mask loss) and additional losses that consider the differences between the predicted and target depth maps (depth loss), normal maps (normal loss), and regularization terms specific to FlexiCubes. By training the model in these two stages, *InstantMesh* can effectively learn to reconstruct high-quality 3D meshes from sparse-view images while leveraging the benefits of both NeRF and mesh representations.

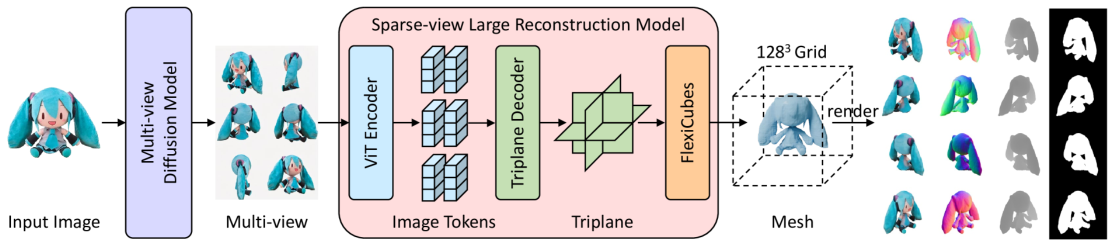


Figure 2.12: InstantMesh framework

Source: https://tinyurl.com/InstantMesh

3F3D *3F3D*, unlike most existing approaches, is explicitly trained for mesh generation, incorporating a fast UV unwrapping technique that enables fast texture generation rather than relying on vertex colors. The method also learns to predict material parameters and normal maps to enhance the visual quality of the reconstructed 3D meshes. Furthermore, *SF3D* integrates a delighting step to effectively remove low-frequency illumination effects, ensuring that the reconstructed meshes can be easily used in different illumination conditions. The model (Figure 2.13) is characterized by five interconnected modules: Enhanced transformer, Material

Estimation Network (Material Net), Illumination Modeling Network (Light Net), Mesh Extraction and Refinement and finally Fast UV Unwrapping and Export Pipeline. [12]

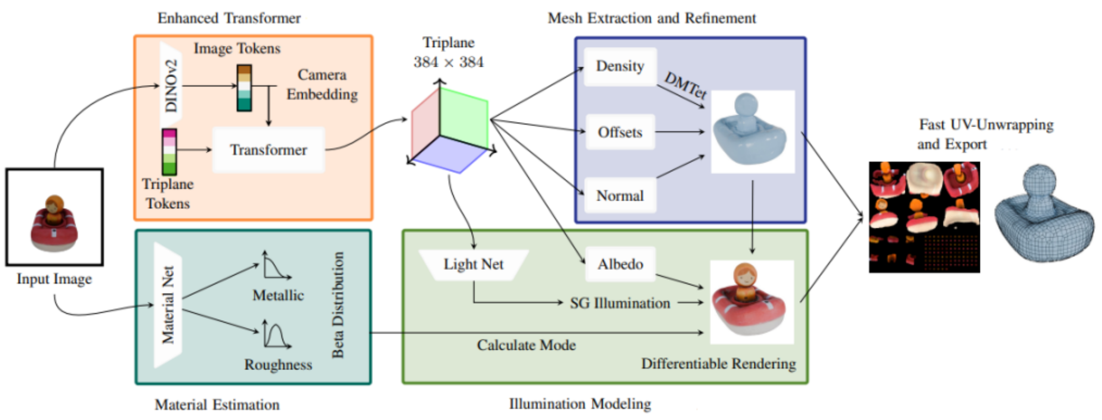


Figure 2.13: 3F3D framework

Source: 3F3D Paper

Additionally, Meta recently introduced a new 3D generation model called *Assets3DGen*, which shows great promise with its high-quality results. This model supports text-to-3D, image-to-3D, and even PBR materials creation. Testing the tool was not possible since no code was ever released.

In the end, worth to underline, is *DonatelloAI*, an experimental tool for 3D asset generation based on AI models, developed by the Plain Concept Research Team. This tool has been developed using *TripoAI* services and Evergine. *DonatelloAI* incorporates many functionalities such as text/image to 3D, auto-rigging and animation of generated models, stylization, optimization of the models and multiple formats exporting settings. *DonatelloAI* is currently available on its own repository [13] allowing users to test it out on their local machines.

2.2.3 3D Animation

The modern film industry heavily relies on 3D animation, which has deeply changed the way stories are presented and how audiences engage with films. Animation has undergone a remarkable transformation, progressing from the simplicity of hand-drawn cartoons to the breathtaking visual complexity of today's 3D animated hits. As a fact, 3D animation plays a crucial role in the VFX pipeline. 3D animation can be defined as the process that brings synthetic objects or characters to life from a car speeding on the road, to a cat that plays with its favorite ball of wall.

After a digital object is modeled and textured as explained in the previous paragraph, the object is then ready to be “rigged”.

The process of "3D rigging" involves creating a skeleton for the 3D model, made of bones and joints interconnected to each other, following a specific hierarchy (Figure 2.14).

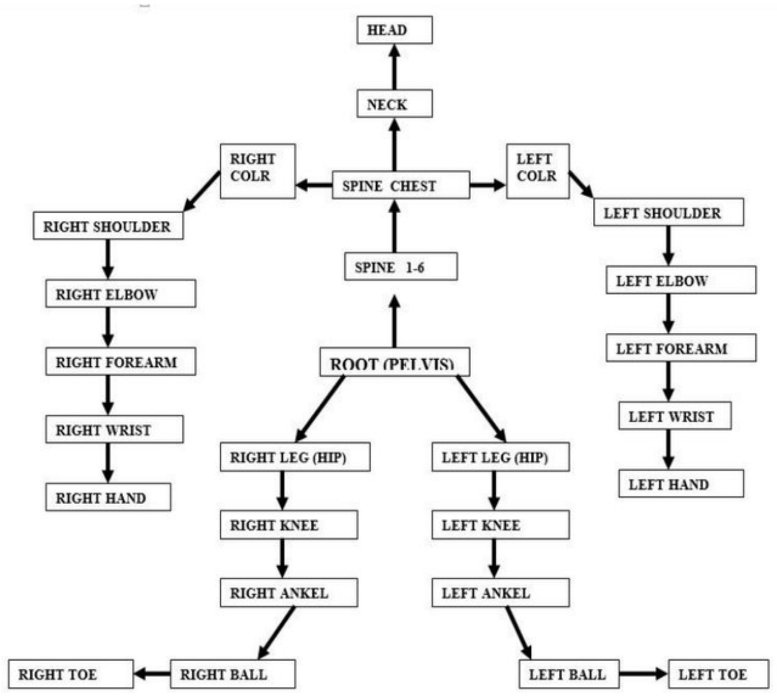


Figure 2.14: Hierarchical Bone Structure for Biped Character

Source: https://tinyurl.com/skel-structure-biped

After the skeleton is created, it is necessary to set-up the control system that is going to be utilized to manipulate the skeleton (e.g., bend a leg, rise one arm) and consequentially animate the 3D model. This process typically involves adding elements such as selectable nulls, that will be used to move the joints and the bones in order to replicate specific poses or movements. After that, the model needs to be “weighted”; in fact in order for the rig to properly control the movement and deformation of the 3D mesh, it is necessary to specify how much each joint influences the deformation of a specific part of the 3D model’s geometry. After all these operations, the so-called “rig” is successfully created (Figure 2.15), and the 3D model is ready to be animated.

3D rigging not only enables control over a model’s movement but also allows for the inclusion of additional elements to the animation, such as facial expressions

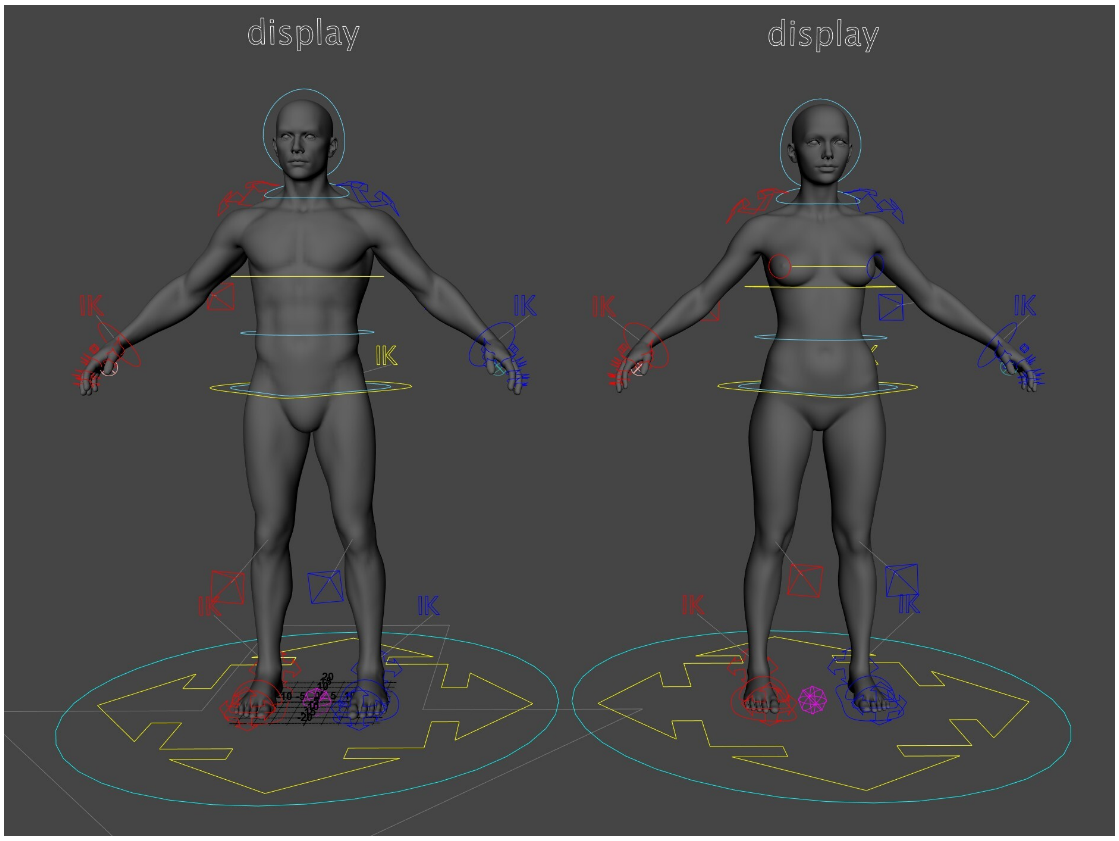


Figure 2.15: Humans' rig examples

Source: https://tinyurl.com/human-rig

or clothing. This is achieved by integrating extra controls into the rig, which can be used to adjust specific parts of the 3D character. For instance, an animator might add controls for the eyes, mouth, and eyebrows to craft a diverse range of facial expressions. A major benefit of 3D rigging is its ability to facilitate the creation of intricate and lifelike animations efficiently. A well-constructed rig allows animators to generate various movements and poses with minimal effort, significantly reducing time and labor. This efficiency is especially advantageous when animating multiple characters, as it enables quick creation and manipulation of multiple rigs simultaneously. [14]

Among the techniques that characterize the animation process, the *keyframe animation* is the one that stands out among others because it offers a high level of control over the overall movement of the digital objects and characters. *Keyframe animation* is a technique that revolves around designing specific "key" poses or frames at pivotal moments in a sequence to outline the start and end of a movement. The intervening frames, which create the illusion of motion, are then automatically filled in by the animation software through a process called "interpolation". In practice, animators begin by meticulously planning a character's motion, pinpointing the most critical poses in the sequence, commonly known as "extremes" or "key poses". These frames serve as the foundation for the raw movement, highlighting essential moments such as the apex of a jump or the initiation of a run. After establishing the keyframes, the animator enhances the motion by introducing "breakdowns" and "in-betweens". "Breakdowns" provide a detailed guide for the transitions between keyframes, shaping the path, speed, and subtle details of the movement. For instance, when animating a character swinging a sword, the "breakdown" might depict the weapon at a specific angle to emphasize the arc and convey its weight and momentum. The software then creates the intermediate frames between these poses to produce a seamless transition. However, this automated process often requires fine-tuning. Animators adjust elements like timing, spacing, and interpolation methods, such as linear, spline, or stepped interpolation, to achieve the intended effect. Spline interpolation, for example, delivers smooth and natural motion curves, while stepped interpolation is useful for abrupt or mechanical movements. *Keyframe animation* stands out for its remarkable precision and the unmatched creative freedom it grants animators. Unlike other techniques (e.g., Motion Capture), this one empowers creators to push boundaries, allowing for the exaggeration or stylization of actions that would be impossible or impractical in reality. This aspect plays a crucial role in character animation, where the personality and the emotions of the character is often defined by the way it moves. A great example of how this technique was successfully used could be Elastigirl from Pixar's *The Incredibles* (2004). This character, in order to express her personality and super powers, needed to be characterized by stretchy and dynamic movements that only the high precision of keyframe animation could give. In this way, animators managed to make her elastic powers feel both plausible and reflective of her character's vibrant personality. Similarly, *The Little Mermaid* (1989) by Disney is a great representation of the power of keyframe animation. Ariel's underwater elegance and flowing hair were animated through carefully designed key poses. These keyframes formed the basis for her smooth, dreamlike movements, beautifully embodying the enchanting underwater setting. [15]

Another technique heavily used by animators in the VFX industry is Motion Capture (MoCap), which will play a crucial role in the present research. Motion Capture, is a technique that allows to transfer real captured motion onto digital characters. Clearly, this process, shows its value and strengths in a context of animating digital characters that are supposed to act like humans. Motion Capture can be used both to replicate full-body motion, hand gestures or intricate face expressions which can be difficult to archive with tradition keyframe animation. James Cameron's

Avatar (2009) serves as a groundbreaking example of the reshaping of cinema using Motion Capture technology. The film introduced revolutionary techniques to capture not just the actors' movements but also the intricate details of their facial expressions and eye dynamics. This allowed for an extraordinary transfer of human performances to the Na'vi characters, seamlessly blending live-action with CGI. Similarly, *The Lord of the Rings* (2001) trilogy showcased the transformative potential of MoCap through Andy Serkis's portrayal of Gollum. By recording every subtle expression and gesture, filmmakers created a character that felt intensely real, setting a new standard for digital performances. Another compelling use of MoCap can be seen in *The Planet of the Apes* (2001) series, particularly in the depiction of Caesar and the other apes. Advanced Motion Capture techniques captured the emotional nuances of the actors' performances, bringing unprecedented authenticity and depth to the digital characters. This innovation bridged the gap between human and digital artistry, allowing audiences to form deep emotional connections with these lifelike creations. [15]

It is important to recognize that the Motion Capture process involves several essential sub-steps that must be completed. In fact, the entire Motion Capture process can be viewed as a distinct pipeline (Figure 2.16), regardless of the specific type of Motion Capture being employed.

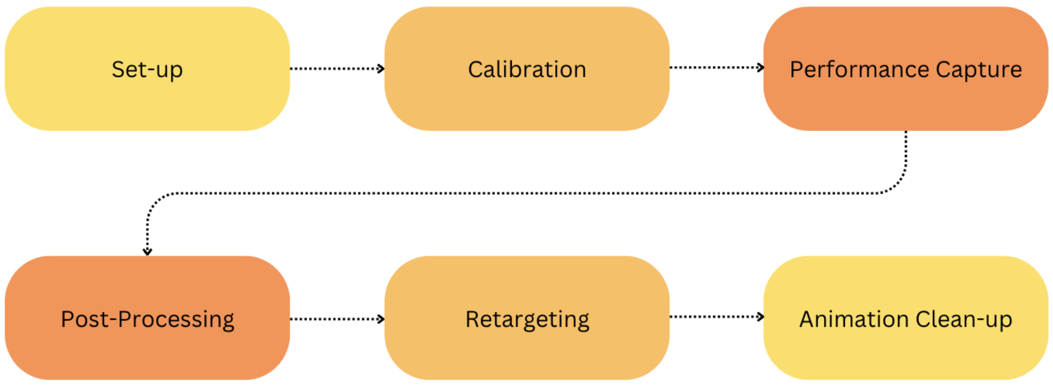


Figure 2.16: Motion Capture Pipeline

1. Set-up: The pipeline begins with ensuring the proper functioning of the selected MoCap tracking system. This often involves configuring the orientation of the cameras within the capture space. To achieve this, specific markers are typically placed at precise locations to assist with calibration and alignment.

2. Calibration: This phase is essential for establishing the initial position of the

captured subject within the space. It typically involves the actor standing in a specific "bind pose" (such as an A-pose or T-pose) for a brief period of time. This allows the software to accurately identify the precise position of the subject's joints, ensuring their movements are tracked correctly throughout the capture process.

3. Performance Capture: after the calibration is successfully completed the actor can start acting and moving. This is the phase where the actual movement is captured.
4. Post-Processing: The post-processing stage focuses on analyzing the performance capture data and refining the numerical information. During this phase, issues like foot-sliding or unnatural movements are addressed.
5. Retargeting: Phase in which the recorded movements are successfully mapped onto a digital skeleton model. This process is usually done automatically by the MoCap software, but in VFX companies is common practice to retarget the animations manually onto specific third characters.
6. Animation Clean-Up: At this stage, the animated skeleton is exported to external animation software, where further refinements can be made to enhance the animation. This phase typically focuses on smoothing, fine-tuning, and adding greater precision, resulting in a more realistic and polished final animation.

As mentioned above, there are different types of Motion Capture, which can be classified into different subtypes according to their way of tracking movements. The classification is explicitly shown in the figure below (Figure 2.17).

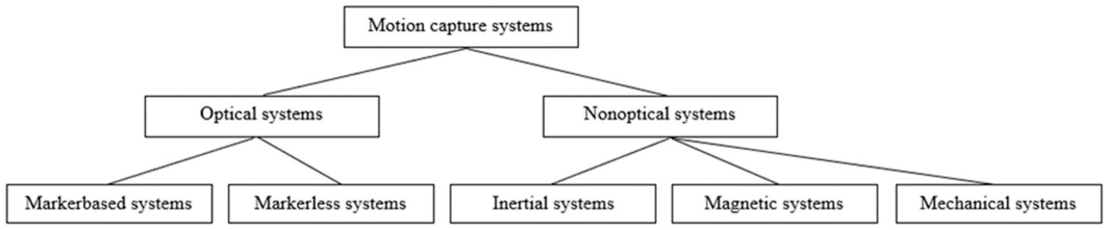


Figure 2.17: Motion Capture Systems classification

Source: https://tinyurl.com/mocap-systems

MoCap systems can be distinguished by optical and non-optical systems. The firsts, as the name suggests, utilize cameras for tracking motion and include marker-based systems and markerless ones. On the other hand, there are non-optical systems which include inertial, magnetic and mechanical systems to track the motion.

Mechanical MoCap Systems Mechanical MoCap Systems operate through interconnected mechanical structures with joints that house sensors, enabling the calculation of the position and orientation of the end-effector: the final point in the chain. These devices represent the earliest form of tracking technology. Their primary strengths lie in their exceptional and consistent accuracy throughout the workspace, as well as their low latency. Furthermore, they are resistant to drift and unaffected by environmental interference, including optical, magnetic, or acoustic disruptions. However, the mechanical nature of these trackers presents challenges. They must be securely anchored to withstand reaction forces, and their operational range is limited to the physical reach of the structure. Additionally, their bulky and heavy design often compromises ergonomics.

Magnetic MoCap Systems Magnetic systems rely on an array of sensors, referred to as receivers, usually ranging from 12 to 20 in number, which are attached to the actor. These receivers measure their spatial relationship relative to a magnetic field generated by a stationary emitter. The magnetic field is produced by applying an electric current to a coil, and the receivers detect the field. Their measurements vary depending on their position and orientation relative to the emitter. To accurately determine the position, the system sequentially generates three distinct magnetic fields aligned along the x, y, and z axes. This sequential approach prevents interference. Each receiver contains three orthogonal metal coils, which measure the magnetic field and convert it into voltage readings based on their spatial orientation and position. By the end of the cycle, nine voltage readings are obtained, corresponding to nine equations involving three positional and three orientational variables. However, one major limitation of this technology is its susceptibility to magnetic interference. AC transmitters can induce eddy currents in surrounding metallic objects, creating secondary magnetic fields that disrupt measurements. The key advantage of these systems is their ability to calculate both translation and rotation components in real time. Typically providing six degrees of freedom (6 DOF), these sensors deliver highly accurate results, especially when integrated with optical systems that usually require a greater number of trackers.

Intertial MoCap Systems Inertial systems are characterized by micro electromechanical systems (MEMS) such as accelerometers, gyroscopes, and magnetometers, which offer nine degrees of freedom (9 DOF): 3 for spatial positioning, 3 for rotational orientation, and 3 for magnetic field measurement. Angular velocity is calculated through three orthogonal gyroscopes, with temporal integration used to determine orientation angles. Spatial accelerations, measured by accelerometers aligned with the gyroscopes, allow for the calculation of position through double integration, once gravitational forces are accounted for. These systems are commonly integrated into wearable devices that can be easily applied to various body parts, offering advantages such as immunity to external signal interference, compact design, ease of use, and no need for a controlled environment or elaborate setup. However, the use of integration and double integration introduces challenges. Noise in angular velocity and acceleration data accumulates, rather than dissipates, over time. Double integration amplifies this issue, resulting in significant drift errors that increase exponentially. For example, positional drift can reach up to 40 mm within two seconds. To mitigate this, the system's output can be periodically reset using complementary trackers. Regular calibration is also necessary to ensure consistent accuracy, as performance deteriorates over time due to drifting. One prominent system that maximizes the potential of this technology is *Xsens* by Movella known for its precision, freedom of movement, and resistance to interference. Additionally, the inclusion of integrated buffers in the sensors ensures data capture even during brief signal losses.

Optical MoCap Systems The optical MoCap systems that use markers to track motion are called “marker-based optical systems”. These systems operate using a set of cameras that detect markers motion placed on the suit worn by the actor, strategically positioned near the joints. This method is extremely precise and allows to effectively capture fast and complex movements. Motion tracking can be classified into 2 subcategories: "inside looking in", which uses passive or active markers, depending on camera resolution; and "inside looking out", which works thanks to RGB sensors (also known as depth sensors). However, this kind of MoCap, brings a large set of challenges due to the hard camera calibration process and the characteristics of the capture environment that has to be well illuminated and not reflective. Also, these kind of systems are extremely expensive. A few examples can be found in *Optitrack* system and *Vicon* systems.

All the challenges just described can be overcome through the development of AI technologies that allow the realization of optical markerless systems. These systems eliminate the need for suits and markers, allowing for more natural performances at a more accessible price. Markerless optical systems in fact can reconstruct the motion performed by an actor recorded with RGB cameras that are placed strategically around them. These cameras capture the motion and through an analysis done by specialized software that use advanced image processing algorithms, computer vision and ML, it is possible to detect and track distinctive features like shape, color, texture and movement. Markerless MoCap systems bring with them few limitations though, such as such a poor reconstruction of the movement due different reasons such as the poor resolution of the cameras used to record the movement, occlusions, or too fast and complicated actions. For this reasons this technology can be used only in specific case scenarios. [16]

Markerless MoCap systems are easily accessible because they can be used basically everywhere and they don't require an expensive equipment, often a phone with a good quality camera and a tripod is enough. Recently this technology has been integrated in proprietary software that have been commercialized and that can be used and tested by companies and common users through the use of freemium plans.

The ones the most stand out in the current market are:

- *Rokoko Vision* by Rokoko
- *MoveAI*
- *Meshcapade*
- *Animate3D* by DeepMotion

These services are usable through online interfaces and process the MoCap data on their proprietary servers. The motion reconstruction models are private and not possible to be analyzed in deep, however in the section below their basic functioning can be observed.

Rokoko Vision

Rokoko Vision offers a platform to perform markerless Motion Capture, powered by AI and ML algorithms. *Rokoko Vision* allows users to perform Motion Capture through a simple 3 steps process:

- Record the movements with a personal webcam, an external camera (phone cameras, DSRL and APSC cameras are supported) or two cameras (e.g., webcam and iPhone, 2 DSLR cameras, 2 APSC cameras, 2 iPhones) for better accuracy; or upload a video file of pre-recorded footage, as long as there's a clear view on the subject's body motion;
- View the MoCap data in *Rokoko Studio*, a free software that allows to view and clean-up the MoCap data, by using AI-powered filters, and finally upload a 3D rigged character to retarget it on the mocap animation;
- Export the file with the chosen extension (e.g., .FBX, .BVH). The software allows the users to choose a skeleton type (e.g., Mixamo, HIK etc) and use the file directly in the 3D tool of choice (e.g., *Maya*, *Blender*, *Houdini*, *Unreal* etc).

The free plan includes both the single-camera set-up and the dual-camera set-up. Also, it allows the upload of a pre-recorded footage, with a maximum length of 15 seconds, and the only clean-up filter available is the foot-locking one. Advanced AI-filters, and the chance to capture animations longer than 15 seconds, are available only with the Pro subscription.

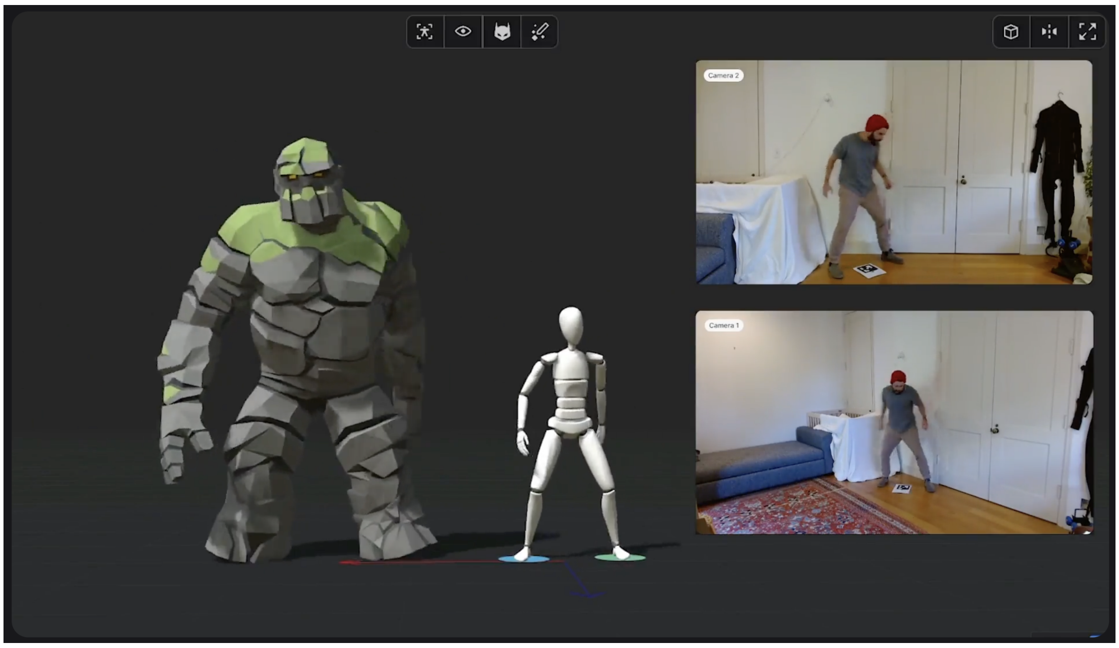


Figure 2.18: Rokoko Vision functioning preview - dual camera set-up

Source: https://www.rokoko.com/products/vision

MoveAI

MoveAI uses AI technologies to generate 3D motion data from a 2D video footage. AI is used, similarly to other software, to identify points on the human body and extract its motion from the video footage captured on the digital camera device. *MoveAI* can reach this result through algorithms that implement computer vision, physics, biomechanics and AI to identify the points of movement on the body. *MoveAI* offers few different ways to perform their markerless MoCap:

- *MoveOne*: an app specifically designed for iOS devices, that allows to capture 3D movement through the use of a single camera footage. *MoveOne* allows users to directly record the actor's movement by using the iPhone (or iPad) camera (Figure 2.19). After the action is recorded, *MoveOne* will process the data in their owned servers and then allows users to download the MoCap data in their favorite file format, onto a *MoveOne* skeleton (e.g., .FBX).

- *MovePro*: *MovePro* is a premium markerless Motion Capture solution tailored for professionals and enterprise teams in industries like gaming, film, sports, and biomechanics. Built for scalability and precision, *MovePro* supports multi-camera, multi-person setups, delivering unparalleled performance with advanced AI algorithms and robust integrations. *MovePro* supports up to 8 cameras to record single or multi-person actions. The whole capturing process can be divided into 2 main phases: acquisition, in which the calibrated cameras work together to capture the movement of the performer, and data processing where the 2D videos captured by the cameras are sent to *MoveAI* servers in order to be processed by the AI algorithms so they can be transformed into 3D animation data, that will be then available to download. This kind of subscription is available just for companies and the pricing has to be discussed directly between them.

- *MoveLive*: *MoveLive* is a real-time markerless Motion Capture solution built for delivering immersive experiences at live events, retail activations, and experiential campaigns. Designed for multi-camera, multi-person setups, *MoveLive* seamlessly streams motion data into real-time engines like *Unreal Engine*, enabling dynamic visualizations that captivate audiences in the moment. *MoveLive* supports up to 4 people captured and up to 10m x 10m volume. Captures at 110fps and supports 4 to 8 cameras, and performs with a latency of 100ms.

Meshcapade

Meshcapade is a platform that by combining computer vision, graphics and machine learning offers users the chance to create realistic 3D human characters from any source data, including photos, videos, body measurements, 3D scans, Motion Capture and text. These avatars come in combination with a rig, so they can be animated, or can come already animated, though the services offered by the platform such as the text to animation and their Motion Capture markerless system. *Meshcapade* can be used though their Automated Processing Platform, “digidoppel”, a one-stop-shop where users can input their complex data from many different sources (photos, 3D scans etc) and instantly receive an accurate 3D avatar. This avatar can then be imported into a variety of commercial tools to be further worked on. *Meshcapade* leverages 3D parametric models of the human body and its movement to gain insights into human behavior and interactions. These models enable the analysis of various ways individuals engage with their

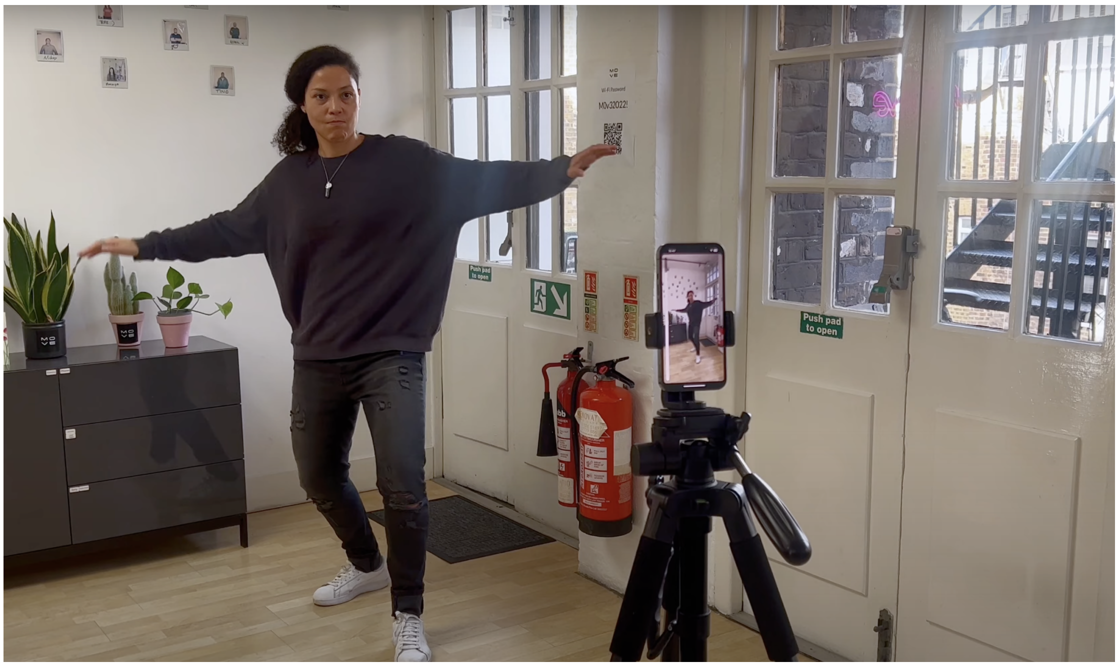
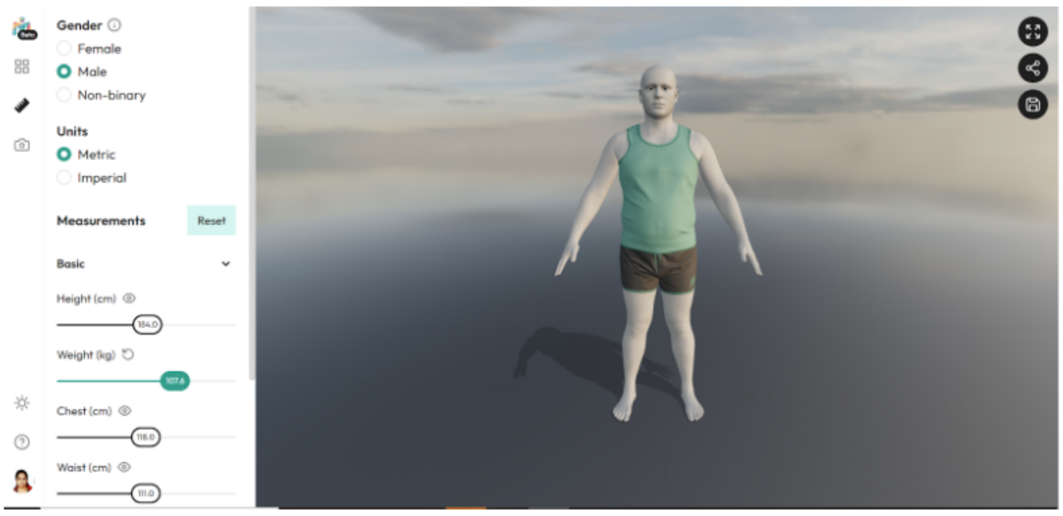


Figure 2.19: MoveOne usage

Source: https://tinyurl.com/move-one

environment, including human-object interactions, human-to-human contact, the effects of external forces like compression, and factors such as posture, gait, and emotional expressions. By employing these models, *Meshcapade* can elucidate the relationship between the 3D human form and the surrounding 3D world. The company develops realistic, parametrizable 3D body models, where shape and pose can be adjusted independently. This is achieved by disentangling body shape variations from pose changes, a process supported by analyzing thousands of high-resolution 3D scans.

The SMPL model, a prominent creation, has become a benchmark for studying human body shape and pose. Its adaptability makes it particularly valuable for research, as its parameters can be optimized to fit data from diverse sources such as 3D scans, RGB-D images, Motion Capture systems, and videos. SMPL can be defined as a parametric model of human behavior that compresses complex human actions into just 100 numbers. The model allows to capture motion, soft tissue movement and more, and provides a light-weight representation of human behavior understandable by large language models. Furthermore, SMPL outputs integrate seamlessly with standard 3D graphics software, enhancing applications in different industries such as animation or gaming. Beyond SMPL, *Meshcapade* has developed



Gender
Female
[✓ Male]
Non-binary

Units
[✓ Metric]
Imperial

Measurements

Basic

Height (cm) 184.0
Weight (kg) 107.6
Chest (cm) 118.0
Waist (cm) 111.0

Figure 2.20: Meshcapade interface

Source: https://tinyurl.com/meshcapade-interface

additional models, including MANO, a 3D hand model built from approximately 2,000 hand scans in various poses, and FLAME, which employs a novel dataset of 4D facial sequences to capture realistic facial features like head shape, jaw movement, eye motion, blinking, and expressions. These models converge in SMPL-X, a comprehensive framework combining full-body, facial, and hand models into one cohesive system. The latest advancement in this series, STAR, introduces an updated parametrization, making it even more compact and user-friendly for AI-driven workflows, while preserving the core functionality and flexibility of its predecessors. In Figure 2.21 is shown the SMPL-X part segmentation.

Moreover, the skeleton layout of the human model created, is defined in the Python model in Rodrigues formulation. Each triplet of pose-parameter corresponds to one skeleton joint, and the joint name for each pose-parameter-triplet is defined in the mapping below. Note that for other SMPL variants the whole structure is different.

```
0: 'pelvis',
1: 'left_hip',
2: 'right_hip',
3: 'spine1',
4: 'left_knee',
5: 'right_knee',
6: 'spine2',
 7: 'left_ankle',
 8: 'right_ankle',
 9: 'spine3',
10: 'left_foot',
11: 'right_foot',
12: 'neck',
13: 'left_collar',
14: 'right_collar',
15: 'head',
16: 'left_shoulder',
17: 'right_shoulder',
18: 'left_elbow',
19: 'right_elbow',
20: 'left_wrist',
21: 'right_wrist',
22: 'jaw',
23: 'left_eye',
24: 'right_eye',
25: 'left_index1',
26: 'left_index2',
27: 'left_index3',
28: 'left_middle1',
29: 'left_middle2',
30: 'left_middle3',
31: 'left_pinky1',
32: 'left_pinky2',
33: 'left_pinky3',
34: 'left_ring1',
35: 'left_ring2',
36: 'left_ring3',
37: 'left_thumb1',
38: 'left_thumb2',
39: 'left_thumb3',
40: 'right_index1',
41: 'right_index2',
42: 'right_index3',
43: 'right_middle1',
44: 'right_middle2',
45: 'right_middle3',
46: 'right_pinky1',
47: 'right_pinky2',
48: 'right_pinky3',
49: 'right_ring1',
50: 'right_ring2',
51: 'right_ring3',
52: 'right_thumb1',
53: 'right_thumb2',
54: 'right_thumb3'
```

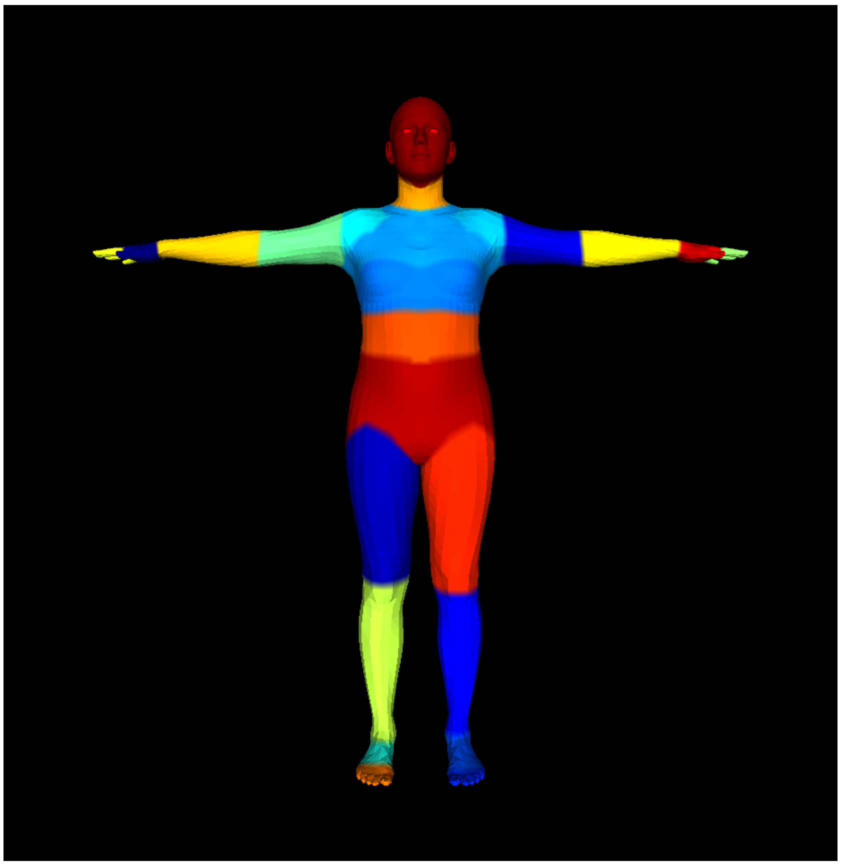


Figure 2.21: SMPL-X segmentation of the human body

Source: https://meshcapade.wiki/SMPL

Meshcapade platform is credit-based and offers both a Free Plan and a Pro Plan. With the Free Plan is included a limited daily quota and basic features, like the creation of a single avatar per-day and a 10 seconds limit for MoCapade, their markeless Motion Capture service. With the Pro Plan on the other hand, are included higher video limits (30 seconds for MoCapade) and early access to new features. For companies interested in using their tools professionally there's the chance to activate an enterprise plan.

Animate3D

Deepmotion is a innovative platform that aims to revolutionize Motion Capture and animation by offering a seamless and accurate way to animate digital humanoid characters through the application of AI-powered algorithms. *Deepmotion* offers two main services: *Animate-3D* and *Saymotion*. *Saymotion* uses the power of Generative AI to transform text prompts into 3D animations, without the need of specific hardware or software, making this process extremely accessible to everyone.

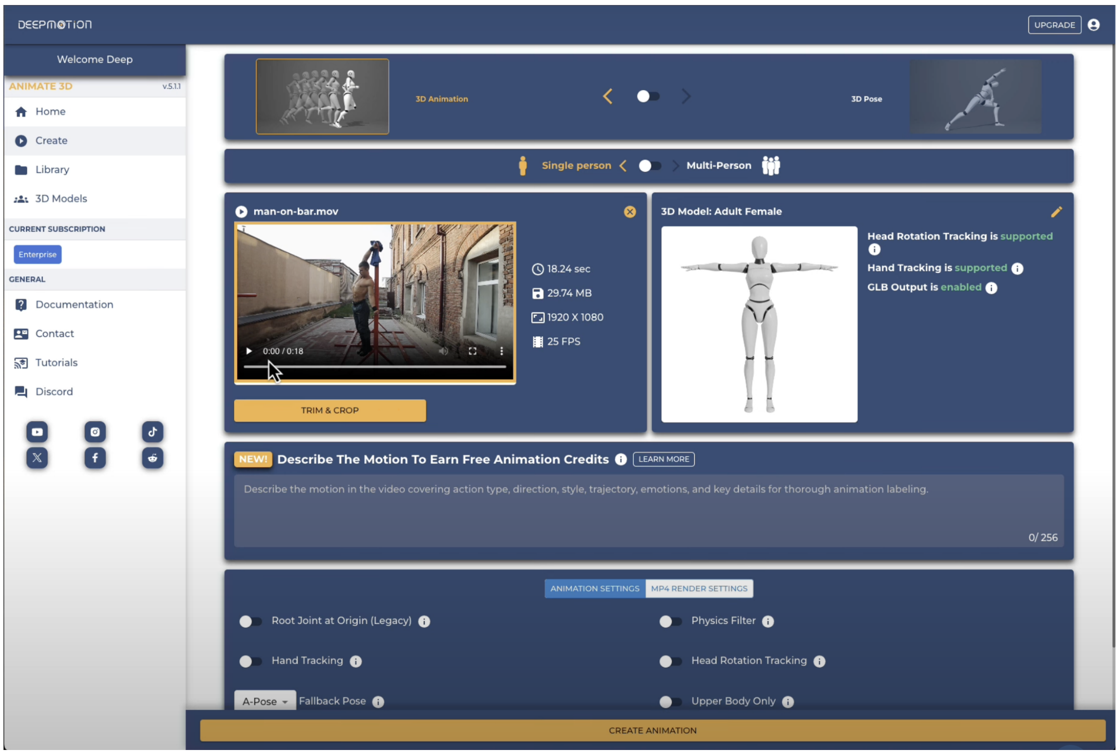


Figure 2.22: Animate3D interface

Source: https://www.deepmotion.com/animate-3d

Saymotion is currently an Open Beta and as today, the platform is optimized for desktop only. Also, it currently supports full body animation, no hands and face animations are possible to be made. *Animate-3D* on the other hand, incorporates *Deepmotion*'s markerless Motion Capture, by using advanced AI algorithms. With

Animate-3D it's possible to convert ordinary videos, coming from various sources, into 3D realistic animations. This systems tracks full body movements, facial expressions, hand gestures and multiple subjects. *Animate-3D*, like the other tools previously analyzed, captures the motion by analyzing the video source, and after processing the data on proprietary servers, transfers the movement into a 3D mannequin, perfectly rigged and animated. This data can be then exported with the preferred format, such as .FBX, so that can be easily integrated in commercialized 3D and animation software.

Chapter 3

Crowd generation process

3.1 Crowd definition and examples

In the context of Visual Effects, the term "crowd" refers to a large group of characters, often digital, that are created to populate scenes in film, television, or video games. These crowds are typically used to represent large gatherings (usually composed by at least 20 elements) of people, animals, or creatures, and are a crucial element in creating realistic, immersive environments for scenes that require hundreds or even thousands of characters, such as battlefields, stadiums, or cityscapes.

One of the key pioneers in the field of crowd simulation for computer graphics is Craig Reynolds. His groundbreaking work, *Boids*, was presented in the SIGGRAPH 1987 proceedings under the title "Flocks, Herds, and Schools: A Distributed Behavioral Model". This research introduced flocking mechanisms, which were later showcased in the short film *Breaking the Ice* by Stanley and Stella, which premiered at SIGGRAPH 1987. The product demonstrated the flocking behaviors of birds and fish. Reynolds continued his exploration in 1999 by extending the concept to include individual behaviors, notably steering. Another significant contribution came from Daniel Thalmann and Soraia Raupp Musse, whose research, starting in 1997, focused on a crowd behavior model that incorporated both individual actions and emergent behaviors. Their work, compiled in the book *Crowd Simulation* (2007/2013), addresses key challenges such as path planning, navigation graphs, potential-based methods, and the importance of gaze attention in crowds, as well as the ongoing study of real-time simulation.

Additionally, crowd simulation has been explored through various models across disciplines, from particle systems to artificial intelligence. A notable example is *Massive*, a crowd simulation software developed in 2001 by Wētā Digital for *The Lord of the Rings* films. Using agents equipped with "brains," this system allows characters to react to each other and their environment using "Fuzzy Logic", revolutionizing the way large crowds were simulated in film. In particular, the term "Fuzzy Logic" refers to a theory introduced by Dr. Lotfi Zadeh in 1965 at the University of California at Berkeley, that uses "degrees of truth" rather than the traditional binary true or false values. The theory allows for the definition of rules that interpret varying degrees of truth into crisp, computable values, a process known as "defuzzification". In crowd simulation, "Fuzzy Logic" plays a crucial role in systems like *Houdini* and *Massive*, where it helps manage the behavior of crowd agents, enabling them to make decisions based on complex, non-binary conditions. "Fuzzy Logic" enhances the realism of crowd interactions by simulating nuanced behaviors in response to their environment and other agents, significantly improving the quality of simulations. [17]

The primary objective of crowd simulation is to effectively represent large groups of individuals that exhibit both distinct individual behaviors and cohesive group dynamics. In the context of cinematic production, the simulation and rendering of crowds serve a critical purpose: to populate scenes with a sense of scale and activity while complementing the central narrative focus and the actions of primary characters. Crowds are often used as visual elements to create a dynamic and immersive environment, enriching the storytelling and adding depth to the scene without drawing attention away from the main action. To achieve this, filmmakers frequently opt to portray crowds from elevated perspectives, utilizing short focal lengths that emphasize the breadth and movement of the group while minimizing the need for intricate details in individual figures. This approach allows the audience to perceive the crowd as a collective entity rather than focusing on individual members. As a result, the level of precision and detail required by visual effects artists in these wide or distant shots is relatively lower compared to close-up sequences. In close shots, actors or animated characters are placed at the center of the viewer's attention, necessitating a much higher degree of accuracy and realism to maintain the integrity of the scene and ensure consistency with the narrative's visual demands. By strategically balancing detail and perspective, crowd simulations achieve their purpose while optimizing resources for the scenes that require the most meticulous craftsmanship.

A thorough exploration of *EDI*'s previous work on crowd simulations reveals remarkable examples showcasing diverse types of crowds. These can generally be categorized into two distinct groups: calm, composed crowds and action-oriented, dynamic crowds, depending on the requirements of the scene and shot.

A noteworthy example of the first category is the ceremonial fire scene crafted by *EDI* for the series *Romulus II* (2022). The breakdown of this scene (Figure 3.1) illustrates the meticulous process behind its creation. Real actors, intended to be the focus of the shot, were filmed on set to ensure authenticity in their appearance and movements. Meanwhile, the remainder of the crowd was generated entirely using computer graphics to populate the background and expand the scene seamlessly. The result is a convincing portrayal of a large assembly of villagers, participating in a ceremonial event. At the center of the scene, a wooden pyre burns brightly, serving as the focal point of the gathering. Surrounding the fire, the digitally enhanced crowd is depicted standing closely together, observing the event with calm and solemn expressions. The subdued and attentive demeanor of the crowd emphasizes the ceremonial and reverential tone of the scene, giving it a high level of realism.

Another compelling example, this time from the second category — dynamic, action-oriented crowds, can be observed in a battle scene from the same series, *Romulus II* (2022). The breakdown of this scene (Figure 3.2) demonstrates a completely synthetic crowd, showcasing the full potential of CG in creating intense and visually complex sequences. In this particular shot, the scene depicts a dramatic clash between warriors in the midst of battle. Unlike the ceremonial crowd, where real actors were prominently featured, only the bodies of fallen warriors were filmed on set to provide realistic visual references for the aftermath of the combat. The rest of the crowd, consisting of active, fighting warriors, was entirely generated using CG. This approach allowed the filmmakers to choreograph a large-scale battle with fluid interactions between the digital characters and their environment. The animated agents in the crowd are shown engaging dynamically with one another, exchanging blows, reacting to impacts, and interacting with weapons and terrain. The result is a vivid and chaotic depiction of combat, represented with a high level fidelity to reality.

A common scenario requiring the creation of synthetic crowds includes concert scenes or sporting events set in large venues such as stadiums or expansive parks. One of the most outstanding examples of such an endeavor is the production of *Bohemian Rhapsody* (2018) (Figure 3.3), a film whose visual effects were meticulously crafted by *DNEG*. To deliver photorealistic, dynamic, and highly directable rock concert audiences, the studio developed an innovative crowd simulation solution leveraging multi-view video capture and image-based modeling techniques. The process began with the acquisition of over 350 choreographed performances by individual crowd extras, captured on set using a specialized video camera array. This resulted in a total of more than 70 hours of high-quality footage. The captured data was subsequently processed through a custom system that converted the video content into lightweight 3D sprites — virtual representations of individuals — that could be seamlessly integrated into large-scale scenes. These 3D sprites were designed for efficient layout, synchronization, editing, and rendering, enabling the creation of a vast, lifelike crowd. To facilitate this process, *DNEG* developed scalable video processing technology and artist-friendly workflow tools, ensuring that even team

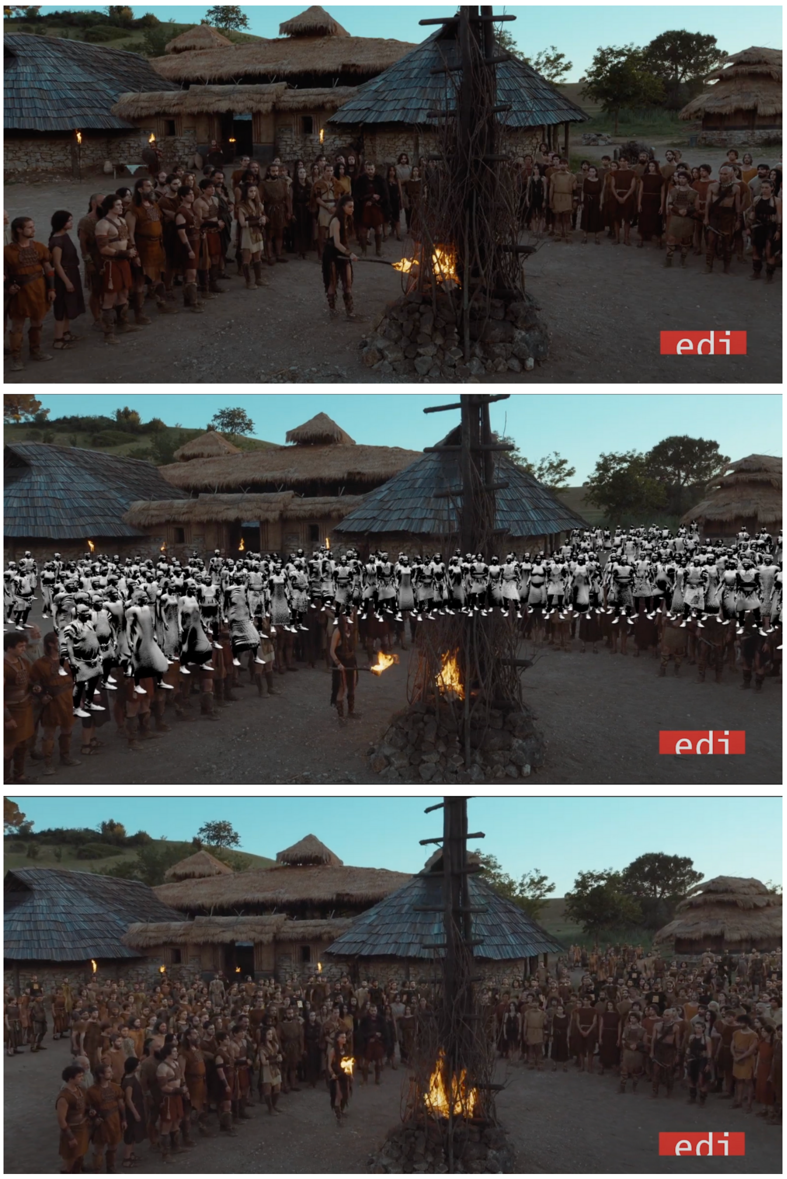


Figure 3.1: Crowd breakdown - Cerimonial fire scene - Romulus II

Source: EDI's FX Reel

members with minimal prior experience in crowd simulation could populate a virtual Wembley Stadium with thousands of cheering, dancing spectators. The crowd's behavior was carefully synchronized with Freddie Mercury's performance, achieving an authentic and visually stunning recreation of one of rock music's most iconic moments. This innovative approach highlights the intersection of technical

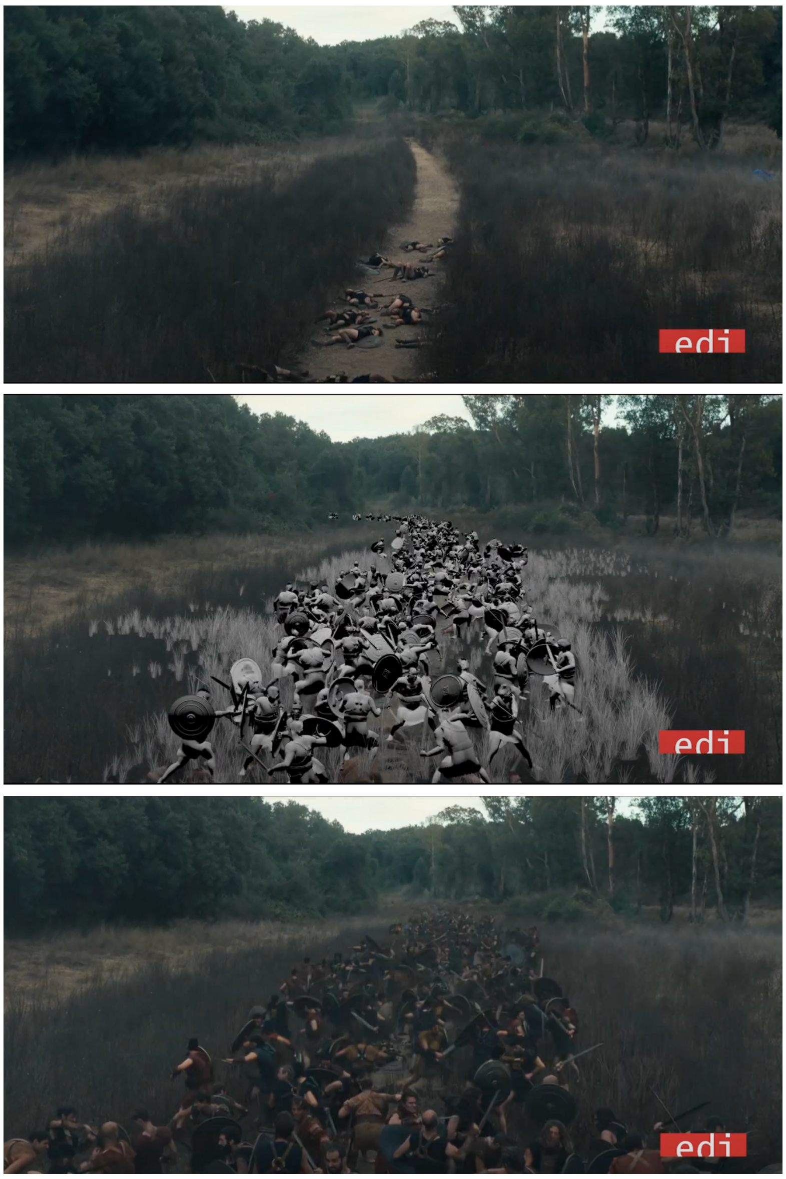


Figure 3.2: Crowd breakdown - Battle scene - Romulus II

Source: EDI's FX Reel

ingenuity and artistic excellence in modern visual effects. [18]

To gather the necessary data for crowd simulation, the *DNEG* team employed an innovative framework involving an array of six networked Arri AlexaXT cameras arranged around a green screen stage. The camera positions were meticulously designed to maximize coverage and enable the extraction of dense stereo depth

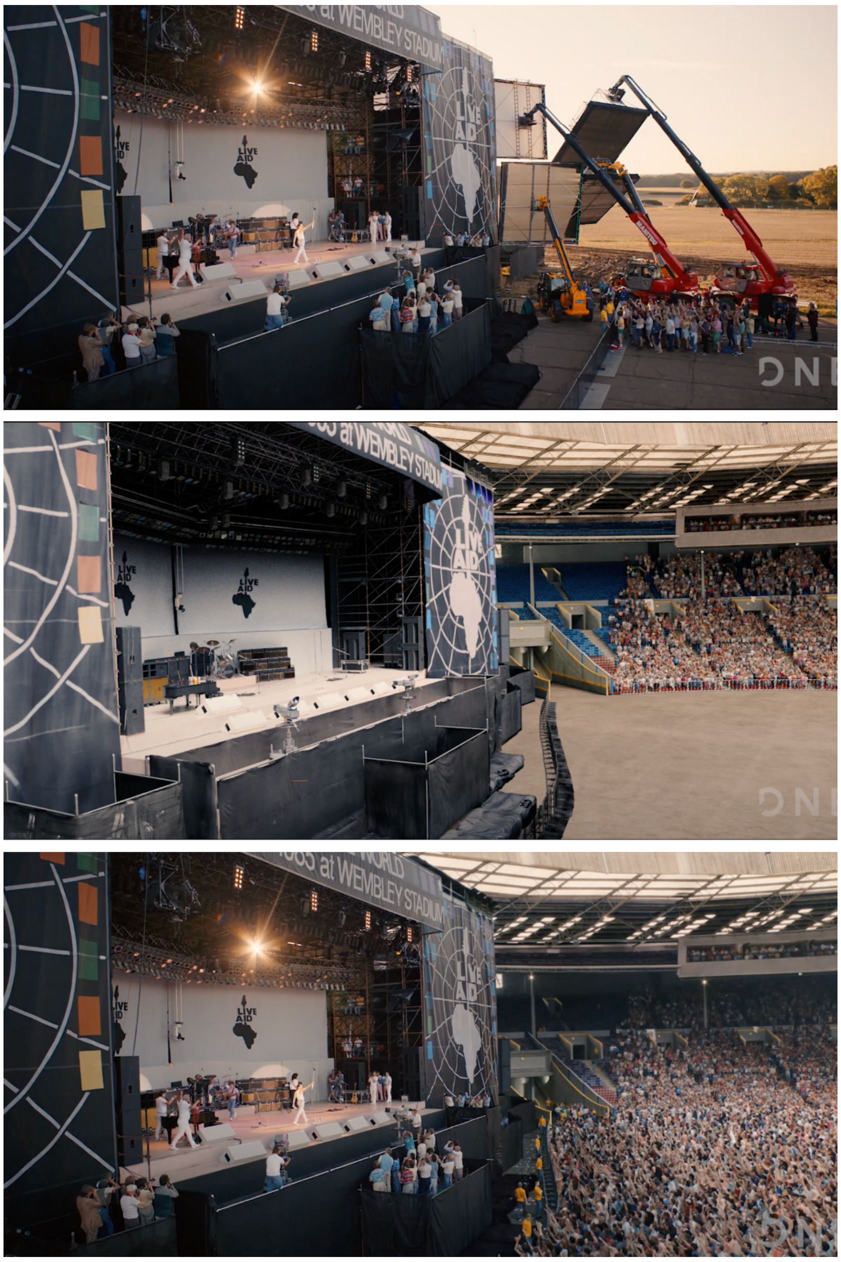


Figure 3.3: Crowd breakdown - Bohemian Rapsody

Source: https://tinyurl.com/DNEG-breakdown

information. These cameras were programmed to shoot at 48 fps to meet the requirements for slow-motion shots, ensuring flexibility for various cinematic needs. A carefully curated guide track, consisting of key verses and choruses from Queen songs likely to feature in the final film, was edited to optimize the variety of crowd actions captured while minimizing the duration of each take. Voiceover instructions were added to the track to prompt performers with specific actions tailored to the energy and tone of each song. Individual crowd extras were then called to the stage one at a time, where they performed choreographed routines synchronized to the guide track, ensuring a wide array of dynamic crowd movements. Once the live-action footage was collected, the next step was to generate 3D sprites for each crowd agent to populate the scene. To achieve this, the team relied on image-based modeling techniques, supported by a separate calibration stage designed to solve camera positions using video of a reference object captured during the shoot. The captured footage was then procedurally keyed and de-spilled to isolate the performer. A depth-matching algorithm was applied to the stereo video pairs, generating a surface point cloud. This point cloud was converted into a surface mesh using the *Poisson* reconstruction method. Subsequent processing steps involved refining the mesh by removing unwanted geometry, reducing noise, and optimizing the polygon count for efficiency. Finally, de-spilled texture data was projected onto the polygonal faces of the mesh, resulting in detailed 3D sprites. The completed assets were then written to an Alembic cache, allowing seamless integration into the virtual crowd simulations. [18]

3.2 Workflow for generating crowds

In the context of VFX production, crowd simulation typically falls under the responsibility of the internal FX department, as crowds are often treated as an extension of particle system simulations. This approach allows for the efficient management and animation of large numbers of agents with complex behaviors, making it a natural fit for FX workflows. To create these types of effects, the VFX industry commonly relies on *SideFX Houdini*, a powerful and versatile software specifically designed for procedural modeling, animation, and simulation. Due to its robust capabilities in handling particle systems and its advanced tools for crowd simulation, *Houdini* has become the industry standard for such tasks.

From this point forward, all information regarding the process of creating digital crowds will reference a *Houdini*-based workflow, as this is the software utilized by *EDI* for its crowd-related projects.

Crowds are essentially animated characters attached to particles. The movement of the characters in a crowd simulation is based on an underlying particle simulation in Dynamic node networks (DOPs).

In a crowd simulation, the characters are called "agents", and at any given time, the agents are in a particular "state", such as standing, walking, or running. Events can happen in a crowd simulation and they cause the change of the state of the agent, for example, from walking to running. These events are called "triggers". Crowd simulations require both a geometry (SOP - Surfce Operator) network and a simulation (DOP - Dynamic Operator) network:

- A geometry network is used to create and define the agents. Through SOP nodes, agents can be displayed and attached to certain objects such as the terrain.
- A simulation network is necessary to run the actual crowd simulation. The DOP crowd simulation network holds the logic that controls the movement of each particle. Within the DOP network, agents can interact with other dynamic elements.

For example, to simulate a crowd hanging around, agents can be assigned to one of two states: "idle" or "walk." In the "idle" state, the point to which an agent is attached remains stationary, while in the "walk" state, the point moves following a certain path, previously defined. A trigger mechanism monitors, for example, the duration each agent spends in its current state. When an agent exceeds the predefined time threshold, a state transition is triggered, switching the agent from "idle" to "walk" or vice versa. This dynamic switching between states creates the natural appearance of a crowd in motion. [19]

In order to deeply understand the functioning of the crowd generation workflow inside *Houdini*, it is necessary to define few concepts:

- Agents: the actual "actors" of the crowd. Agents are actual 3D characters, with an associated geometry, skeleton and animation.
- Crowd source: the "source" for a crowd simulation are the points in a SOP network. Corresponding to each point, an agent is instantiated. The position of those points define the starting position of each agent in the simulation.
- Clips: "clips" correspond to single portions of actions (e.g. walk cycle). They are played when an agent is in a specific state (e.g. "idle" or "walk").
- States: a "state" is defined by an animation clip that is played for the agents in that state, and the "behavior nodes" that affect the movement of the underlying particles that the agents are attached to.
- Triggers and transitions: "Triggers" define a condition that *Houdini* periodically checks in order to trigger a state transition (e.g. if the agent is in the "idle" state for more than 2 seconds, the triggers activates the "walk" state). The shift between two states is regulated by a transition, in which *Houdini* handles its duration by blending automatically the animation clips that characterize the two states.

- Weights: *Houdini* provides a variety of POP (Particle Operators) forces that can act as "behavior nodes", guiding the movement of the particles that agents are linked to. These "steering force" nodes function like particle forces but come with the added feature of weights, enabling realistic crowd behaviors such as "move to a specific point" or "follow a designated path". An agent can be influenced by multiple forces at the same time, like "stay on this path" while also "avoiding obstacles". The crowd solver manages these overlapping forces by normalizing their weights, meaning it adjusts them so their total equals 1. Each force is then scaled by its normalized weight, ensuring a balanced and coordinated effect. Along with the crowd steering force nodes, there are also default forces on the crowd solver that have the concept of weights.
- Obstacles: specific DOP objects that agents are supposed to avoid.
- Terrain: object in which agents are displayed (a static RBD - Rigid Body Dynamics - object). [19]

In order to set up a crowd simulation inside *Houdini*, it is necessary to follow three main steps:

1. Agent definition
2. Crowd source
3. Crowd simulation

3.2.1 Agent definition

An agent is defined in a *agent definition network*, a geometry SOP network where the artist can build a specific node tree, in order to prepare and characterize the agent that will act in the simulation. A generic tree that builds a correct agent definition can be seen in Figure 3.4.

The "Agent" node creates a primitive for the agent, and allows the artist to set basic information (e.g. name). Here the primitive can be imported in multiple formats, like .FBX or similar. The "Agent clip" nodes allow to import different animations clips to the agent definition. These clips have to be made specifically for the skeleton-type associated to the agent primitive. Note that if the animation clips don't respect this important point, then the animation will not work. The "Agent layer" node allows the insertion of other pieces of geometry to the agent (such as a hat or a sign meant to be held by the agent), which is possible to randomize when the crowd simulation starts. The "Null" node, in this example called "OUT_AGENT_DEFINITION", is simply a reference used by the crowd simulation network to grab the agents. The *agent definition cache* node allows

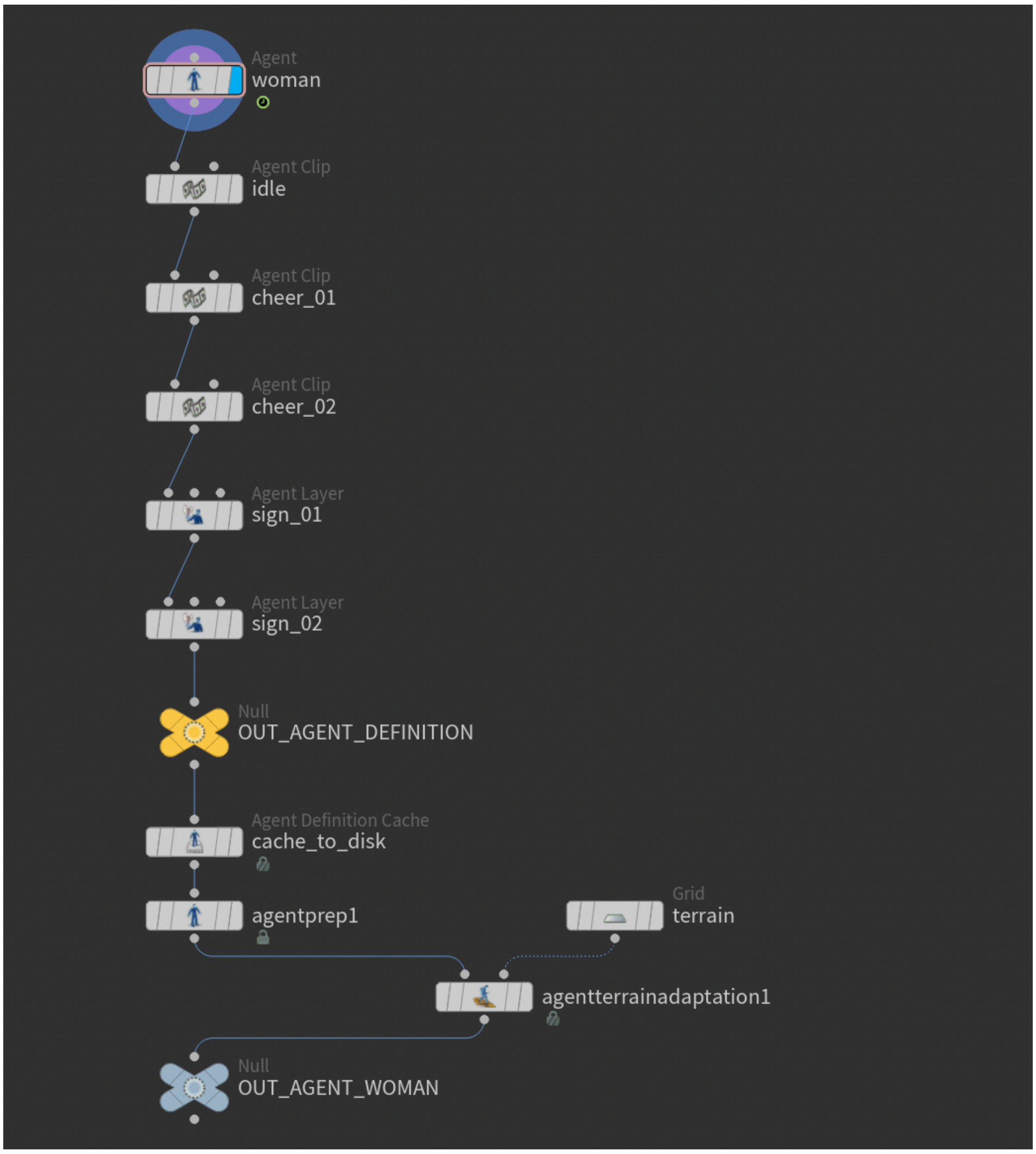


Figure 3.4: Agent definition node tree Houdini

the agent to be written and correctly saved to disk, in order to be available for the next processes. This node is used to speed up the agent definition network if the "cooking" process takes too long. The "Agent prep" node sets up additional information on the agent, such as the names of important joints and the transforms to use for the different limbs. This node is necessary if then a node like "Agent terrain adaptation" is meant to be used. This kind of node requires *Houdini* to know the joints that represent each body part, because they're going to be used by the crowd solver in order to perform certain task, such as foot placement. Note that this node comes after the "Agent definition cache" node because the computational work in this case is always fast. The "agent terrain adaptation" node needs also a new geometry as the second input (in this case a simple "Grid"), that will perform as the terrain to which the agent has to be placed into. In conclusion, the final "Null" node called in the example "OUT_AGENT_WOMAN", is a marker that will be then used as a reference from other networks.

3.2.2 Crowd source

In order to create a crowd, before starting up a simulation, a *crowd source network* is needed. This network, a geometry SOP, scatters points over a given area, and attaches agents onto the points. Figure 3.5 displays the node tree that characterizes this network.

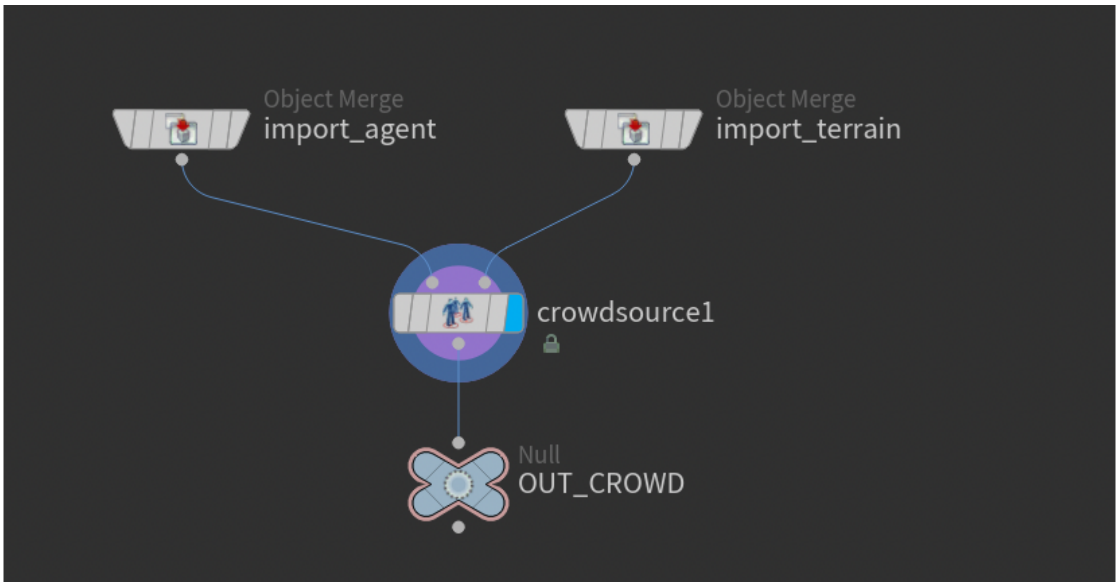


Figure 3.5: Crowd source node tree Houdini

In this structure, the most important node is the "Crowd source" one. This node is responsible of copying the agents onto points. The agents are fed into the node through the first input, thanks to the "Object merge" node that allows importing references from other networks. In this example the "Object merge" node is used to import the agents and the terrain from the *agent definition network*. The crowd source node offers in its setting a panel dedicated to the set-up and a panel dedicated to randomizing the attributes that characterize the crowd source. In the set-up panel, if no points or surface have been wired to the node, the artist can modify the crowd by controlling the number of agents scattered, the seed, and also how much each agent is supposed to be distant from the other. In this panel it is also possible to determine which of the *agent clips* is the *initial state* and also the direction of the agents. In the randomize panel, *Houdini* offers the chance to choose which of the attributes that characterize the *crowd source* are meant to be "randomized" in order to give the crowd a more realistic look. The "randomizable" attributes are the following: agent primitive, initial state, clip time, current layer, scale, initial velocity, heading or the up vector. This panel is extremely useful for crowds that behave in a simple way (maybe where locomotion is not required) and that don't need to start a simulation. This can happen whether the motion of the crowd agents is minimum or whether the shot does not require the agents to interact with each other or the surroundings.

3.2.3 Crowd simulation

The final step of the crowd creation process, is represented by the set-up of the *crowd simulation*. To start a simulation, a DOP network must be added to make the crowd move and execute behaviors. Inside the "DOP network" node are hosted the following nodes: "Crowd solver", "Crowd object", "Crowd source", the "Crowd states" and the "Crowd transitions". Diving deep into the single meaning of each one of the these nodes:

- *Crowd object*: allows the import of the data that the crowd systems has to process. Here the crowd data is imported from other environments.
- *Crowd source*: this is a POP Source DOP that references the agents created in the crowd source network. The node contains the particles that represent the agents.
- *State and behaviors*: here are collected all the "states" that characterize the agents in the simulation. The state name has to match the name given to the *animation clip* that the agents are supposed to play when they're in that state. The states node can be followed by *behaviors nodes* that affect the agent's motion. These nodes are useful for avoiding obstacles for example.
- *Triggers and transitions*: a "Crowd trigger" node specifies a condition that causes a change of state. Triggers can be combined with the use of "Crowd trigger logic" nodes that allow to specify logic relations between conditions, like "and", "or" or "not". Triggers are followed by "Crowd transition" nodes, which are responsible for switching the agents from one state to another when the trigger condition is met.

- *Crowd solver*: the core of the DOP network. This node implements the crowd system. It offers controls for default behaviors that affect all agents or a subgroup of them (when specified), such as agent collision avoidance and terrain following.

In Figure 3.6 an example of node tree that characterizes the crowd simulation can be observed.

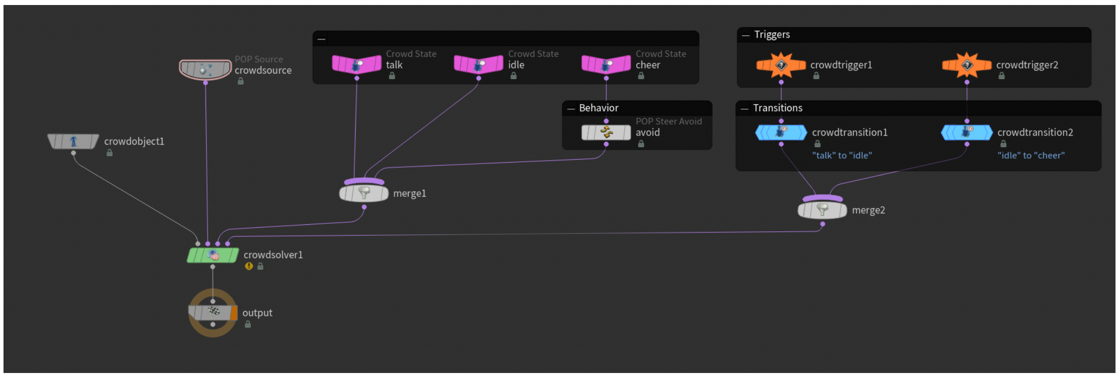


Figure 3.6: Crowd simulation node tree Houdini

3.3 How to make a realistic-looking crowd

To create a realistic-looking crowd, several key principles must be considered. As mentioned above, scenes featuring crowds are often wide shots designed to convey the impression of a heterogeneous group acting collectively. Achieving this effect requires careful attention to the variations that define the crowd as a whole and, by extension, each individual within it. Variation should be approached from two perspectives: aesthetics and animation. To avoid the need for an excessive number of unique models, it is standard practice to use techniques that generate variations for each agent, ensuring sufficient visual diversity across the crowd. This can be achieved by customizing the appearance of the generic agents — typically a "generic man" and a "generic woman" — that form the basis of the crowd. Adjustments to hairstyles, clothing, and overall looks are essential for creating distinct individuals. Clothing props, in particular, offer significant flexibility. These can be toggled on and off or combined in various ways to produce a wide range of unique appearances. Additionally, it's crucial to introduce variation in the agents' scale and, whenever possible, their body types. By incorporating these changes, the crowd will appear more organic, dynamic, and visually compelling.

AI-powered technologies, such as *image-to-3D* tools, can significantly streamline and enhance the process of creating crowds for visual productions. Let's consider, for instance, a company tasked with producing a simple crowd scene for a car-racing movie. In this scenario, the shot needs to show hundreds of spectators in the stands, erupting in joyful cheers as the winning car speeds by. In such cases, AI can be a game-changer. With just a few clicks and minimal time investment, it is possible to generate hundreds of human models — fully clothed in context-appropriate attire — ready for rigging and integration into a synthetic crowd created using traditional techniques, thanks to the photos taken on-set. These AI-generated agents are not intended for prominent positions in the shot but rather for background placement, where they will remain out of focus or softened by a "bokeh" effect.

On the animation side, it's crucial to balance the appearance of the crowd as a unified, homogeneous group with the individuality of each agent's movements. Each agent must display subtle variations in behavior to enhance realism. Traditionally, creating accurate and detailed animations for human motion is a challenging and time-consuming process due to the complexity of replicating natural human movement. One widely used method is motion capture, which, while effective, comes with significant costs. These include the expense of specialized software and hardware, as well as the hiring of performers to act out the motions. Additionally, the process often requires extensive clean-up, which can be labor-intensive and may not always yield perfect results. For a crowd of several hundred individuals, animators generally need at least five variations per state for each agent. For example, if a scene requires three states — idle, talking, and cheering — there must be variations within each. In the idle state, a person might shift their weight from one leg to another or scratch their head. For the talking state, one person might speak with their arms crossed while another might be seated and chatting. In the cheering state, one agent might clap their hands, while another jumps with excitement. With three states and five variations per state, a total of at least 15 distinct animations are required, all tailored to specific rigs.

In this context, AI markerless motion capture technology represents a transformative advancement. It allows companies to record diverse motion efficiently and at a fraction of the cost of traditional motion capture. Markerless systems work effectively with readily available equipment, such as consumer-grade cameras (DSRLs or APSCs), multiple GoPros, or even iPhones cameras, eliminating the need for expensive set-up. Moreover, Deep Learning algorithms could add another layer of efficiency by generating diverse motions from a single reference animation. These tools could automatically create variations, significantly reducing the time and effort needed while maintaining natural and fluid movement. By combining markerless motion capture with AI-powered motion generation, animators can achieve a diverse and dynamic crowd animation with remarkable speed and cost-effectiveness.

Moreover the animations defining each agent's state should be designed as long "looped cycles". This approach is essential not only for achieving a high level of variety in the agents' behaviors (due to the extended length of the loops) but also for ensuring smooth transitions between animations or seamless repetition of the same animation over time. Industry-standard crowd simulation software, such as *SideFX Houdini*, incorporates features that randomize the clip time for each agent. This randomization ensures the crowd appears diverse and natural, even when composed of only a limited number of animations. "Looped" animations refer to sequences where the starting and ending poses are identical, allowing the animation to replay seamlessly without any noticeable interruptions. If the starting and ending positions of an animation do not align, the result would be a jarring "snappy" effect when the action loops back to the beginning. Achieving perfectly looped animations with traditional methods is a complex and time-intensive process. This is where AI-powered tools could revolutionize the process by extracting diverse looped cycles from a single, long, and complex animation. Such tools could break down and repurpose segments of a larger animation into distinct, naturally looping cycles, significantly reducing the effort required to manually craft looped sequences. By automating this process, AI can enable animators to focus on creative refinement while still producing a dynamic and varied crowd efficiently.

Lastly, another critical aspect to consider when preparing data for creating a realistic human crowd is the animation retargeting process. This process is essential for transferring animations designed for one skeleton structure onto another. However, retargeting can introduce unwanted artifacts, especially when the source and target skeletons differ in bone structure or proportions — for example, transferring an animation from a male skeleton to a female skeleton with shorter legs or a smaller frame. These discrepancies often result in unnatural movements, distortions or unwanted interpenetration. Due to these challenges, many companies opt to create unique, specific animations for each skeleton type, which can be both time-consuming and resource-intensive. Automating or streamlining this workflow — whether through AI, automatic, semi-automatic, or procedural methods — could significantly accelerate the animation process and reduce the associated costs. Animation retargeting is particularly important when using systems like motion capture. The motion data captured by the hardware, such as a suit or sensors, is typically mapped to a standard skeleton, depending on the software that processes the data. This captured motion must then be retargeted to fit the specific skeleton of the agent being animated. Without an efficient retargeting workflow, this step can become a bottleneck, leading to delays and inconsistent results. Developing advanced retargeting tools or workflows that minimize artifacts and ensure smooth adaptation across skeletons would not only improve efficiency but also enhance the realism of human crowds. By addressing this issue, studios could more easily integrate motion capture data or traditional animations across diverse skeletons, streamlining production while maintaining high-quality animations. [20]

3.4 EDI's workflow weaknesses

The previous paragraphs outlined the general characteristics of industry-standard workflows for generating human crowds. To correctly structure this research, *EDI* identified several weaknesses in their current pipeline that required improvement to enhance both the efficiency and quality of their output.

Rather than addressing the crowd-making process as a whole, this discussion focused specifically on the critical stages of gathering the necessary data for creating human crowds, emphasizing the modeling and animation aspects. *EDI* highlighted the need for a solution that could deliver more, diverse, output in less time without compromising quality. Specifically, the company sought a robust and reliable system capable of generating realistic, high-quality animations that could be seamlessly transferred to crowd agents of any type via an efficient and stable retargeting system. They emphasized the need for a workflow capable of producing multiple variations of a single animation quickly while ensuring that looped animations were free from visual artifacts such as interpenetration, snapping, or foot-sliding. Furthermore, *EDI* expressed a strong interest in developing a system that could automatically extract clean, looped animations from complex input motions, reducing manual intervention and time expenditure.

On the modeling side, the company, wanted me to study tools and algorithms that could rapidly generate diverse and high-quality 3D models to populate their crowds, further streamlining the data preparation process.

To address these challenges, *EDI* proposed integrating AI, machine learning, and deep learning systems into their pipeline. They tasked me with researching and identifying solutions that leveraged these technologies to resolve their specific problems.

In the present research, I studied and examined numerous systems and technologies capable of addressing *EDI*'s needs. The most effective solutions were carefully combined to create a tailored, AI-powered workflow that met their requirements. This workflow was first implemented in a specific case-study project, which I developed to demonstrate the potential outcomes. The project allowed for a comparative analysis of various tools, technologies, and approaches, providing *EDI* with a clear understanding of their options in terms of cost-efficiency and quality of results. The following chapter will detail the project, its methodologies, and the results achieved, offering insights into the viability of this AI-enhanced workflow.

Chapter 4

Stadium Crowd Project

4.1 The case study

The case study for the present research is a stadium crowd scene, generated entirely through synthetic methods. The objective is to simulate a large-scale audience, consisting of hundreds of thousands of agents, joyfully cheering during a soccer game.

To construct this scene, the following resources were utilized:

- A 3D model of *San Siro* Stadium, previously developed and owned by *EDI*.
- Primitive agent models representing a generic male and female, created by *EDI* and commonly used in the company's crowd simulation projects.
- *SideFX Houdini* software, chosen for its robust procedural workflow capabilities.

These requirements were established to enable *EDI* to evaluate a practical workflow using the company's standard assets for crowd simulations, with the goal of seamlessly integrating the process into their existing pipeline.

The project aimed to demonstrate the effectiveness of the AI-driven workflow for crowd generation, which formed the core of my research.
As a result, this project primarily focused on the process of obtaining the necessary animation data for individual agents within the crowd. This aspect was identified by *EDI* as the most time-consuming and problematic phase prior to my research. Consequently, all testing was conducted with this objective in mind and followed a structured series of phases:

1. In-depth evaluation of each tool listed in Chapter 2, analyzing their strengths and weaknesses in relation to their AI-powered motion capture system, to

determine the most suitable option for *EDI* in terms of cost-effectiveness and quality.

2. Development of a procedural workflow for cleaning-up the animations generated in the previous phase to achieve production-ready results, addressing potential artifacts such as foot sliding and mesh penetration.

3. Design of a procedural workflow for efficiently retargeting the animations produced in the first phase onto *EDI*'s generic man and generic woman rigs.

4. Study and implementation of a system capable of generating diverse animation variations from a single input animation.

5. Development of a method for extracting high-quality animation loops from the processed animations.

6. Evaluation of the results and tools tested, assessing their effectiveness in relation to the project's objectives. The highest-quality animations were selected for integration into the stadium scene.

7. Construction of the stadium crowd, following the workflow described in Chapter 3.

8. Selection of camera angles to construct an effective sequence that highlights the strengths and outcomes of the AI-powered workflow proposed and implemented in the project.

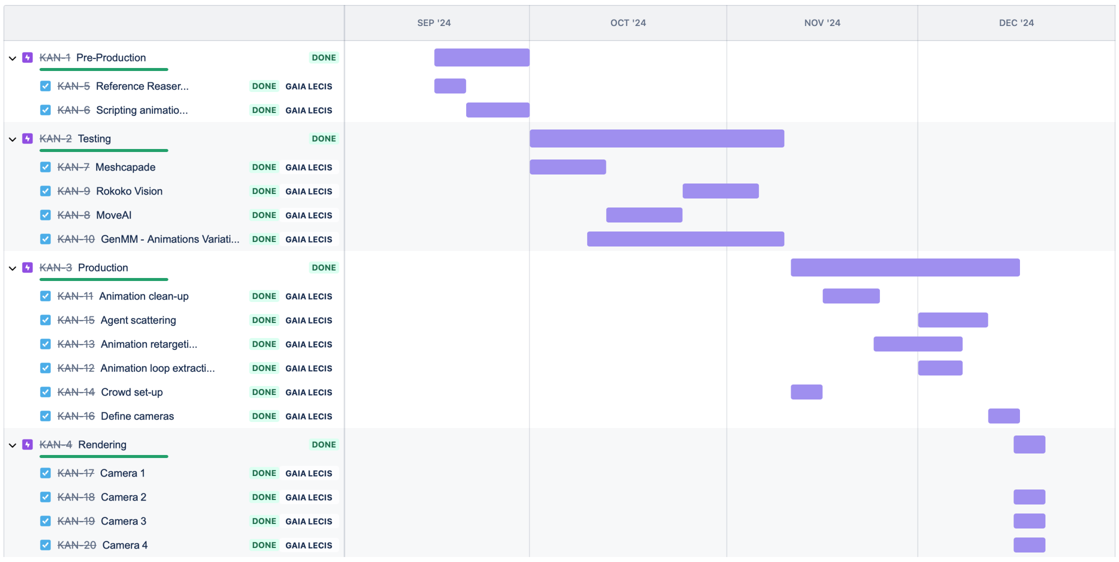
9. Rendering of the final scene.

4.2 Project organization

To successfully execute the project, it was essential to establish an organized workflow to minimize potential setbacks and optimize time and resource management. To achieve this, each phase was systematically structured using *Jira*, which enabled me to break down the project into distinct phases, each with specific sub-goals, and manage them following an agile-based approach. In this framework, "sprints" were represented by smaller sub-tasks that collectively contributed to the completion of the entire project.

Additionally, for each phase, I was required to document the ongoing activities through detailed reports. These reports were designed to be clear and well-structured, ensuring that *EDI* employees could easily reference them in the future to understand the methodologies applied in the research and seamlessly integrate them into their existing pipeline. For documentation purposes, *Confluence* was utilized — a collaborative workspace and documentation tool developed by Atlassian. Widely used by teams for organizing and sharing content such as project documentation, meeting notes, and knowledge bases, *Confluence* facilitates the creation of structured pages, real-time collaboration, and seamless integration with other tools like *Jira*.

Figure 4.1 illustrates the structure of the "sprints" that defined the development of the project.



~~KAN-1~~	Pre-Production	DONE	
~~KAN-5~~	Reference Reaser...	DONE	GAIA LECIS
~~KAN-6~~	Scripting animatio...	DONE	GAIA LECIS
~~KAN-2~~	Testing	DONE	
~~KAN-7~~	Meshcapade	DONE	GAIA LECIS
~~KAN-9~~	Rokoko Vision	DONE	GAIA LECIS
~~KAN-8~~	MoveAI	DONE	GAIA LECIS
~~KAN-10~~	GenMM - Animations Variati...	DONE	GAIA LECIS
~~KAN-3~~	Production	DONE	
~~KAN-11~~	Animation clean-up	DONE	GAIA LECIS
~~KAN-15~~	Agent scattering	DONE	GAIA LECIS
~~KAN-13~~	Animation retargeti...	DONE	GAIA LECIS
~~KAN-12~~	Animation loop extracti...	DONE	GAIA LECIS
~~KAN-14~~	Crowd set-up	DONE	GAIA LECIS
~~KAN-16~~	Define cameras	DONE	GAIA LECIS
~~KAN-4~~	Rendering	DONE	
~~KAN-17~~	Camera 1	DONE	GAIA LECIS
~~KAN-18~~	Camera 2	DONE	GAIA LECIS
~~KAN-19~~	Camera 3	DONE	GAIA LECIS
~~KAN-20~~	Camera 4	DONE	GAIA LECIS

Figure 4.1: Project's timeline

In the following sections, each one of the phases listed on the project's timeline, will be explained and examined in deep.

4.3 Pre-production

To develop a believable project, it was crucial to have an in-depth understanding of how human beings move and behave in a scenario similar to the one I aimed to depict. Therefore, the pre-production phase was carefully divided into two key sub-tasks: reference research and animation scripting.

For reference research, I analyzed real-life stadium crowd scenes, observing how people naturally reacted and interacted while watching a sporting game. This provided valuable insights into the authentic actions and behaviors that define a stadium audience. Once the core concept was solidified, I moved on to scripting the animations that would bring the synthetic crowd to life. To ensure a structured approach, I categorized the different types of animations into distinct groups and compiled them into a detailed table, outlining each action that would contribute to a realistic and dynamic crowd simulation.

Category	**Anim name**	**Description**	**Class**
Idle	SitIdle01	Seated looking around	Seated
Idle	SitIdle02	Seated with hands on laps	Seated
Idle	StandIdle01	Stand Idle in place, looking around	Standing
Idle	StandIdle02	Stand Idle in place, scratch head	Standing
Idle	StandIdle03	Stand Idle weight switching, looking around	Standing
Idle	StandIdle04	Stand Idle weight in place, hands on hips	Standing
Cheer	SitCheer01	Seated cheer single arms with hands on fist pump	Seated
Cheer	SitCheer02	Seated cheer with both arms that go up and down	Seated
Cheer	SitCheer03	Seated cheer with both arms down and forward	Seated
Cheer	SitCheer04	Seated clapping looking around	Seated
Cheer	SitCheer05	Seated waving with both arms up, then up and down	Seated
Cheer	StandCheer01	Stand jumping with single arm up, then both	Standing
Cheer	StandCheer02	Stand jumping with both arms up	Standing
Cheer	StandCheer03	Stand, point forward then jump with both arms up	Standing
Cheer	StandCheer04	Stand cheer with left arm up waving	Standing
Cheer	StandCheer05	Stand cheer with both arms up waving	Standing
Cheer	StandCheer06	Stand clap bent forward then stops looking around	Standing
Cheer	StandCheer07	Stand clap arms up in both directions	Standing

Cheer	StandCheer08	Stand wave both arms up in both directions	Standing
Cheer	StandCheer09	Stand yelling with hands on mouth	Standing
Cheer	StandCheer10	Stand swing flag from left to right	Layer
Cheer	StandCheer11	Stand still holding sign slightly waving it in both directions	Layer
Mixed	SitToJump01	Seated to jump with arms up	Transition
Mixed	SitToJump02	Seated to jump with single arms on fist pump	Transition
Mixed	SitToStand01	Seated to stand looking around	Transition
Mixed	StandToSit01	Stand to sit looking around	Transition
Mixed	StandToSit02	Stand with arms up cheering to sit calm with hands on laps then down	Transition

Table 4.1: Animation Scripting Table

Animations were organized into categories: Idle, Cheer and Mixed. Each one of the animations were briefly described in order to be prepared for the capturing moment. In the end, animations where also labeled into different classes, which were selected in order to better organize the transition graph that the agents are supposed to follow during the simulation.

4.4 Testing

Once the planning of activities and the pre-production phase was completed, it was time to move on to the second main phase: Testing. This stage involved evaluating various AI tools, open source or available on the market, analyzing their strengths and weaknesses to identify the most suitable solution for the workflow. Key factors such as ease of use, cost, and output quality were carefully considered during the selection process.

As outlined in Chapter 2, most of these tools were accessible through freemium offerings, requiring either a subscription or the purchase of credits to unlock their full potential. To initiate the testing phase, I leveraged the free trials provided by these platforms (when they were not open-source). Although these trials came with limitations — such as reduced performance and a restricted number of generations — they proved valuable in optimizing the selection process. The constraints encouraged a strategic approach, allowing me to focus on a subset of animations from the previous phase to directly assess the real capabilities of each tool.

The following sections will provide a detailed breakdown of the workflow applied to each tested tool, along with an in-depth analysis of the results obtained.

4.4.1 Meshcapade

Meshcapade was tested by using its two main functionalities: animations from prompt (*Motion from text*), and animations from RGB videos (*MoCapade*).

In this case, the free trial, comprehended a sign up bonus of 2500 credits, with a daily top-up of 500 credits. With this subscription, *MoCapade* would process videos with a maximum length of 10 seconds.

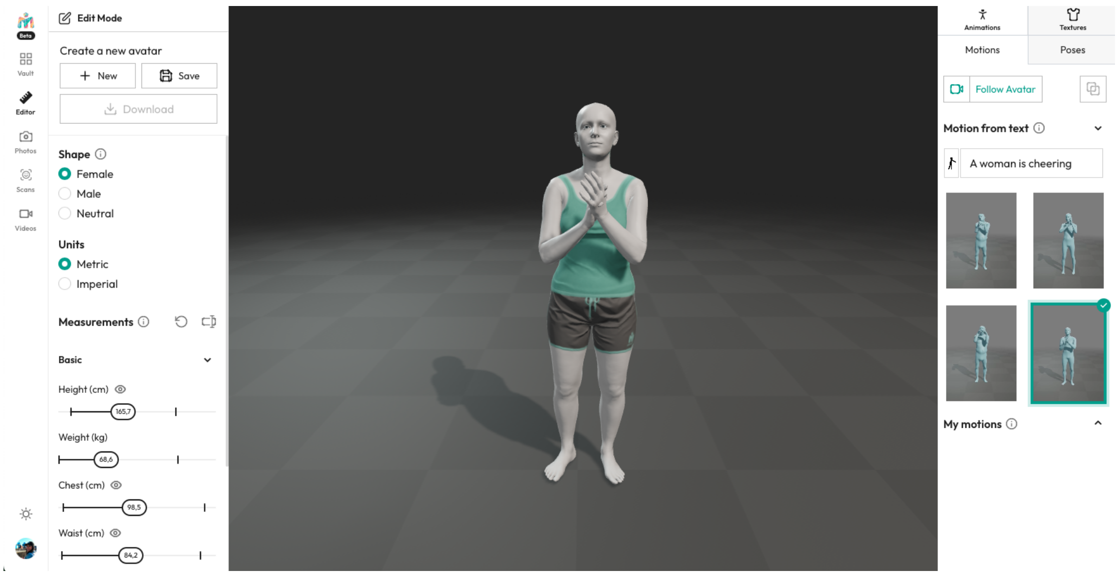
Let's focus on the *Motion from text* functionality. To generate an animation using this feature, the process begins with creating a reference avatar through the platform's intuitive "Editor" panel. Users can select from basic human shapes — male, female, or neutral — and customize parameters such as height, weight, chest, and waist for precise adjustments. Once the avatar is set-up, animations can be generated via a simple text prompt. The prompt structure requires a subject and an action (e.g., "A person is walking"). Each prompt generates four animation variations, which can be previewed directly on the avatar. If the results are unsatisfactory, users can refine their prompt and re-run the generation without consuming credits.

The interface is user-friendly and well-organized. On the right side of the "Editor" panel, the Motion section allows users to initiate animations, while the "Poses" panel provides options to apply standard poses — A-pose, T-pose, U-pose, or W-pose. Users can also refine hand positioning, selecting between flat, relaxed, curled, or fist poses. Additionally, the "Textures panel" offers eight visualization styles, enabling customization of the avatar's appearance and clothing for a tailored look.

To download the character along with its animation, the avatar must first be saved before proceeding with the download. The entire process costs 100 credits, so it is crucial to ensure the avatar meets expectations before finalizing the save to avoid unnecessary credit usage. Avatars can be downloaded in multiple formats, including FBX and OBJ. The FBX format includes a rig and may offer additional export options, such as *Unreal Engine* compatibility. Additionally, textures can be downloaded from the "Textures" tab in the right-side panel of the "Editor". When

hovering over the desired texture, a download icon appears in the top-right corner. Clicking this icon downloads three essential texture files: the *Diffuse map*, which provides the base color; the *Normal map*, responsible for surface details; and the *Roughness map*, which defines material reflectivity.
In this scenario three main tests are worth to be analyzed.

Test 01 Prompt: **"A woman is cheering"**. The animations generated predominantly showcase a clapping movement, with various subtle differences in the starting and ending positions as well as the velocity of the action. Despite these variations, a consistent issue across all the generated animations is mesh penetration, particularly around the hands. These mesh intersection problems become especially noticeable during the clapping motion, where the hands often overlap unnaturally. Figure 4.2 illustrates the results of these generated animations.



Shape: Female [✓ Female] Male Neutral
Units: [✓ Metric] Imperial
Measurements — Basic
Height (cm) 165,7
Weight (kg) 68,6
Chest (cm) 98,5
Waist (cm) 84,2
Motion from text: A woman is cheering

Figure 4.2: Meshcapade Test 01 Result

Test 02 Prompt: **"A woman is cheering and jumping"** In this case, the animations generated are less accurate than the ones from the Test 01. All the variations proposed, show weird artifacts in the movements, with the avatar assuming strange poses and a movement style that does not match the prompt description. Also in this case, mesh penetration problems persist.

Test 03 Prompt: **"A woman screaming joyfully"** Even with a more specific prompt, the results obtained are weird and not very close to the prompt description.

Here the animations generated show weird movements, that don't align at all with the action described by the prompt. Instead of yelling, the avatar in some variations claps its hands or mimics a strange movement with its hand close to its mouth. Besides the poor coherence with the prompt, all animations present, mesh penetration, as highlighted in the previous tests, but also jittering problems and scattering movements.

These tests show the important limitations that this functionality offers. The main problem is represented by the poor fidelity of the animations generated to the prompt description. Further tests, showed that when the prompt describes more "popular" motions, such as walking, running; the reconstruction of the movement is more precise and presents a higher quality. This happens because the prompt description strictly depends on the database of motions used for training the model, that is supposed to follow a Large Language Model architecture. Whenever the prompts present keywords that are not inserted into the training database, the model tries somehow to find and reconstruct similar actions by combining the ones recognized by the algorithm. This leads to poor results, and a small degree of control over the final result, making this kind of service not usable by companies that constantly have to confront the requests of clients.

Further tests were conducted then to dive deep into *MoCapade*, *Meshcapade*'s Markerless Motion Capture. It is possible to access the *MoaCapade* panel through the vertical navigation bar on the left side of the interface. *MoCapade* gives users the chance to transform simple RGB videos into 3D animations. This functionality is available by simply clicking the "new" button inside the editor. This feature allows to upload videos in MP4 or MOV formats, with a length of maximum 10 seconds. Once the video is uploaded, it's possible to re-name the avatar and select the human shape the avatar is supposed to have - female, male or neutral. Once this process is completed, it is possible to actually create the avatar at the cost of 500 credits. At this point the information of the avatar start to be processed by *Meshcapade* proprietary servers, and usually after 1-2 minutes the avatar will be successfully uploaded to the viewport.

The official documentation does not specify any particular requirements for the videos to be uploaded, suggesting that the feature works with any type of RGB video. However, to improve the algorithm's performance, it is good practice to use high-quality videos, preferably recorded with a tripod to ensure stability. This helps the model track and reconstruct the subject's joints more accurately in the synthesized motion. Additionally, the accuracy of motion capture can be improved if the performer wears tight-fitting clothing or, ideally, keeps key joints uncovered. The best outfit for this purpose would be a snug tank top and short pants to allow for better joint tracking.

In order to test this functionality, a brief MoCap session was organized. During this session, the animations listed in Table 4.1 were recorded. These videos have been recorded with an iPhone 14 Pro, with the 1.0x lens. Since no tripods were available, the phone was placed in a stable position, by using generic objects to support it. For this reason the inclination of the camera was not perfect, and this was reflected in the results. All the videos were recorded by keeping in mind that the animations were supposed to be looped. So, to facilitate the loop extraction, the movements were meticulously scripted in order to start and end with a similar pose. These videos have been then edited so that their duration would not exceed the 10 seconds limitation supported by the free subscription.

In this case, numerous tests were conducted, but two, in particular, stand out.

Test 04 The source video captures a person performing a cheering motion. It begins with their hands raised in the air, clenched in a fist pump, followed by alternating repetitions of the same motion with the right and left hand. The video was processed remarkably quickly, taking approximately one minute, and the animation was promptly available in the viewport. The quality of the reconstructed movement is pretty high: all motions were replicated with impressive fidelity, with no mesh penetration, and even the movement of the fingers was seamlessly reconstructed. The only noticeable weaknesses are foot sliding and minor jittering in the animation. However, these issues could be taken care of during the post-processing phase. The most striking aspect of this process is its exceptional speed — within just a minute, a high-fidelity animation is ready for download. It is important to note that when downloading the animation, no rest pose is attached to the rig. Therefore, to facilitate the retargeting process, it is necessary to download both the animated character and the same character in a T-pose. This requirement results in a total cost of 600 credits per animation. Figure 4.3 illustrates the results of this test.

Test 05 The source video, in this test, captures a person performing a sit-to-cheer action. In the beginning the person is seated and looks around, then after seeing something exciting, they jump up with their right hand raised in a fist pump before sitting back down and resting their arms on their lap. Clearly, here the motion that is supposed to be reconstructed is more complex: the presence of a prop - the chair - could obstacle the AI-model to correctly reconstruct the movement. Additionally, the jumping action is highly dynamic and hard to reconstruct due to the poor visibility of the joints, covered by the shoes. The outcome though, is again remarkable: the motion is very well reconstructed and complicated motions like the jumping one is synthesized with a decent level of accuracy. However, unlike the previous test, a slight mesh penetration can be observed in the end of the action,

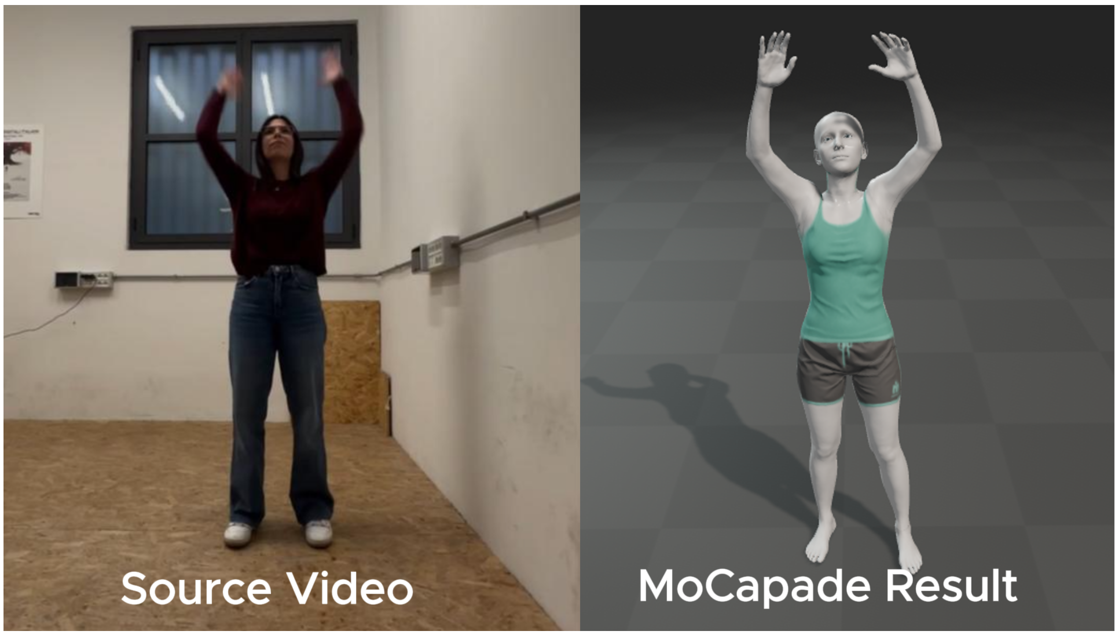


Figure 4.3: Meshcapade Test 04 Result

when the hands rest on the the laps. Foot sliding can be seen as well, though it seems to be less pronounced then expected. Nevertheless, these issues could be efficiently corrected during the post-processing phase. The results of this test are illustrated in Figure 4.4.

Based on the results of these tests, *Meshcapade* proves to be an effective tool for companies looking to create brief and simple animations. While the *Text-to-Motion* feature does not particularly stand out in terms of output quality and output control, *MoCapade* — their markerless motion capture system — is both highly accessible and capable of producing good-quality results in just one minute. However, relying solely on *MoCapade* for large-scale projects may become cost-prohibitive and the output quality may not be sufficient.

4.4.2 MoveAI

MoveAI was presented as one of the most promising tools for Makerless Motion Capture, due to its solid algorithm, and results presented though its demos. This tool could not be tested entirely with the free subscription, in fact no free trial for the multi-cam option was available. For this reason the testing was limited to *MoveOne*, the free app that the company released for iOS users, that unlocks the Markerless Motion Capture with a single camera. *MoveOne* gives new subscribers 30 credits with the free plan, that do not expire. A single credit is consumed for

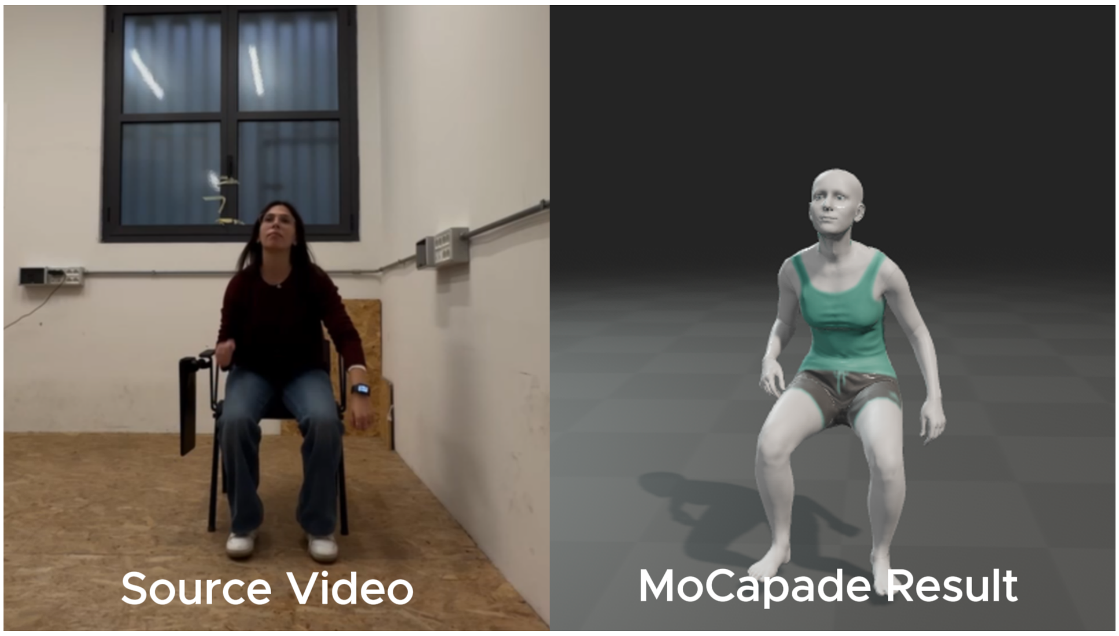


Figure 4.4: Meshcapade Test 05 Result

one person animated for 1 second - this means that for a 10 seconds long animation, 10 credits will be spent.

Unlike *Meshcapade*, *MoveOne* provides clear guidelines for camera and actor positioning to achieve the best possible results. With *MoveOne*, the action must be recorded within the app. Once the record button is pressed, the actor must align with the displayed A-pose figure to ensure proper camera calibration. After this setup, a countdown begins, signaling the start of motion recording. Once the recording is complete, converting the video into an animation is straightforward due to the app's intuitive and user-friendly interface. However, processing time can vary significantly depending on server load. When the animation is ready, it can be downloaded in FBX format, allowing easy transfer to the preferred 3D software for further refinement or direct use. All animations exported from the app are generated at 60 FPS.

Due to limited credit availability, only one test was conducted. No specialized equipment was required; the motion was captured using an iPhone 14 Pro, mounted on a tripod.

Test 01 The source video captures the performance of a dynamic jump cheer, characterized by consistent jumping, with hands raised and alternately waved. The action culminates in an overhead clap. While the recording experience itself was exceptionally smooth and the action easy to interpret, the data processing phase presented significant delays. Despite waiting several hours for the process to complete, it was ultimately still unfinished. The final output was not available until several days later. However, the end result was impressive: the animation appeared fluid and natural, with no noticeable foot-sliding, and even the finger movements were tracked with precision. Despite the impressive quality of the final output, a few issues emerged, primarily related to mesh penetration. This was particularly noticeable during the overhead clap action, where the mesh slightly distorted. Figure 4.5 provides a visual representation of these issues.

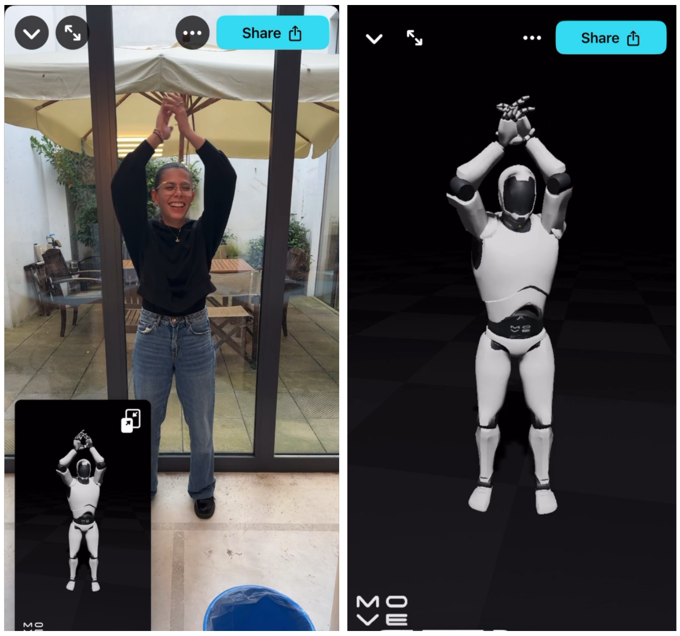


Figure 4.5: MoveOne Test 01 Results

MoveAI largely met expectations, delivering high-quality output even when the source video was suboptimal. However, the tool exhibited notable weaknesses, primarily in its slow data processing and the high subscription cost required to access premium features. Among these features, the multi-camera setup promised even higher-quality animations but remained inaccessible without a significant financial investment. Due to these limitations, *MoveAI* ultimately failed to meet *EDI*'s requirements and was therefore excluded from further testing or integration into the workflow.

4.4.3 Rokoko Vision

Rokoko Vision was tested in two phases. The first phase focused on evaluating its capabilities within the single-camera AI motion capture system. In the second phase, I was able to test the dual-camera setup for 3D animation generation, which yielded particularly favorable results and was well received by *EDI*.

The free subscription allows new users to test the single-camera and dual-camera setup with minimal limitations. Unlike other platforms that use a credit-based system, *Rokoko Vision* enables users to create animations of up to 15 seconds and access a few basic clean-up tools within *Rokoko Studio* free software. These features made it possible to conduct an in-depth assessment of *Rokoko Vision*'s functionalities. Once an animation is recorded using just one camera, two cameras, or an external video is uploaded, the data is processed on *Rokoko*'s proprietary cloud servers. Upon completion, the animation can be viewed through the web interface or further refined within the *Rokoko Studio* desktop application, which offers specialized clean-up tools, such as the foot-locking filter. Once the desired adjustments are made, the animation can be downloaded in a preferred file format and with a preferred skeleton option (e.g Mixamo, HumanIK etc).

In particular, the dual-camera system implies the completion of a set-up phase that includes camera and ground calibration (Figure 4.6). This process is very intuitive, thanks to a step-by-step guide offered by the interface.

With the Plus subscription, the workflow remains similar but includes additional features. Once the calibration process is completed, animations can be recorded without time restrictions. As with the free subscription, the data is processed on *Rokoko*'s cloud servers and subsequently accessed in *Rokoko Studio.* Moreover, Plus subscribers gain access to advanced functionalities, such as the ability to import custom characters and automate retargeting.

Within this framework, several tests were conducted. The following paragraphs will detail the most significant findings from these evaluations.

Test 01 The first test aimed to evaluate the performance of the single-camera AI motion capture system. In this scenario, an animation recorded during the initial motion capture session was uploaded to the platform (these videos were recorded with an iPhone 14 Pro 1,0x camera). The recorded video depicted a simple cheering action, beginning with a relaxed idle pose and transitioning into an enthusiastic arm-raising motion. The processing phase was relatively fast, with the animation becoming available for review within approximately two minutes. To maximize the tool's performance, the animation was further refined within *Rokoko Studio.* The software enabled edits such as adjusting the animation length via the timeline and

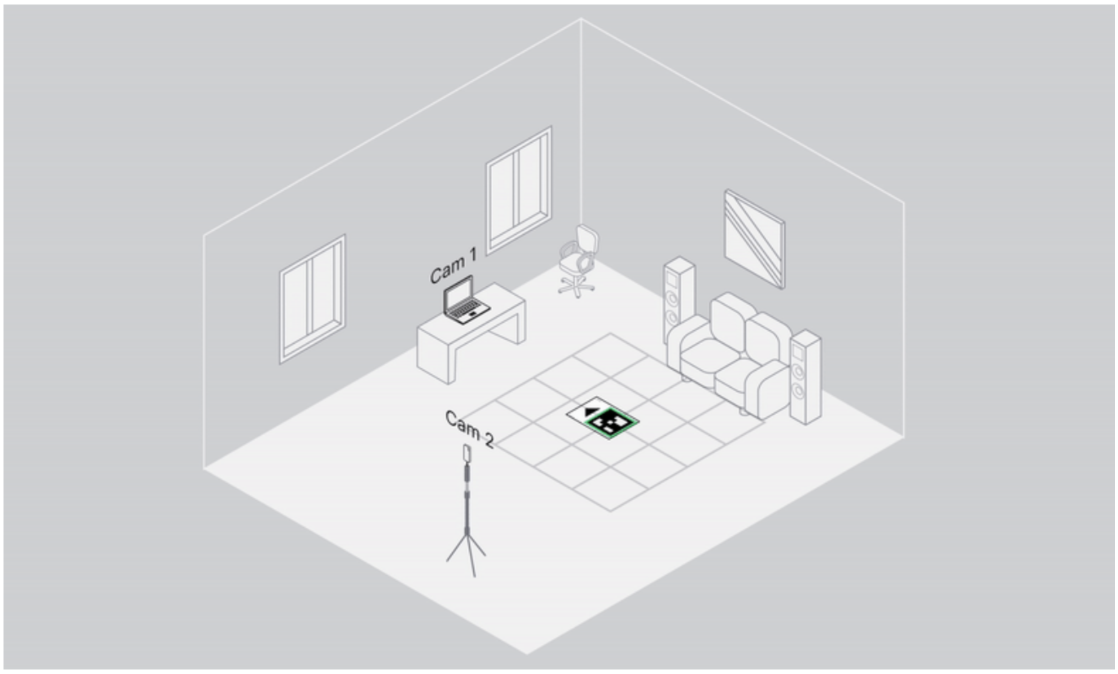


Figure 4.6: Calibration Phase preview

Source: Rokoko Vision Guide

applying specialized filters designed to enhance motion quality. Among these, the actor filters, accessible on the right-hand panel, played a crucial role in refining the final output:

- The "Locomotion Filter" simulates the character's position by estimating ground contact points. Users can manually adjust foot contact keys during playback for improved accuracy.
- "The Treadmill Filter" locks the actor's hip position in place, preventing unintended spatial movement.
- The "Toe Bend Filter" automatically bends the actor's toes as they transition in and out of foot locks, ensuring smoother motion.

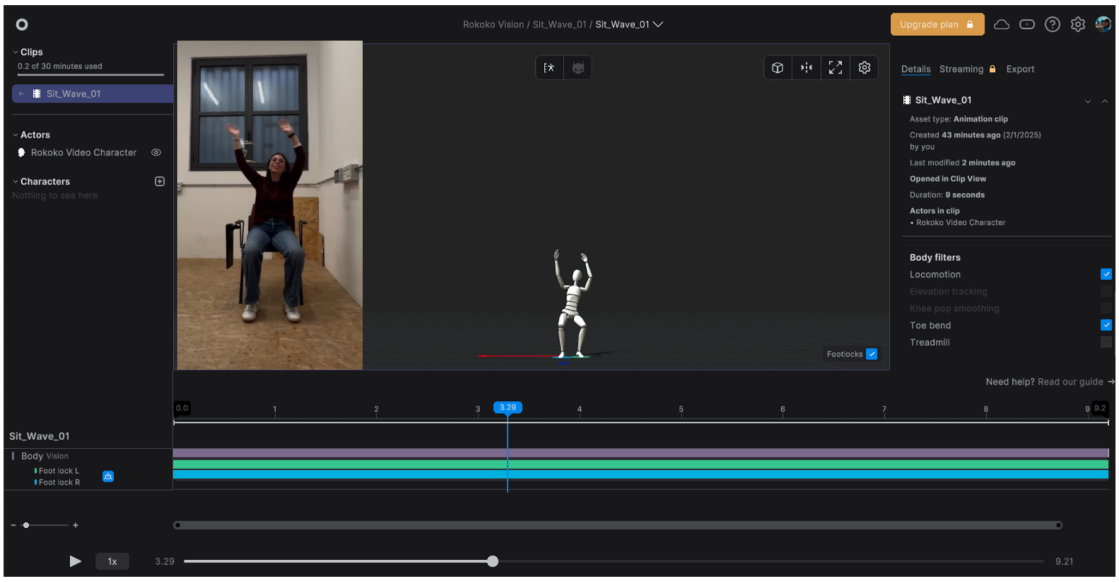
The relevance of these filters depends on the nature of the captured movement. For this specific animation, the Locomotion and Toe Bend filters were applied to improve motion fidelity. The overall animation quality was satisfactory, with decent foot-locking and fluid motion. However, some noticeable artifacts were present, including rigid character movement and few unnatural joint rotations (e.g., elbows and knees). These issues can be addressed in the next phase of post-processing.

One notable limitation of *Rokoko Vision*, in contrast to other motion capture tools, is its inability to track finger movements, which restricts its application in some scenarios.

Test 02 The second test involved using a source video that captured a seated cheering action. This test aimed to evaluate the performance of the tool when external props, such as a chair, were present in the scene. The action was relatively simple: the actor performed a swinging motion with arms raised in the air. As in the previous test, the processed animation was further refined using the filters available in *Rokoko Studio* desktop application. In this case, the "Locomotion Filter" proved essential, as the character was not correctly positioned on the ground without it. The "Toe Bend Filter" did not produce noticeable differences in this animation, as the feet were consistently detected on the ground regardless of whether the filter was activated. As a result, the filter did not provide any explicit improvements to the animation. The "Treadmill Filter", however, was not suitable for this scenario. When activated, it caused the character's movement to appear as though it was “floating” above the chair, which was not desirable. Overall, the generated movement was of acceptable quality, but some issues remained, such as noticeable rigidity in the character's motion and abnormal joint rotations. Additionally, minor mesh penetration and general jittering were observed. As a result, the presence of an external prop in the scene did not influence the quality of the output, as no many differences form the previous tests are noticeable. The results are displayed Figure 4.7.

Test 03 This test and following one, were done in order to try out the capabilities of the dual set-up camera system. With this system higher quality results were expected, because recording the same action from two different perspectives should avoid obstructions and guarantee a better movement reconstruction. In order to obtain the best results from this set-up, *Rokoko* furnishes a precise guide where good practices are listed, from the positioning of the cameras, to the best camera settings to use, to the clothing that the actor should wear when recording the action. For these tests, the guidelines were followed precisely in order to see the best performances of the software. The hardware and room set-up is briefly described into scheme represented in Figure 4.8.

This test was conducted using a webcam as the first camera and an iPhone 14 Pro as the second, both set to 24 fps, which resulted in some motion blur and noise. Both cameras were mounted on tripods, and due to poor room lighting, an additional light source was used to provide balanced illumination. The light was directed towards the ceiling to ensure proper diffusion, as direct lighting would have caused harsh shadows on the actor. Additionally, the actor's clothing was selected



Sit_Wave_01
Asset type: **Animation clip**
Created **43 minutes ago** (2/1/2025) by you
Last modified **2 minutes ago**
Opened in Clip View
Duration: **9 seconds**
Actors in clip
• Rokoko Video Character

Body filters

Locomotion	✓
Elevation tracking	
Knee pop smoothing	
Toe bend	✓
Treadmill	

Figure 4.7: Rokoko Vision Test 02 Results

to facilitate the algorithm in tracking joint movement: tight-fitting clothes were worn, and the knees and feet were left uncovered. In this test, the actor performed a joyful cheering action, characterized by continuous jumping with one arm raised in the air, up and down. Notably, the arm raised in the air was not fully visible to both cameras: while it was clearly visible to the first camera (Camera 1), it was only partially visible to the second (Camera 2). This was done intentionally to assess the system's performance when the movement was partially obstructed. The overall results were satisfactory. The movement appeared smooth and coherent with the captured action, and foot locking was effective, even though the jumping motion posed a challenge for accurate reconstruction. No mesh penetration was observed. The movement of the bending arm was reconstructed reasonably well, although the arm's rotation appeared somewhat unnatural. This issue was anticipated, as the arm was partially obstructed and not directly visible to both cameras. Both the "Locomotion" and "Toe Bend" filters were essential due to the jumping motion. While the animation still exhibited some rigidity, the overall result was nonetheless satisfying.

Test 04 In this test, another cheering action was performed. The actor began by jumping alternately on the right and left foot while raising both arms in the air and moving them in a circular motion. The action culminated in an overhead clap while continuing to jump. This more complex motion was tested to assess

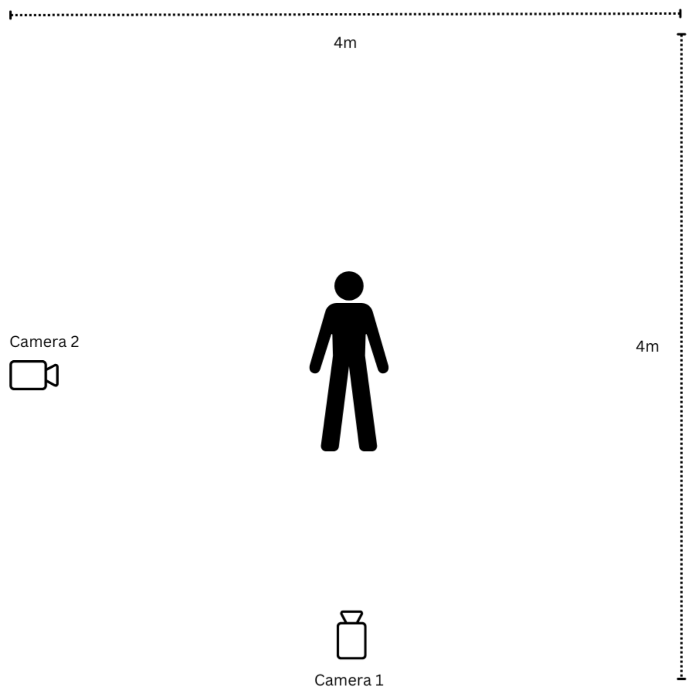


Figure 4.8: Room set-up scheme

the performance of the dual-camera setup in handling intricate actions, such as hand clapping. Similarly to the previous test, a portion of the movement was obstructed, as it was not fully visible to both cameras simultaneously. Given the circumstances, the results were reasonably good. The jumping movement was accurately reconstructed, although the knee-bending action was slightly unnatural due to some approximations in the tracking. The clapping motion, however, was only partially reconstructed, as finger tracking was entirely absent. Notably, some mesh penetration occurred during the clapping portion of the action. Despite these issues, the animation can still be further refined during the post-processing phase, making the overall result satisfying. Results are displayed in Figure 4.9.

Upon analyzing the results obtained from the preliminary tests, *Rokoko Vision* demonstrated good performance, though not without limitations. However, due to the extensive capabilities offered by the free subscription, it was possible to thoroughly test the tool, adjusting the capture parameters to achieve the best possible results.

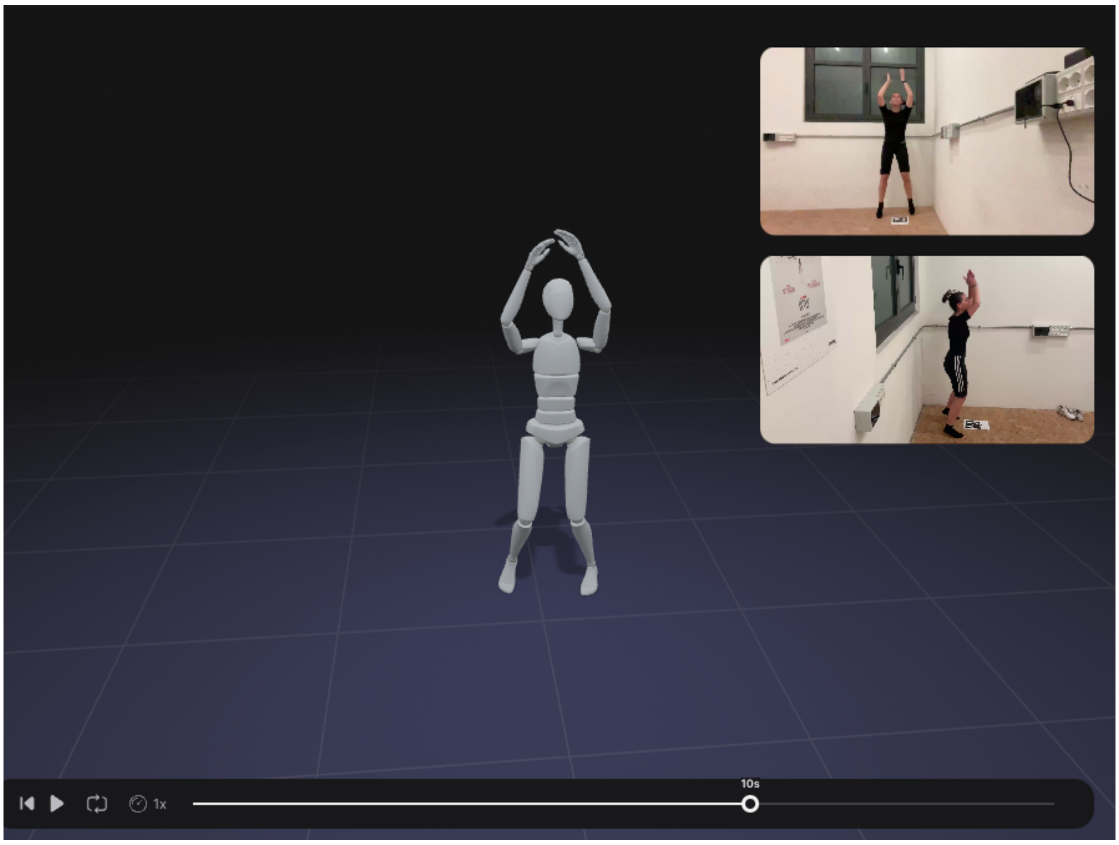


Figure 4.9: Rokoko Vision Test 04 Results

The first attempt involved modifying the hardware set-up. I observed that high-speed dynamic movements resulted in significant motion blur. I hypothesized that using higher-quality camera settings might mitigate this issue and lead to better results. Consequently, further tests were conducted using an iPhone 14 Pro as the primary camera and a Sony Alpha 6500 as the secondary camera, both set to a recording frame rate of 50 fps. This frame rate was expected to reduce motion blur and, in turn, improve motion reconstruction. Despite these adjustments, no significant improvements were observed in the output.

I then hypothesized that the main issue affecting motion reconstruction was the partial obstruction of movement. To address this, I repositioned the actor relative to both cameras. The goal was to ensure that both arms and legs were fully visible to the cameras. In these tests, the actor was rotated 45° towards the second camera rather than directly facing the first. This set-up minimized the obstruction of the movement and resulted in better output, particularly in the reconstruction of joint rotations, such as knee bending and elbow movement. This configuration was identified as the most effective for optimal performance.

Additionally, a few locomotion animations were tested to evaluate *Rokoko Vision*'s performance with walking, running, and other complex locomotion actions. However, these tests did not yield solid results due to the limited capture space available. The room was too small to record an entire walking cycle; after just two steps, either one of the cameras failed to detect the subject, causing the capture process to fail. This type of action will require further testing in the future.

Another key consideration during the hardware set-up process was ensuring a stable connection between the cameras and the computer used for capture. I found that a wired connection was preferable, as it provided greater stability. In contrast, relying on wireless connections resulted in inconsistent camera connectivity, which negatively impacted the output quality.

Overall, *Rokoko Vision* proved to be a promising AI motion capture tool, offering impressive results given the quality of the outputs and the relatively low cost of the hardware and software subscription. The results obtained from these tests were highly appreciated by *EDI*, who subsequently invested in a Plus subscription for further testing and integration into their animation pipeline. Thanks to this investment, I was able to conduct additional tests capturing longer and more complex movements. The results exceeded expectations, and *Rokoko Vision* was identified by the company as the best tool for generating high-quality animations in a short time frame. It is important to note, however, that all the results generated using this tool are not yet production-ready. They require further clean-up and refinement, which will be discussed in greater detail in subsequent sections.

4.4.4 GenMM

In order to address the problem of obtaining many variations from a single and specific animation, *GenMM* stood out as one of the most remarkable researches that could solve this challenge.

GenMM is a generative model designed to extract a wide range of diverse motions from a single input sequence. It was developed based on previous GAN-related research, which often required long offline training times and produced outputs with visual artifacts, struggling to handle large and complex skeletons effectively. Unlike those models, *GenMM* leverages the training-free nature and high-quality output of the *Motion Matching* method [21]. Due to its training-free approach, *GenMM* can generate high-quality motion within seconds, even for complex skeleton structures. At the core of *GenMM*'s generative framework lies the generative motion matching module, which employs bidirectional visual similarity as a generative cost function for motion matching. This module operates in a multi-stage framework, refining an initial random guess through exemplar motion matches. [22]

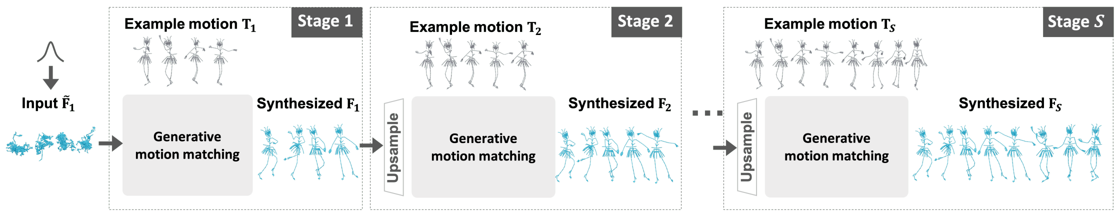


Figure 4.10: GenMM generative framework

Source: GenMM Paper

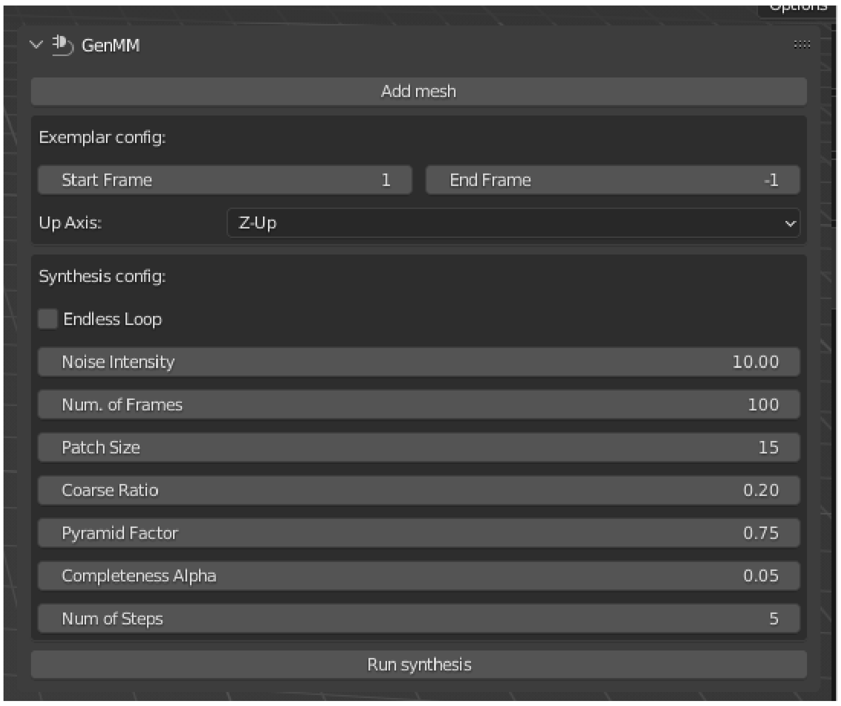
Figure 4.10 illustrates the *GenMM* generative framework. Beginning at the coarsest stage, each generative motion matching stage (s) refines an up-sampled version of the previous stage's output, using motion patches from the example motion T_s to generate a more detailed motion sequence F_s. Notably, the initial synthesis at the coarsest stage is purely generative, as it starts from noise sampled from a Gaussian distribution. [22]

Given its lightweight nature, the researchers implemented this framework in Python and developed a simple *Blender* add-on, making *GenMM* easily accessible. This add-on allows users to upload a source animation and quickly generate diverse, high-quality variations.

By following the instructions on the project's GitHub page [23], I was able to install and test the tool successfully. The add-on was specifically designed for *Blender 3.22.0*, *Python 3.8*, and *PyTorch 1.12.1*. The interface of *GenMM* is shown in the figure below (Figure 4.11).

The interface allows users to modify the following parameters:

- Start Frame and End Frame: Define the duration of the input animation. The default values encompass the entire animation;
- Up Axis: Specifies the character's up direction;
- Noise Intensity: Controls the noise introduced at the coarsest stage;
- Number of Frames: Determines the length of the synthesized motion;
- Patch Size: Defines the temporal length of motion patches used in generative matching and blending. This value is multiplied by the coarse ratio;
- Coarse Ratio: Sets the ratio between the patch size and the length of the coarsest-stage motion sequence, controlling the initial receptive field size;



Exemplar config:

Start Frame 1

End Frame -1

Up Axis: Z-Up

Synthesis config:

Endless Loop

Noise Intensity 10.00

Num. of Frames 100

Patch Size 15

Coarse Ratio 0.20

Pyramid Factor 0.75

Completeness Alpha 0.05

Num of Steps 5

Figure 4.11: GenMM Blender Interface

- Pyramid Factor: Determines the up-sampling factor applied at each stage;
- Completeness Alpha: Ensures all motion patches from the example sequence are incorporated into the synthesized motion;
- Number of Steps: Specifies the number of optimization steps used to refine the synthesized motion at each stage.

The interface also includes an option to generate looped motion from the example sequence, which can be enabled via a toggle. However, for this functionality to work, the generated animation must be shorter than the example sequence. As indicated in the documentation, the sample animation should be at least 500 frames long in order for the tool to perform at its best.

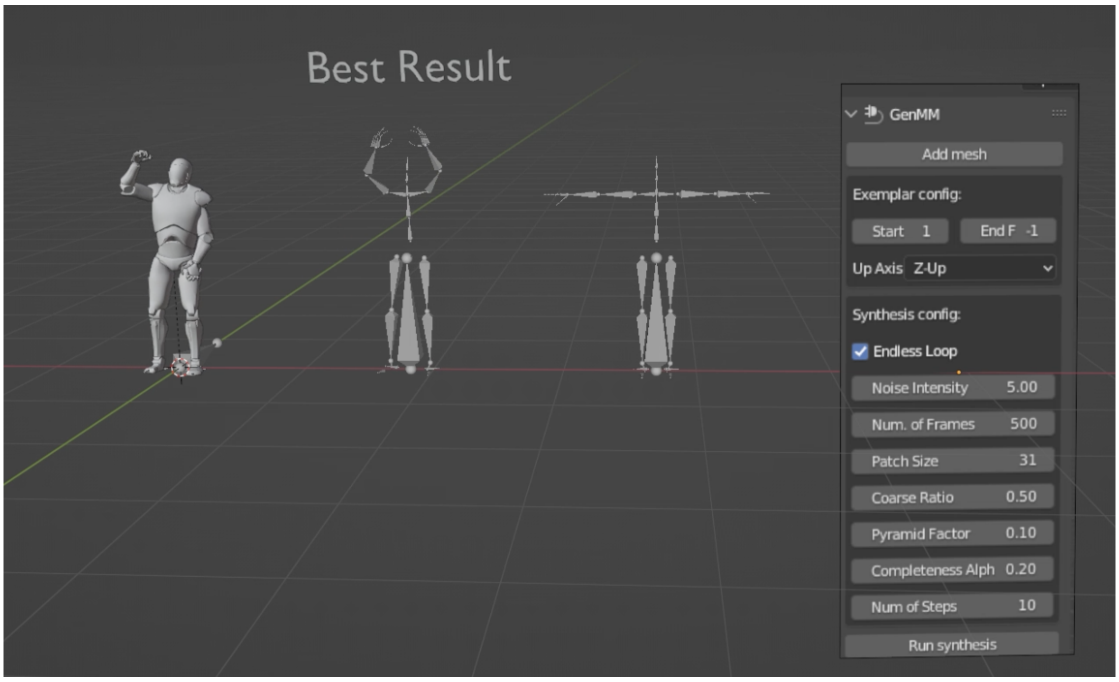
Adjusting these parameters is essential to achieve the desired motion output, depending on the source animation and the intended results.

Once *GenMM* was installed, I tested it using sample animations from previous experiments. The tool performed as expected, generating infinite motion variations in a fraction of a second. However, I encountered a few issues:

- Each synthesized animation was scaled by 100 along the x, y, and z axes and rotated by -90° on the x-axis. These transforms had to be manually fixed for each animation.

- While the synthesized motion appeared to retain the same skeleton structure as the sample animation, the bone rotations differed. This caused complications during the retargeting process (which will be discussed later), as retargeting requires identical bone orientations for both the bind pose and animated sequence. To mitigate this, a "synthesized" T-pose must be generated to match the bone orientations of the new animation.
- Selecting the right parameters was extremely challenging, often requiring multiple synthesis attempts to achieve the desired outcome.
- Some synthesized motions exhibited artifacts, such as unintended root bone movements or random unnatural motions.
- The infinite loop generation feature did not work for all animations.

Despite these challenges, I was able to archive some good results. These can be appreciated in the example below (Figure 4.12).



Exemplar config:
Start 1
End F -1
Up Axis Z-Up

Synthesis config:
☑ Endless Loop
Noise Intensity 5.00
Num. of Frames 500
Patch Size 31
Coarse Ratio 0.50
Pyramid Factor 0.10
Completeness Alph 0.20
Num of Steps 10

Figure 4.12: GenMM Test Results

In this test, the sample motion was represented by the animation generated though *MoveOne*, as it featured a more dynamic movement incorporating various actions as jumping, clapping hands, and alternately spreading arms in the air. On the left, the animation generated with *MoveOne*, is displayed. The parameters chosen for producing the final output are shown in the figure, and they have been chosen as the final ones after 5 different attempts. In this example, the "Endless Loop" parameter was enabled, and in this case, worked perfectly, resulting in a seamless looped synthesized animation. On the right side, the synthesized T-pose is displayed.

In the end *GenMM*, although the difficulties encountered, demonstrated to be a great tool for generating animation variations in a immediate way, and, for case scenarios like crowd scenes, *GenMM* could be a game changer for the entire animation process.

4.5 Production

The production sprint was fundamental in order to prepare the animation data, gathered in the previews phase, for the stadium crowd sequence. The work in this phase was not linear, in fact, in order to overcome some difficulties, I had to dynamically switch from one sub-task to another, as the results reached did not fully satisfy me.

4.5.1 Animation Retargeting

The first sub-task I focused on was animation retargeting. As discussed in previous sections, this process is essential for efficiently transferring motion from one skeleton to another. Based on *EDI*'s requirements, I aimed to develop a procedural workflow within *SideFX Houdini*. The key advantage of a procedural approach is its reusability — once set-up, it can be applied to various scenarios, including different target skeletons. To achieve this, I studied and analyzed the *KineFX* workflow [24].

Using this system was extremely important because, thanks to its robustness and flexibility, I could handle many case scenarios with different characteristics, such as characters with different number of joints, characters with different limb proportions or characters with different joint orientations. The basic workflow proposed by *SideFX*, is clearly schematized in Figure 4.13.

Let's now describe how each node of tree works.

On the top of the tree it can be found a "FBX Character Import" node: the latter is used to import the target character into the scene. Note that other kinds of "Import Nodes" could be employed for further animation type files (e.g MoCap data). On the other side another "Import" node is required, in order to correctly bring in the source animation.

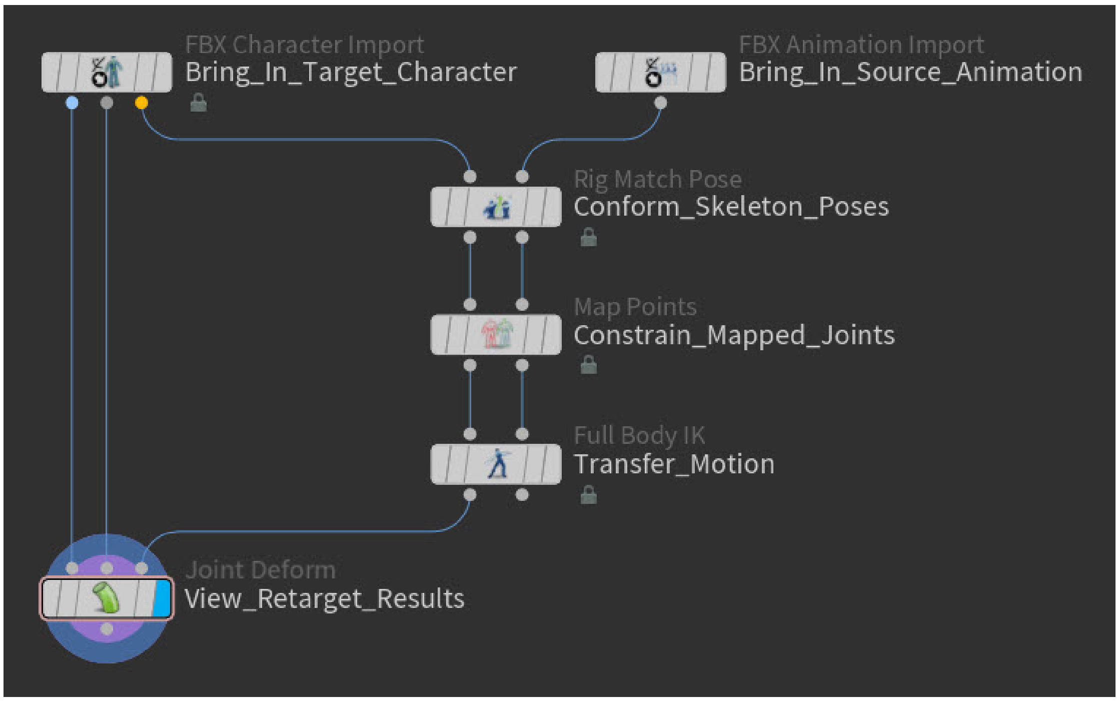


Figure 4.13: KineFX core retargeting workflow

https://tinyurl.com/retargeting-workflow

The next node, the "Rig Match pose", is responsible for effectively matching the poses of the source and the target skeleton. Moreover, this node allows to conform the source and the target skeletons' poses in world space. It then stores the skeletons' transform values or "rest poses" as point attributes. In addition, this node takes care for matching the scale between target and source if different. By selecting this node it's then possible to access the "Rig Match Pose" mode, where transforms can be applied to both target and sources' skeletons. The goal is to match as more accurately as possible the pose of the target skeleton.

The following step is to map source points to target points. This is possible due to the "Map Points" node, which creates references between source-target joint pairs and then stores those associations as mapping attributes. The source-target joint pairs have a constraint-like relationship. Once the correct wires are connected to the right inputs, it's possible to access the "Mapping" mode. Here both the source and target's skeleton are displayed into the viewport, the joints are represented as small dots. The objective is to associate the corresponding body parts between targets and sources by simply sliding them from one skeleton to the other. If some joints are missing from the target skeleton it does not imply an issue since the solver will take care of the movement reconstruction. What is important, though, is to connect the most important joints which are: the root, the hips, the ankles, the shoulders, the wrists and heads. At the end of this process the source skeleton and the target one should be fully mapped.

The "Full Body IK" SOP node is responsible for computing the offsets between the joint-pairs defined in the previous step, in the rest poses defined with the "Rig Match Pose" SOP. In addition it uses the offsets and source's transforms to determine the transform target positions for the joints in the target skeleton. Furthermore, it moves or transforms the target skeleton's joint to the target positions. All these operations are done for each frame of the source animation. When this process is completed, by selecting the node the result of the retargeting is visible.

Finally, it is possible to view the source animation successfully transferred to the target character through to the "Joint Deform" SOP node. This node allows the visualization of the skinned target character reproducing the animation meant to be transferred.

Whenever dealing with AI generated animations (e.g the ones created in the previous section), it's likely not to have source animations combined with standard bind poses such as a T-pose. In this scenario the node tree just described could not be sufficient for an accurate retarget, because the system could struggle to build a good match between poses. For this reason, it is necessary to slightly modify the node tree previously explained. To address this issue two nodes in particular have to be inserted as shown in Figure 4.14.

The second "FBX Animation Import" node allows the import of the source skeleton's bind pose. It is important to specify inside the editor of the node that the file linked to it represents a rest pose. This node, together with the first "FBX Animation Import" SOP, has to be wired to the new "Rig Stash Pose" node. This node contains the state of a valid rest pose by cooking the input and cashing the geometry in a data parameter. [25]

When transferring an animation to one skeleton source to another, it can happen to have few artifacts: the most common one is to have "sloped" shoulders on the target skeleton. This issue though is improvable, just by adding few settings inside the "Map Points" SOP, or by adding another node soon after that one. It's common practice to separate single operation as much as possible, so let's consider adding another node: a "Configure Joint" SOP, that will explicit the target joints offsets. Inside the configuration panel a new joint configuration group must be created. At this point the "Rotation weights" parameter has to be modified basing on the result desired. This parameter adds rotation limits to the joints part of the group on the target skeleton.

If the result is still not satisfying, it's possible to add another node in the tree node,

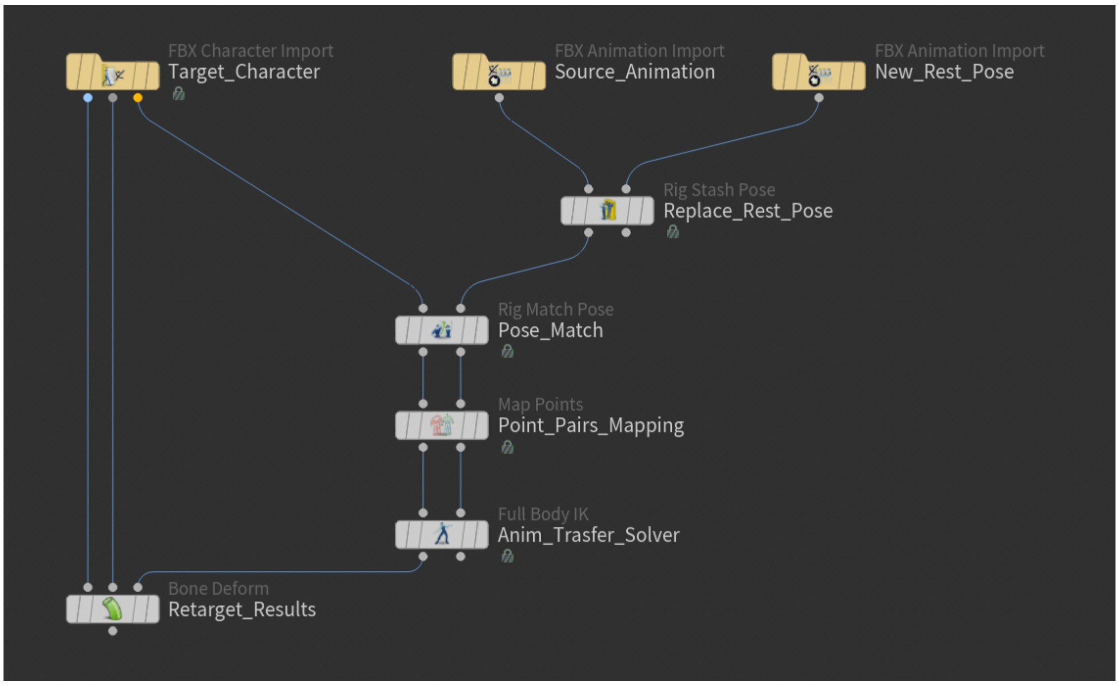


Figure 4.14: KineFX retargeting workflow with custom bind pose

called "Rig Pose". This operator allows to modify manually the position, rotation or translation of the joins selected. This tool, for example, can be extremely useful if mesh penetration happens during the animation. Each parameter can obviously be animated with keyframes for a more precise result.

4.5.2 Animation clean-up and Loop extraction

The animation clean-up phase comes before the retargeting one. The purpose of this process is to make the raw animation, as smooth as possible by eliminating the jittering, and fix the foot-sliding within the bounds of possibility. In order to organize the work better, it's convenient to operate inside a separate geometry node.

The first passage here, is to import the raw animation from the disk, and convert the animation into *motionclip* format, through the "Motionclip" SOP. This node takes as input a skeleton animation and evaluates and stores each frame of the input animation using the given frame range and sample rate. The animation gets converted into a list of packed primitives, where the first one contains the skeleton topology evaluated at the given rest frame and the rest of them represent a cached frame, containing a list of points that constitute the position of each joint. [26] Inside the motionclip panel, in order to help out the animation loop extraction, it is convenient to select the frame range of the animation that is meant to be looped.

Once the animation is successfully converted to a "motionclip", the animation is ready to be looped. There are two ways that allow loop creation. The easier one, is to re-convert the "motionclip" back to its original format through the "MotionClip Evaluate" node. Inside this node, by selecting "Loop" inside the "End behavior" parameter, the clip will play an infinite loop of the frame range selected. No controls about the blending mode for the reproduction can be selected though. Therefore this kind of solution works well in the only case where the start frame and end the end frame of the animation are identical. This scenario however, is impossible to have whenever handling with MoCap data. In fact, for efficiently addressing this issue, there is the need to add another node to the workflow. The node "MotionClip Cycle" takes as input a motionclip and a frame range, and produces a clip containing the specified number of repetitions, blending between each of them. Inside this node the method of blending must be specified:

- *Preserve length*, keeps the total length of each cycle the same as the length of the input motionclip. Blends the end of the cycle to be the first frame of the next one;

- *Overlap sequences*, overlaps the end of end cycle with the beginning of the previous one;

- *Insert Blend Region*, inserts a region between the cycle where blending is done.

The node also offers the chance to enable the feature "Blend Start to Ends" which enables blending from the end of the last cycle to the start of the first, allowing for seamless looping of the clip. Also, it is possible to choose between the different blend types: linear, ease-in, ease-out or ease-in ease-out. [27] For the animation used in the *Stadium Crowd Project*, I composed the loop just by reproducing two entire cycles, in order to keep the animation file light. This animation will be in fact played in loop whenever converted into an agent clip, in the next phases.

After converting the animation, it's time to work on fixing the foot-sliding. Note that this process doesn't always work: it is fundamental to have a raw animation that appears as more stable as possible within the feet movement. Through the "Stabilize Joint" node, specific joints can be selected, and locked in their initial position. This process though can cause some mesh distortion, so it is extremely important to analyze the specific case scenario and find the best locking values in order to reach a realistic movement. Afterward, the whole animation is ready to be smoothed out through the "Smooth Motion" SOP node, which allows to smooth out unwanted noise in a series of point attributes by applying a "Butterworth Filter", to remove frequencies that exceed a specific cut-off frequency. [28] Note that if the cut-off frequency is set too high, the whole motion will be ruined.

Once the animation is completed, the "Motion Evaluate" node can be inserted if the "motionclip" is meant to be transformed into "traditional" animation again. Otherwise, it's more convenient to cache the animation as a "motionclip", as it will be easily read during the agent set-up phase. In the end an optional "Rig Pose" node could be inserted, in order to fix some basic mesh penetration. Note that this issue could be addressed directly during the retargeting phase. The node tree built for this phase can be viewed in the following image (Figure 4.15).

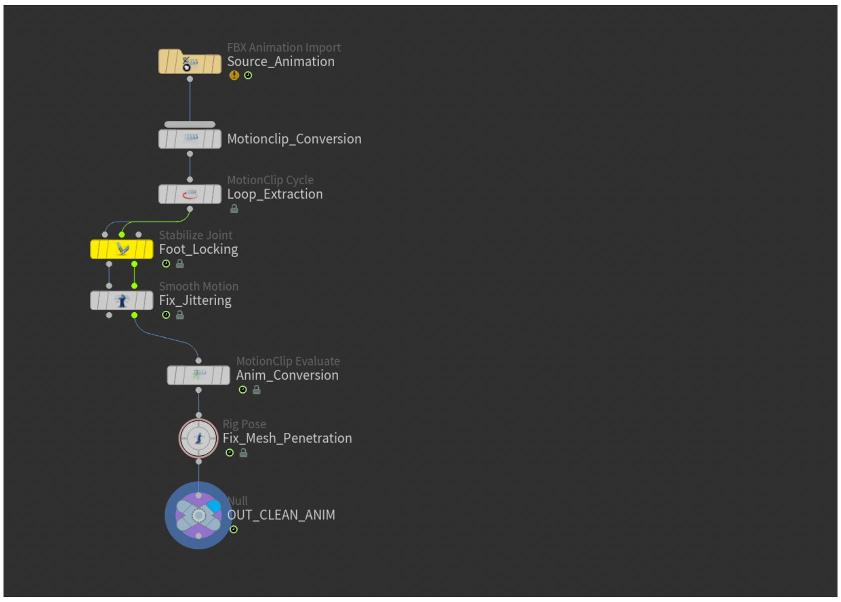


Figure 4.15: Animation clean up and loop extraction workflow

4.5.3 Agents and Crowd Set-up

Before setting up the project, it was fundamental to select out the best animations obtained during the tests previously analyzed, relatively to each different tool. I chose the best animations that came out from each tool, that, after the processing and the retargeting phase, didn't present any visual artifacts. The majority of them were obtained through *Rokoko Vision* as it was the tool that best handled foot sliding which was the hardest problem to address in post processing. Specifically, 23 animations were produced through *Rokoko Vision*, 4 animations through *Meshcapade* and just one with *MoveOne*. Moreover 2 new synthesized animations variations were produced through *GenMM* using the *Meshcapade*'s ones as sample motions.

The agent set-up process was pretty simple and replicated precisely what described in chapter 3. Each animation, after being correctly processed and retargeted to both the man and the woman character, was associated as agent clip to the correspondent agent. Extremely important in this phase, was to use a correct name convention for the clips, that should contain the agent associated to it (e.g woman or man), the class of animation (e.g sit, stand), the type of movement (e.g idle, cheer, jump) and the version. At the end of the agent set-up process, each agent was then cached to disk, in order to avoid long "cooking" processes in the next phases.

The crowd set-up phase was crucial in order to have a first look on the overall behavior of the crowd. In the crowd source node, particular attention was given to the "randomize" panel. The latter was crucial to set-up, in order for the crowd behave naturally. The following actions were done:

- Randomization of the agents' primitives, with different weights. A different amount of woman and men were scattered to form the crowd.
- Randomization of the agent clips, with different weights. In order for the crowd to look natural, it's important that all agents are characterized by a different movement.
- Randomization of the initial state. Each agent starts to reproduce the clip associated to it from a different frame. Here the seed was modified until the result wasn't satisfying.
- Randomization of the clip time, by applying random shifts for each agent clip.

The agents, in this phase, were not yet scattered along the stadium mesh; but a custom simpler scheme was used in this preliminary phase to display a simulation of a small section of the stadium seats.

4.5.4 Agents scattering and Camera views definition

The agent scattering phase presented significant challenges. The objective of this phase was to distribute points where each agent would be placed. Before proceeding, a thorough analysis of the stadium's geometric topology was essential.

Despite the suboptimal geometry, a procedural workflow was successfully developed. The initial steps involved grouping all primitive faces with normals oriented along the y-axis, as *Houdini* designates the y-axis as the upward direction. The stadium featured two tiers of seats, and based on this structure, it was divided into seven distinct areas. Each area was identified using a specific naming convention that indicated its location within the stadium. The same workflow was applied to each section (Figure 4.16).

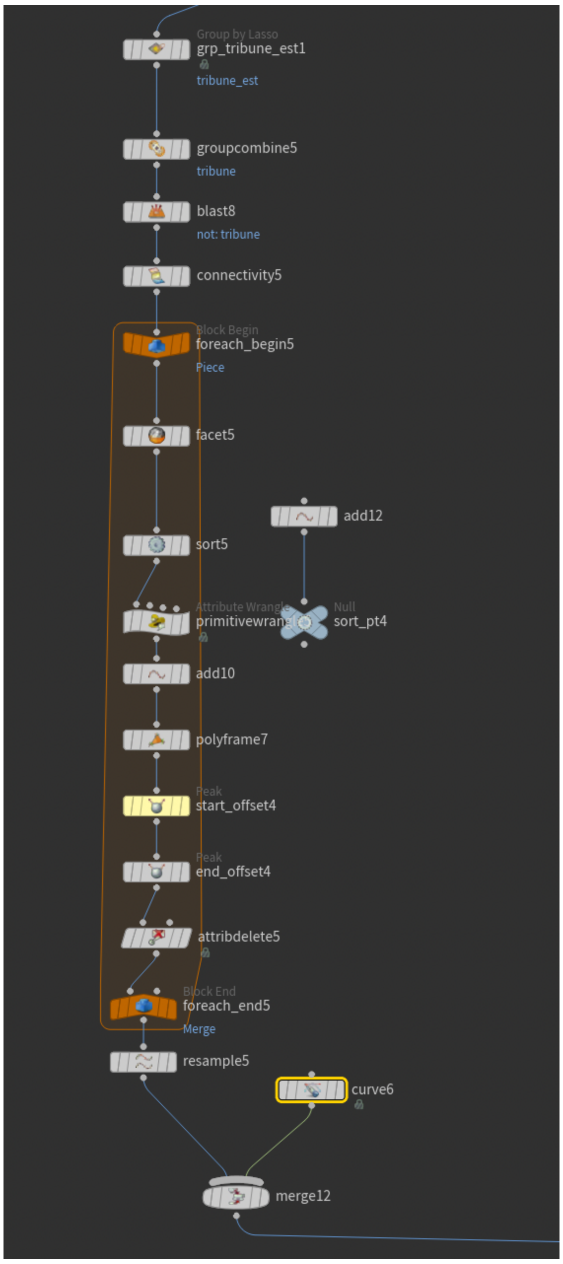


Figure 4.16: Agent scattering workflow

This procedural workflow enabled the selection of upward-facing faces for each stadium area using the "Group by Lasso" node. From each selected face, a set of points was extracted and then sorted based on a specified reference point. Subsequent steps focused on refining the scattered points to ensure no unnecessary points were included. Finally, after completing all the steps shown in Figure 4.16, it was essential to define a facing direction for each selected point. To ensure that the agents were oriented toward the field, an "Attribute Wrangle" node was used to assign the appropriate facing direction. The VEX code responsible for this operation is as follows:

```
vector primuv;
int primnum;
float dist = xyzdist(1, @P, primnum, primuv);
vector closestPos = primuv(1, "P", primnum, primuv);
v@N = normalize(closestPos-@P)*set(1,0,1);
```

For the stadium areas where this workflow could not be applied, the lines along which the points needed to be scattered were drawn manually. The final result can be observed in Figure 4.17.

Regarding the camera set-up, the process was fairly straightforward. Before positioning the cameras, a brief storyboard was created to provide an initial preview of the sequence. Once the storyboard was defined, the cameras were placed within the 3D space and animated to achieve the desired movement.

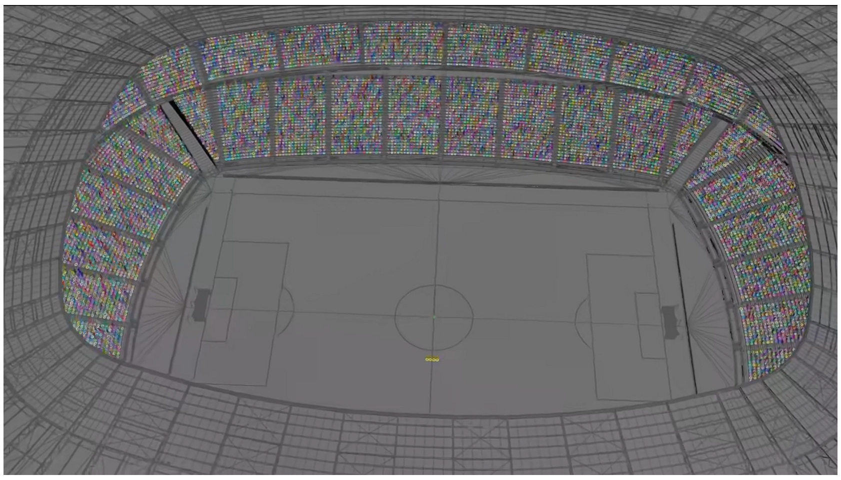


Figure 4.17: Agents scattering preview

4.6 Rendering

The rendering phase was particularly challenging due to several factors:

- The large amount of data to process, with approximately 50,000 agents scattered;
- The need to run renders locally.

Despite these challenges, a straightforward rendering workflow was implemented in the *Solaris* environment. This node tree (Figure 4.18) efficiently imported the cameras, the stadium geometry, and the specific portion of agents to be rendered, set-up the lighting and defined the right settings for the renders. To ensure the rendering process could be handled locally, the crowd agents were segmented into multiple groups.

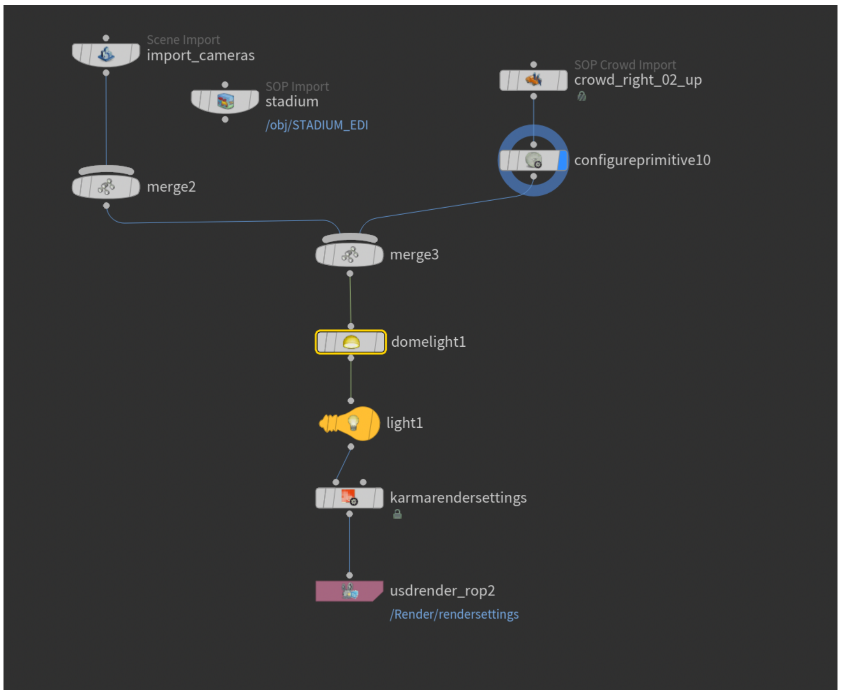


Figure 4.18: Rendering workflow

Despite these precautions, it was not possible to complete a full-quality render within the time available. As a result, the final output is presented as a low-quality rendering preview, captured using *Houdini*'s "Flipbooks".

Chapter 5

Evaluation of results

5.1 The proposed workflow

The present research aimed to develop an AI-powered workflow integrating various tools to optimize, refine, and expedite the data collection process essential for synthetic human-crowd projects within the Visual Effects industry.

To construct the stadium crowd sequence, extensively detailed in the preceding chapter, I systematically examined and finalized the phases that define this workflow. Moreover, the evaluation of the different tools employed (i.e. *Meshcapade*, *MoveAI*, *RokokoVision* and *GenMM*) provided critical insights, enabling me to make informed decisions regarding the most suitable tools for a VFX company. These selections were based not only on the quality of the results but also on the financial investment required. Additionally, it was imperative to design a workflow that seamlessly integrates into the existing pipeline, ensuring accessibility for future users. By taking these factors into account, I successfully formulated a comprehensive workflow proposal that addresses the key challenges and requirements identified in collaboration with *EDI* during the preliminary phase. A schematic representation of this proposed workflow is presented in the following figure (Figure 5.1).

This workflow represents the culmination of extensive testing conducted throughout the research process. The structured sequence depicted in the accompanying figure has been meticulously designed to automate, to the greatest extent possible, each phase involved in generating the necessary data for human-crowd projects. Consequently, every stage—ranging from mesh generation and auto-rigging to animation, has been integrated into the workflow. However, the application of this workflow remains flexible, allowing it to be adapted based on the specific requirements of a given shot. For instance, the initial phase, "Mesh Generation", may be omitted if the project necessitates highly detailed agent primitives, as

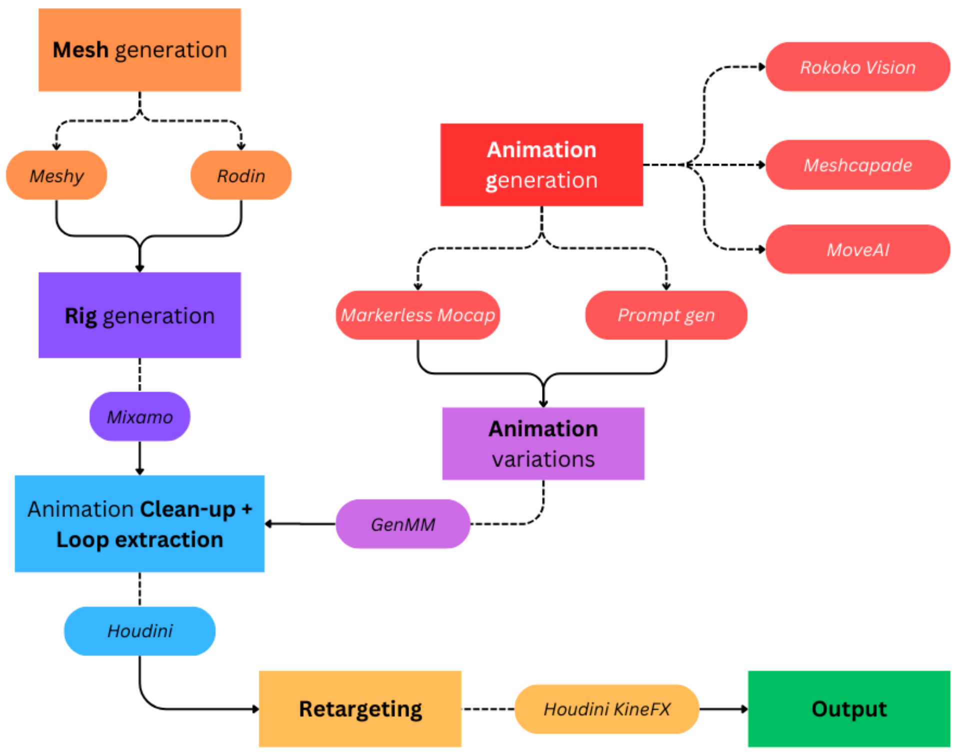


Figure 5.1: AI-powered workflow proposed

current *image-to-3D* tools lack the precision required for close-up shots. Conversely, when creating a standard crowd scene, such as in the stadium crowd project examined in the present research, the focus can shift directly to the animation phase. Additionally, if a scene features a distant group of people ultimately obscured by a bokeh effect, the entire workflow may be utilized without modification.

The schematic representation of the workflow highlights seven key operational stages:

1. Mesh Generation – This phase employs *image-to-3D* or *text-to-3D* algorithms to create digital character models. Achieving satisfactory results necessitates the use of high-quality reference images, with subjects posed in a standardized rest position, such as a T-pose. This approach ensures greater ease in the subsequent rigging process.

2. Rig Generation – This step involves rigging the synthesized character produced in the previous phase. While no AI-based rigging tool was thoroughly explored

in this study, *Mixamo* was identified as a widely used auto-rigging solution. This tool facilitates the creation of a basic human skeleton by manually positioning markers at key anatomical joints — such as the chin, wrists, elbows, knees, and groin — enabling full auto-rigging within minutes.

3. Animation Generation – As the most critical phase of the workflow, this step focuses on generating motion data to be applied to digital agents in the crowd. AI-powered animation tools, including those provided by platforms such as *Rokoko Vision* and *Meshcapade*, were utilized for this purpose. Motion sequences can be obtained through Markerless Motion Capture techniques or generated via text-based prompts.

4. Animation Variations – This phase introduces procedural motion diversity using *GenMM*, which enables the generation of an extensive range of motion variations from a single animation sequence. Additionally, infinite animation loops can be created to enhance realism and adaptability in crowd simulations.

5. Animation Clean-up – A crucial phase in which synthetic AI-generated animation data undergoes refinement to optimize its quality for retargeting. At this stage, human intervention is essential to ensure a high level of accuracy and detail. The cleaner and more precise the animation data, the more refined and realistic the final output will be.

6. Animation Retargeting – In this step, AI-generated animation is transferred onto the target character's skeleton. Although this process does not inherently involve AI, the use of procedural techniques — such as the ones offered by *Houdini KineFX* tools described in the previous chapter — can significantly enhance efficiency and reduce manual effort.

7. Output – Once all phases have been successfully completed, the animated character is finalized and designated as an agent, forming part of the digital human crowd.

5.1.1 Quantitative Analysis

This section evaluates the output of the motion capture animations generated using the AI tools applied in the previous chapter. Specifically, two AI tools are analyzed: *Rokoko Vision* and *Meshcapade*. *MoveAI* is excluded from this evaluation, as *EDI* had already decided against integrating it into their pipeline due to its excessive cost.

Foot-locking analysis A key objective of this evaluation is to assess the performance of these tools with a particular focus on one of the most problematic artifacts: foot-locking. As highlighted in Chapter 4, both *Rokoko Vision* and *Meshcapade* exhibited noticeable foot-sliding in their output animations. To determine which tool performs better, it is crucial to analyze their handling of feet movement and its impact on animation quality.

To conduct this evaluation, two distinct animations were considered for each tool: a cheer motion and an idle motion. Both animations share a critical characteristic — the feet remained fully locked to the ground throughout the capture process. To quantify the error introduced by each tool, the positions of the feet and toes joints were tracked and analyzed (data can be visualized into Appendix A and Appendix B). These data points were then visualized in a line graph. The closer the graph approximates a straight line, the cleaner the output and the less foot-sliding is present.

The first tool analyzed is *Meshcapade*. This motion capture system utilizes single-camera tracking, which suggests that its accuracy may be lower compared to *Rokoko Vision*, which operates with a two-camera tracking system. To assess the tool's performance, feet positions were recorded at specific frames: 1, 24, 48, 72, 96, 120 and 144 (if the animation was long enough), corresponding to a frame rate of 24 fps. These data points were compiled into a table and subsequently visualized in graphs, illustrating the feet joint positions for both the cheer and idle animations. The following figures (Figure 5.2 and Figure 5.3) present these graphical representations, providing insight into the extent of foot-sliding and overall animation accuracy.
As expected, foot-locking is a noticeable issue in both animations. The lines representing the x, y, and z positions of the feet joints deviate significantly from a straight trajectory, indicating a low level of accuracy in *Meshcapade*'s feet-movement reconstruction. Due to the poor quality of these results, animations generated with this tool would not be suitable for use in a production environment. The extent of foot-sliding observed is too pronounced to be corrected in subsequent refinement stages, making the output impractical for professional applications.

Next, the performance of *Rokoko Vision*'s AI-based motion capture system is analyzed. This system utilizes a dual-camera set-up, which is expected to provide more accurate motion reconstruction compared to single-camera solutions. Although *Rokoko Vision* offers also a single-camera tracking, the present analysis evaluates only the dual-camera set-up, as it is the configuration that *EDI* would to use for its projects, given its expected superior performance. In this case, the animations were generated at a frame rate of 30 fps. Consequently, feet joint positions were recorded at frames 30, 60, 90, 120, and 150 (for animations of sufficient length). The results of this analysis are presented in Figures 5.4 and 5.5, where the reconstructed feet movement can be observed and evaluated.

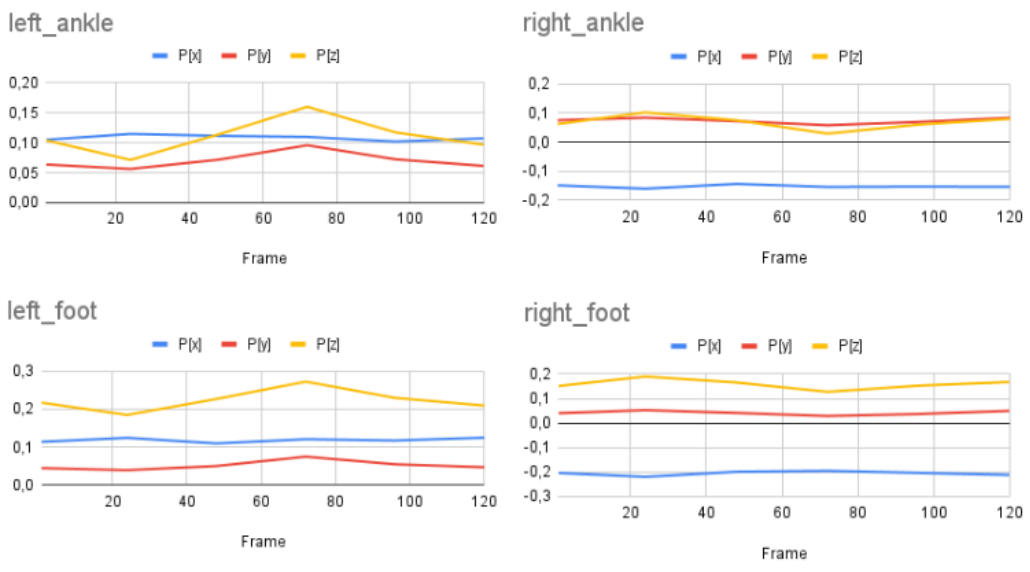


Figure 5.2: Idle animation - Feet movement - Meshcapade

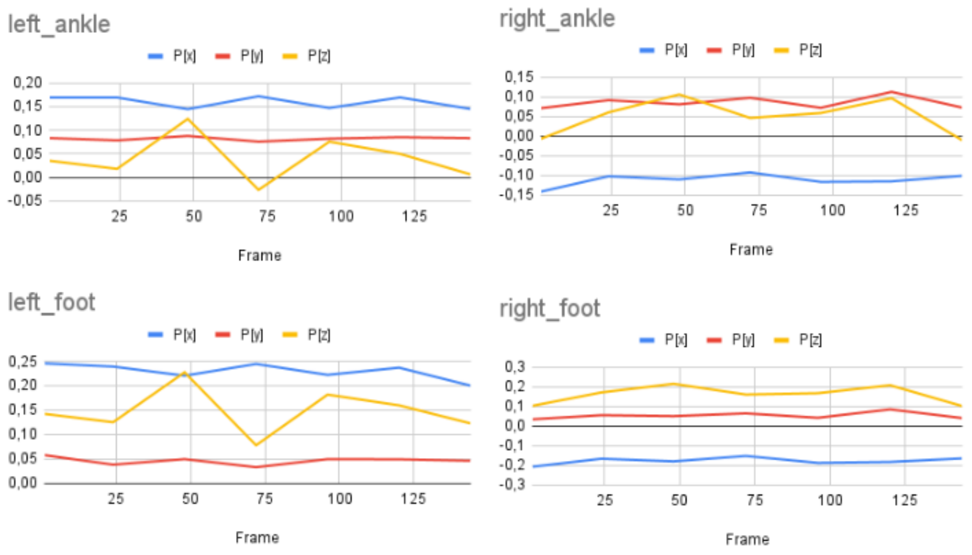


Figure 5.3: Cheer animation - Feet movement - Meshcapade

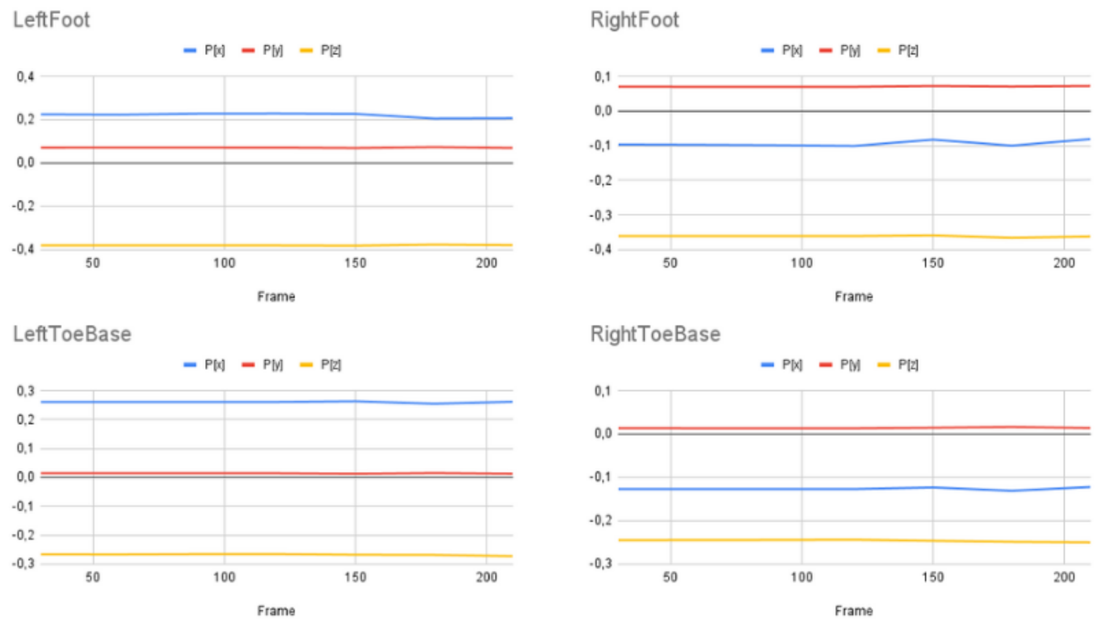


Figure 5.4: Idle animation - Feet movement - Rokoko Vision

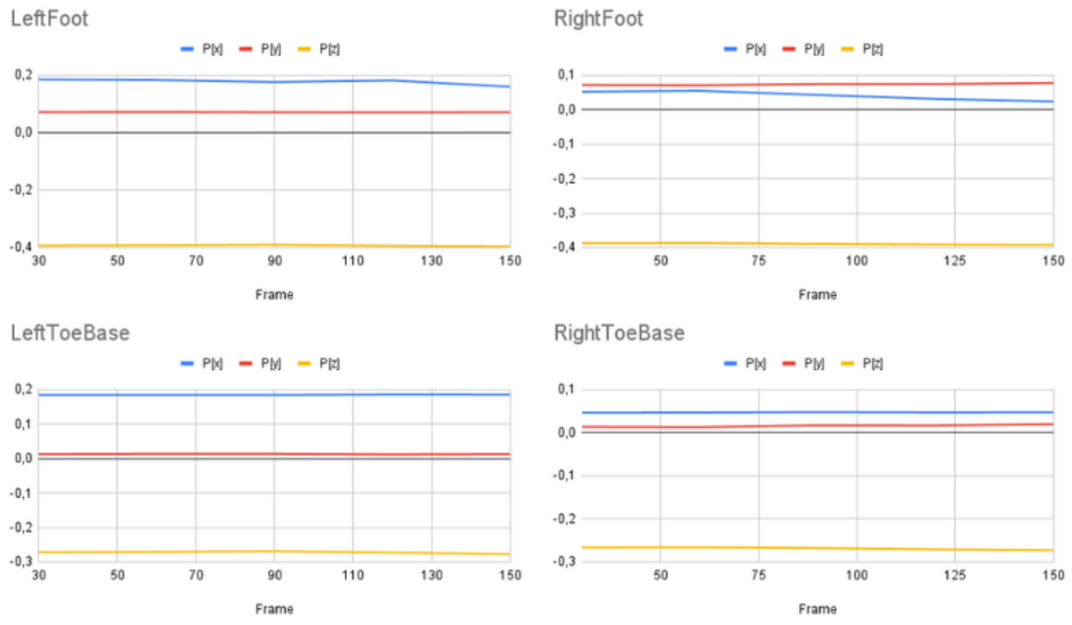


Figure 5.5: Cheer animation - Feet movement - Rokoko Vision

In contrast to the previous case, the curves in this analysis closely approximate a straight-line trajectory, indicating that the error introduced by *Rokoko Vision*'s algorithm is minimal. These results demonstrate that *Rokoko Vision* more accurately tracks and reconstructs feet movement, leading to only minor foot-sliding, which can be easily addressed during the post-processing phase. Due to this superior performance, *EDI* selected *Rokoko Vision* as the preferred tool for integration into their animation pipeline. It will be primarily used for generating simple animations that do not involve locomotion.

Time effort analysis Another crucial aspect to consider in the present section is the time required to produce an animation. This analysis compares the time spent to create two animations of the same type using two different methods: the traditional keyframing technique and the AI-driven approach proposed in this research.

To ensure a fair comparison, two types of animations are considered:

1. Idle Movement: A low-dynamic animation depicting a relaxed character action. The character looks around by rotating their head while shifting their weight between one leg and the other. To achieve a natural-looking motion, it is necessary to incorporate micro-movements across all body joints.

2. Cheer Movement: A highly dynamic animation involving a combination of multiple actions, such as raising the arms alternately up and down in a celebratory gesture, clapping, and continuous jumping.

For this analysis, it is assumed that the animations are created at 30 fps, with a total duration of approximately 30 seconds (i.e. 900 frames). Additionally, the animations are considered to be loopable, allowing for immediate integration into a potential crowd project.

The AI-driven animation process follows the workflow outlined in the previous section. Specifically, the animation is generated using a markerless motion capture system, followed by animation clean-up, loop extraction, and subsequent retargeting onto a production-ready skeleton. The evaluation focuses on the processing times associated with *Rokoko Vision* for consistency. The key stages for time assessment in this methodology are as follows:

- Hardware set-up time for recording the animation
- Data processing time
- Preliminary animation clean-up within *Rokoko Studio*
- Animation clean-up and loop extraction using a procedural workflow in *SideFX Houdini*

- Retargeting phase using a procedural workflow based on *Houdini KineFX* tools

The traditional process, on the other hand, exclusively considers the time required to produce the animation using the traditional keyframing method. The time calculation also includes the additional time needed for refining the animation after it has been reviewed by project supervisors.

Idle animation According to the traditional method, the necessary time to produce an animation of this kind is resumed in the following table (Table 5.1).

Operation	**Time Estimation**
Study of references	1 h
Main Poses definition	2 h
Preliminary animation	5 h
Polished animation	8 h
Total Time	16 h

Table 5.1: Idle animation - Time estimation table - Traditional method

The AI-powered method on the other hand is characterized by the following timing (Table 5.2).

Operation	**Time Estimation**
Study of references	1 h
Hardware set-up	10 min
Motion Capture	15 min
Data Processing	5 min
Preliminary clean-up	10 min
Animation post-processing	1 h
Retargeting	20 min
Total Time	3 h

Table 5.2: Idle animation - Time estimation table - AI powered method

As can be clearly observed, the time required when using the second method is significantly lower than that of the first. Specifically, it results in a time savings of 13 hours—equivalent to one and a half working days. This reduction would lead to a substantial acceleration in the overall production process, enhancing efficiency and productivity.

Cheer animation Similarly, the tables below (Tables 5.3 and 5.4) provide estimates of the processes involved in creating this type of animation, which, compared to the previous one, presents a higher level of complexity.

Operation	**Time Estimation**
Study of references	1 h
Main Poses definition	3 h
Preliminary animation	12 h
Polished animation	8 h
Total Time	24 h

Table 5.3: Cheer animation - Time estimation table - Traditional method

In this case, it is evident that the time required for animation creation increases. As animations become more complex, it is reasonable to expect that the production process will extend by at least one additional workday. Furthermore, refining such animations is a tedious process that demands an exceptionally high level of detail. Moreover, the overall quality of the animation heavily depends on the animator's skill in incorporating subtle micro-movements that characterize human motion, adding a significant level of realism. This is particularly crucial in cases where the animation consists of a combination of multiple actions. Additionally, the creation of seamless loops should not be underestimated, as it often proves to be highly challenging.

Operation	**Time Estimation**
Study of references	1 h
Hardware set-up	10 min
Motion Capture	15 min
Data Processing	5 min
Preliminary clean-up	10 min
Animation post-processing	3 h
Retargeting	20 min
Total Time	5 h

Table 5.4: Cheer animation - Time estimation table - AI powered method

As shown in Table 5.4, it is evident that the time required to create the cheer animation is not significantly different from that needed for the idle animation. The only noticeable increase is in the post-processing phase, which takes approximately two additional hours. This is due to the greater number of actions involved in the cheer animation, requiring a higher level of attention in fixing overall artifacts - in these cases in fact, mesh penetration could be challenging to handle. However, the time spent on the remaining phases remains unchanged. This workflow offers substantial advantages in terms of production efficiency, leading to a significant boost in overall productivity.

It can therefore be concluded that the proposed workflow brings considerable benefits in terms of production time. A comparison of the two examples reveals an approximately 80% reduction in animation time. This is because the presented workflow is largely independent of the type of source animation, making it easily applicable to various case scenarios. From a VFX studio perspective, this not only simplifies the work of the animation department but also enhances the production team's ability to accurately estimate work schedules, ultimately improving project planning and management.

5.1.2 Qualitative evaluation

The qualitative evaluation is based on the considerations made by *EDI*.

In selecting an AI-driven tool for integrating Markerless MoCap into their pipeline, the decision was guided by three key factors:

- The quality of the results,
- The efficiency of animation production,
- The overall cost.

Taking into account the findings from the quantitative evaluation, *Rokoko Vision* emerged as the optimal compromise between timing, cost, and quality. This tool demonstrated strong performance in generating simple animations (with no locomotion) and was recognized as a valuable accelerator for the animation department. Additionally, the workflow for refining raw data was well received, particularly the ability to extract motion loops efficiently and cleaning-up the animation from all the jittering introduced by the tracking system. The system proved effective in mitigating artifacts in raw animations without excessive time or effort.

Similarly, the retargeting workflow developed using *Houdini*'s *KineFX* tools was well regarded, as it allowed for seamless application across various scenarios (e.g skeletons with different proportions and sizes).

Regarding the use of *GenMM*, *EDI* acknowledged its strengths while also recognizing its current limitations. While the tool delivered promising results, it presented certain weaknesses that require further refinement, such as the random root-movement found in some tests. Moreover, *GenMM* is not universally applicable, as its effectiveness is not guaranteed in all scenarios. A significant drawback is its compatibility constraint, as it currently operates exclusively on a specific version of *Blender*, which is not aligned with *EDI*'s existing pipeline. This limitation ultimately discouraged the company from incorporating it into their pipeline. However, despite these criticisms, the team appreciated the results achieved through its use.

The workflow outlined at the beginning of this chapter also introduced an AI-driven approach for generating 3D-assets from reference images. In this context, two tools were tested: *Meshy* and *Rodin*. Both demonstrated satisfactory performance, but *Rodin* stood out due to two key features: the ability to generate models from multiple viewpoints of the same subject and its high degree of customization, which allows for precise control over the final output. As a result, *EDI* opted to subscribe to *Rodin* and integrate it into their modeling pipeline, recognizing its potential to significantly accelerate and enhance the 3D-asset creation process.

In conclusion, the proposed workflow, with all its phases, was highly valued by the company as it successfully addressed the challenges identified during the preliminary phase. With this workflow, *EDI* can now efficiently generate an extensive range of high-quality looped animations in a short time-frame, retarget them to different skeletons without concerns regarding size and proportion variations, and even produce 3D-assets with acceptable topology, ready for texturing and shading. This workflow is particularly beneficial, for example in projects with tight deadlines and moderate detail requirements, such as large-scale crowd scenes in stadiums or concerts.

Chapter 6

Conclusions

6.1 AI in Visual Effects: Challenges and Future Prospects

The present thesis has enabled an in-depth exploration of the role of Artificial Intelligence in production environments, particularly its integration into the Visual Effects industry. The results obtained highlight the immense potential of these algorithms while also underscoring their inherent limitations. Although Artificial Intelligence and Machine Learning can serve as powerful accelerators for various processes, the current level of automation achievable in this industry remains insufficient to fully replace traditional workflows. This research has demonstrated that while AI can significantly enhance processes such as animation, it still requires substantial human intervention to meet the quality standards demanded by the industry.

Moved by these reasons, the present study has proposed an integration strategy that combines these emerging tools with established methodologies, which I consider to be the key to achieving an optimal balance between innovation and tradition. Advanced testing has further confirmed that the effective use of these technologies must still be guided by skilled professionals with extensive expertise and experience in the field.

Additionally, the present thesis has specifically examined the application of AI tools in a well-defined case study: the creation of digital human stadium crowd. However, it also leaves room for future research and potential applications in other domains, such as projects that prioritize a higher level of detail over the need to generate large amounts of data in a short period. As AI technologies continue to evolve, new approaches could emerge to facilitate the automation of more complex animation tasks while preserving the artistic control essential to high-end productions.

Following the same trajectory that has shaped advancements in 3D modeling over the past few years, the field of animation continues to evolve with the integration of AI-driven tools, which act both as accelerators and facilitators of the creative process. Ongoing research consistently introduces novel approaches and methodologies to address existing challenges and expand the possibilities of AI-assisted animation. A particularly promising extension of the workflow proposed in this study involves exploring the potential of Motion In-betweening. An innovative study presented at SIGGRAPH 2024 by researchers Cohan, Tevet, Reda, Peng, and Van de Panne — "Flexible Motion In-betweening with Diffusion Models" — introduced a unified model capable of generating precise and diverse motion sequences while accommodating a broad range of user-specified constraints. [29] This approach could represent a significant advancement in animation synthesis, offering an alternative solution to the challenge of efficiently producing varied animations from a minimal set of key poses. By leveraging diffusion models, this method could enhance the fluidity and adaptability of AI-generated animations, making it a valuable tool for both automated and artist-guided workflows.

Additionally, future research could focus on refining AI-driven tools to further enhance traditional animation techniques. For instance, the implementation of advanced motion filters — such as those found in *Cascadeur* [30] — could improve the accuracy and realism of keyframing, reducing the manual effort required for secondary motion and dynamic adjustments.

Another promising research direction involves the further integration of AI into procedural animation techniques. By leveraging AI-driven physics simulations and reinforcement learning, future tools could generate highly realistic character motions in real time, adapting dynamically to different environments and interactions. This could significantly improve areas such as crowd simulation, where AI-driven agents could respond intelligently to obstacles, terrain variations, and other characters in a scene. Moreover, AI-powered physics simulations could streamline the creation of secondary motion, such as muscle deformations, cloth dynamics, and hair simulations, reducing the need for manual adjustments.

In addition, further performance improvements in the tools tested in the present study would be highly beneficial. One key area of interest would be developing solutions to fully eliminate the foot-sliding artifacts introduced by markerless motion capture tracking algorithms, potentially through further optimization of the underlying methodologies. Another avenue for improvement involves enhancing the efficiency of AI-generated locomotion animations, which were beyond the scope of this research but still highly utilized in crowd scenes. Furthermore, AI-driven 3D model generation algorithms, whether from single images or multiple images, still could be optimized. While these systems have already reached a high level of efficiency, the refinement of output files remains necessary. A particularly compelling research direction would be the development of high-resolution texture generation techniques that can be seamlessly projected onto low-poly models, further enhancing their visual fidelity.

Finally, as AI-generated animations become more sophisticated, ethical considerations will also need to be addressed. The use of AI in VFX production raises important questions regarding authorship, artistic integrity, and job displacement within the industry. Future studies could explore the development of ethical frameworks and guidelines to ensure that AI tools serve as creative enhancers rather than replacements for skilled artists. Additionally, AI could play a role in developing new accessibility solutions within animation, such as real-time motion adaptation for inclusive character performances, enabling greater representation of diverse physical abilities.

With these future perspectives in mind, the integration of AI into VFX traditional processes, presents both exciting opportunities and critical challenges. While AI has the potential to revolutionize workflows that guide this industry, its success will ultimately depend on how well it can be harmonized with human creativity, industry standards, and evolving technological advancements.

Appendix A

Rokoko Vision Data

The present section presents the tables containing the data utilized for the quantitative analysis, specifically detailing the positions of the feet joints throughout the analyzed animations made with *Rokoko Vision*. This data was subsequently mapped into the graphs outlined in Chapter 5. The tables provide the positions along the x, y and z axes, based on the *Houdini* coordinate system, and were extracted through the Geometry Spreadsheet. All measurements are expressed in meters which is *Houdini*'s fundamental unit of measurement.

Tables A.1, A.2, A.3, A.4 represent the data from the Cheering animation. Tables A.5, A.6, A.7 and A.8 represent the data from the Idle animation.

Frame	**P[x]**	**P[y]**	**P[z]**
30	0,184239	0,0706726	-0,392309
60	0,182304	0,0711512	-0,391396
90	0,175492	0,0701722	-0,389514
120	0,180934	0,0695289	-0,393558
150	0,159296	0,0702192	-0,395896

Table A.1: Left Foot positioning, cheering animation

Frame	**P[x]**	**P[y]**	**P[z]**
30	0,183772	0,0123784	-0,271338
60	0,183772	0,0131527	-0,270298
90	0,183772	0,0131085	-0,268398
120	0,185056	0,0116441	-0,27246
150	0,184795	0,0124724	-0,277374

Table A.2: Left Toe Base positioning, cheering animation

Frame	**P[x]**	**P[y]**	**P[z]**
30	0,0515763	0,0711173	-0,387197
60	0,0543024	0,0703747	-0,386662
90	0,0423019	0,0737804	-0,389234
120	0,0306791	0,0737174	-0,390957
150	0,0234709	0,077015	-0,391855

Table A.3: Right Foot positioning, cheering animation

Frame	**P[x]**	**P[y]**	**P[z]**
30	0,0460803	0,0131412	-0,266197
60	0,0464704	0,0124155	-0,265783
90	0,0474329	0,0169123	-0,267693
120	0,0467885	0,016582	-0,270506
150	0,0470366	0,0197913	-0,27268

Table A.4: Right Toe Base positioning, cheering animation

Frame	**P[x]**	**P[y]**	**P[z]**
30	-0,0966757	0,0697978	-0,360809
60	-0,0976045	0,0696888	-0,360865
90	-0,0988818	0,0696428	-0,360869
120	-0,100917	0,0696058	-0,360827
150	-0,082871	0,0719783	-0,358785
180	-0,100111	0,0704108	-0,365981
210	-0,0812313	0,0720416	-0,361978

Table A.5: Right Foot positioning, idle animation

Frame	**P[x]**	**P[y]**	**P[z]**
30	-0,126935	0,0130238	-0,244774
60	-0,126933	0,0128802	-0,244608
90	-0,126975	0,0128387	-0,244304
120	-0,127063	0,0128007	-0,243811
150	-0,123438	0,0142334	-0,246433
180	-0,130919	0,016103	-0,248914
210	-0,12237	0,0135712	-0,250211

Table A.6: Right Toe Base positioning, idle animation

Frame	**P[x]**	**P[y]**	**P[z]**
30	0,224544	0,0707427	-0,380715
60	0,222758	0,0708723	-0,380539
90	0,227683	0,0708076	-0,380785
120	0,228345	0,0707358	-0,380837
150	0,226444	0,0687131	-0,381958
180	0,204976	0,0730558	-0,376731
210	0,206415	0,0689925	-0,379487

Table A.7: Left Foot positioning, idle animation

Frame	**P[x]**	**P[y]**	**P[z]**
30	0,260606	0,0142152	-0,266228
60	0,260559	0,0144277	-0,266573
90	0,260791	0,0143303	-0,265384
120	0,26084	0,0142522	-0,265265
150	0,263274	0,012406	-0,267607
180	0,254546	0,0149511	-0,268243
210	0,261603	0,0126259	-0,272808

Table A.8: Left Toe Base positioning, idle animation

Appendix B

Meshcapade Data

Similarly to the previous section, the present one reports the tables related to the animations made with *Meshcapade*.

Tables B.1, B.2, B.3, B.4 represent the data from the Cheering animation. Tables B.5, B.6, B.7 and B.8 represent the data from the Idle animation.

Frame	**P[x]**	**P[y]**	**P[z]**
1	0,168943	0,0832264	0,0352982
24	0,169038	0,078355	0,0180735
48	0,144368	0,0877645	0,123814
72	0,171627	0,0756027	-0,0263963
96	0,1469	0,0819373	0,0755087
120	0,169147	0,085107	0,0496953
144	0,144673	0,083118	0,00622036

Table B.1: Left Ankle positioning, cheering animation

Frame	**P[x]**	**P[y]**	**P[z]**
1	0,245715	0,0575723	0,14228
24	0,239214	0,0380129	0,125054
48	0,22049	0,0494835	0,227434
72	0,244539	0,0328165	0,0777679
96	0,222214	0,0494748	0,181673
120	0,236963	0,049141	0,159716
144	0,200102	0,0461195	0,122649

Table B.2: Left Foot positioning, cheering animation

Frame	**P[x]**	**P[y]**	**P[z]**
1	-0,141263	0,0707968	-0,0072585
24	-0,1021	0,0910598	0,0600768
48	-0,110034	0,0805074	0,105568
72	-0,0923976	0,0972085	0,0460155
96	-0,116409	0,0718248	0,058463
120	-0,115139	0,112171	0,0961673
144	-0,100848	0,0722913	-0,0109028

Table B.3: Right Ankle positioning, cheering animation

Frame	**P[x]**	**P[y]**	**P[z]**
1	-0,206901	0,0339756	0,101988
24	-0,166561	0,0546216	0,17015
48	-0,179922	0,0498957	0,214093
72	-0,152295	0,0637847	0,159568
96	-0,187576	0,0415646	0,166253
120	-0,182835	0,0842328	0,206782
144	-0,164858	0,0399955	0,100716

Table B.4: Right Foot positioning, cheering animation

Frame	**P[x]**	**P[y]**	**P[z]**
1	0,105158	0,0639644	0,10436
24	0,114962	0,0565168	0,0715815
48	0,111789	0,0719723	0,114255
72	0,109968	0,0960208	0,160268
96	0,102117	0,0725253	0,11737
120	0,10761	0,0615405	0,0968661

Table B.5: Left Ankle positioning, idle animation

Frame	**P[x]**	**P[y]**	**P[z]**
1	0,113875	0,0445408	0,217142
24	0,124402	0,0391814	0,184645
48	0,109632	0,0501156	0,226908
72	0,120759	0,0747741	0,272541
96	0,117317	0,0546771	0,229724
120	0,124799	0,0469948	0,20941

Table B.6: Left Foot positioning, idle animation

Frame	**P[x]**	**P[y]**	**P[z]**
1	-0,14932	0,0750803	0,0621201
24	-0,160748	0,0840715	0,102158
48	-0,144263	0,0719655	0,0750696
72	-0,154728	0,05807	0,0292542
96	-0,153448	0,069332	0,06028
120	-0,15443	0,0835654	0,0799256

Table B.7: Right Ankle positioning, idle animation

Frame	**P[x]**	**P[y]**	**P[z]**
1	-0,203248	0,0402803	0,151286
24	-0,219146	0,0528853	0,189833
48	-0,198594	0,0413451	0,165515
72	-0,19485	0,0290923	0,127337
96	-0,203045	0,037134	0,152872
120	-0,211424	0,0500027	0,167646

Table B.8: Left Foot positioning, idle animation

Bibliography

[1] Synapse Studio. *The growing role of AI in visual effects creation.* Accessed 14 January 2025. 2025. URL: https://synapsestudiovn.com/the-growing-role-of-ai-in-visual-effects-creation/#:~:text=Artificial%20Intelligence,%20particularly%20machine%20learning,into%20film%20and%20television%20productions (visited on 01/14/2025) (cit. on p. 1).

[2] Built In. *AI Basics.* Accessed 14 January 2025. 2025. URL: https://builtin.com/artificial-intelligence/ai-basics (visited on 01/14/2025) (cit. on p. 7).

[3] Chris McGrowan. *Rapid Evolution at the Intersection of AI and VFX.* Accessed 14 January 2025. 2023. URL: https://www.vfxvoice.com/rapid-evolution-at-the-intersection-of-ai-and-vfx/ (visited on 01/14/2025) (cit. on p. 9).

[4] Numalis Communication team. *The Magic of AI in VFX: Enhancing Visual Effects.* Accessed 20 January 2025. 2024. URL: https://numalis.com/the-magic-of-ai-in-vfx-enhancing-visual-effects/ (cit. on p. 10).

[5] Foundry. *Copycat: Machine Learning in Nuke.* Accessed: 2025-01-20. 2021. URL: https://www.foundry.com/insights/film-tv/copycat-machine-learning-nuke (cit. on p. 10).

[6] HQ VFX. *3D Assets Development for VFX Production.* Accessed: 2025-01-14. 2023. URL: https://hqvfx.com/3d-assets-development-for-vfx-production/ (cit. on p. 11).

[7] Jorge Cantón. *Generating 3D Assets Using AI.* Accessed: 2025-01-20. 2024. URL: https://www.plainconcepts.com/generating-3d-assets-using-ai/ (cit. on p. 12).

[8] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. «Structured 3D Latents for Scalable and Versatile 3D Generation». In: *arXiv preprint arXiv:2412.01506* (2024) (cit. on p. 12).

[9] Shay Pomeroy. *New 3D Models: Trellis 3D, Tripo AI, and Rodin.* Accessed: 2025-01-20. 2024. URL: https://www.layer.ai/blog/new-3d-models-trellis-3d-tripo-ai-rodin/ (cit. on p. 12).

[10] MLWires. *Triposr: Creates detailed 3D objects from single images in split seconds.* Accessed: January 8, 2025. 2024. URL: https://www.mlwires.com/triposr-creates-detailed-3d-objects-from-single-images-in-split-seconds/ (cit. on p. 14).

[11] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. *InstantMesh: Efficient 3D Mesh Generation from a Single Image with Sparse-view Large Reconstruction Models.* 2024. arXiv: 2404.07191 [cs.CV]. URL: https://arxiv.org/abs/2404.07191 (cit. on p. 15).

[12] Mark Boss, Zixuan Huang, Aaryaman Vasishta, and Varun Jampani. *SF3D: Stable Fast 3D Mesh Reconstruction with UV-unwrapping and Illumination Disentanglement.* 2024. arXiv: 2408.00653 [cs.CV]. URL: https://arxiv.org/abs/2408.00653 (cit. on p. 17).

[13] Jorgemagic. *DonatelloAI.* Accessed: January 8, 2025. 2024. URL: https://github.com/Jorgemagic/DonatelloAI (cit. on p. 17).

[14] EJ HASSENFRATZ. *What is 3D Rigging for Animation?* Accessed: January 15, 2025. N.D. URL: https://www.schoolofmotion.com/blog/what-is-3d-rigging-for-animation (cit. on p. 19).

[15] Frank Govaere. *6 Animation Techniques.* Accessed: January 15, 2025. 2024. URL: https://www.linkedin.com/pulse/6-animation-techniques-frank-govaere-xyhaf (cit. on pp. 20, 21).

[16] M. Caroline Wibowo, S. Nugroho, and A. Wibowo. «The use of motion capture technology in 3d animation». In: *International Journal of Computing and Digital Systems* 15 (1 2024), pp. 975–987. DOI: 10.12785/ijcds//150169 (cit. on p. 25).

[17] Deborah R. Fowler. *History of Crowds.* Accessed: January 21, 2025. 2017. URL: https://www.deborahrfowler.com/CrowdsResources/CrowdsHistory.html (cit. on p. 35).

[18] Ted Waine, May Leung, and Paul Norris. «Directable stadium crowds from image based modelling for "Bohemian Rhapsody"». In: *ACM SIGGRAPH 2019 Talks.* SIGGRAPH '19. Los Angeles, California: Association for Computing Machinery, 2019. ISBN: 9781450363174. DOI: 10.1145/3306307.3328155. URL: https://doi.org/10.1145/3306307.3328155 (cit. on pp. 38, 40).

[19] SideFX. *Crowds Basics Documentation.* Accessed: January 21, 2025. URL: https://www.sidefx.com/docs/houdini/crowds/basics.html (cit. on pp. 41, 42).

[20] Susan Zwerman Jeffrey A. Okun. «The VES handbook of Visual Effects». In: Elsevier Inc, 2010. Chap. 7, pp. 690–694 (cit. on p. 49).

[21] Simon Clave Michael Büttner. *Motion Matching - The Road to Next Gen Animation.* 2015. URL: `https://www.youtube.com/watch?v=z_wpgHFSWss` (cit. on p. 68).

[22] Weiyu Li, Xuelin Chen, Peizhuo Li, Olga Sorkine-Hornung, and Baoquan Chen. «Example-based Motion Synthesis via Generative Motion Matching». In: *ACM Transactions on Graphics (TOG)* 42.4 (2023). DOI: `10.1145/3592395` (cit. on p. 69).

[23] *Example-based Motion Synthesis via Generative Motion Matching, ACM Transactions on Graphics (Proceedings of SIGGRAPH 2023).* 2023. URL: `https://github.com/wyysf-98/GenMM` (cit. on p. 69).

[24] SideFX. *Houdini KineFX Documentation.* Accessed: 02-02-2025. URL: `https://www.sidefx.com/docs/houdini/character/kinefx/index.html` (cit. on p. 72).

[25] SideFX. *Houdini KineFX Retargeting Documentation.* Accessed: 02-02-2025. URL: `https://www.sidefx.com/docs/houdini/character/kinefx/retargeting.html` (cit. on p. 74).

[26] SideFX. *Houdini MotionClip geometry node.* Accessed: 03-02-2025. URL: `https://www.sidefx.com/docs/houdini/nodes/sop/kinefx--motionclip.html` (cit. on p. 75).

[27] SideFX. *Houdini MotionClip Cycle geometry node.* Accessed: 03-02-2025. URL: `https://www.sidefx.com/docs/houdini/nodes/sop/kinefx--motionclipcycle.html` (cit. on p. 76).

[28] SideFX. *Houdini Smooth Motion geometry node.* Accessed: 03-02-2025. URL: `https://www.sidefx.com/docs/houdini/nodes/sop/kinefx--smoothmotion.html` (cit. on p. 77).

[29] Setareh Cohan, Guy Tevet, Daniele Reda, Xue Bin Peng, and Michiel van de Panne. *Flexible Motion In-betweening with Diffusion Models.* 2024. arXiv: `2405.11126 [cs.CV]`. URL: `https://arxiv.org/abs/2405.11126` (cit. on p. 94).

[30] *Cascadeur: Physics-based animation software.* URL: `https://cascadeur.com` (cit. on p. 94).

Acknowledgements

I would like to express my sincere gratitude to my supervisors for their invaluable guidance, support, and encouragement throughout this research. Their expertise and insightful advice have been fundamental to the completion of this work.

I extend my heartfelt thanks to *EDI - Effetti Digitali Italiani* for providing me with the opportunity to approach the VFX industry for the first time. In particular, I am deeply grateful to Daniele De Maio, my mentor and supervisor, for his unwavering support and guidance, as well as to Anna Maria Nigro for giving me this incredible opportunity. Their trust and belief in my abilities have been truly motivating.

Finally, I would like to express my profound appreciation to my family and friends. Throughout these challenging years, their unwavering support, patience, and encouragement have been a constant source of strength and inspiration.