Practical Resources for Developing Idiosyncratic Generative Systems for Dance

Daniel Bisig
Center for Dance Research, Coventry University, Coventry, United Kingdoms
e-mail: daniel.bisig@coventry.ac.uk
Institute for Computer Music and Sound Technology, Zurich University of the Arts, Zurich, Switzerland
e-mail: daniel.bisig@zhdk.ch

Abstract

Computer-based generative approaches possess a great creative potential in Contemporary Dance, in particular for artistic realisations that combine dance and technology. At the same time, the adoption and dissemination of generative approaches in dance is hampered by the fact that Dance and Technology is a small subfield within Contemporary Dance, with Generative Dance occupying an even smaller niche within this subfield. The work presented in this paper tries to ameliorate this situation by supporting artistic communities in Contemporary Dance and Generative Art with practical resources in the form of source code, dance data, educational articles, and documentations of exemplary artistic realisations. This material is meant to motivate and facilitate the selection from and adoption of a wide range of computational techniques and their use as foundations for realising dance specific generative systems. These techniques include both computer simulations and machine learning models that have proven useful in the author's own collaborations with dancers and choreographers for translating embodied creation principles into generative procedures. With regards to the integration of generative systems into the creative process, the provided material differentiates itself from other existing tools and collections in that it supports artists in devising their own idiosyncratic generative systems instead of working with a readily available but inscrutable software. Accordingly, this material aligns with artistic approaches that attribute a central role to the ideation and development of a generative system within a creative process.

Keywords: Dance, Generative Art, Machine Learning, Creative Coding

1. Introduction

Computer-based Generative Art provides a fertile ground for artistic experimentation in Contemporary Dance. But the number of artists who are active in this field and t he number of tools available to them is very small, especially when compared to other artistic domains such as music and fine arts.

This article hopes to increase the popularity of generative approaches in Contemporary Dance by contributing a set of tools that facilitate the development of generative systems. These tools take the form of programming libraries and source code examples. This form has been chosen to foster a c reative approach in Contemporary Dance that is already embraced by many practitioners in Generative Art, that is to situate the development of generative systems at a core of creative practice.

The tools cover a br oad range of generative techniques and include both simulation-based and dat a-driven methods. The tools are made available alongside educational material and artistic case studies. The educational material introduces not only the technical principles of each generative techniques but also provide context about embodied forms of creativity and the methods employed for bringing dance into digital form. The artistic case studies illustrate the adoption of the tools in collaborative creative productions, all of which have resulted in public performances.

The tools and the accompagnying materials represent the main tangible outcomes of a t wo and half years long fellowship during which the author has collaborated with dance scholars and practitioners. These collaborations served the purpose of identifying principles of embodied creativity which can be adopted for the development of generative techniques.

2. Background

Computer-based generative approaches possess a niche status in Contemporary Dance. The cause for this are conceptual and pr actical issues that are inherited for the most part from the wider field of Dance and Technology. The background section briefly introduces some of the most prominent issues and provides an ov erview of existing tools that support generative approaches in Contemporay Dance.

2.1 Dance and Digital Technology

The relationship between dance and technology is the subject of several debates, some of which raise principled concerns. These concerns are based on fears that technology is dismissive of the human body [31], incompatible with the ephemeral characteristics of dance [15], and alienates dancers and audiences from the experiential and i ntellectual aspects of movement [38]. These fears are countered by opinions that technology opens up new avenues for understanding [15, 29], envisioning [34], and experiencing dance [26]. Other concerns deal with practical issues such as the removal of essential context in digital representations of dance [40], the blindness of sensing technology to nuanced and hi dden aspects of dance [28], and the difficulty of formalizing dance as highly idiosyncratic creative practice [1, 41]. These concerns contrast with opinions that emphasise the proximity between choreographic thinking and algorithmic approaches [21, 1, 32].

The author is convinced that generative approaches possess the potential to contribute to these debates in a practical and constructive manner. This is because practitioners in Generative Art possess the expertise to formalise creative processes and br ing them into the computational domain while preserving their core idiosyncratic properties. This is also because generative systems can operate in a non-deterministic manner and produce results that are similarly variable and ephemeral as live performances. Finally, it is also because Generative Art often draws from sophisticated techniques for modeling biological, cognitive, and social principles and is thereby able to synthetically recreate both experiential and c ultural aspects of bodily creativity.

2.2 Dance and Generative Tools

Most generative systems that have been developed for dance specifically cater to the creative approach of a s ingle choreographer or dancer and ar e not intended to be us ed by other dance practitioners. The overview provided in this article focuses on the small number of generative systems that were developed with an application by the wider dance community in mind. A more exhaustive survey of generative systems that covers both idiosyncratic systems for individual dancers and more generic systems for a w ider dance community has been previously published by the author [3].

2.2.1 Dance Instructions

Several systems have been dev eloped for automatically creating instructions for human dancers.

The *Adaptive/Responsive Movement Approach* (*A/RMA*) is a gener ative tool for collaborative projects including dance and new media [30]. This system draws from system theory, computational programming protocols, and directed improvisation techniques. It provides a language for defining a trigger-based logic based on which dancers respond to the presence and activities of other dancers, audiences, and media on stage. The *A/RMA* systems is taught in workshops and can easily be adopt ed and modified by its users.

Terpsicode is a prototype programming language for live coding algorithms that generate choreographic patterns [35]. The language builds on t op of a vocabulary for describing movement, timing, and phrasing. The generated choreographic patterns are shown to dancers as a succession of photographs of dance poses. The dancers are free to interprete these poses and ex plore different transitions between poses. While *Terpsicode* is restricted to pattern-based approaches, it can be combined with different vocabularies.

2.2.2 Creativity Support

Several systems have been developed to support the creative process of choreographers.

Scuddle serves as co-creative tool that assists choreographers in the discovery of novel body movements [18]. It employs a genetic algorithm to generate incomplete movement proposals as catalists for ideation. The fitness function evaluates the proposals according to body symmetry, position, and levels and favours contralateral movements and unstable levels. This system was evaluated as stand-alone software and its behaviour cannot be modified.

Cochoreo has been realised as submodule for the choreographic software *idanceForms.* It can be used to generate full body poses as key-frames on a choreographic timeline [19]. *Cochoreo* employs the same evaluations as part of its fitness function as *Scuddle*. But contrary to *Scuddle*, the user can change the contributions of the individual evaluations to the fitness value by weighting them differently.

Hsieh and Luciani have developed a tool for generating dance movements based on a physical simulation of energy propagation [27]. The tool is based on the assumption that choreographers think in terms of energy transmission instead of keyframes when designing dance movements on a computer. The tool employs the *Cordis-Anima* dynamics simulation to deconstruct a dancing body into a minimal set of interactions between masses. Following this approach, a basic set of dance verbs has been chosen and for each of them a minimal set of masses and the dynamics of energy propagation were defined. The tool can in principle be extended to simulate other types of dance movements.

The *Body-part Motion Synthesis System* (*BMSS*) allows users to synthesize and sequence body motions into short choreographies for a single dancer [36]. The system provides a co-creative workflow in which the user specifies a whole body motion and body part categories and the system determines suitable body part motions, timings, and blendings with successive motions. The system provides some flexibility by varying the balance between manually and automatically generated motions.

The *chor-rnn* system creates synthetic motions for a single dancer [20]. The system employs an autoregressive neural network for pose sequence continuation that can be trained on motion capture data. After training, *chor-rnn* is able to generate novel choreographic material in the language and style of an individual choreographer. The authors of *chor-rnn* propose a creative workflow in which the system and a choreographer take turns and either continue each other’s motion sequences or use them as inspiration. This system is released as open source code and can be retrained or modified.

Petee et al. have released several machine learning-based tools that are meant to augment a choreographer’s workflow [29]. These tools include an autoencoder for poses, a variational autoencoder for pose sequences, and an autoregressive neural network for pose sequence continuation. These models can be trained on motion capture data of a single dancer. The authors discuss applications of these models for production, creation, introspection, and teaching. These systems are released as open source code and can be retrained and modified.

2.2.3 Interactive Media

Several systems have been developed for controlling and/or generating interactive media.

Among these, the *ViFlow* system is the only one that is specifically geared towards dance [16]. *ViFlow* employs a particle simulation to generate interactive visuals for live performance. This system provides gestural controls for interacting during performance and also for authoring the simulation and visualisation. The authors claim that the system can be eas ily integrated into an embodied creation workflow and alleviates the need f or dancers to collaborate with engineers.

Embodied forms of interacting with and authoring of digital media are also faciliated by several machine learning based tools such as Wekinator [23], ml.lib [17] and M arcelle [24]. Most example applications for these tools deal with the design of digital musical instruments [22]. Nevertheless, these tools can also be adopt ed for dance-specific purposes. The tools support a create workflow for designing gestural interfaces that exploit tacit and embodied knowledge about movement [25]. The tools offer different machine learning models that users can chose from, parametrise, and train. To foster iterative and explorative workflows, the models provided are simple enough to operate in real-time, handle low dimensional input and output data, and c an be trained on very small datasets. These models cannot directly generate synthetic media but are instead used to establish mappings between interfaces and parameters for controlling an external media generation system. The flexibility of Marcelle exceeds that of Wekinator and ml.lib since it allows users to add models of their own.

The tools that have been dev eloped by the author complement the generative tools listed above in that they cover a much broader range of generative techniques and ar e provided in the form of programming libraries and source code examples instead of stand-alone applications. Accordingly, these tools are aimed at artists who are interested in developing their own generative systems and require a maximum amount of flexibility for this purpose.

3. Tools

The tools are provided in the form of open source libraries and source code examples for the two programming environments *openFrameworks*[1] and *PyTorch*[2]. *openFrameworks* is a C++ programming environment for developing creative applications. *PyTorch* is a deep-learning framework that provides a Python and C++ interface. These environments have been c hosen because openFrameworks is particular popular in the creative coding community and *PyTorch* is frequently used by researchers for developing and s haring novel machine learning methods. Furthermore, models implemented in *Py-Torch* can easily be ex ported and integrated into C++ code which makes this environment suitable for real-time and interactive applications. All tools are accessible via *Github* repositories.

3.1 Simulation-based Generative Tools

Several software libraries have been developed in the form of *addons* for *openFrameworks*. These libraries facilitate the development of simulation-based generative systems. Currently, libraries are provided for simulating mass-spring systems, articulated rigid bodies, and flocking behaviours.

3.1.1 Mass-Spring Systems

1 openFrameworks: https://openframeworks.cc/

2 PyTorch: https://pytorch.org/

The addon[3] provides functionality to simulate mass-spring-damper systems. The simulation implements regular springs which exhibit a restitution force whenever they deviate from their rest lengths and in which damping forces oppose the velocities of mass-points. The simulation also implements directional springs which exhibit a restitution force whenever their relative directions with regards to a preceding spring deviates from their relative rest directions (fig. 1 top). The simulations also supports the applications of external forces to mass-points which can either be randomised or deterministic. For numerical integration, both Euler and Leapfrog integration schemes are provided. The simulation can be employed to construct mesh topologies in which mass-points are organised in lattices (fig. 1 middle) and cross-linked via multiple springs or to create branching topologies (fig. 1 bottom).

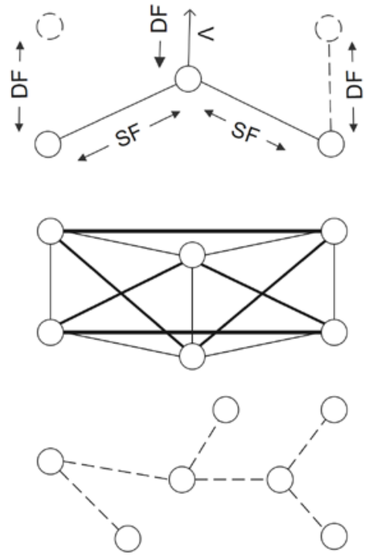

Figure 1: Mass-Spring Systems. In these graphical depictions, mass-points are shown as outlined circles, regular springs as solid lines, and directional springs as dashed lines. The figure at the top depicts the forces acting on mass-points. These forces are shown as solid arrows and are labeled as follows: SF stands for regular spring restitution force, DF stands for directional spring restitution force, DF stands for damping force. Velocity is shown as outlined arrow and abbreviated as V. The figure in the middle depicts mass-points and regular springs that are organised in a mesh topology. The figure at the bottom depicts mass-points and directional springs that are organised in a branching topology.

3.1.2 Articulated Rigid Bodies

The addon[4] employs the Bullet rigid body dynamics engine on top of which it provides classes for importing, configuring, and actuating articulated morphologies. Morphologies consist of body parts which are connected to each other via joints (fig. 2). The body parts are rigid and can possess arbitrary shapes. The joints can either be passive or active. Active joints act operate in one of the following modes: as freely spinning motors, as servo motors that possess a target rotation, and as spring motors with stiffness and damping properties. Bodies can exhibit autonomous behaviours. Behaviours are routines that periodically change some of the physical properties of the body parts or joints that are assigned to them. These changes can

3 ofxDabSpring: https://github.coventry.ac.uk/ad5041/ofxDabSpring

4 ofxDabPhysics: https://github.coventry.ac.uk/ad5041/ofxDabPhysics

either be deterministic or random. Behaviours can also operate on joints or body parts that don't belong to only one body and thereby generate coordinated movements among several bodies.

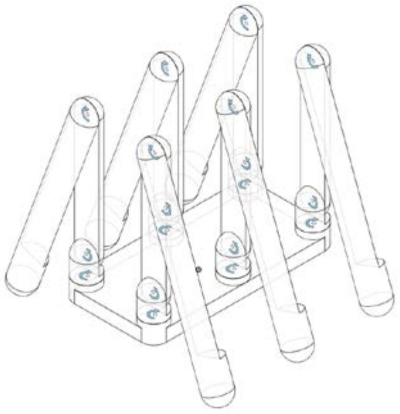

Figure 2: Articulated Morphology. The graphical rendering depicts a morphology that consists of six legs which are attached to a bas e platform. Each leg possesses three joints that provide one rotational degree of freedom. In the rendering, body parts are shown as outlined shapes and rotational joints as curved arrows.

3.1.3 Flocking Behaviours

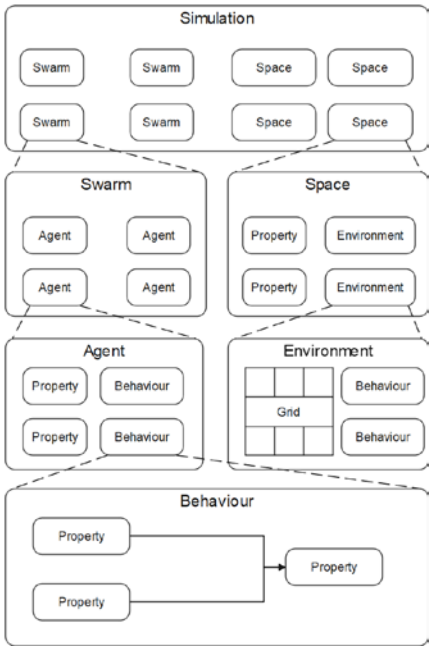

Figure 3: Flocking Behaviours. The schematic image displays the hierarchical organisation of the main elements that make up the simulation.

The addon[5] provides functionality to simulate flocking behaviours. This simulation has originally been developed as generative mechanism for computer music [9, 8, 33, 6]. The simulation models the behaviours of agents that organise in single or multiple swarms (fig. 3). The agents possess properties. The properties can be as signed to spaces to calculate neighborhood relationships among them. Spaces can also possess spatially distributed properties which can change dynamically based on c ellular automata rules. Both agents and spaces can possess behaviours which read and modify properties. The simulation is highly generic in that the number of

5 ofxDabFlock: https://github.coventry.ac.uk/ad5041/ofxDabFlock

agents, swarms, and s paces, the type and dimensionality of properties, and the effects of behaviours can be chosen freely.

3.2 Machine-Learning based Generative Tools

Several generative models have been implemented for the *PyTorch* framework. The models include sequence continuation networks, generative adversarial networks, and adv ersarial autoencoders. All models are available online[6]. With the exception of sequence continuation networks, different versions of each model have been c reated for image, pose, and pos e sequence data. The models are provided with example training data. In the case of images, data has been collected by searching the social media platform *Flickr*[7] for photographs of dancers. In the case of poses and pos e sequences, data has been obtained by conducting motion capture recordings of a professional dancer. The poses and pos e sequences consist of joint orientations that are represented as unit quaternions.

3.2.1 Sequence Continuation

The sequence continuation networks are provided in two versions. Both versions are autoregressive, consist of a Long-Short Term Memory Memory (LSTM) network, and take as input a sequence of poses for which they predict as output a potential sequence succession. The simpler version is deterministic and directly outputs a pos e (fig. 4 top). The more complicated version is probabilistic and passes the output from the LSTM into a mixture density network (MDN) (fig. 4 bot tom). This network outputs for each pose dimension the mean, standard deviation, and w eighting for several Normal distributions. An MDN decreases the risk that a pr edicted sequence continuation stagnates after a while.

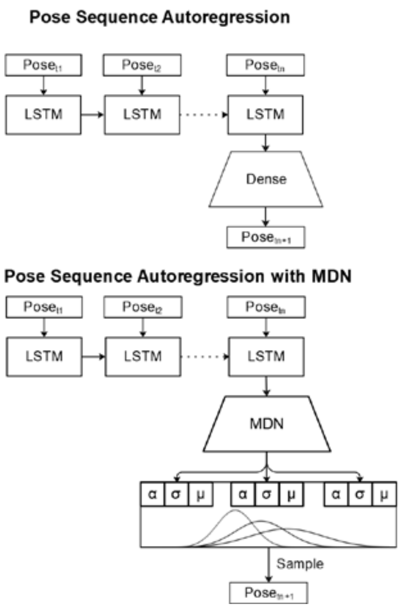

Figure 4: Autoregressive Models for Pose Sequence Continuation. The model depicted at the top is deterministic. The model depicted at the bottom is probabilistic. Following conventions, the LSTM network is shown unrolled in time with time running from left to right.

For both models, two different training schemes are available, one t hat always provides the correct pose as model input (Teacher Forcing) and one that occasionally provides the model's own output as input (Without Teacher Forcing). Without Teacher Forcing, the model becomes better at correcting its own errors during sequence continuation.

6 ML Models: https://github.coventry.ac.uk/ad5041/PyTorch_ML_Tutorials

7 Flickr: https://www.flickr.com/

3.2.2 Generative Adversarial Networks

Three different Generative Adversarial Networks (GAN) are provided, one for poses (fig. 5 top left), one for pose sequences (fig. 5 bot tom), and one f or images (fig. 5 top right).

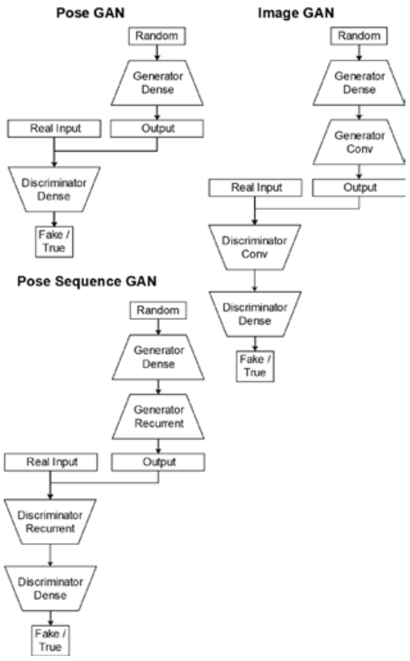

Figure 5: Generative Adversarial Networks. Shown are three networks: one for generating poses (top left), one for generating images (top right), and one for generating pose sequences (bottom). Each network consists of two models, a Generator which is shown on the right and a D iscriminator which is shown on the left of each model.

Each GAN consists of two models, a Generator that takes as input a vector of random values and produces as output a data instance, and a D iscriminator that takes as input a data instance and produces as output a bi nary value that classifies the instance as either real instance is from the dataset) or fake (instance is from the Generator). The models in the GAN that operates on poses consist of a m ultilayer perceptron (MLP) only. The two other GANs add t o their models a net work that is either convolutional for images or recurrent for pose sequences. During training, the two models compete with each other. The Generator tries to improve its capability to produce data instances that the Discriminator mistakenly categorises as real. The Discriminator tries to improve its capability to distinguish between real and fake data instances. After a successful training, the output of the Generator is indistinguishable from instances that stem from the dataset.

3.2.3 Adversarial Autoencoders

Three different Adversarial Autoencoders (AAE) are provided, one for poses (fig. 6 top left), one f or pose sequences (fig. 6 bottom), and one for images (fig. 6 t op right). An autoencoder is a m odel that operates as information bottleneck by encoding and m apping high-dimensional data instances into a low-dimensional latent-space. Mathematical operations can be conducted in latent-space and the result of these operations can be converted back through decoding into data instances.

AAEs adopt the use of a D iscriminator from GANs. Here, the task of the Discriminator is to classify latent dimension vectors as real when following a true Normal distribution or fake when output by the Encoder-part of the autoencoder. Controlling the distribution of latent vectors ensures that the latent space is free of gaps and t hat distances within it represent a m easure of similarity. This in turn guarantees that arbitrarily chosen latent vectors can be converted by the Decoder into meaningful data instances. For all three AAE, the Discriminator model consists of an MLP only. In case of the AAE that operates on pos es, the Encoder and Decoders also consist of MLPs only. In case of the other AAE, an additional network is added that is either convolutional in case of images or recurrent in case of pose sequences. During training, the autoencoder not only enters into an adversarial competition with the Discriminator but also tries to improve its capability to reconstruct instances of the training data.

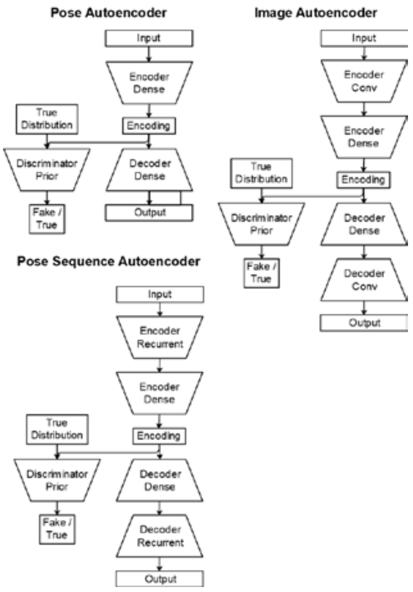

Figure 6: Adversarial Autoencoders. Shown are three autoencoders: one f or generating poses (top left), one f or generating images (top right), and one for generating pose sequences (bottom). Each autoencoder consists of three models, an Encoder which is shown on the top right, a D ecoder which is shown on the bottom right, and a Discriminator which is shown on the left of each model.

4. Educational Material

The release of the tools is accompanied by an educ ational blog[8]. This blog provides a large number of articles that are meant to deepen and br ing together an understanding for Contemporary Dance with skills in both simulation-based and data-driven forms of Generative Art. The topics covered by the articles include creative embodied practice, methods for digitising and analysing dance, generative methods for simulating dance, and m achine learning models for synthesising dance movements. Also included are tutorials for all the tools that are described in this article. The blog not only helps with learning how to work with these tools but also situates the tools firmly in Contemporary Dance, either as a r ich resource of inspiration for creating generative artworks, or as domain of application for Generative Art.

5. Artistic Case Studies

Most of the tools have emerged from or were employed for the development of generative systems used in artistic realisations. This article presents a selection of the most recent realisations.

5.1 Strings P

Strings P is an audi o-visual concert [13] that involves three instruments: an acoustic violin, a s ynthetic acoustic instrument, and a s ynthetic visual instrument. These instruments are related to each other conceptually, technically, and aes thetically by sharing the same physical principle of a vibrating string [14]. In case of the synthetic

8 Educational Blog: https://wp.coventry.domains/e2edu/

instruments, vibrating strings are simulated using mass-spring systems.

The generative system that controls sound synthesis simulates several one-dimensional arrays of interconnected springs. These arrays are sonified following a direct audification approach [37] by mapping the deflection of mass-points into amplitudes of a waveform.

The generative system that controls image synthesis simulates two-dimensional meshes of interconnected springs. These meshes are created dynamically from a live camera image by detecting salient image points and tessellating them into triangulated surfaces (fig. 7). The surfaces are visually rendered with a colouring that depends on the live camera image and an opacity that depends on the amplitude of the springs' oscillations.

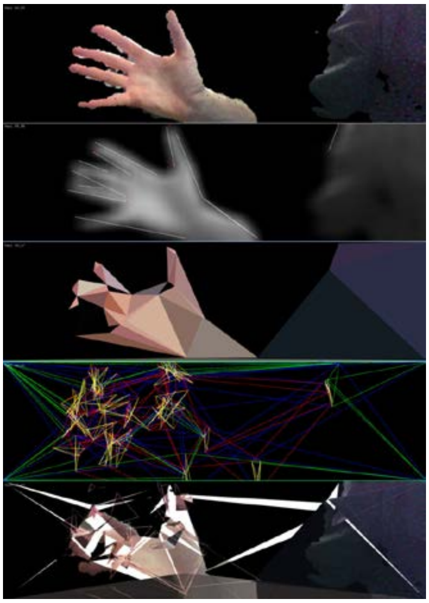

Figure 7: Conversion of a Camera Image into a Mass-Spring System. From top to bottom: The camera image, salient contours and feature points in the camera image, Delaunay triangulation derived from feature points and coloured according to the camera image, a Mass-Spring System created from the triangulation, and the final visual rendering of the Mass-Spring System.

Interaction between the three instruments is based on acoustic resonance. The acoustic output of the violin is recorded by a microphone and its most prominent spectral peaks are identified. Based on the amplitude and frequency of these peaks, periodic motions are imposed on those simulated springs that possess a matching resonance frequency. The resonance frequency is derived from the rest lengths of the simulated springs.

This setup establishes a performance situation in which the violinist can exploit his familiarity and virtuosity with the acoustic instrument to simultaneously play two novel synthetic instruments (fig. 8). A video recording of the performance is available online[9].

9 Strings P Performance Video: https://youtu.be/eUwZuc2OxHs

Figure 8: Still Image of a Video Recording of a Rehearsal.

5.2 Artificial Intimacy

Artificial Intimacy is an installation that creates a duet between a single human dancer and an artificial dancer. The human dancer acts as puppeteer who controls a single limb of the artificial dancer while its remaining limbs remain under the control of a machine learning model. The human dancer interacts with the artificial dancer by means of a wearable sensor that measures absolute orientation. The measured orientation is either translated into a target rotation for one of the artificial dancer's limbs or into a target position towards which the artificial dancer reaches with one of its limbs.

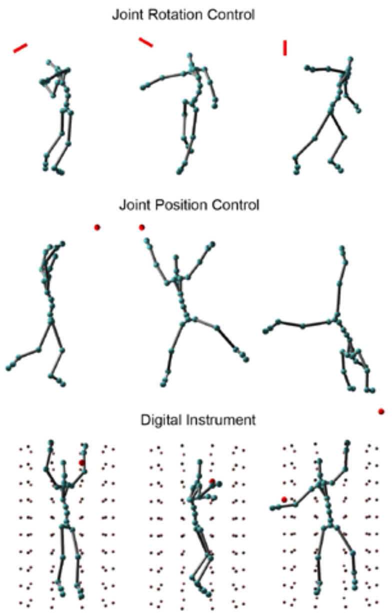

Figure 9: Machine Learning-based Artificial Dancer. The images on the top and middle row depict the artificial dancer responding to interactively controlled changes of its right shoulder rotation (top row) or the target position of its right hand (middle row). The bottom row depicts the artificial dancer surrounded by resonant filters depicted as small spheres.

The artificial dancer is based on a generative system named *Granular Dance*. This system combines an Adversarial Autoencoder trained on pose sequences with a mechanism to seamlessly concatenate multiple pose sequences [4]. For realising this installation, *Granular Dance* has been extended with two novel methods for interactively creating synthetic motions. These methods operate on the level of the motion itself rather than its encoding. The first method combines the control of the orientation of a joint with iterative autoencoding [12] (fig. 9 top row). The second method combines the control of the target position of a joint with forward kinematics and the application of latent difference vectors [11] (fig. 9 middle row).

For visualisation, the Ray Marching method is employed to display the artificial dancer as smooth surface that varies between a humanoid and amorphous appearance (fig. 9). The artificial dancer is also displayed acoustically using a virtual musical instrument. This instrument simulates a vibrating surface by means of a bank of resonating filters. The filters are arranged cylindrically and surround the artificial dancer (fig. 9 bottom row). The instrument emits sounds when the artificial dancer approaches the filters.

Video recordings of a rehearsal are available online[10].

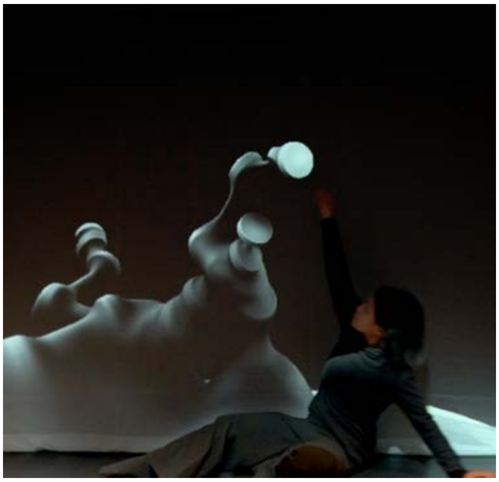

Figure 10: Still Image of a Video Recording of a Dance Rehearsal.

5.3 Embodied Machine

Embodied Machine is a performance for a single human dancer and a stage that acts as an extension of the dancer's body or as an autonomous dance partner. At the core of *Embodied Machine* are movement qualities that have been developed by choreographer Muriel Romero. These qualities are used to establish a common ground between the human dancer, music, and light.

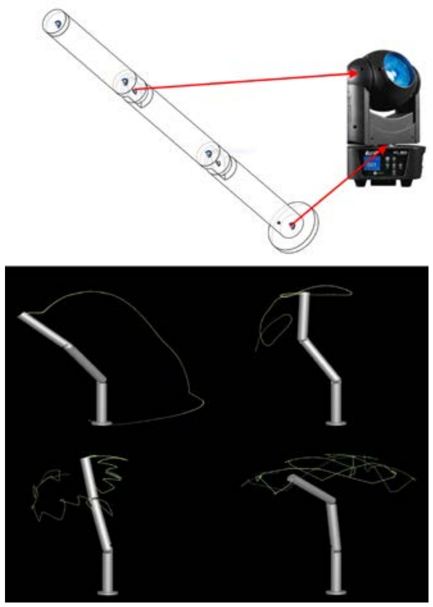

Figure 11: Simulated Articulated Bodies. The figure at the top depicts the mapping of joints between a simulated body and a robotic moving light. The figure at the bottom depicts a simulated body performing movements according to different movement qualities.

Several robotic moving lights have been used in the performance. These lights are endowed with the capability to execute autonomous movements that express specific movement qualities [5]. For this purpose, the robotic lights have been modelled and simulated as articulated rigid bodies (fig. 11 top). The

10 Artificial Intimancy Rehearsal Video: https://vimeo.com/724839655

simulation has been extended with two behaviours. A behaviour named *Force Behaviour* generates forces that impact externally on body parts. A behaviour named *Rotation Behaviour* specifies target angles towards which body joints rotate to. By assigning body parts and joints to these behaviours, and by choosing appropriate settings for the parameters of the bodies and behaviours, the desired movement qualities could be imitated (fig. 11 bottom). These movement qualities were then transferred on the robotic lights by mapping the joint rotations from the simulated bodies on the robotic lights.

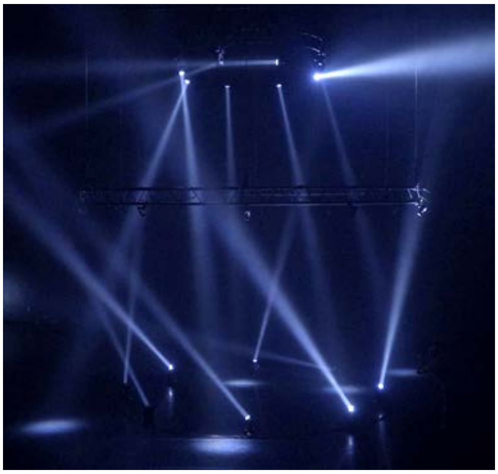

Figure 12: Still Image of a Video Recording of the Scene Progression.

The autonomous robotic lights were employed in two dance scenes. A scene named *Approximation* plays out as a series of duets between a human dancer and each robotic light. The dancer approaches one robotic light after the other. Once the dancer is sufficiently close to a robotic light, this lights starts to emit light and chooses a movement quality according to which it behaves. In a scene named *Progression*, the robotic lights follow their own choreography independently of the activities of the human dancer. The choreography progresses through several stages during which different movement qualities are combined or juxtaposed (fig. 12).

Video recordings of excerpts of the scenes *Approximation* and *Progression* are available online[11] [12].

6. Discussion

The tools that have been introduced in this article form part of the author's attempt to foster and facilitate the exchange and collaboration between Generative Art and Contemporary Dance. The tools constitute the practical complement to the author's previously published taxonomy of generative approaches in dance [3].

The tools are unique with regards to the breath of generative methods they employ. Some the tools make use of simulation-based methods and others of data-driven methods. Each category of methods comes with their own benefits and drawback. Simulation-based methods usually excel at reproducing natural phenomena but struggle with capturing unique stylistic or expressive aspects. Data-driven methods possess the opposite properties.

In the following, the application potential of each method is briefly addressed. Simulations of mass-spring systems have been popular for a long time since

11 Approximation Videos:
https://youtu.be/6LYVv1aq5hQ
https://youtu.be/XBfekygb1c8

12 Progression Videos:
https://youtu.be/_bTh7102zfE
https://youtu.be/h5SkN97wiKc

they are easy to implement, fast to compute, and can mimic a broad range of physical phenomena. Such phenomena include flexible surfaces such as textiles, vibrating bodies such as acoustic instruments, or elastic morphologies of plants or invertebrates. Simulations of the constrained dynamics of rigid bodies are widely employed in computer games for character and vehicle animation. In the context of dance, these simulations can be used for modelling the behaviour of artificial dancers. Since the simulated bodies don't necessary have to be anthropomorphic, they can also be used to represent and control actuated machinery on stage. Simulations of flocking behaviours are among the most canonical forms of multi-agent systems. Originally, these simulations have been employed to model the coordinated movements of animals such as flocks of birds or schools of fish. But likely any form of coordinated spatial motion can be modelled with these simulations, regardless of whether it is of physical, biological, or speculative origin. Accordingly, these simulations offer great artistic flexibility for creating and controlling the spatial distribution of media and for designing interactions. Machine-learning models for sequence continuation have gained some prominence as generative tools for choreography. A model that has been trained on example movements of a specific dancer or choreographer can generate synthetic dance movements that mimic with a high degree of fidelity the stylistic and expressive properties of the example movements. Unfortunately, sequence continuation models offer little room for interactive control unless they are conditioned during training on specific control parameters. Generative adversarial networks have gained tremendous popularity as generative machine learning models. State of the art versions of GANs such as *StyleGANs* excel at producing output that is both novel and realistic. Furthermore, GANs can cope with more or less any type of data and are therefore attractive for a wide range of artistic applications. On the other hand, at least the canonic versions of GANs which are included as tools are challenging to train and make it difficult to control specific aspects of the generated data instances. Autoencoders have been superceeded in popularity by GANs because they produce less realistic results. Neverthelss, autoencoders remain attractive for artistic purposes. They offer a wide range of possibilities for generating new data instances whose similarity or novelty compared to original training data can be finely tuned. Autoencoders also offer interesting forms of interaction within latent space of data encodings or within data space itself.

The tools presented in this article are also fairly unique in that they are provided as software libraries and example source code rather than stand-alone applications. Accordingly, these tools are highly flexible and can be thoroughly modified and extended for new artistic realisations. This flexibility is meant to cater to creative practices that embrace idiosyncratic approaches which includes those in Contemporary Dance and Generative Art. This also means, that the tools will mainly appeal to artists for whom the development of generative systems forms a core element of their creative practice. As a consequence, at least some of the tools have a relatively high entry barrier and require programming skills that might be out of reach for artists who are not familiar with creative coding. For those creative practitioners that lack these skills, the tools might only be of use if they manage to recruit creative coders as collaborators. Fortunately, the involvement of technical experts in creative productions is very common in the field of Dance and Technology. Such forms of collaboration have been thoroughly explored and doc umented in the creative case studies presented in this article [14, 39, 10].

7. Outlook

The tools presented in this article are very diverse not only with regards to the generative methods they represent but also their immediate usefulness for artistic workflows. The most useful tools have undergone an i terative process in which the development of the tool and the creation of new works mutually influenced each other. This is the case for the simulations of mass-spring systems, articulated rigid bodies, and flocking behaviours and for the autoencoders that operate on pos e sequences. The other tools have yet to benefit from such an i terative process. An upcoming dance production provides the opportunity to evaluate and improve the tools for sequence continuation. It is hoped that future productions will also lead to further developments of the GAN-based tools.

At the moment, each of the tools has been used in isolation. It would be interesting to experiment with generative approaches that combine multiple tools. In particular, the combination of simulation-based and dat a-driven methods represents a very promising and largely unexplored domain of activity. One approach that employs such a combination is reinforcement learning. The author has previously conducted research with this approach [2] and plans to translate some of the results of this research into additional tools for Contemporary Dance and Generative Art.

While the artistic case studies that made use of the tools involved several collaborators, it has mostly been the author himself who directly worked with the tools. The few exceptions that prove the rule include a media artist who worked with the mass-spring simulation to create most of the visual content for a precursor version of the piece *Strings P* and a computer musician who combined his code for sound synthesis with the autoencoder employed in the piece *Artificial Intimacy*. Motivating other artists to experiment with these tools and contribute to their development is a high priority for future activities. First steps in this direction are currently undertaken in the form of ongoing and upc oming teaching activities about AI and Art and a recently started European research project that deals with the integration of machine learning and ex tended reality into dance and theatre productions[13].

Finally, a further research and development direction aims at lowering the tools' barriers for artists with limited programming skills. This direction will likely involve the tools operating as servers running generative systems whose state can be configured and modified remotely by sending and receiving messages. While this approach

13 HORIZON-CL2-2021-HERITAGE-01-04: https://cordis.europa.eu/project/id/101061303

doesn't allow to fundamentally change or extend the functionality of a tool, it nevertheless permits an artist to experiment with the tool and integrate it into his or her digital work pipelines. Since these messages can be generated algorithmically with any tool the artist is already familiar with, such a server-based approach can reduce the difficulties in building a customised generative system. This approach has been employed by the author for the *Interactive Swarm Orchestra* system [7] from which the Flocking Behaviour tool is derived.

8. Acknowledgments

The research presented in this publication has been conducted during a Marie Curie Fellowship (H2020-MSCA-IF-2018-Grant agreement no: 840465).

9. References

[1] Sarah Fdili Alaoui, Kristin Carlson, and Thecla Schiphorst. "Choreography as mediated through compositional tools for movement: Constructing a historical perspective". In: Proceedings of the 2014 International Workshop on Movement and Computing. 2014, pp. 1–6.

[2] Daniel Bisig. "Expressive Aliens-Laban Effort Factors for Non-anthropomorphic Morphologies". In: International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar). Springer. 2022, pp. 36–51.

[3] Daniel Bisig. "Generative Dance-a Taxonomy and Survey". In: Proceedings of the 8th International Conference on Movement and Computing. 2022, pp. 1–10.

[4] Daniel Bisig. "Granular Dance". In: Ninth Conference on Computation, Communication, Aesthetics & X. i2ADS. 2021, pp. 176– 195.

[5] Daniel Bisig. "Simulating Idiosyncratic Movement Qualities". In: Proceedings of the 11th EAI International Conference: ArtsIT, Interactivity & Game Creation. Faro, Portugal, 2022.

[6] Daniel Bisig and Philippe Kocher. "Tools And Abstractions For Swarm Based Music And Art". In: ICMC2012 Non-Cochlear Sound. 2012, pp. 297–300.

[7] Daniel Bisig and Philippe Kocher. "Tools and Abstractions for Swarm based Music and Art". In: Proceedings of the International Computer Music Conference. Ljiubljana, 2012.

[8] Daniel Bisig and Martin Neukom. "Swarm Based Computer Music - Towards a Repertory of Strategies". In: Proceedings of the Generative Art Conference. Milano, Italy, 2008, pp. 101–111.

[9] Daniel Bisig, Martin Neukom, and John Flury. "Interactive swarm orchestra-a generic programming environment for swarm based computer music". In: Proceedings of the International Computer Music Conference. Vol. 2008. 2008.

[10] Daniel Bisig, Muriel Romero, and Pablo Palacio. "Embodied Machine". In: Embodied Machine. 2022. URL: https : / / wp . coventry . domains /embodiedmachine/.

[11] Daniel Bisig and Ephraim Wegner. "Puppeteering AI - Interactive Control of an Artificial Dancer". In: Proceedings of

the Generative AI and H CI - CHI 2022 Workshop. New Orleans, USA, 2022.

[12] Daniel Bisig and E phraim Wegner. “Puppeteering an AI - Interactive Control of a M achine-Learning based Artificial Dancer”. In: Proceedings of the XXIII conference on Generative Art. Rome, Italy, 2021, pp. 315–332.

[13] Daniel Bisig and Ephraim Wegner. “Strings”. In: Eighth Conference on Computation, Communication, Aesthetics & X. i2ADS. 2020, pp. 299–312.

[14] Daniel Bisig, Ephraim Wegner, and Harald Kimmig. “Strings P”. In: Ninth Conference on Computation, Communication, Aesthetics & X. i2ADS. 2021, pp. 546–553.

[15] Hetty Blades. “Creative computing and the re-configuration of dance ontology”. In: Electronic Visualisation and the Arts (EVA 2012) (2012), pp. 221–228.

[16] Taylor Brockhoeft et al. “Interactive augmented reality for dance”. In: Proceedings of the Seventh International Conference on C omputational Creativity. 2016, pp. 396–403.

[17] Jamie Bullock and A li Momeni. “Ml. Lib: robust, cross-platform, open-source machine learning for max and pure data.” In: NIME. 2015, pp. 265–270.

[18] Kristin Carlson, Thecla Schiphorst, and Philippe Pasquier. “Scuddle: Generating Movement Catalysts for Computer-Aided Choreography.” In: ICCC. 2011, pp. 123–128.

[19] Kristin Carlson et al. “Cochoreo: A generative feature in idanceForms for creating novel keyframe animation for choreography”. In: Proceedings of the Seventh International Conference on Computational Creativity. 2016.

[20] Luka Crnkovic-Friis and Loui se Crnkovic-Friis. “Generative choreography using deep l earning”. In: arXiv preprint arXiv:1605.06921 (2016).

[21] Scott DeLahunta. “Software for dancers: coding forms”. In: Performance Research 7.2 (2002), pp. 97–102.

[22] Rebecca Fiebrink and B aptiste Caramiaux. “The machine learning algorithm as creative musical tool”. In: arXiv preprint arXiv:1611.00379 (2016).

[23] Rebecca Fiebrink and Perry R Cook. “The Wekinator: a s ystem for real-time, interactive machine learning in music”. In: Proceedings of The Eleventh International Society for Music Information Retrieval Conference (ISMIR 2010)(Utrecht). Vol. 3. 2010.

[24] Jules Françoise, Baptiste Caramiaux, and Téo Sanchez. “Marcelle: Composing Interactive Machine Learning Workflows and I nterfaces”. In: The 34th Annual ACM Symposium on U ser Interface Software and T echnology. 2021, pp. 39–53.

[25] Marco Gillies. “Understanding the role of interactive machine learning inmovement interaction design”. In: ACM Transactions on Computer-Human Interaction (TOCHI) 26.1 (2019), pp. 1–34.

[26] Mark BNHansen. Bodies in code: Interfaces with digital media. Routledge, 2012.

[27] Chi-Min Hsieh and A nnie Luciani. “Generating dance verbs and assisting computer choreography”. In: Proceedings of the 13th Annual ACM

international Conference on M ultimedia. 2005, pp. 774–782.

[28] Erin Manning. “Prosthetics Making Sense: Dancing the Technogenetic Body”. In: The Fibreculture Journal 9 (2006).

[29] Mariel Pettee et al. “Beyond imitation: Generative and variational choreography via machine learning”. In: arXiv preprint arXiv:1907.05297 (2019).

[30] Megan Pitcher. “Adaptive/responsive movement approach: dance making as interactive system design”. In: Proceedings of the 2nd International Workshop on Movement and Computing. 2015, pp. 76–79.

[31] Richard Povall. “A little technology is a dangerous thing”. In: Moving history/ dancing cultures: A dance history reader (2001), pp. 455–458.