Reinforcement Learning Based Dance Movement Generation

Markus Toverud Ruud
markustr@ifi.uio.no
University of Oslo
Oslo, Norway

Tale Hisdal Sandberg
talehs@ifi.uio.no
University of Oslo
Oslo, Norway

Ulrik Johan Vedde Tranvaag
ujtranva@ifi.uio.no
University of Oslo
Oslo, Norway

Seyed Mojtaba Karbasi
mojtabak@ifi.uio.no
University of Oslo
Oslo, Norway

Benedikte Wallace
benediwa@ifi.uio.no
University of Oslo
Oslo, Norway

Jim Torresen
jimtoer@ifi.uio.no
University of Oslo
Oslo, Norway

ABSTRACT

Generating genuinely creative and novel artifacts with machine learning is still a challenge in the world of computational science. A creative machine learning agent can be beneficial for applications where novel solutions are desired and may also optimize search. Reinforcement Learnings' (RL) interactive properties can make it an effective tool to investigate these possibilities in creative contexts. This paper shows how a Reinforcement learning-based technique, in combination with Principal Component Analysis (PCA), can be utilized for generating varying movements based on a goal picking policy. The proposed model is trained on a data set of motion capture recordings of dance improvisation. Our study shows that the trained RL agent can learn to pick sequences of dance poses that are coherent, have compound movement, and can resemble dance.

CCS CONCEPTS

• **Computing methodologies** → **Reinforcement learning**; • **Applied computing** → **Performing arts**.

ACM Reference Format:
Markus Toverud Ruud, Tale Hisdal Sandberg, Ulrik Johan Vedde Tranvaag, Seyed Mojtaba Karbasi, Benedikte Wallace, and Jim Torresen. 2022. Reinforcement Learning Based Dance Movement Generation. In *8th International Conference on Movement and Computing (MOCO'22), June 22–24, 2022, Chicago, IL, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3537972.3538007

1 INTRODUCTION

Movement generation using Artificial Intelligence (AI) is a complex task, and artistic movement such as dance introduces an additional set of challenges as the space of potentially interesting solutions is infinite and subjective. Dance generation has been explored previously using deep learning sequence prediction models such as LSTMs [2, 15, 17], and the Transformer model [7, 8, 16]. However, using a supervised learning approach enforces an expectation that there is a single correct next state given any starting position. When generating artistic output, this is not the case. In generative AI applications, this is often addressed through the use of Generative Adversarial Networks [1], but this method requires large data sets and can be difficult to train effectively. A promising alternative is RL [11]. However, finding an appropriate reward function and evaluating the agents performance remains a challenge. Recently several works have used RL to generate artistic outputs like music and paintings [3, 4, 9, 13, 21]. The idea of using RL to generate novel creative artifacts was explored by J. Schmidhuber using the concept of curiosity and intrinsic motivation (internal rewards as opposed to external rewards, affecting behaviour) [12]. One way of achieving intrinsic motivation is to use Hierarchical RL (HRL) algorithms. Kulkarni et al. [6] propose an approach for dealing with delayed rewards in some video games, using HRL based on temporal abstraction. Our work is inspired by the HRL approach and uses predefined goals to generate dance sequences which are inspired by the dance samples observed by the agent.

MOCO'22, June 22–24, 2022, Chicago, IL, USA

ACM ISBN 978-1-4503-8716-3/22/06.
https://doi.org/10.1145/3537972.3538007

2 METHODS

Using a data-set of dancers captured with full-body motion capture, we analyze the data, frame by frame, and feed it to our RL. The RL is programmed to generate its own versions of the dance based on a list of predefined goals.

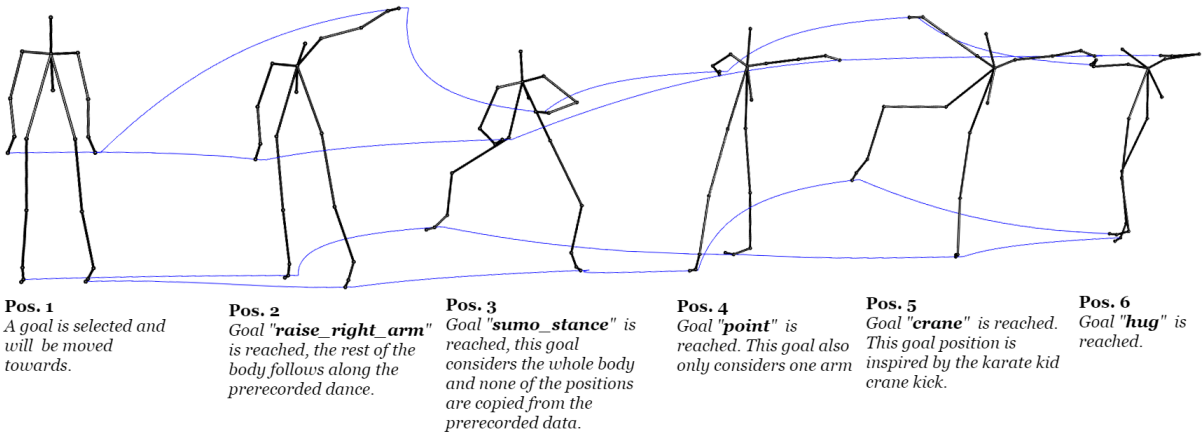
2.1 Movement data, goals and evalution

The training data contains 162 examples of improvised dance from an open access data set[1], previously used by Wallace et al. [17, 18]. The time-series consist of 22, 3-dimensional points on the dancers bodies, captured at a rate of 30 frames per second.

We define a set of goals, G, as a series of movement frames. The goals are stored as a dictionary, where the name of the goal is the key. We created each goal g by animating the data sets and visually inspecting the animation for positions that seemed interesting to us and were sufficiently different from the rest of our goal positions. Examples are demonstrated in Figure 1. How the goals are structured is shown in Figure 2.

Defining states is a problem when using time-series data with high dimensionality. When deciding on a metric for evaluating the generated movements we use the center point of the 22 joints (See Figure 3a). This center point is the mean value of the dancer's positions in x, y and z dimensions. We use these center points to

[1]Wallace, Benedikte, Nymoen, Kristian, Martin, Charles P., & Tørresen, Jim. (2022). DeepDance: Motion capture data of improvised dance (2019) (1.0) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.5838179



Pos. 1
A goal is selected and will be moved towards.

Pos. 2
*Goal "**raise_right_arm**" is reached, the rest of the body follows along the prerecorded dance.*

Pos. 3
*Goal "**sumo_stance**" is reached, this goal considers the whole body and none of the positions are copied from the prerecorded data.*

Pos. 4
*Goal "**point**" is reached. This goal also only considers one arm*

Pos. 5
*Goal "**crane**" is reached. This goal position is inspired by the karate kid crane kick.*

Pos. 6
*Goal "**hug**" is reached.*

Figure 1: How the agent follows goal positions. Pose names are arbitrary and not of importance to the algorithm and its workings.

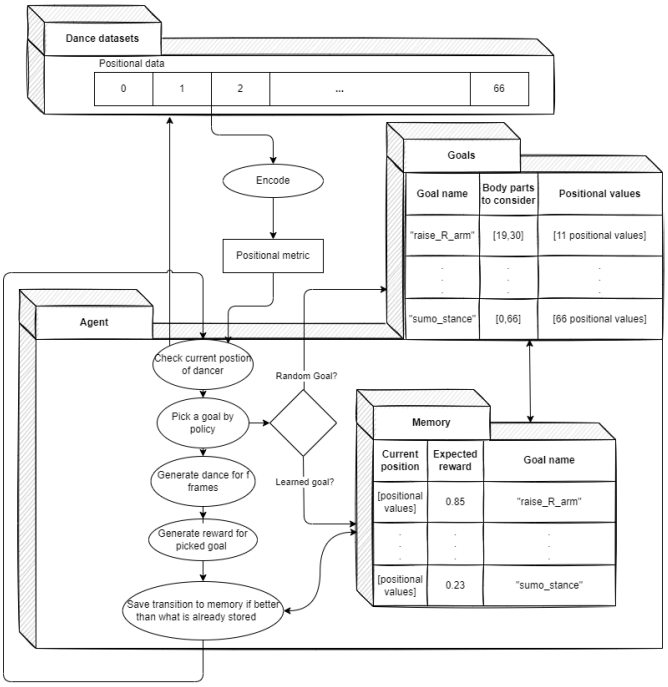


Figure 2: Overview of our reinforcement learning based approach

calculate a single metric for each frame. The metric is the average Euclidean distance from each of the 22 points to the center point. This approach allows for ease of evaluation, but does incur a loss of information.

We construct a memory unit (See Figure 2) to store the data. The unit we use functions as an ordered list, where the "memories" (state transitions) are ordered by the same metric as they are evaluated by. The memory unit has a finite size and will at some point during training be full. When trying to store a new memory while the unit is full, the unit will find the closest memories it has stored and replace one of them if it has worse expected reward than the new memory, if not it will be discarded.

2.2 Rewarding behaviour

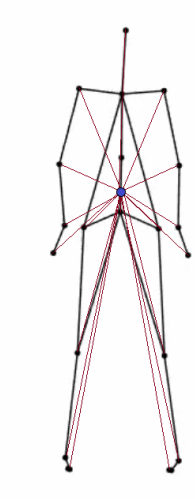
Salleh et. al. [10] suggest a 30/70% explore/exploit-ratio for optimal search in Swarm-Based Metaheuristic Algorithms. We use this information to create a bell curve distributed reward function that motivates 30% divergence, shown in Figure 3b.

2.3 Movement generation

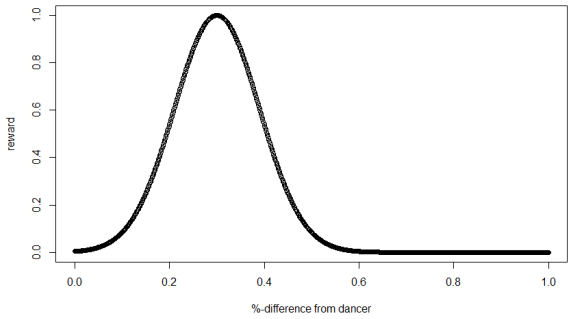
When generating new movement, the agent uses the input data for comparison. We use a hyper-parameter f, as the number of frames for an agent to follow a certain goal. A goal is either picked at random or sampled from the memory unit. After f frames, the movement is evaluated by calculating its difference from the observed dance and thereafter sent to the memory unit. Then a new goal is set using an epsilon greedy policy [19]. When an entire dance sequence is used, a new one is picked and the process repeats (Figure 2). When generating movement, the agent uses the delta values from the goal position to where the points currently are in the 3D-space. It then multiplies these values with a static, tuneable parameter from 0 to 1, which decides how fast the agent moves towards the goal.

2.4 Using PCA for goal-selection

A second approach using PCA has also been implemented. The PCA model is fitted to the full data set, reducing it down to three principal components. During goal selection we look back at the preceding 30 frames of generated dance data and project them down on three dimensions. The three values are averaged over the 30 frames to get the average of the three most principal components



(a) Figure showing what the data looks like when animated. Black dots represent the 22 3D positions of the dancer. The blue dot represents the calculated center point for the frame and the red lines represent the distance from the points representing the dancer to the center point.



(b) The reward function, a bell curve with maximum at a 30% difference.

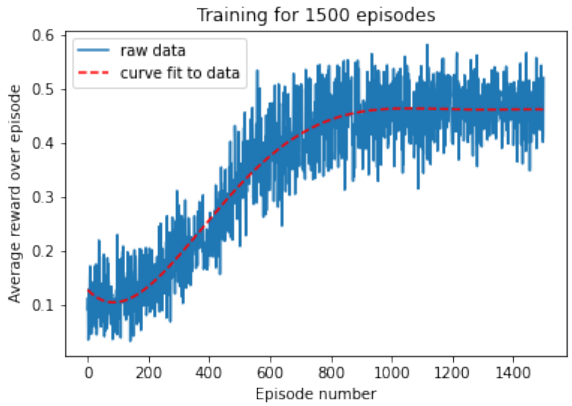
Figure 3: Setup for calculating rewards

for the given time span. These values are used to store the data in our memory unit.

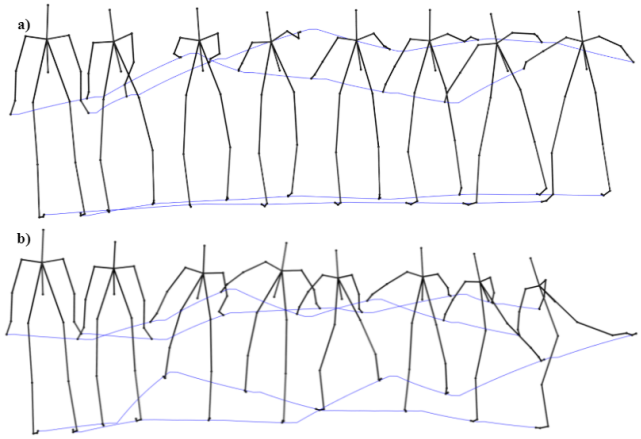
2.5 Visual comparison of generated dance motion

The goal of this comparison is to identify visual differences between the implementations. It is conducted as a side by side comparison by the authors. As a benchmark, a random goal selection strategy is used, in which the agent does not utilize its internal memory but picks goals from the goal pool randomly. For each instance of input data, we compare the three different goal selection strategies (random, PCA and center point difference), as well as the input motion.

3 RESULTS



(a) Results of RL trained agent (without PCA implementation) showing convergence towards a reward value of about 0.46. Individual runs still vary quite a lot, but become better on average over time. The red curve fit to the data shows the general reward trend of the trained agent.



(b) Keyframes of the first couple of seconds of generated dance with the agent. a) Goals are picked at random. b) Goals are picked by the agent (No PCA algorithm. Center point deviation evaluation.).

Figure 4

Results show a steady increase in average reward over the episodes. Since each episode tackles a different dance sequence than the next, the average reward for each subsequent episode varies. The agent optimizes for the general data set, and will, on average, do well on most of the data. On the other hand, we can clearly see that it does not work as well for some of the dance sequences. Figure 4a shows that when in the later stages of the training we still get some episodes with quite low average reward. This run was done by setting a goal for every five frames. The average performance for each episode converges towards a reward of about 0.46. Since we measure the average reward of each episode there is a varying degree of success within them. Note that the rewards presented in Figure 4a were calculated by setting a goal for five frames at a time

while the results used in the visual evaluation come from setting a goal for ten frames at a time. The trend seen in 4a is the same seen for ten frames.

3.1 Visual comparison

Visually, none of the generated dances correlates with the original input motion capture, and all generally maintain an even pace. The randomly generated dance performs small, smooth, and disorderly movements, only within a small scope of its available movement space. The RL-algorithms movements are more compound, resulting in larger, more sudden changes in posture (See 4b), although sometimes sharp and abrupt, breaking the flow. The RL PCA agent falls in between the two former agents. It performs large compound movements without the abrupt changes, in a fixed fast-paced tempo, being perceived as energetic, but not as diverse as the RL implementation. Videos available on YouTube[2].

3.2 Quantitative measurements

To contribute to assessing the flow and thus dance-like quality of the agents generated movement, fluidity values were calculated. The fluidity measurements indicate the level of flow and circularity present in the movement. It gives the ratio between velocity and acceleration of the normalized and averaged data. The larger the fluidity measurement, the greater the fluidity of the movements. The fluidity measurements of the RL agent ranged between 0.1720-0.1803, and 0.1669-0.1769 for the RL-PCA agent. The random agent ranged between 0.1651-0.1699.

We also measured the cumulative distances traveled by each of the markers in the generated movements. The RL agent had an average cumulative distance of 6.4×10^5, RL-PCA agent had a value of 6.7×10^5, and the random agent had a value of 4.8×10^5.

4 DISCUSSION

4.1 Visual comparison

The RL algorithm seems to make improvements, as the two RL implementations, with their more compound movements, ordered transitions, and pose choices perform better than the random algorithm. The RL-PCA is considered the best, being perceived as smoother, more energetic, and lacking irregularities, implying that the PCA provides smoother transitions and better flow. Input dance and generated dance seem unrelated, implying "novelty" in this sense, and is presumably caused by the center point evaluation. More complex methods might decrease the deviation. Also, the 30% differentiation-ratio might be too high. The monotone tempo is presumably caused by the fixed goal-picking pace. The algorithms seems to improve the peformance, but we will not claim that the results are genuinely creative or novel.

4.2 Measurements

On average the RL agent had slightly larger fluidity values than the PCA-RL agent, indicating that the RL agent has the smoothest generated movement. This is surprising due to the visual comparison concluding the PCA-RL to be smoother. Both trained agents had larger fluidity values than the random agent. The differences are small, and fluidity may not be a significant performance measurement. On average the PCA-RL agent had the largest cumulative distance. The PCA-RL agent and the RL agent have very similar average cumulative distances, while the random agent has a much smaller cumulative distance than the other two. This agrees with the visual observations of the trained agents' generated movements being larger and more compound than the random agent's movements.

[2]https://youtube.com/playlist?list=PLtPJR9AsAyVVMruA04l7FbuDjLYCKhoVE

4.3 Limitations and future work

A lot of information gets lost during the center point calculation. Better positional simplification could perhaps solve this. The reward system makes it nearly impossible for the agent to reach full reward, as it requires there to be a 30% perfectly deviating goal to pick for every frame in every sequence. The visual comparison has a very limited selection of five results to evaluate. To get a more holistic impression, more videos should be generated and compared. Identifying creativity is challenging, and often subjective [5]. Having only the authors performing the comparisons brings limitations. Having a larger representation of different, independent assessors may increase the comparisons' quality. A perceptual judgment experiment could be conducted [14]. The Wiggins CSF model [20] may also be a good alternative.

5 CONCLUSION

We have attempted a new implementation of an RL and PCA-based agent for dance generation, with a reward structure that motivates divergence from the original data set. It learns by observing improvisational-dance motion recordings. When presented with new dance sequences, the trained agent will try to improvise its own novel and coherent dance. Defining creativity in generative AI is challenging. From our results, it is hard to define anything the agent does as creative. Movements generated by the RL and the RL-PCA implementations are larger, more compound and coherent than of the random agent. Although lacking the expected flow and choreography of a dance performance, they can be considered as some form of dance. There is still a long way before this approach can create truly human-like dance performances, but it provides a groundwork that can be expanded and developed.

ACKNOWLEDGMENTS

This work was partially supported by the Research Council of Norway through the Collaboration on Intelligent Machines (COINMAC) project, under grant agreement no. 309869 and its Centres of Excellence scheme, project number 262762.

REFERENCES

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

[2] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[3] Biao Jia, Chen Fang, Jonathan Brandt, Byungmoon Kim, and Dinesh Manocha. 2019. Paintbot: A reinforcement learning approach for natural media painting. *arXiv preprint arXiv:1904.02201* (2019).

[4] Seyed Mojtaba Karbasi, Halvor Sogn Haug, Mia-Katrin Kvalsund, Michael Joseph Krzyzaniak, and Jim Torresen. 2021. A Generative Model for Creating Musical Rhythms with Deep Reinforcement Learning. In *2nd Conference on AI Music Creativity.*

[5] James C Kaufman and John Baer. 2012. Beyond new and appropriate: Who decides what is creative? *Creativity Research Journal* 24, 1 (2012), 83–91.
[6] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. 2016. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *Advances in neural information processing systems* 29 (2016), 3675–3683.
[7] Hsin-Ying Lee, Xiaodong Yang, Ming-Yu Liu, Ting-Chun Wang, Yu-Ding Lu, Ming-Hsuan Yang, and Jan Kautz. 2019. Dancing to music. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 3586–3596.
[8] Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. 2021. Learn to Dance with AIST++: Music Conditioned 3D Dance Generation. arXiv:2101.08779 [cs.CV]
[9] Qinggang Meng, Ibrahim Tholley, and Paul W. H. Chung. 2014. Robots learn to dance through interaction with humans. *Neural Computing and Applications* 24, 1 (01 Jan 2014), 117–124. https://doi.org/10.1007/s00521-013-1504-x
[10] Mohd Najib Mohd Salleh, Kashif Hussain, Shi Cheng, Yuhui Shi, Arshad Muhammad, Ghufran Ullah, and Rashid Naseem. 2018. Exploration and exploitation measurement in swarm-based metaheuristic algorithms: an empirical analysis. In *International conference on soft computing and data mining*. Springer, 24–32.
[11] Jürgen Schmidhuber. 2006. Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. *Connection Science* 18, 2 (2006), 173–187.
[12] Jürgen Schmidhuber. 2010. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE transactions on autonomous mental development* 2, 3 (2010), 230–247.
[13] Benjamin D Smith and Guy E Garnett. 2012. Reinforcement learning and the creative, automated music improviser. In *International Conference on Evolutionary and Biologically Inspired Music and Art*. Springer, 223–234.
[14] Bob L Sturm and Oded Ben-Tal. 2017. Taking the models back to music practice: evaluating generative transcription models built using deep learning. *Journal of Creative Music Systems* 2, 1 (2017).
[15] Taoran Tang, Jia Jia, and Hanyang Mao. 2018. Dance with melody: An LSTM-autoencoder approach to music-oriented dance synthesis. In *Proceedings of the 26th ACM international conference on Multimedia*. 1598–1606.
[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.
[17] Benedikte Wallace, Charles P. Martin, Jim Torresen, and Kristian Nymoen. 2020. Towards Movement Generation with Audio Features. In *Proceedings of the 11th International Conference on Computational Creativity*.
[18] Benedikte Wallace, Charles P Martin, Jim Tørresen, and Kristian Nymoen. 2021. Learning Embodied Sound-Motion Mappings: Evaluating AI-Generated Dance Improvisation. In *Creativity and Cognition*. 1–9.
[19] Christopher John Cornish Hellaby Watkins. 1989. Learning from delayed rewards. (1989).
[20] Geraint A Wiggins. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems* 19, 7 (2006), 449–458.
[21] Ning Xie, Hirotaka Hachiya, and Masashi Sugiyama. 2013. Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. *IEICE TRANSACTIONS on Information and Systems* 96, 5 (2013), 1134–1144.