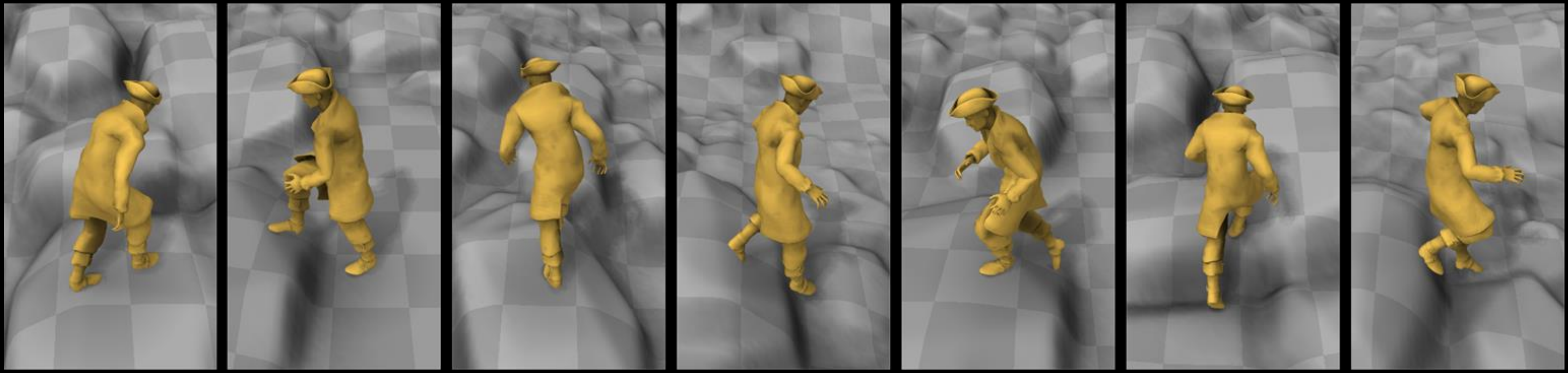
30 JULY – 3 AUGUST Los Angeles
SIGGRAPH 2017

PHASE-FUNCTIONED NEURAL NETWORKS FOR CHARACTER CONTROL

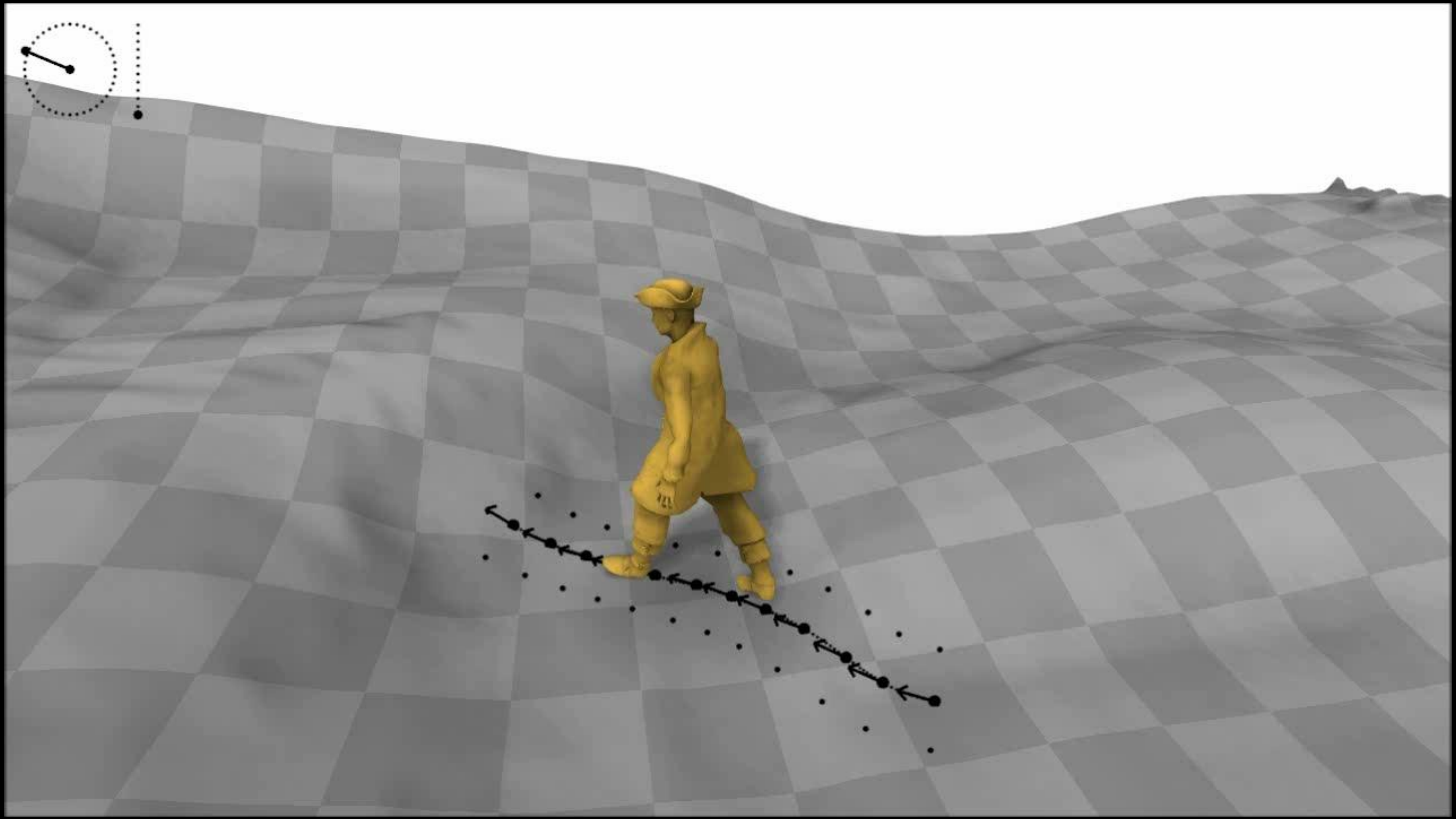
DANIEL HOLDEN, UNIVERSITY OF EDINBURGH
TAKU KOMURA, UNIVERSITY OF EDINBURGH
JUN SAITO, METHOD STUDIOS

RESEARCH GOAL

Fast, compact, expressive character controller for games



FINAL RESULT



OVERVIEW

Background

Data Capture

Neural Network

Results

Conclusion

PREVIOUS WORK – MOTION X

Motion Graphs - Motion Fields - Motion Matching

[Kovar et al. 2002]
[Lee et al. 2002]
[Arikan et al. 2002]

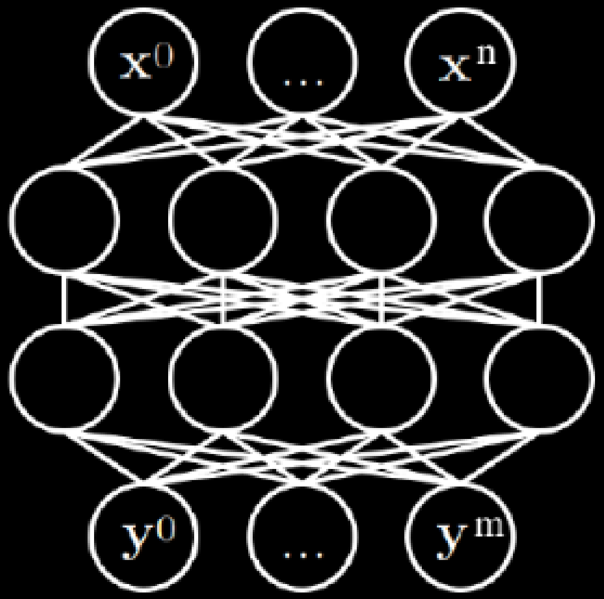
[Lee et al. 2010]

[Büttner 2015]
[Clavet 2016]

- **Scalability:**
  - Require full motion database to be stored in memory.
  - Require manual processing of data by artists (often).
  - Require tricky acceleration structures (e.g. kd-tree).

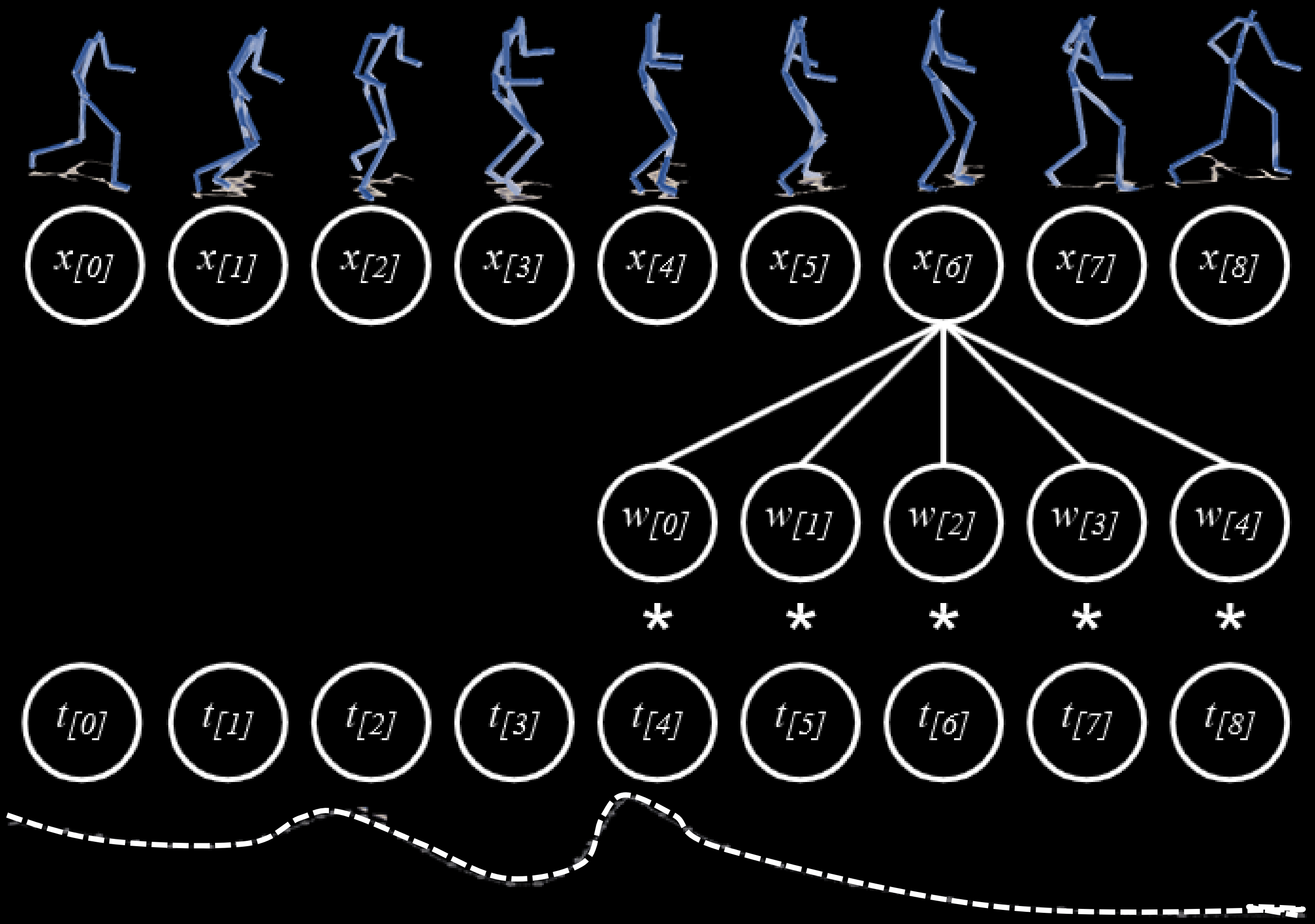
CAN NEURAL NETWORKS HELP?

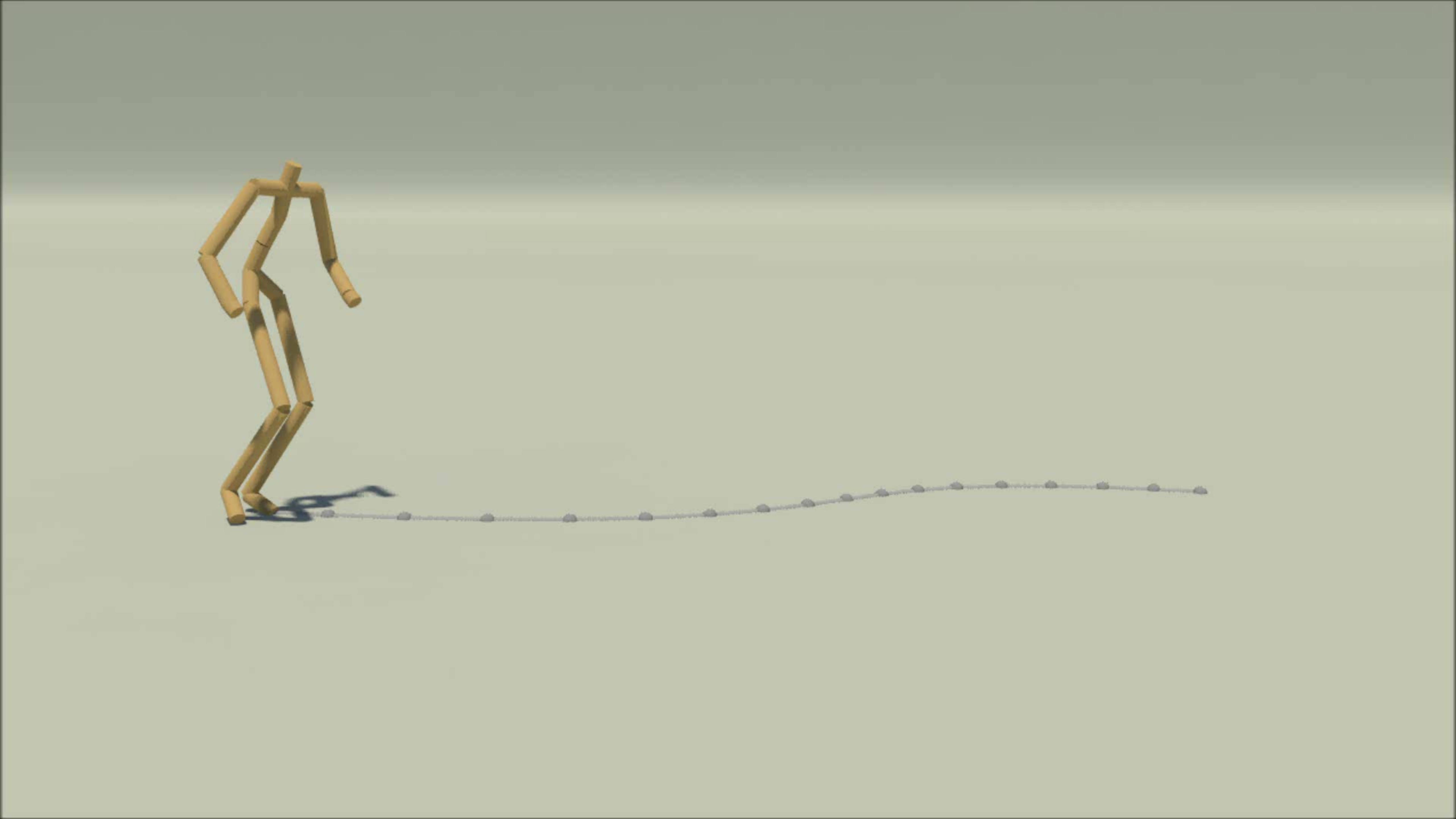
- **High Scalability**:
  - Virtually unlimited data capacity.
  - Fast runtime / low memory usage.
- **But how can they be used for motion generation?**



CONVOLUTIONAL NEURAL NETWORKS

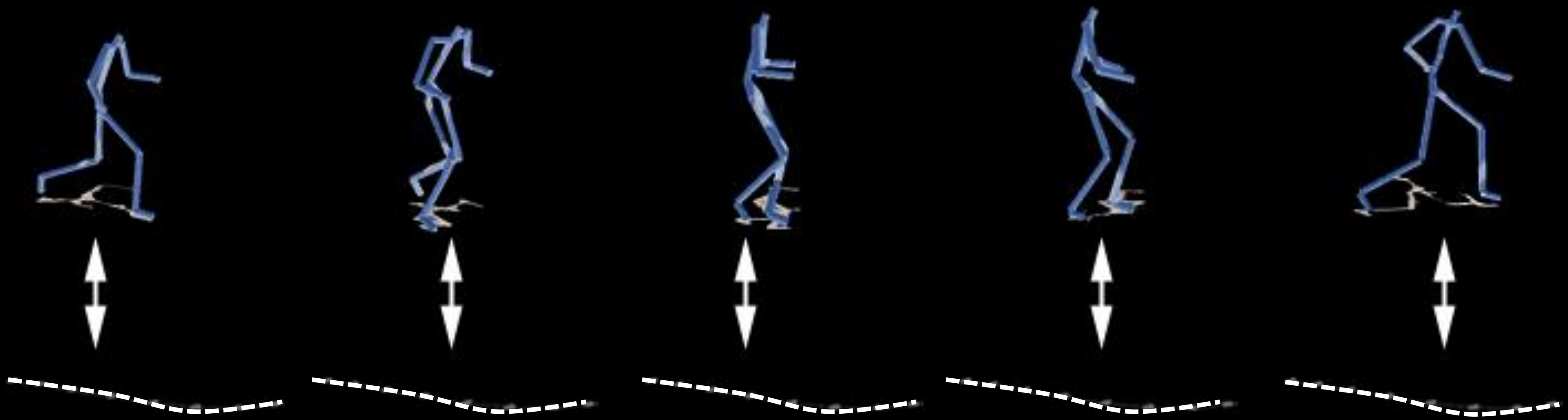
Learn a mapping from a user control signal to a motion





WHAT HAPPENED?

- **Ambiguity**: same input maps to multiple different motions.

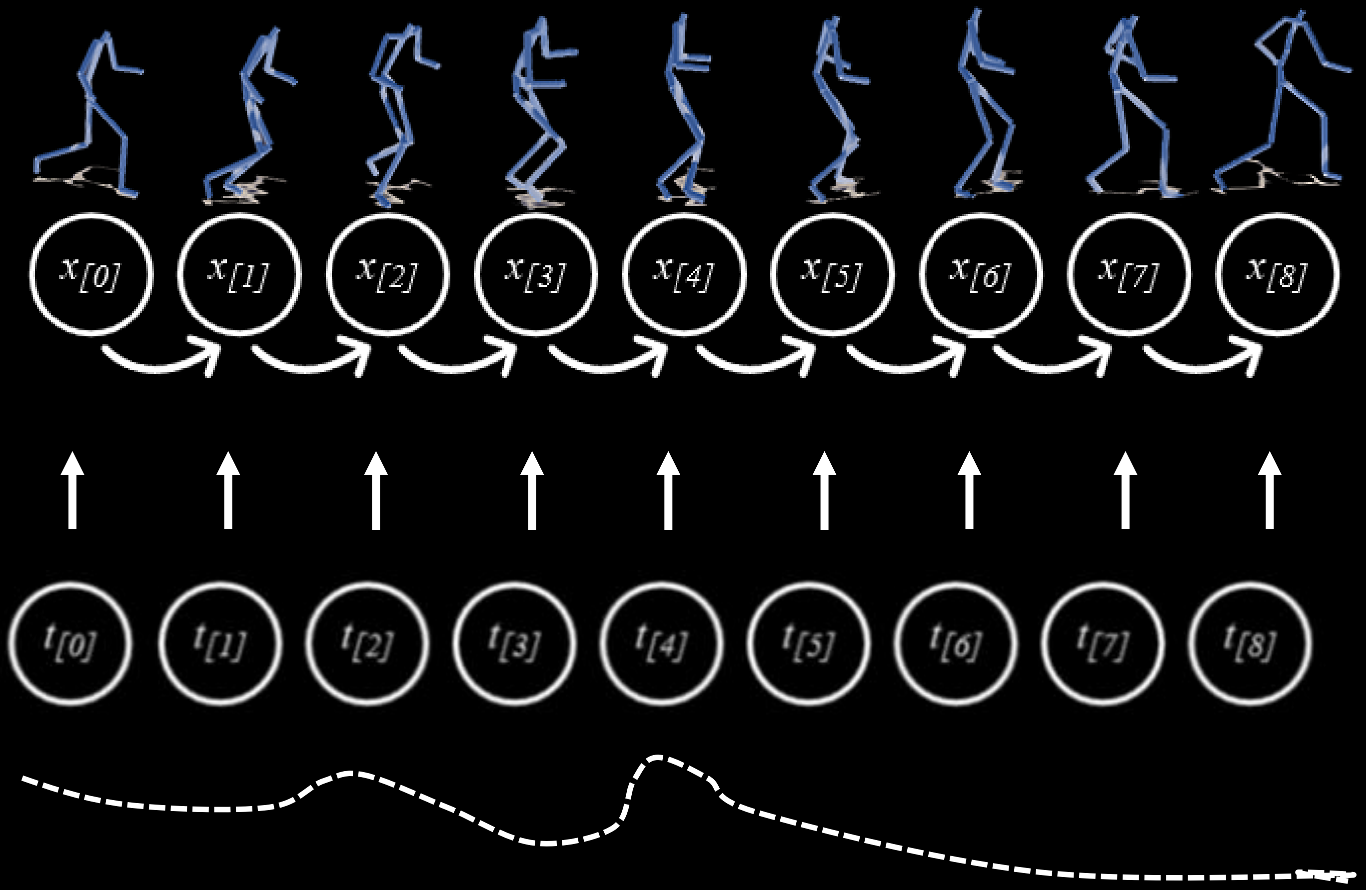


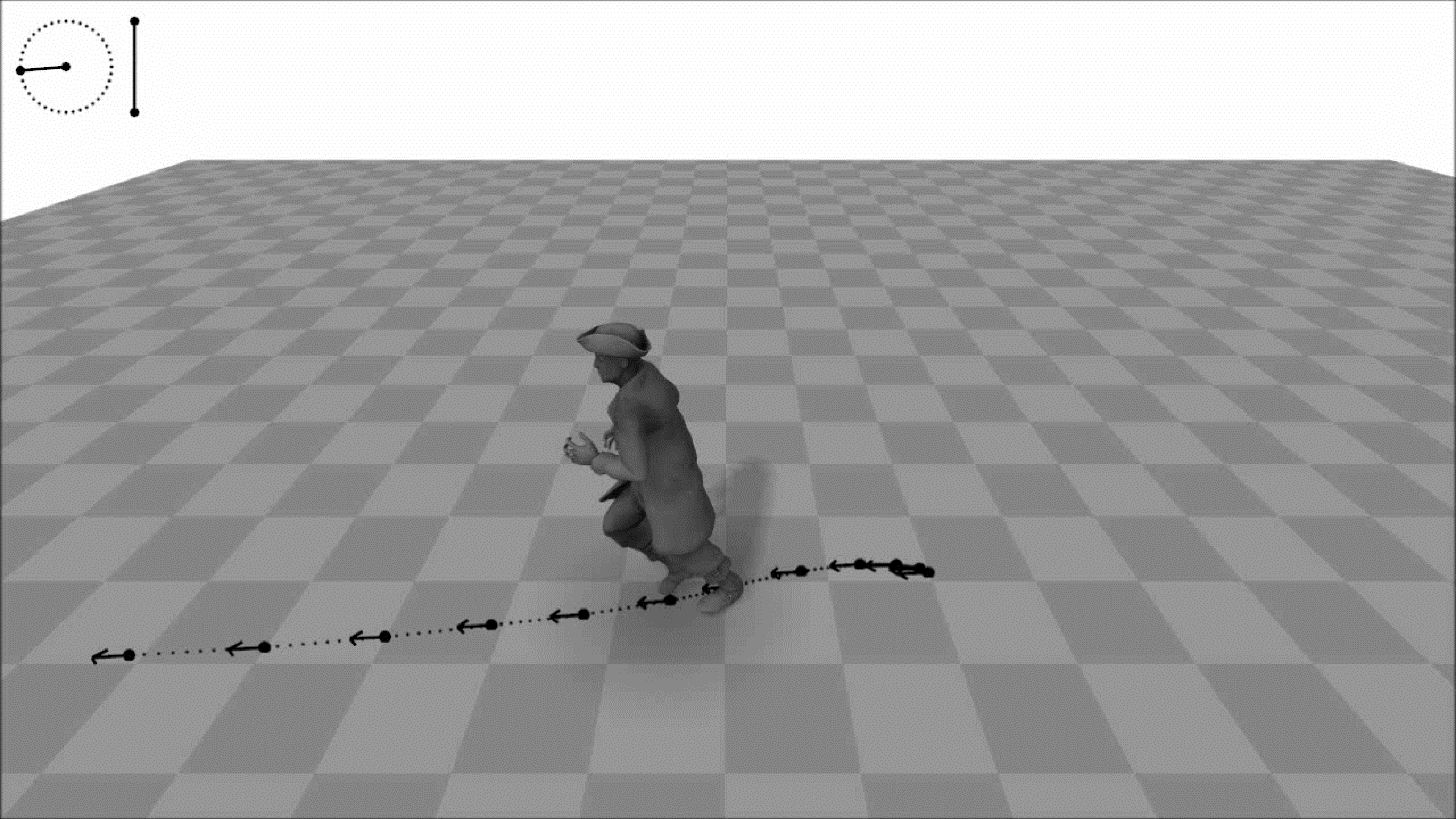
CONVOLUTIONAL NEURAL NETWORKS

- **Practicality:**
  - Require a trick to remove the ambiguity in the input [Holden et al. 2016].
  - Whole input trajectory must be given beforehand.
  - Multi-layer CNNs are still too slow for games.

RECURRENT NEURAL NETWORKS

Learn a mapping from the previous frame(s) to next.





RECURRENT NEURAL NETWORKS

- **Quality:**
  - State of the art produces ~10 seconds before “dying out”
    [Fragkiadaki et al. 2015]
  - Difficult to avoid “floating”.
  - Still has issues of ambiguity.

SUMMARY

- **Scalability**:
  - How can we scale to large amounts of data?

- **Ambiguity**:
  - How do we solve the ambiguity problem?

- **Quality**:
  - How can we make the generated motion look good?

OVERVIEW

Background

Data Capture

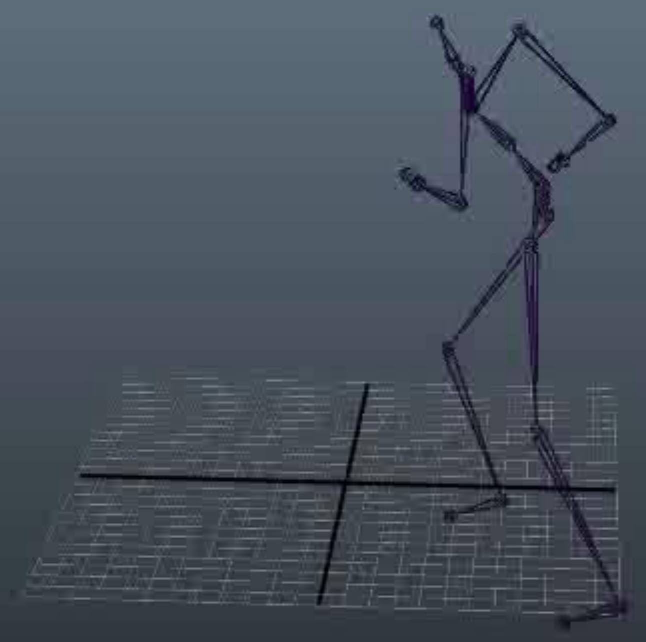
Neural Network

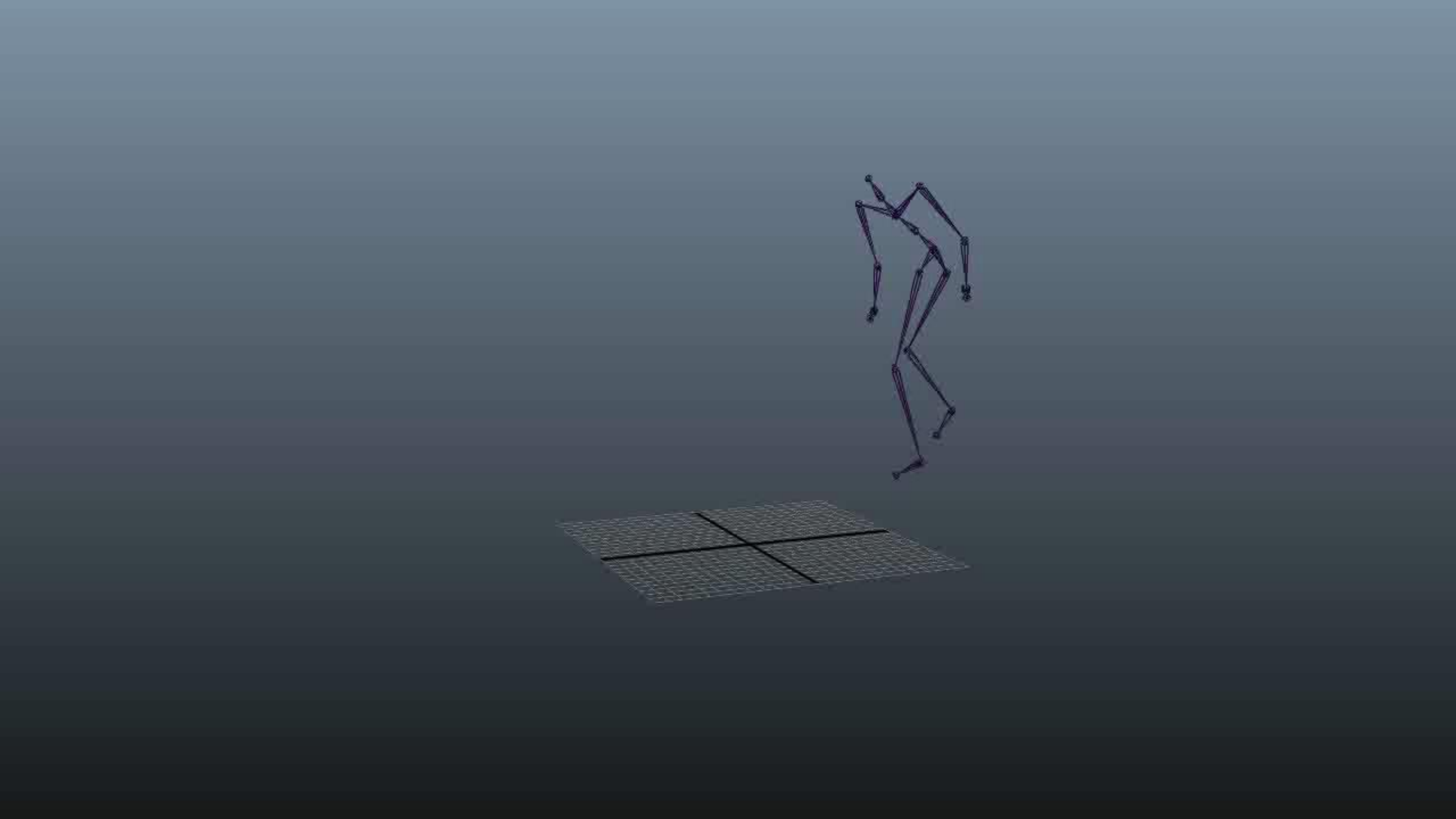
Results

Conclusion

DATA CAPTURE

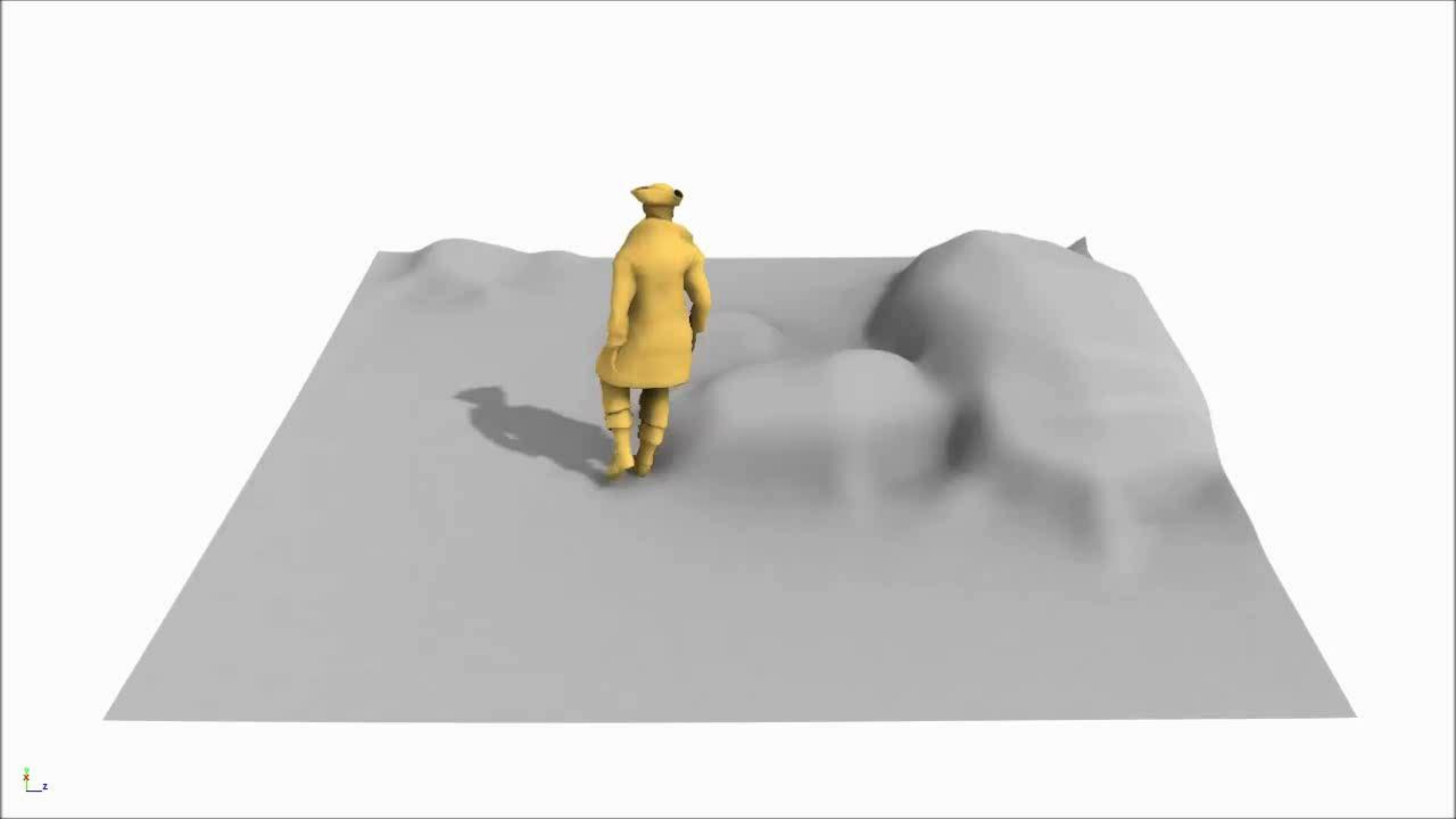
- **Unstructured data capture:**
  - Around 2 hours of raw locomotion mocap data (~1.5 GB).
  - Each capture is around 10 minutes long.
  - Each contains a mixture of gaits, facing directions, etc.
  - We placed chairs, tables in capture volume to climb over.





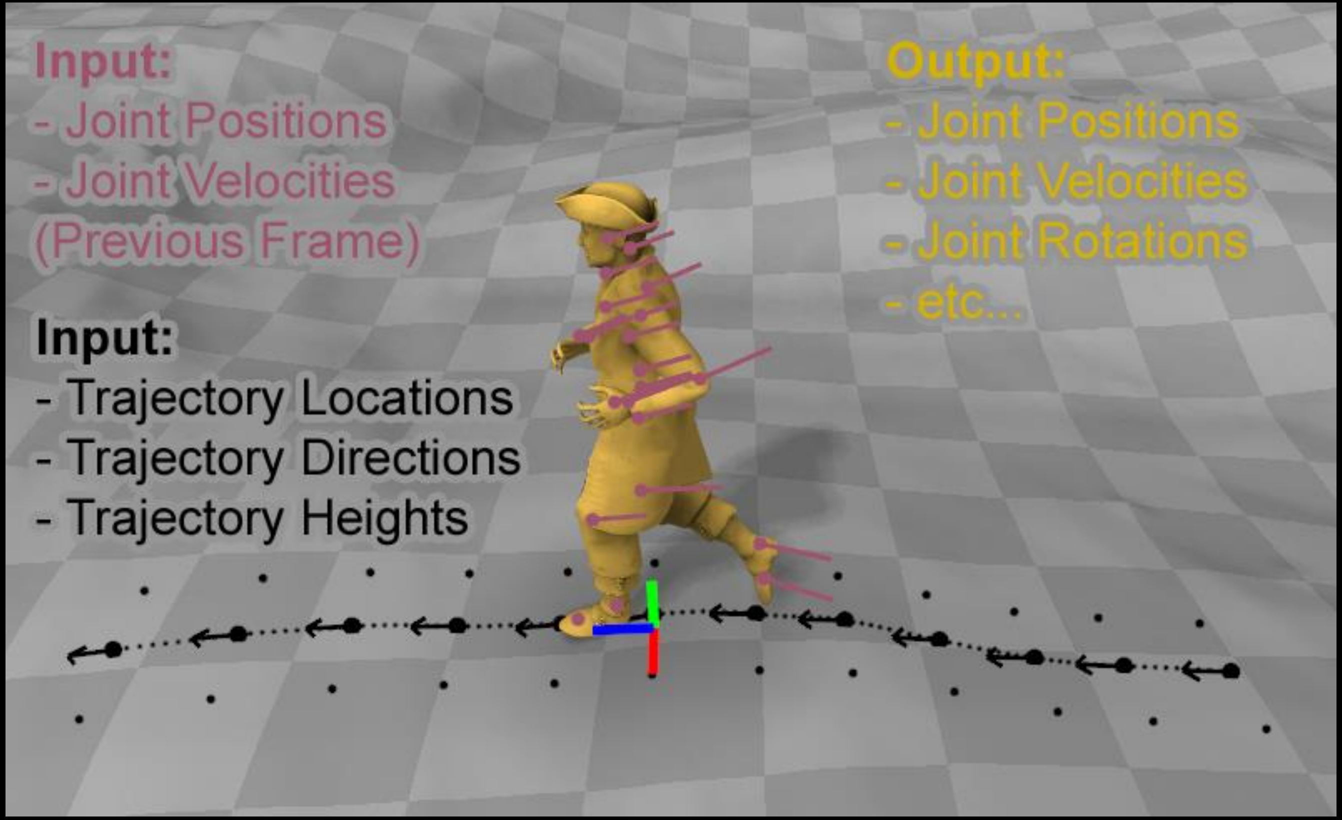
TERRAIN FITTING

- We want to have terrain geometry to learn from alongside motion.
- But capturing motion and geometry together is difficult.
- Make a database of heightmaps and fit patches from it to each locomotion cycle.



PARAMETERISATION

- Has a large effect on the final quality.
- Window of the trajectory local to the character.
- We add **gait**, **terrain height**, and other variables.



OVERVIEW

PHASE-FUNCTIONED NEURAL NETWORK

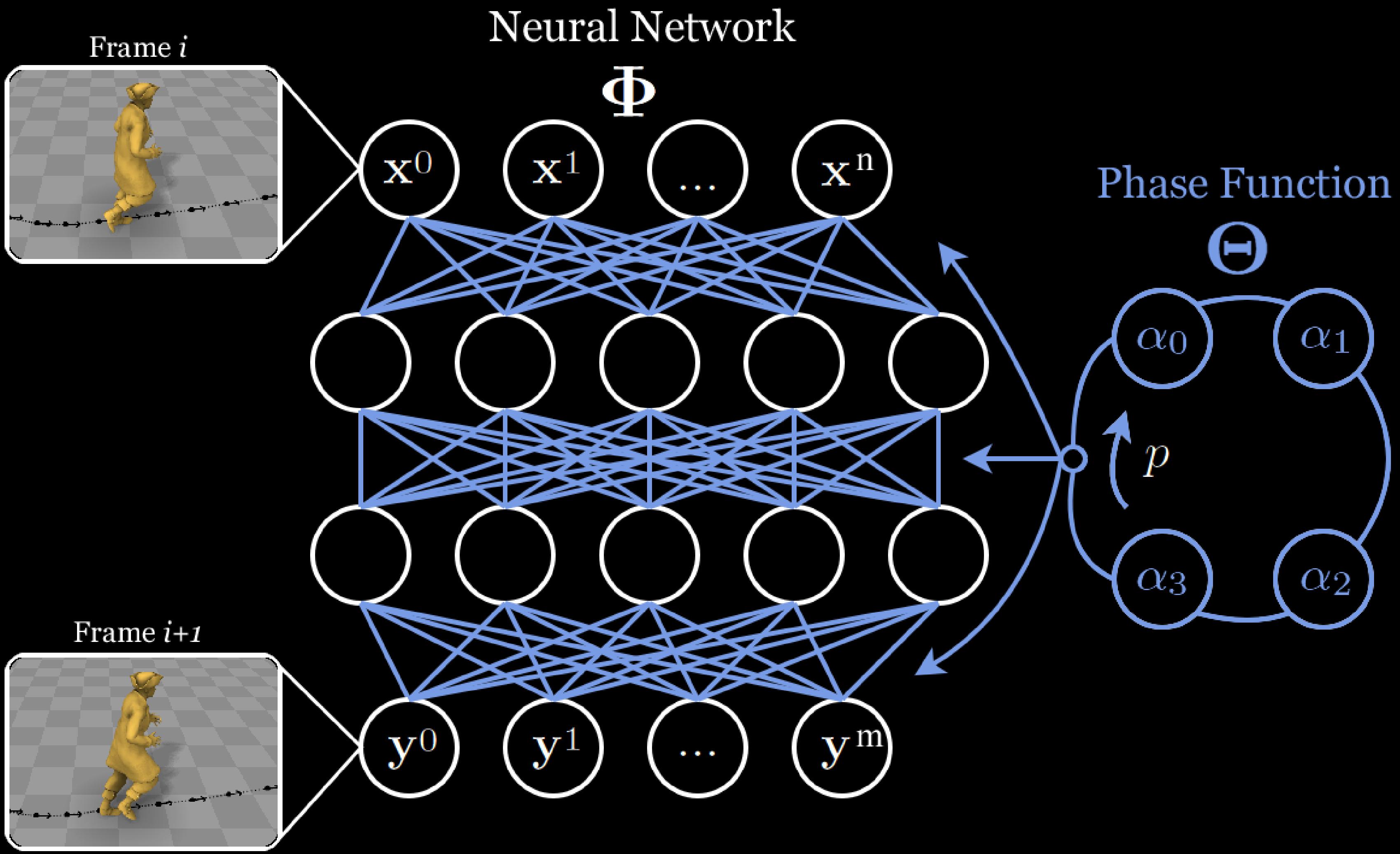
A Neural Network where the weights are generated as a function of the phase.

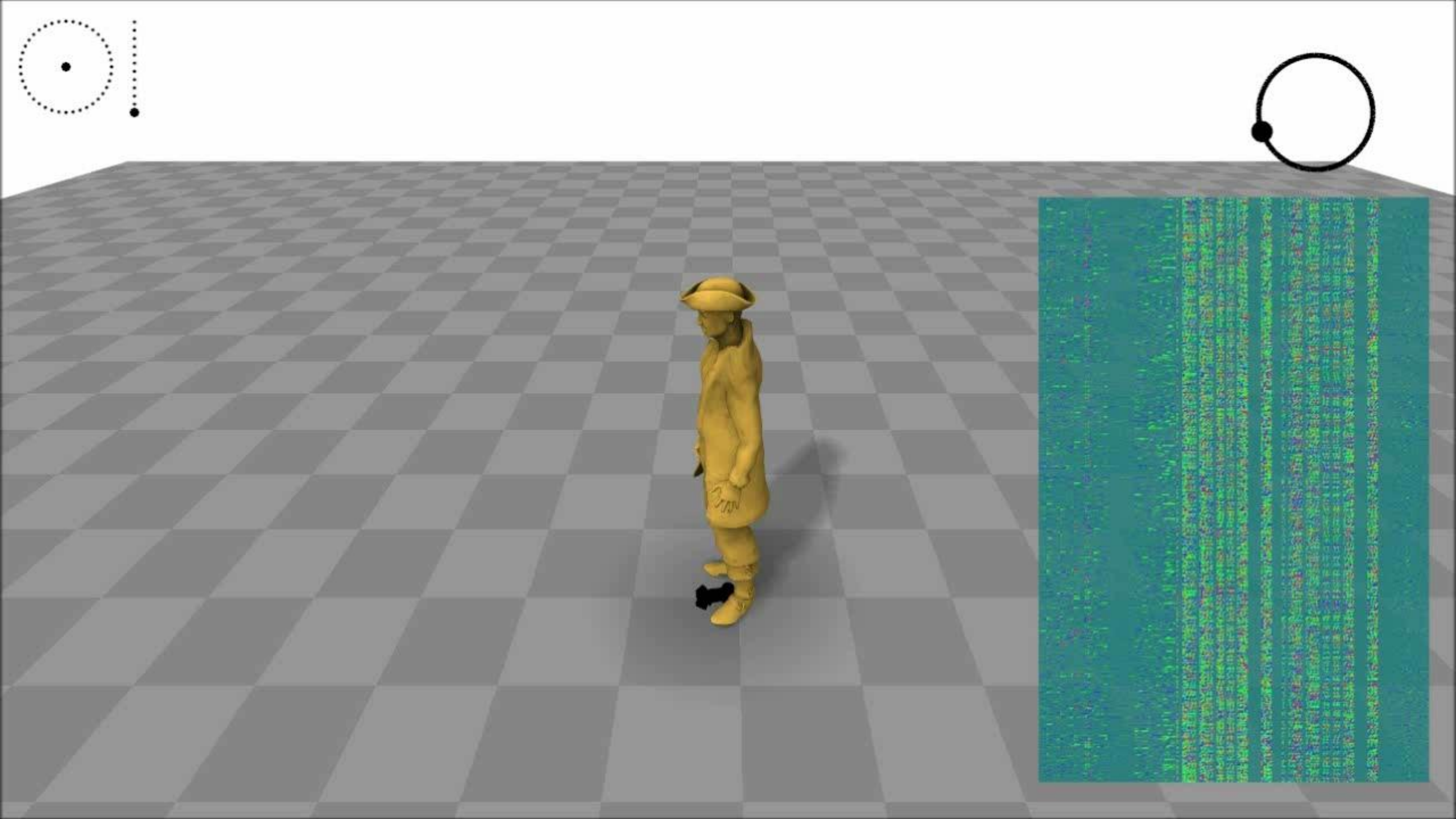
THE PHASE

The “phase” is the scalar variable in the range 0 to 2π representing the point in time of the current pose in the locomotion cycle.

- **Given the phase:**
  - The pose of the character is far less ambiguous.
  - The space of poses is smaller and more convex.
  - The average pose is not the character “floating”.

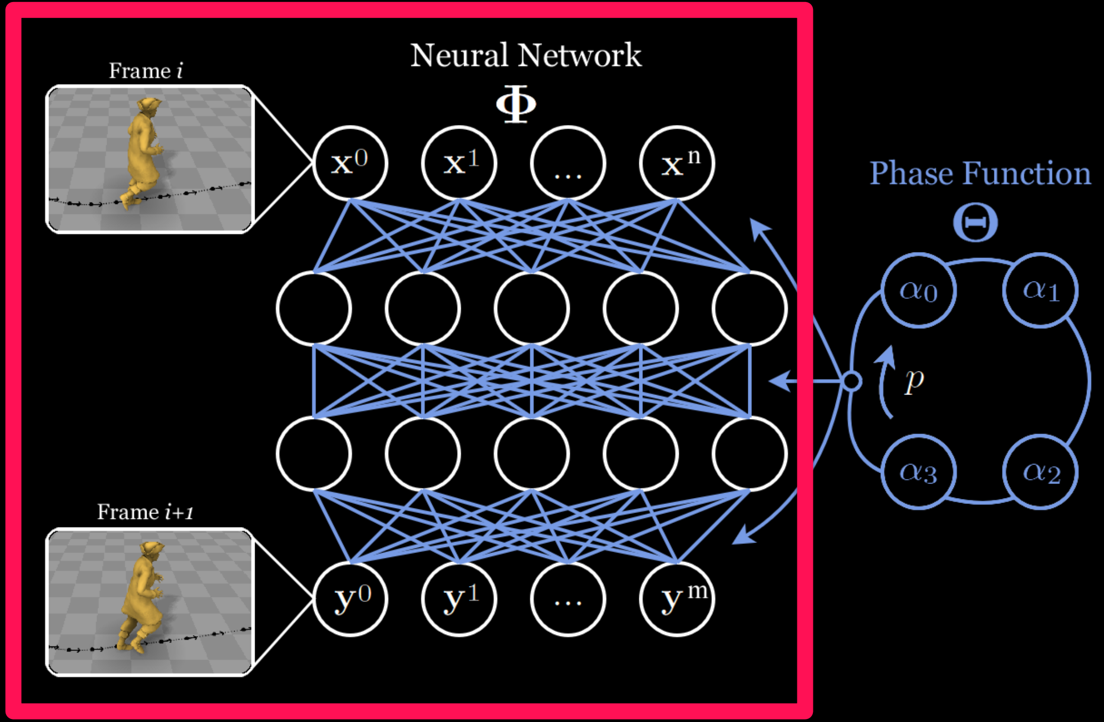
PHASE-FUNCTIONED NEURAL NETWORK





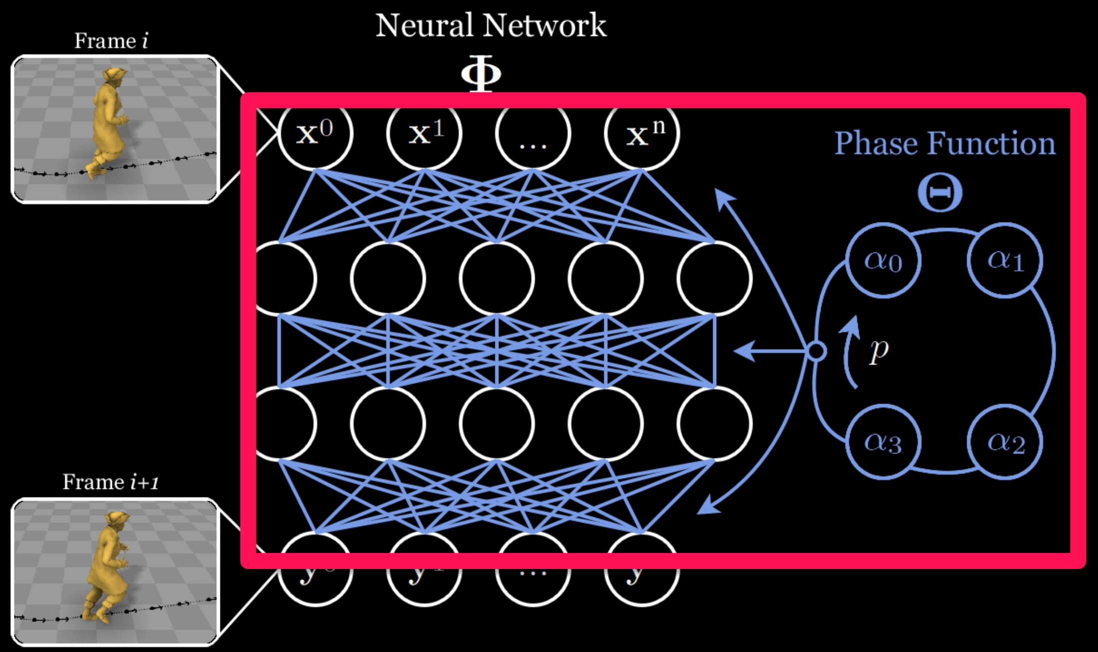
NEURAL NETWORK

- Feed-Forward Neural Network.
- 2 hidden layers.
- 512 hidden units per layer.
- ELU activation function.



PHASE FUNCTION

- Outputs neural network weights.
- Cyclic cubic spline function interpolating 4 *control points*.
- Each *control point* is effectively a set of neural network weights.



TRAINING ALGORITHM

1. Input phase p in phase function to generate network weights $\boldsymbol{\alpha}$.

2. Using weights $\boldsymbol{\alpha}$, input $\boldsymbol{x}$ into neural network to generate output $\boldsymbol{y}$.

3. Measure error in output $\boldsymbol{y}$.

4. Back-propagate error through **both** neural network **and** phase function to update values of *control points*.

OVERVIEW

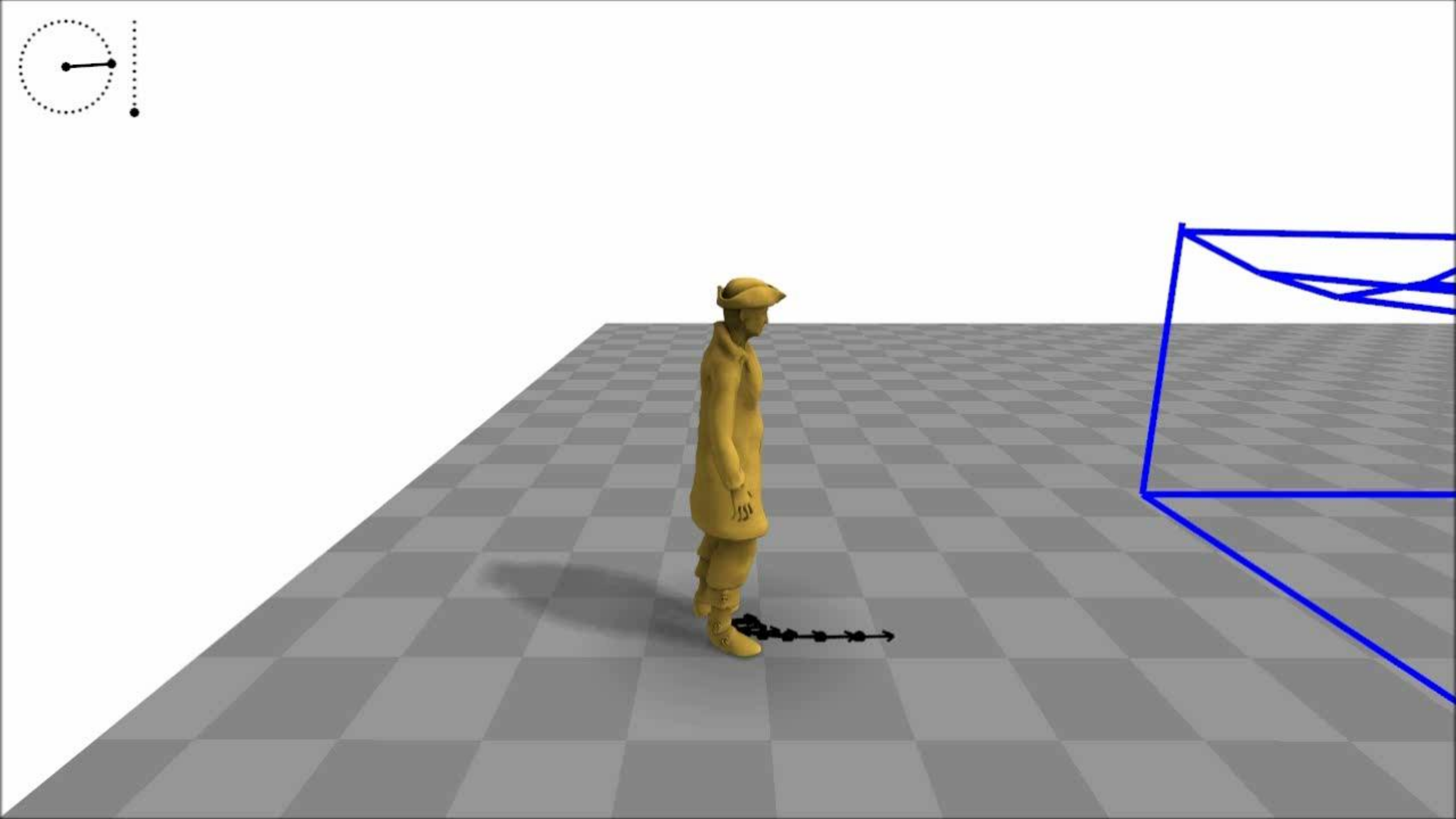
Background

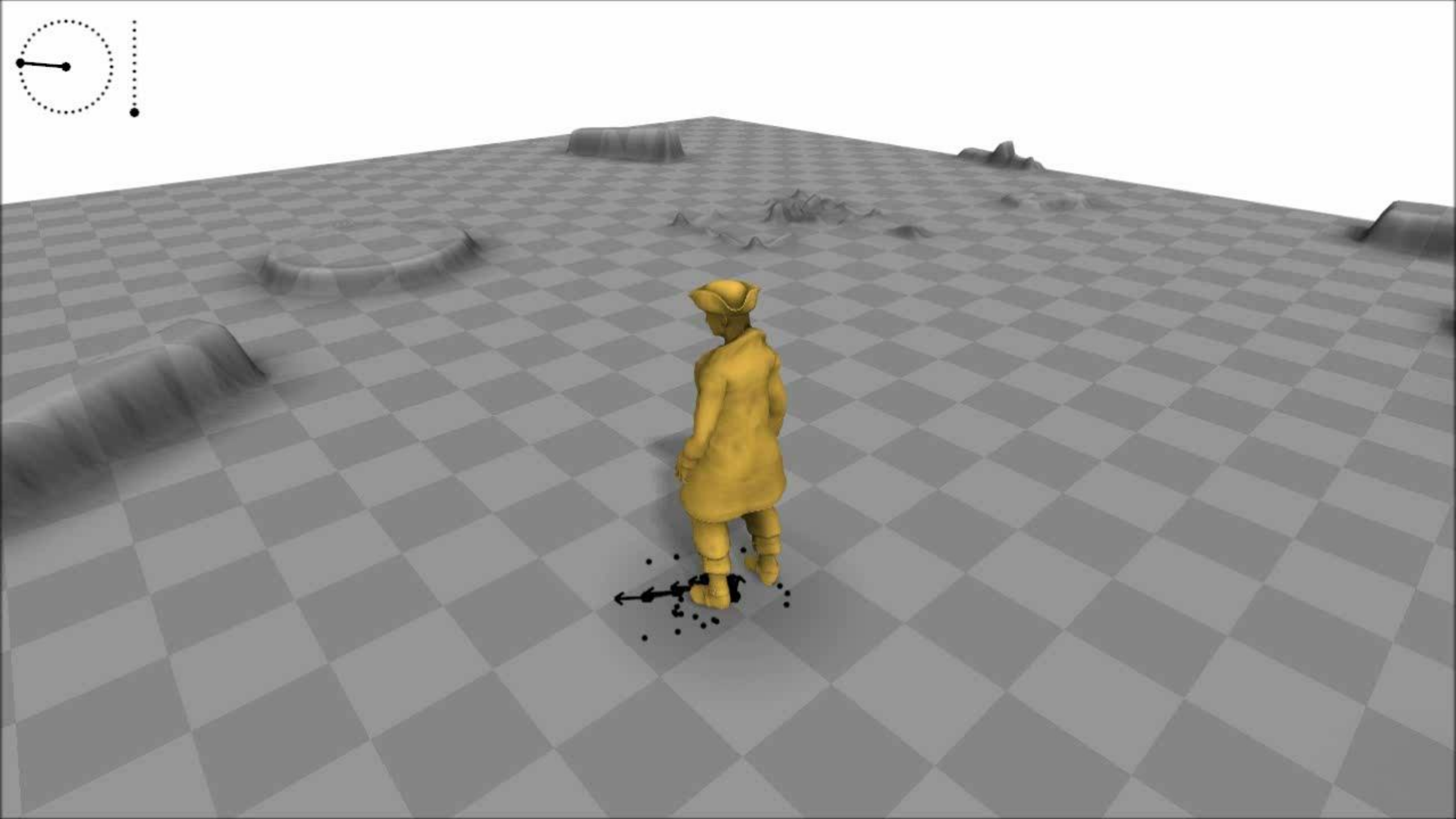
Data Capture

Neural Network

Results

Conclusion





OVERVIEW

Background

Data Capture

Neural Network

Results

Conclusion

PHASE-FUNCTION PRECOMPUTATION

Computation of phase function is a bit slow (for games).

- **Pre-compute phase function:**
  - Phase is scalar in range $0 \leq p \leq 2\pi$.
  - We can pre-compute phase function in this range.
  - Interpolate pre-computed values at runtime.
  - Provides a trade-off between memory and speed.

PERFORMANCE

Runtime	Memory
1.8 ms	10 mb
or	
0.8 ms	125 mb

NEGATIVES

- **Training Time:**
  - Longer than usual as each mini-batch item has different phase.
- **Artistic Control:**
  - Difficult for artists to direct / edit outcome of this kind of setup.
- **Unpredictability:**
  - Difficult to predict what the results will be like and why.

POSITIVES

- **Scalability**:
  - Neural Networks can easily scale to huge amounts of data.
- **Ambiguity**:
  - Factoring out the phase very effectively reduces ambiguity.
- **Quality**:
  - Good parametrisation and simple structure helps control quality.

QUESTIONS?