

Developer

/ Documentation

/ Unreal Engine ▾

/ Unreal Engine 5.4 Documentation

/ Making Interactive Experiences

/ Physics

/ Cloth Simulation

/ Machine Learning Cloth Simulation Overview

Machine Learning Cloth Simulation Overview

Overview of Machine Learning Cloth Simulation in Unreal Engine.



Introducing the Machine Learning Cloth Simulation System



The **Machine-Learning (ML) Cloth Simulation** system provides high-fidelity, highly performant real-time cloth simulation in Unreal Engine.

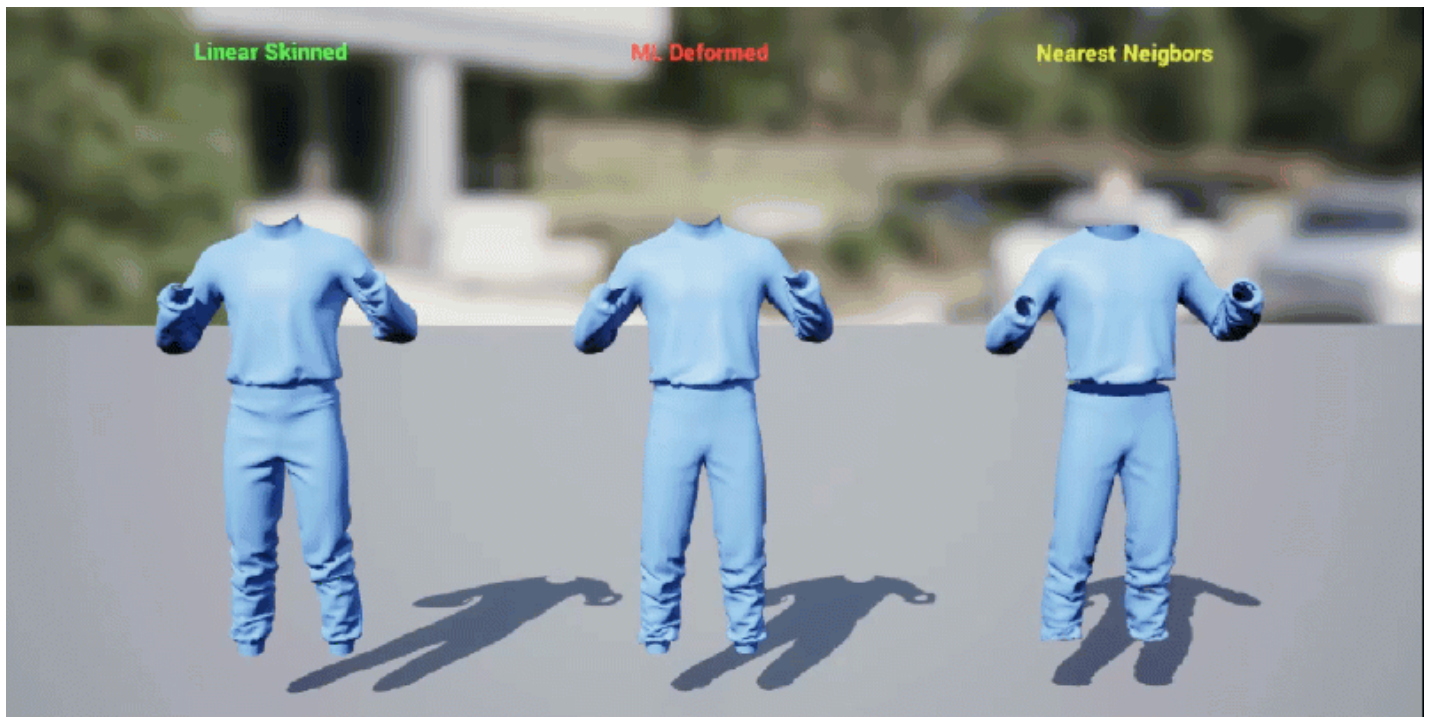
The system provides higher fidelity than the traditional physics-based model by using a trained dataset that can be used in real-time to produce results that were previously only achievable with offline simulation.

Benefits of using Machine Learning

In traditional game development, you simulate character clothing by using a physics solver. However, this process can be computationally intensive due to the time-consuming updates for elasticity, and the complicated interactions between cloth and body.

A machine learning system can assist users in training a model using high-quality simulation data generated in advance. This system is capable of producing clothing meshes that are of similar quality to pre-simulated data, while being fast and efficient in terms of memory usage.

Technical Implementation



For each frame, the ML Cloth Simulation system takes the character's skeletal pose as input and predicts a final cloth pose by deforming a base cloth mesh using linear blend skinning. The deformation has two major components.

The first component is a **low-frequency deformation** similar to the neural morph model.

This component is modeled with a multi-layer perceptron (MLP) network, which predicts a set of coefficients using the Skeletal Mesh pose as input. The low-frequency deltas are computed as:

```
low_frequency_deltas = mean_delta + coeffs * basis
```

 Copy full snippet


The basis can be learned at training time or pre-computed using principal component analysis (PCA).

To train this component, you should prepare a dataset of random poses and simulate clothing to settle on each pose. For this component, we recommend 5000 poses for a humanoid character.

The second component is a **high-frequency deformation** computed using the **Nearest Neighbor** search.

This component takes the coefficients from the first component and applies a nearest neighbor search in a Nearest Neighbor dataset, namely:

```
high_frequency_deltas = NearestNeighborSearch(coeffs)
```

 Copy full snippet

For this component, one should prepare a small, but diverse set of poses and simulate clothing to settle on each of the poses. Ideally the poses should be taken from some key frames of the animations in-game. For optimal results, between 50 and 100 poses are recommended for a humanoid character.

Using this technique, the vertex delta is calculated as follows:

```
vertex_delta = low_frequency_deltas + high_frequency_deltas
```

 Copy full snippet

For the trained dataset, you can generate the final cloth position for each pose by using the **Chaos Cloth Generator** tool under the **ML Deformer Editor** in Unreal Engine. This will simulate the cloth and generate a Geometry Cache that can be used in real-time.

Alternatively, you can use external cloth simulation solvers (such as Houdini Vellum). For each pose, simulate the cloth until settlement and save the settled cloth in the Geometry Cache. You can import the cache into Unreal Engine as an Alembic file.



Follow the [ML Cloth Generation tutorial](#) to learn how to generate the Geometry Cache inside Unreal Engine.

For best results, you should manually choose the most relevant poses for the dataset. However, it is possible to do automatic pose selection with the **KMeans Pose Generator**. The generator takes an animation sequence and runs the KMeans algorithm to generate a list of entries for the Nearest Neighbor dataset.

Current Limitations

The current system implementation has one main limitation - The clothing is assumed to be quasi-static. This means the system can predict shape details like wrinkles but cannot predict dynamic motion like swinging or draping.

This system is currently best suited for tight-fitting clothing like pants and T-shirts, but not loose garments like dresses or capes.

You can learn how to use the ML Cloth system by following the [ML Deformer - Nearest Neighbor Model](#) tutorial on the Epic Developer Community.