SimpleMeshDeformer

Developed by: Daniel Rammer
Web: https://blitzzart.github.io/projects/

Make 3D models reactive to music or approximation - or both. This mesh deformation plugin is created to allow exactly that - fast and easy.

The two main features of this asset are:

- Music Visualization. Easily use music to animate several or all 3d objects in your game.
- Deformation based on the proximity of an object. Imagine a first-person game - when the player comes close to a wall the wall, mirror or any other object, it starts wobbling.

It comes with one test audio clip and a couple of tools - tiny scripts.

Setup Guide

Link to a Step-by-Step video tutorial: https://youtu.be/csnS29Sgp6k
Further videos which show SimepleMeshDeformer in action:
https://www.youtube.com/playlist?list=PLti3TdQr7h5XwndZqBFEPeUbEd1MoLktL

Getting Started

Import Asset to your project

- Best you include all files

Now you can already start playing around with the sample scenes.

- ***01_Simple*** has a *MasterMesh* object with 3d mesh, an *AudioSource,* and the two Scripts *SimpleFFT* and *MeshDeformer* attached to it. When you hit play the *Audiosource* starts playing, *SimpleFFT* is analyzing the spectrum of the played audio clip and *MeshDeformer* is utilizing the FFTs outcome to deform the 3d mesh.
- ***02_MasterSlave*** has a couple of *SlaveMeshes* in addition to to the *MasterMesh*. All the *MeshDeformerSlave* Script needs is a 3d mesh with the same vertex count as the *MasterMesh* object and a reference to the *MasterMesh*. The slave meshes apply the data calculated by the master. This way all are deformed exactly like the master. The master-slave mechanism also makes the processing cheaper.

- ***03_Attractor*** shows a setup with two master meshes which react on the approximation of a Transform - called *Attractor* (the violet sphere). Meshes are only deformed when they get close to their attractor.
  See it in action (ITERUM Game): https://youtu.be/jV_JLquGKuY
- ***04_SmallShow*** is just a small music visualization demo scene with a master and some slaves moving around and a moving camera.

Set up your own scene

Basics

1. Import Asset to your project
2. Open the scene you want to add *SimpleMeshDeformer* or create a new scene
3. Have the 3d object you want to deform in your scene.
   - **IMPORTANT:** the 3d object must have ***Read/Write Enabled*** checked. In *Projects* Tab - click the 3d object (your3dObject.obj) - find and check ***Read/Write Enabled***.
4. Click on the GameObject which has the *Mesh Filter* and the *Mesh Renderer*. And add the *MeshDeformer* Script.
   - Hit Play. The 3d object already starts to wobble.
   - Now you can start using the different sliders of the *MeshDeformer* component and see what they do.

Make MeshDeformer react to music

1. Add *SimpleFFT* script to the same GameObject
2. Add an *AudioSource* to the same GameObject
   a. Set an AudioClip
   a. Hit Play.

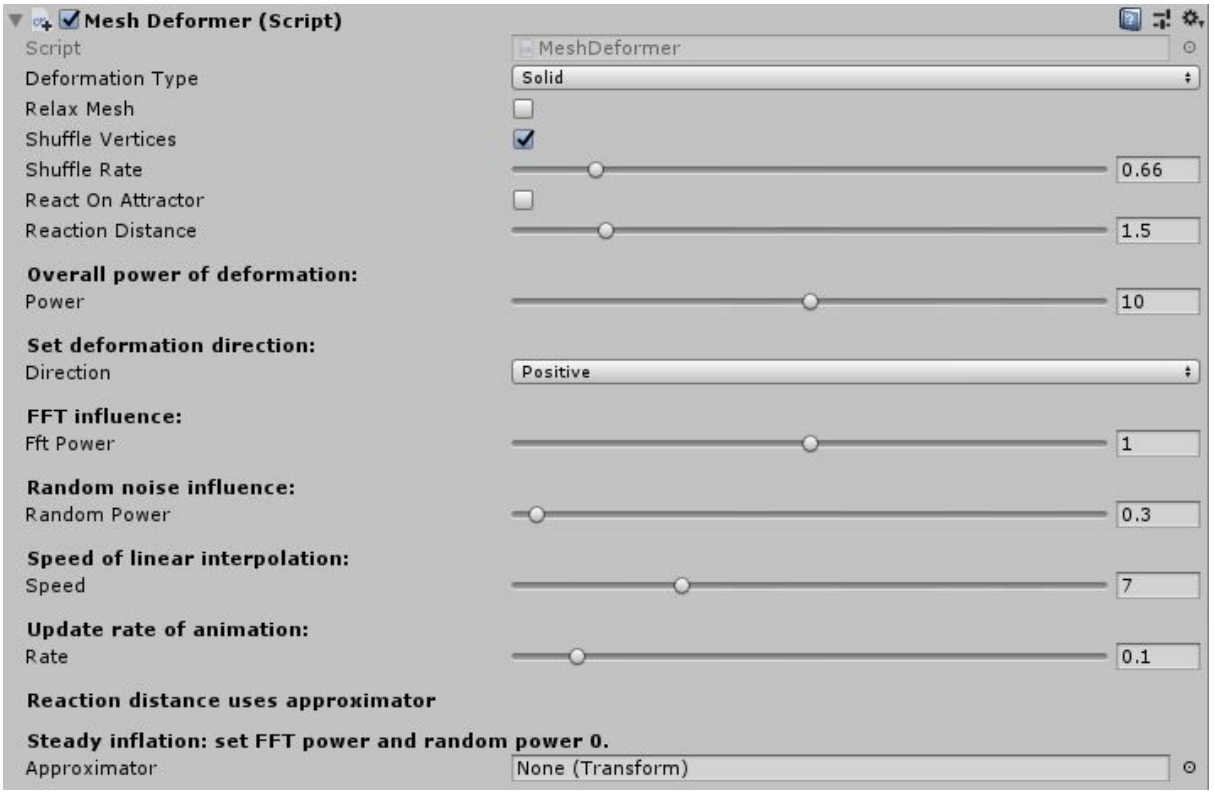
Add Slaves

1. Click on another 3d mesh which is equal to the master mesh or has at least the same vertex count
2. Add the *MeshDeformerSlave* Script to the GameObject, which has the *MeshFilter* and *MeshRenderer* attached.
3. Set the Master Property of the *MeshDeformerSlave* Component by dragging the GameObject with the *MeshDeformer* Script on it to it.
4. Hit Play. Now the slave (all slave) acts exactly like the master mesh.

Add Approximator

1. Create a new GameObject.
   E.g. right-click in Hierarchy - 3D Object - Sphere.
2. Set this GameObject as *Approximator* in the *MeshDeformer* Component. And check ***React On Attractor*** on the *MeshDeformer* Component.
3. Hit Play. If the Attractor (the GameObject you set as Attractor) comes close to the master its closer vertices start to react.

Settings



Deformation Type	Solid: The mesh stays whole while being deformed. Fractured: The mesh is separated in its triangles/explodes. Turn of **Shuffle Vertices** to keep the order of vertices.
Relax Mesh	et the mesh go back to original form if no deformation is applied.
Shuffle Vertices	Randomly shuffles vertex order to achieve a varying behavior when FFT is applied.

Shuffle Rate	Determines the rate on which vertices are reordered.
React on Attractor	Makes the MeshDeformer responsive to a Attractors proximity.
Reaction Distance	Determines the reaction distance of an Attractor to each vertex of the master object.
Power	The global strength of the deformation.
Direction	Determines the direction in which the vertices are translated.
FFT Power	The weight of the FFT data to the total deformation.
Random Power	The weight of randomly generated per-vertex values to the total deformation.
Speed	The speed with which the vertices are translated.
Rate	Determines the time in between the vertex transformations in seconds.
Approximator	The Transform, which is utilized for proximity-based vertex translation.

Have Fun!