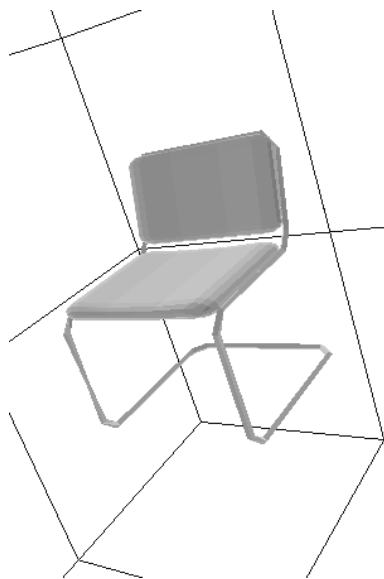


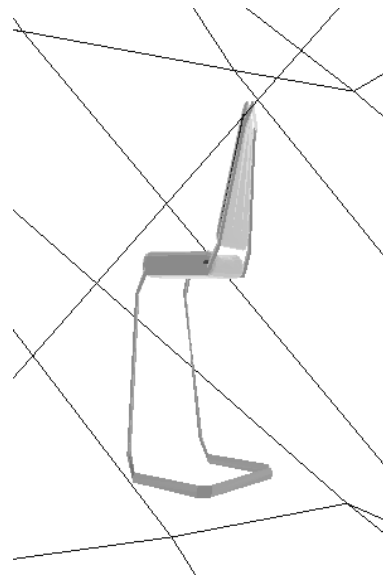
Adaptive Free Form Modeling and Animation Clint Chua and Ulrich Neumann

Static objects populate most virtual environments. While this may be sufficient for some applications, the addition of deformable objects to a scene will not only increase realism but also broaden the application's utility. The limiting factors for including deformable objects in an environment are object specific deformation software, lack of standard deformation primitives and complex mesh representations of deformable objects. The goal of our research is to propose a standard geometric deformation primitive and simplify complex deformable object representation.

Free Form Deformation (FFD) is a simple and versatile geometric deformation tool used to model and animate deformable objects. Unlike physically based deformation systems such as mass-spring models or Finite Element methods, FFD has no underlying physical model simulation when deforming an object. Instead, the FFD method manipulates the object's vertices directly according to the animator's direction and thus is very fast compared to its physically based counterparts. The FFD mechanism can be thought of as embedding an object into a piece of Jell-O and as the Jell-O twists or stretches, the object follows the deformation.



Embedding an object
in a lattice.



Moving the lattice
deforms the object

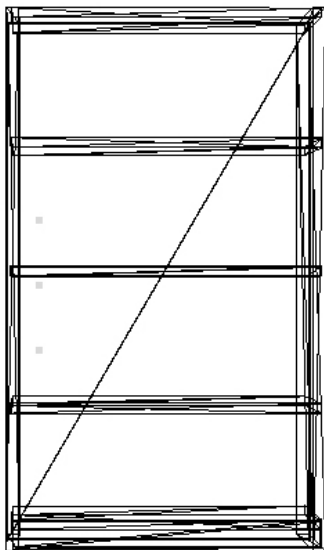
The benefits of choosing FFD as a deformation primitive are two-fold. First of all, the mathematics behind FFD can be easily implemented in hardware as a pre-processing step in a standard OpenGL graphics pipeline. The second benefit is that the FFD can serve as a primitive application program interface (API) for other deformation techniques. This allows the programmer to augment the FFD deformation behavior by adding mass-spring systems or skeleton animation systems on top of the FFD API.

Rendering a deformable object can take up a substantial amount of rendering time. This is because a deformable object tends to have a lot of curved surfaces and hence needs a highly tessellated mesh to accurately represent the object. With adaptive subdivision coupled with FFD, the animator can work with a lower resolution mesh and then add triangles to the mesh only in portions of the object that require it.

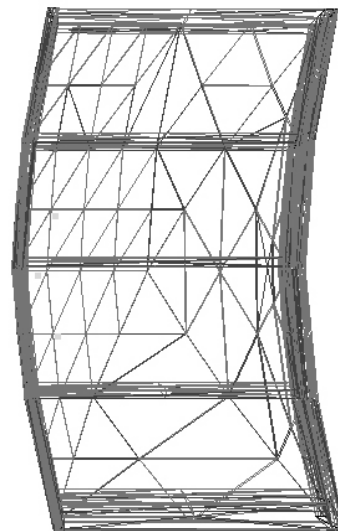
Current subdivision methods are uniform which means that the whole model is subdivided regularly. The drawback of this method is that triangles are added to regions that may not require additional subdivision. Coupled with the fact that each triangle is subdivided into 4 or more triangles, this method generates an exponential amount of triangles for each subdivision step. The end result is an excessively tessellated model that takes time to render.

Other adaptive subdivision methods propose to only add triangles in high curvature areas and refrain from subdividing low curvature regions. While this is also the intent of our method, current adaptive subdivision schemes apply the subdivision on a static model. This means that if a low curvature region that was not previously subdivided begins to deform, the model is no longer capable of representing the deformation accurately.

Our method aims at generating additional triangles on the fly and only in regions that require more subdivision. As the FFD deforms an object, the curvature of the affected surface is checked if subdivision is necessary. Triangles are added only when this criterion is met. Thus, our method only adds the triangles in regions that require additional subdivision and, in addition, all deformed regions are checked to insure that subdivision occurs on the appropriate regions.



Model in its rest position



Model is subdivided based on deformation