International Journal of Scientific & Engineering Research Volume 8, Issue 5, May-2017
ISSN 2229-5518

Survey of trends in 3D animation

Prof. Puja Agarwal, Nishant Adhikari

Abstract— 3D animation is one of the most interactive and useful application of computer graphics, programming and mathematics. It is also widely used in fields like gaming, movie, AI, virtual reality, etc. 3D animation involves complex algorithms and data structures as per the requirement of artist, engineer, designer and machine. Some techniques are simple to understand, some are simple to implement but on the hand. Some are difficult to understand or implement. But there are few techniques that are in trend. This paper focuses on how some techniques, which are in trend are used to convert a 3D design into a 3D object on the screen. And also on challenges associated with each along with future development

Index Terms— 3D animation, 3D rendering, Animation technique, cage, mesh parameterization, multichart technique, texturing

1 Introduction

In every computer graphics oriented application, it is a common practice to create 3D models (Fig 1, top) as a way to obtain realistic feel. After creating the objects, animators then usually deform and/ or animate them (Fig 1, right) to create the poses most appropriate for each given scenario or application. Finally, once the 3D objects are ready, the scene is visualized with a rendering algorithm (Fig 1, bottom) to produce synthetic scene images [1]. Another technique is texturing, however a texture can be loosely defined as: texture can be a texture in the usual sense (cloth, wood, gravel) or a detailed pattern repeated many times to tile the plane-or, more generally, it can be a multidimensional image mapped in multidimensional space. Another relevant technique is Texture mapping, which means the mapping of a function onto a surface in 3D. Hence texturing, deformation, and visualization are all key aspects of the computer graphics [2]

When 3D artists need to apply texture mapping to 3D models, they usually do it with the help of mesh parameterization techniques [3]. Of those techniques, some of the most popular are multichart approaches, which break a continuous 3D model into a set of disconnected pieces called charts. This way, general and complex 3D models are mapped from a 3D space to 2D space, in which artists can easily apply images or directly paint. This procedure, although useful, produces the appearance of texture discontinuities where the charts meet. This cause serious problems for common applications such as regular texture filtering (interpolation from a set of pixels), continuous simulations in texture space, and more advanced tasks like subdivision surfaces.

After texturing, animators are often unsatisfied with the initial 3D model poses and need methods and software to easily deform and/or animate 3D models. From the many existing deformation approaches, cage-based methods [8] have gained popularity in recent years among the computer graphics community, mainly because of their simplicity and speed. This feature is important for modelers, who require fast feedback on their work. Those methods are characterized by the use of a single cage that encloses the model to be deformed. This cage must be of a similar shape to the input mesh, but much simpler, and it drives the final deformation.

To apply deformations to the input mesh from the deformed cage, a set of coordinates are computed, producing a binding from cage vertices to mesh vertices. Several types of coordinates exist, each with their own advantages and drawbacks, and depending on the one used, we will obtain different results. By using a single cage, we are restricted to the use of just one type of coordinate, and our deformations are global. This means that we cannot combine the strengths and flexibility of several coordinate types and cannot apply deformations at different levels of detail (from the whole model to just one small detail). Moreover, using several cages produces discontinuities at cage boundaries, eliminating the smoothness of the input mesh and even producing cracks. However, this is a need for a method that allow animators to deform a 3D model to produce high-quality deformations with fine control, while preserving the smoothness of the input mesh.

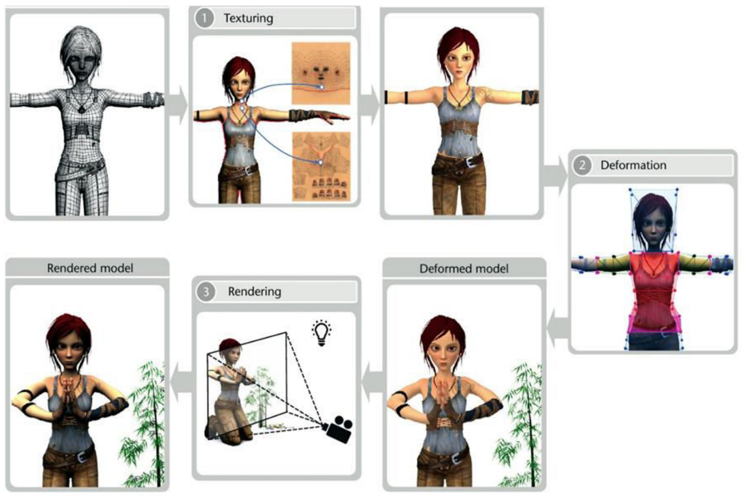


Fig 1: Computer graphics pipeline. A 3D model is first textured by an artist and then deformed by an animator to create new poses. Finally, the model is rendered from various points of view or requirements.

• *Prof. Pooja Agarwal is assistant professor in Thakur Institute of Management Studies Career Development and Research, Mumbai, India.*
E-mail: puja.mca07@gmail.com
• *Nishant Adhikari is currently pursuing master's degree in Computer Applications in Thakur Institute of Management Studies Career Development and Research, Mumbai, India.*
E-mail: nishantadhikari.994@gmail.com

International Journal of Scientific & Engineering Research, Volume 8, Issue 1, January-2017
ISSN 2229-5518

Once a scene is correctly textured and deformed, we usually want to visualize it as fast as possible. While preserving the image quality of the rendering algorithm used. Many techniques can help accelerate the visualization of a 3D scene, from common rasterization to more complex global illumination approaches. Some approaches present new data structures to accelerate the creation of the final rendered image, [9] whereas others try to avoid computations by exploiting similarity between 3D models [10] once the scene is projected onto the screen. One key strategy of those methods is to compute some small amount of information (pixel colors) and reuse it as much as possible. When reusing pixel information, it is important to only reuse pixels from smooth and similar regions, avoiding the reuse of pixels coming from discontinuous or dissimilar features. A bad reuse approach, usually some kind of interpolation, could lead to visible artifacts and, as a consequence, decrease the final image quality.

2 Techniques used in 3D animation

Some techniques that will improve existing state-of-the-art approaches related to continuity and interpolation of texture space (or texturing), object space (or deformation), and screen space (or rendering) are listed below by Francisco González García[1].

2.1 Texture Space: Contunintiy Mapping

Multichart parameterizations are among the most commonly used methods for texturing in current computer graphics pipelines. Those methods introduce discontinuities, called seams, by breaking the models into charts. In practice, this process produces spatial discontinuities as triangles, where neighbors in 3D are not neighbors in texture space (2D) anymore (see Fig 2), and a sampling mismatch of textures at chart boundaries occurs because every chart in 3D space is mapped to the texture space with different scales and rotations, causing texels (or texture element) from different charts not to match up at chart boundaries (see Fig 2)

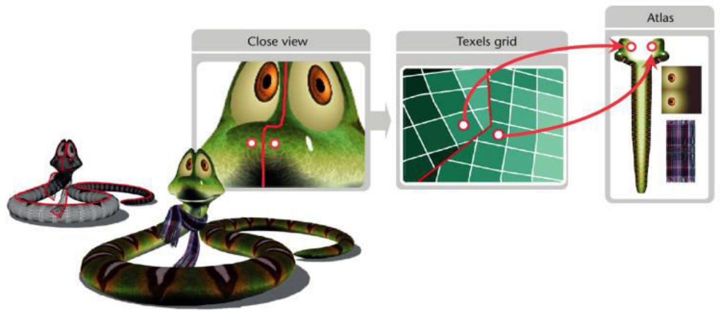


Fig 2: Multichart parameterizations discontinuities. Neighboring points in 3D are mapped into different areas in 2D (spatial discontinuities) with different scale and rotation (sampling mismatch).

Up until now, those problems were hidden by increasing the texture resolution, applying padding procedures (extending chart boundaries in 2D), or transferring the multichart texture information to another seamless parameterization. This does not solve the problem, however, and it can even produce loss of detail and artifacts from the original multichart textures.

To address this, we developed Continuity Mapping, a technique that is able to provide a continuous mapping for any multichart parameterization without modifying the original artist's content. Continuity mapping consists of two related techniques: traveler's map and sewing the seams, both applied in a preprocessing stage.

First, traveler's map solves the texture spatial discontinuities problem at chart boundaries by defining a correspondence, in texture space, of any point outside a chart with its corresponding point inside the neighboring chart.

Second, the sewing the seams technique solves the sampling mismatch at chart boundaries by literally sewing the seams, creating a thin border of virtual triangles (not real object geometry) in texture space to correctly interpolate and filter texture values through charts boundaries.

Continuity mapping allows for some applications that were not previously possible, such as seamless texture filtering, continuous simulations in texture space like fluid simulations and reaction-diffusion simulations, and multichart relief mapping (see Fig 3)

(a) (b) (c)

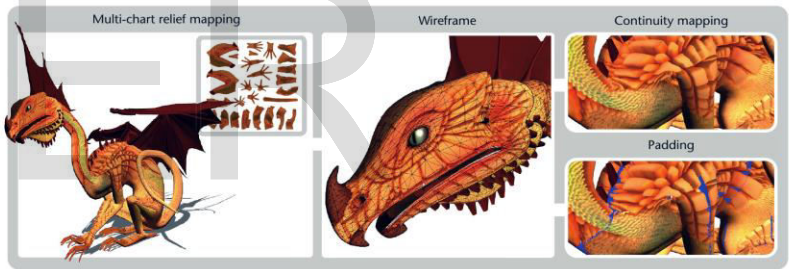


Fig 3: Multichart relief mapping. (a) The dragon model with multichart relief mapping applied and its parameterization. (b) A close-up view of the head model and its base geometry. (c) A comparison between continuity mapping and padding when relief mapping is applied. The blue fragments represent regions outside charts.

2.2 Object Space: *Cages

Cages are the line drawn around the object to be rendered (typically putting object inside an imaginary cage for easy computation), cages can be treated as separate or single as per the need. As discussed earlier, all previous cage-based deformation approaches [8] used a single cage to drive the final deformation of a mesh. This presented some common problems:

■ Locality: By using a single cage for the deformation, local modifications cannot be applied to the 3D models.
■ Time and memory consumption: The global behavior of those methods increases the memory consumption and the number of evaluations to perform.
■ Smoothness: All previous methods have continuity problems at cage boundaries, which produces non smooth deformed meshes at those locations.
■ Coordinate combination: The deformations produced by all the methods are different, and there is no way to combine them.

International Journal of Scientific & Engineering Research, Volume 8, Issue 1, January-2017
ISSN 2229-5518

■ Usability: The design and modeling of a single cage is less user-friendly than a small set of simpler cages.

To overcome the above limitations, Francisco González García[1] has proposed *Cages in his work Dissertation Impact. *Cages, are generalization of traditional single cage-based systems that uses a set of hierarchical cages, in which the leaf cages bound the object in a piecewise manner without any intersection (see Fig 4). Each cage can use a different set of coordinates, which increases the flexibility available to artists. They can select the coordinate type for any cage and, as a consequence, produce several differ- [illegible] e- ng al ne ss

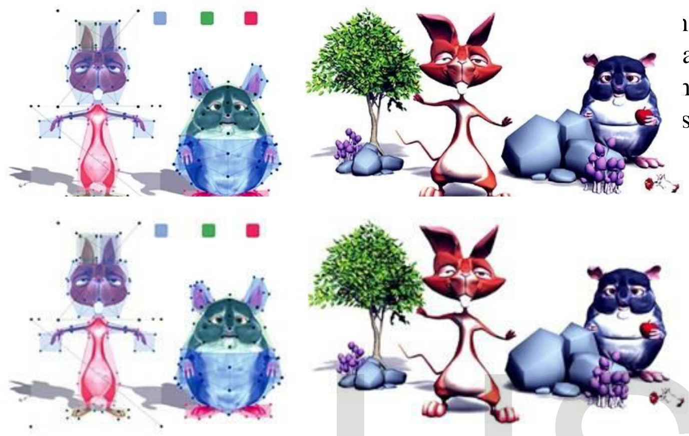


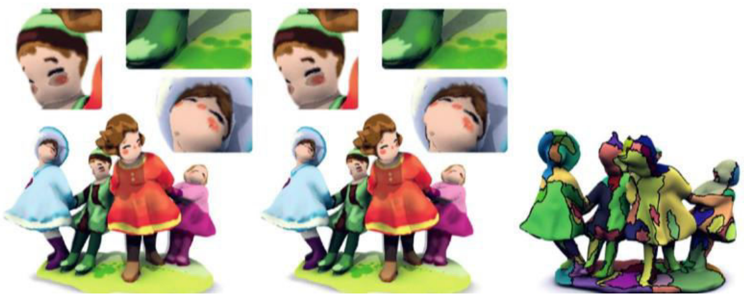
Fig 4: Applying *Cages to the squirrel and chinchilla models. The blue cages use mean value coordinates, green cages use green coordinates, and red cages use harmonic coordinates. [10] Notice especially the hierarchy of cages applied on the squirrel model.

2.3 Screen Space: I-Render

The Continuity Mapping and *Cages techniques brought new methods to texturing and mesh deformation to avoid discontinuities in texture and object space, respectively. This last method drives reuse and interpolation as a way to speed up the generation of the final rendered image. We observed that many of the scenes used in production have many regions that share the same characteristics or information. Because their final rendering is going to be nearly the same, they could thus share some information to make their computation more efficient. Taking into account this key point, we presented a two-stage technique called I-Render. In the first stage, we use information theoretic tools and measures to cluster the scene geometry (see Fig 5) by a similar set of features (such as visibility, orientation, or texture color). This set of features is the key characteristic that we want to keep and, as a consequence, share between scene objects. Then, in real time, the rendering stage uses the previously computed clusters to evaluate the final image's colors, using a multiresolution approach.

This method first renders the scene in a low-resolution buffer and then iteratively computes larger resolutions until it reaches the final desired resolution. Every iteration reuses low-resolution pixel information coming from previous passes. At cluster boundaries, in which a discontinuity between features exist, the samples are fully evaluated and reused in further iterations of the method. Given the iterative nature of the method, we provided an automatic control mechanism that determines the number of passes (intermediate low-resolution images) to be computed to avoid situations in which we could underperform traditional ray tracing i.e., situations with a low level of similarity or interpolation. We can implement this technique on top of a GPU-based ray tracing engine, and our results show that I-Render can render up to 12 times faster than traditional ray tracing [1], while preserving the image quality (see Fig 5).

(a) (b) (c)
Fig 5: Final scene rendering. Comparison of (a) traditional ray tracing and (b) I-Render using the (c) previously computed clusters.



3 Current trends

The first presented technique, Continuity Mapping, appeared as a solution to avoid the discontinuities resulting from all multichart parameterizations. From the beginning, we knew that most of the 3D models used in the computer graphics industry (videogames, virtual reality, film, and so on) were using this type of parameterization, and we also knew that this type of parameterization still resulted in discontinuities in texture space.

Thus, the main goal was to propose a new method that could convert any discontinuous multichart parameterization into a seamless one, avoiding any modification or loss of detail of the textures painted by the artists. This constraint was important because computer graphics companies invest a lot of their resources in the modeling and texturing of 3D models. As a consequence, destroying or modifying that previous investment made no sense. Based on this primary goal, we can retain the original multichart parameterization and to try to make it seamless. This led us to a solution that, by using a small extra memory footprint, was able to visualize any multichart parameterization as a seamless one without any kind of re-parameterization or modification in the artist's provided textures. Continuity Mapping was tested in real environments by a few videogame companies, and even though the results were good, in the end the extra computations needed for the method to work were harmful to its final applicability in the next wave of videogame engines.

A videogame is an interactive application with numerous physics, AI, and graphic computations occurring for every single frame, so it is necessary to carefully invest engine resources. Even though this technique was efficient in both memory and time consumption, videogame companies were not ready to assume the extra cost that seamless texturing benefits can offer. In addition, Continuity Mapping is less applicable to the 3D film industry because more expensive parameterizations could be used without the memory and time restrictions needed for interactive environments. As a result, we learned that sometimes research turns out to be less applicable that hoped, but it still opens the door of possibilities for other researchers.

From the early stages of the *Cages development, researcher felt that it could be a useful tool for modelers. Being able to offer a hierarchical and continuous approach for cage-based deformation that could suit any type of the existing cage coordinates, taking advantage of their strengths and avoiding their main limitations, would allow them to provide a useful tool for mesh deformation. Pixar had developed one of the previous approaches, [8] which made them conscious of the high level of interest that this type of research could have from the 3D computer animated film industry.

Last but not least, I-Render is not ready, in its current form, to be integrated into commercial rendering engines (such as Maxwell Render, V-Ray, and Arnold). This is mainly because of its robustness as a technique, which has to be applied into a large and different range of scenarios and situations, while still providing the high level of image quality that commercial renderers demand. To be able to achieve this, more research is necessary in the clustering stage. First, we need to provide a more general framework to accommodate all kinds of features to be preserved and interpolated. Second, we must improve the performance of this stage; in its current form, it can take several minutes to preprocess moderately complex scenes. A parallel implementation would be an appropriate approach. Nevertheless, I-Render is a promising proof of concept that we hope to continue extending in the future.

4 Future Scope

When doing research related to texturing, we need to keep in mind that this task is close to the 3D artist's workflow. Artists usually prefer to paint 3D models on 2D (textures), instead of working directly on 3D models. In addition, artists usually need to tweak the initial parameterization to achieve better mappings of selected areas and features. Those demands impose some serious constraints on the future of parameterization research. We can hope that the next wave of texturing approaches should allow artists to work seamlessly from 3D to 2D, and vice versa, while they take care of the possible discontinuity issues appearing in the process. This would provide them with a lot of flexibility and allow them to focus on painting the 3D models rather than hiding the seams

Mesh deformation tools are going to evolve into hybrid methods in which the user is not restricted to a single deformation approach. Animators are going to be able to select between various methods, depending on their needs, providing the user with more flexible approaches without having to deal with the discontinuities that could appear when using different deformation techniques. Alec Jacobsen [10] has already taken a first step in that direction by allowing the user to choose between a ranges of handles, from skeletal bones to cages, to deform a 3D model. *Cages could perfectly fit into that trend, as a good candidate to replace single cage-based handles. However, in addition to being free to select a deformation approach, animators still need a more general framework, such as the one *Cages proposes for cage-based deformations, in which they are not limited to a single coordinate type and, as a consequence, to a single way to deform a mesh.

Although many techniques exist that could improve the performance of current commercial renderers, a promising direction is to take advantage of the similarity and coherence present in 3D scenes. Defining similarity is going to continue to be a key challenge at many levels - for performance, in which a coherent, similar, and efficient computation is necessary, and for its definition, because the better we specify this concept, the better the results will be. Most of that new research will focus on screen space because it will allow researchers to decouple the method's complexity from the scene's complexity, thus making the research only coupled to the resolution of the final image.

5 Conclusion

For better understanding of 3D animation and its applications, we surveyed the literature in this area. We listed the techniques which are in trend and discussed current approaches to implement them. We found that, the research done in this domain is really useful in multiple domain which needs animation as a tool. Available tools and techniques must be revised or at least reviewed for improvement. Need of techniques used is changing with demand of quality and commercial scale of production.

References

[1] Trends in Continuity and Interpolation for Computer Graphics, Francisco González García, Next Limit Technologies, November/ December 2015, IEEE computer society.
http:/ / ieeexplore.ieee.org/ ielx7/ 38/ 7331155/ 07331160.pdf?tp=&arnumber=7331160&isnumber=7331155

[2] P.S. Heckbert, "Survey of Texture Mapping," IEEE Computer Graphics and Applications, vol. 6, no. 11, 1986, pp. 56–67.
http:/ / ieeexplore.ieee.org/ ielx5/ 38/ 4056747/ 04056764.pdf?tp=&arnumber=4056764&isnumber=4056747

[3] D. Cohen-Or, "Space Deformations, Surface Deformations and the Opportunities In-Between,"
http:/ / jcst.ict.ac.cn:8080/ jcst/ EN/ article/ downloadArticleFile.do?attachType=PDF&id=9101

[4] M.S. Floater and K. Hormann, "Surface Parameterization: A Tutorial and Survey,", Computer Science Department, Oslo University, Norway
http:/ / vcg.isti.cnr.it/ Publications/ 2005/ FH05/ survey_mingle04.pdf

[5] P.V. Sander, "Multi-chart Geometry Images", Microsoft Research 2003
http:/ / hhoppe.com/ mcgim.pdf

[6] BrunoLévy,Least square conformal maps for automatic texture atlas generation, ISA (Inria Lorraine and CNRS), France
http:/ / www.cs.jhu.edu/ ~misha/ Fall09/ Levy02.pdf

International Journal of Scientific & Engineering Research, Volume 8, Issue 1, January-2017
ISSN 2229-5518

[7] Kun Zhou, Iso-charts: Stretch-driven Mesh Parameterization using Spectral Analysis, ACM Transactions on Graphics, Vol. V
http:/ / research.microsoft.com/ en-us/ um/ people/ johnsny/ papers/ isochart_report.pdf

[8] Pushkar Joshi, harmonic coordination for character articulation, Pixar
http:/ / www.cs.jhu.edu/ ~misha/ Fall07/ Notes/ Joshi07.pdf

[9] QimingHou, Real-Time KD-Tree Construction on Graphics Hardware
http://kunzhou.net/2008/kdtree.pdf

[10] Anders Adamson, Adaptive Sampling of Intersectable Models Exploiting Image and Object-space Coherence, Department of Computer Science Darmstadt University of Technology
http://www.cs.rutgers.edu/~nealen/research/asim.pdf