Dynamic Mesh Compression with Clustering

Jan Rus*
Department of Computer Science and Engineering
Faculty of Applied Sciences
University of West Bohemia
Univerzitní 8, 306 14 Plzeň, Czech Republic
jrus@kiv.zcu.cz

Abstract

The use of 3D data in the form of triangle meshes is nowadays a widespread phenomenon. A great many examples of using this kind of data can be found in engineering applications or in film and gaming industry. These examples include not only static, but also dynamic meshes – 3D animations. This kind of data representation is usually voluminous and needs to be compressed for efficient storage and transmission. In this paper, we are dealing with the influence of vertex clustering on dynamic mesh compression. The mesh is divided into vertex clusters based on the vertex movement similarity and compressed per-partes to achieve higher compression performance. We use Coddyac as a basic compression algorithm and extend it by adding well known clustering algorithms to demonstrate the efficiency of this approach. We also deal with what clustering is the most appropriate Coddyac.

Categories and Subject Descriptors

I.3.7 [**Three-Dimensional Graphics and Realism**]: Animation; E.4 [**Coding and Information Theory**]: Data compaction and compression

Keywords

3D dynamic meshes, Data compression, Computer animation, Coddyac, Clustering

1. Introduction

Because storage capacities and transmission speeds are limited, we need to use compression algorithms to reduce data volume and thus reduce hardware requirements for storage and distribution of data. ZIP and RAR are popular compression algorithms but they are not primarily intended for dynamic mesh compression. We can achieve better compression rates using specialised compression algorithms, such as Coddyac [6], which is currently the most efficient compression algorithm for dynamic meshes.

Unlike the ZIP compression, Coddyac compression is a lossy compression algorithm, but this is not an obstacle due to the kind of data. Usual video compression algorithms are also lossy. One of our key observation is, that the efficiency of the Coddyac algorithm directly depends on the dynamic mesh movement complexity. The less complex the movements in animation are, the more efficient the Coddyac algorithms is. Therefore, one possible way to improve the compression ratio is to reduce the movement complexity, which could be achieved by clustering the mesh vertices by similarity of their trajectories.

2. Coddyac Algorithm Overview

Coddyac contains two well-known algorithms, Rossignac's Edgebreaker[2] intended for triangle connectivity compression and Principal Component Analysis[5], which is used for compression of the dynamic mesh geometry.

The Coddyac algorithm is based on representing dynamic meshes as a set of vertex trajectories of individual vertices. By using PCA on these vertex trajectories during compression, we obtain a new description of trajectories: feature vectors. These vectors consist of linear combination coefficients of principal components (significant movements of the mesh), which are ordered by the amount in which they affect the movement of vertices during the whole animation. Reason for such trajectory transformation is the possibility to ignore less important components of feature vectors (less important movements), and this way reduce the amount of data describing the original animation. Size of compressed data depends especially on the rate of compression of feature vectors. Length of these vectors is equal to number of selected principal components and thus efficiency of the Coddyac algorithm – and all PCA-based compression algorithms intended for dynamic meshes in general – directly depends on the dynamic mesh movement complexity.

3. Clustered Coddyac

As noted, the efficiency of the Coddyac compression algorithm directly depends on the complexity of movements of the animated mesh. By movement complexity of a set of vertices we mean differences of their trajectories. The trajectories can be very complex, but they have to be

*Master degree study programme in field Computer Graphics and Virtual Reality. Supervisor: Dr. Libor Váša, Department of Computer Science and Engineering, Faculty of Applied Sciences, University of West Bohemia.

Rus, J. Dynamic Mesh Compression with Clustering. Information Sciences and Technologies Bulletin of the ACM Slovakia, Special Section on the ACM Student Project of the Year 2011 Competition, Vol. 3, No. 4 (2011) 51-52

similar to each other to decrease movement complexity of the set of vertices. It follows that if different parts of the mesh move in a relatively simple manner, but differently, the global movement of the mesh will be complex. Therefore, after application of PCA, all trajectories are described by vectors longer than necessary – we are using combination of more principal components than we need. For example to describe the movement of the torso of the chicken in figure 3 on the left, principal components describing movement of his wings are also used, even though this movement almost does not affect the torso at all.

Figure 1: Dynamic mesh movement simplification. Three important movements for each vertex of chicken mesh on the left, but only one important movement for torso and each wing on the right.

To reduce the length of the feature vectors, we must select those vertices of the mesh, whose trajectories are similar each other and include them in a common group - a cluster. This way the movement complexity in individual clusters is reduced and so is the necessary number of principal components. That leads to shorter feature vectors and better compression ratio.

Many algorithms have been proposed for mesh division into smaller parts depending on the topology or geometry criteria. We want to divide meshes according to their movement complexity and similarity. Movement is a geometrical property of dynamic mesh, thus neither strictly topology-driven nor logical division were considered during our experiments. Geometry-driven division of dynamic meshes can be performed by clustering of vertices using their trajectories, but we have to choose suitable clustering algorithm for Coddyac to maximise its efficiency. It is important to select the appropriate number of clusters as well. The smaller clusters we choose to cover the surface of the model, the better complexity reduction can be achieved. However each cluster requires initialisation data (PCA basis), which negatively affect the final compression ratio. Therefore, we need to find the optimal number of clusters, enough to ensure that movements of vertices included in them is as ordered as possible.

For the proper functionality of discussed approach we had to solve several problems, such as disjointed clusters and selection of an appropriate number of basis vectors for each cluster. Solution of these problems and more details on the clustered Coddyac can be found in [4] and [3].

4. Experimental Results

The clustering improvement of the Coddyac algorithm and its modifications have been implemented and subsequently tested on several 3D dynamic meshes. This experimental data-set consists of artificial animations of single and multiple objects and scanned real data. To compare errors resulting from compression of 3D animated meshes we used KG-Error [1] and STED [7] metric.

We have also experimented with different methods of clustering of dynamic meshes, such as facility location, edge-collapse based clustering and k-means combined with different distance norms. Each tested clustering method has brought some improvement of compression ratio. The best of these methods achieve results in the range of tens of percents lower bit-rate while maintaining the same error, see figure 4.

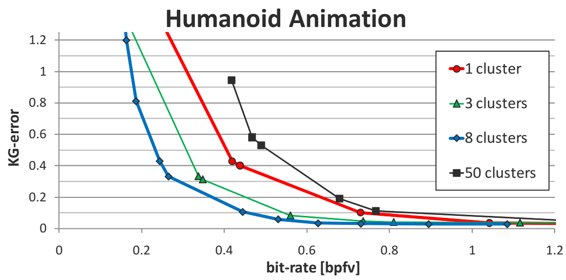

Figure 2: Comparison of different number of clusters. Red - not clustered, blue - the best result.

5. Conclusions

In this paper we have examined the influence of vertex clustering on the Coddyac algorithm, which lies in the suitable division of compressed mesh into smaller parts – clusters. Clusters are then compressed separately to reduce the animation movement complexity and thus improve the compression ratio. We found out that the optimal number of clusters is not dependent on the structure of animated mesh neither the complexity of vertex trajectories, but especially on the differences between trajectories of individual vertices. For most tested meshes we have found that values of 5-10 clusters provide good results. We have tested several methods for clustering of dynamic meshes. Of these methods the best results are in average provided using the k-means clustering algorithm in combination with L_1 distance norm. The use of clustering in the Coddyac algorithm improves its performance in terms of errors and data rate reduction. The bit rate can be reduced by 33%-49% compared to the Coddyac output while maintaining the same error.

Acknowledgements. This work has been supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research program LC-06008 (Center for Computer Graphics).

References

[1] Z. Karni and C. Gotsman. Compression of soft-body animation sequences. *Computers & Graphics*, 28:25–34, 2004.

[2] J. Rossignac. Edgebreaker: Connectivity compression for triangle meshes. *IEEE Transactions on Visualization and Computer Graphics*, 5:47–61, 1999.

[3] J. Rus. Dynamic mesh compression with clustering. Diploma thesis, University of West Bohemia, 2011.

[4] J. Rus and L. Váša. Analysing the influence of vertex clustering on pca-based dynamic mesh compression. In *Articulated Motion and Deformable Objects*, pages 55–66, Heidelberg, 2010. Springer.

[5] L. I. Smith. A tutorial on principal components analysis. Technical report, Cornell University, USA, February 26 2002.

[6] L. Váša and V. Skala. Coddyac: connectivity driven dynamic mesh compression. In *3DTV-CON 2007*, pages 1–4, Piscataway, 2007.

[7] L. Váša and V. Skala. A perception correlated comparison method for dynamic meshes. *IEEE Transactions on Visualization and Computer Graphics*, 17(2):220–230, 2011.