BARRIOL Rémy
TUCHOLSKI Lilian

Compte Rendu #2-3
Projet : Ultimate 3D Crypto-Compressor
Sujet #1 Crypto-compression d'objets 3D

Lien Git : https://github.com/BarriolRemy/Ultimate-3D-Crypto-Compressor

I. Comparaison

Pour pouvoir choisir efficacement la méthode de compression que nous implémenterons, nous allons tout d'abord voir l'ensemble des méthodes et ainsi les comparer sur leur performance. L'article "3D mesh compression: survey, comparisons and emerging trends" [1] a déjà eu pour but de décrire et comparer les différentes méthodes de compressions de modèles 3D.

Pour sélectionner la méthode que nous implémenterons, nous nous baserons sur les résultats de la compression, sur les pertes probables mais aussi sur leur capacité à être implémenter. (Certains articles peuvent se concentrer sur la description des résultats d'une méthode mais ne pas décrire précisément la méthode de compression ce qui augmentera la complexité de l'implémentation) et enfin sur les éléments utilisés (par exemple les octree ou kd-tree que nous avons vu en cours).

A. Single-Rate Mesh Compression

Pour rappel, les algorithmes de compression "Single-Rate" ont pour but de compresser en une seule fois l'ensemble du modèle 3D. Et la décompression donnera un modèle 3D similaire à celui compressé ou en partie. Et les différents algorithmes seront adaptés suivant la constitution du modèle, certains algorithmes seront utilisables pour des modèles composés de polygone quand d'autres seront conçus pour les modèles composés uniquement de triangle.

Table I. Summary of the main single-rate compression algorithms.

Algorithm	Connect. comp. rates (bpv)	Compress polygon meshes	Embed strips	Remarks
Deering [Deering 1995]	11	no	yes	
Topological surgery [Taubin and Rossignac 1998]	6 max. 2.5 to 6	no	no	
Cut border machine [Gumhold and Straßer 1998]	4.4 on avg.	no	no	
Valence coder [Touma and Gotsman 1998]	2.3 on avg.	no	no	Introduced valence encoding & parallelogram prediction
Edgebreaker [Rossignac 1999] [Gumhold 2000]	3.55 max. 2.1 on avg.	no	yes	Many derived schemes
Valence polygonal [Khodakovsky et al. 2002] [Isenburg 2002]	1.8 on avg. for poly. meshes	yes	no	Proven near-optimality of the connect. coding
Valence coder [Alliez and Desbrun 2001a]	2.1 on avg.	no	no	Proven optimality of the connect. coding

Nous pouvons premièrement voir que seul un des algorithmes présent dans le document dans les méthodes Single-Rate permet de compresser un modèle basé sur les polygones, il est par ailleurs l'algorithme avec le taux de compressions le moins élevé, puisque qu'il sera au alentour de 1.8.

Notre choix se porterait donc sur l'algorithme de Deering et le Topological surgery.

B. Progressive Mesh Compression

Pour rappel, les algorithmes de compression de mesh Progressifs permettent d'avoir comme l'objet 3D compressés sous plusieurs niveaux de détail. Dans certains cas, nous pourrions donc afficher les niveaux de détails les plus faibles, pour tendre vers les niveaux de détails les plus hauts, pour pouvoir donc afficher un modèle 3D sans avoir le besoin d'attendre une décompression totale et lourde du modèle dans tous ses détails.

Table II. Summary of the main progressive mesh compression algorithms.

Algorithm	Lossless connect. comp.	Total comp. rates (bpv)	Compress non-manifold meshes	Progr. granularity	Remarks
Progr. meshes [Hoppe 1996]	yes	37 (10 bit)	no	5/5	
Compressed progr. meshes [Pajarola and Rossignac 2000]	yes	22 (10 bit)	no	3/5	
Valence encoder [Alliez and Desbrun 2001a]	yes	21 (12 bit)	no	3/5	
Wavemesh [Valette and Prost 2004b]	yes	19 (12 bit)	no	1/5	Low number of levels of detail
Spectral compression [Karni and Gotsman 2000]	yes	19 (12 bit)	no	3/5	No progr. coding of the connect.
Kd-tree coder [Gandoin and Devillers 2002]	yes	19 (12 bit)	yes	2/5	High distortion at low rates
Octree coder [Peng and Kuo 2005]	yes	15 (12 bit)	yes	2/5	Like above
Incremental parametric refinement [Valette et al. 2009]	yes	15 (12 bit)	no	5/5	Original connect. may fail to be restored
Geometry image [Gu et al. 2002]	no	-	no	-	Can generate cracks in decomp. models
Wavelet compression [Khodakovsky et al. 2000]	no	8 (eq. 12 bit)	no	3/5	Fits well to smooth and dense meshes
Normal meshes [Guskov et al. 2000]	no	6 (eq. 12 bit)	no	3/5	Like above

Notre première réaction est de remarquer que l'algorithme ayant le taux de compression le moins élevé de l'ensemble des algorithmes de compression progressifs à un taux de compression similaire à l'un des meilleurs algorithmes de compression Single-Rate du tableau précédent. Dans le cas où nous voudrions prioriser le taux de compression, nous serions donc tenté de ne pas implémenter un algorithme Single-Rate.

Nous pouvons faire une remarque sur la présence de deux algorithmes utilisant des technologies que nous avons vu en cours, l'Octree coder et le Kd-tree coder.

Nous pouvons aussi remarquer la méthode des Geometry image, qui permettent de compresser des models 3D sous forme d'image, ce qui serait aussi particulièrement intéressant dans le cadre de cette UE. En revanche, que ce soit dans cet article où dans les articles auxquels nous avons pu accéder concernant les Geometry Image, il n'y a pas de taux de compression communiqué, nous avons donc moins d'informations permettant de nous lancer sur cette méthode. De plus, elle possède un défaut qui diminue la qualité du modèle décompressé. Cela nous amène donc à remarquer que certains de ces algorithmes provoquent des pertes, nous aurons donc tendance à nous diriger vers les algorithmes sans pertes, pour avoir des modèles décompressé de la meilleure qualité possible.

C. Random Accessible Mesh Compression et Progressive Random Accessible Mesh Compression

Pour rappel, les algorithmes de Random Mesh Compression ont pour fonction principale de pouvoir décompresser seulement certaines zones du modèles, celle qui nous intéresse. Nous ne perdrons donc pas de temps à décompresser les parties intéressantes pour l'utilisateur.

Les algorithmes Random Progressive Mesh Compression sont les algorithmes précédents mais ajoutant la particularité des algorithmes Progressive Compression. Ils auront donc pour but de d'afficher une ou plusieurs parties du modèles 3D avec un accès à plusieurs niveaux de détails.

Table III. Summary of the main random accessible (first part) and progressive random accessible (second part) mesh compression algorithms.

Algorithm	Lossless connect. comp.	Total comp. rates (bpv)	Compress non-manifold meshes	Random access granularity	Remarks
Streaming random accessible [Yoon and Lindstrom 2007]	yes	28 (12 bit)	no	2/5	Preserve the streaming layout
Hierarchical compression [Courbet and Hudelot 2009]	yes	20 (12 bit)	no	5/5	High random-accessibility granularity
Chart-based compression [Choe et al. 2009]	yes	16 (12 bit)	no	3/5	High compression performance
SQuad representation [Gurung et al. 2011a]	yes	2 references per triangle	no	5/5	First compact representation
Dependency free vertex splits [Kim et al. 2006]	yes	31 (12 bit)	no	5/5	Produce smooth transitions between mesh parts
Kd-tree cell compression [Jamin et al. 2009]	yes	21 (16 bit)	yes	3/5	Need post-processing to stich adjacent mesh parts
POMAR [Maglo et al. 2013]	yes	20 (16 bit)	no	3/5	Produce smooth transitions between mesh parts
Layered Kd-tree compression [Du et al. 2009]	yes	17 (16 bit)	yes	3/5	High distortion at low rates Dependency between parts
Normal mesh compression [Sim et al. 2005]	no	?	no	2/5	Fits well to smooth and dense meshes

Note: the figures in the parenthesis are the number of global quantization bits.

Nous pouvons premièrement remarquer que tous les algorithmes possèdent un bon taux de compression par rapport aux taux de compressions des algorithmes de Single-Rate. Nous pourrions, par exemple, nous intéresser au Streaming Random Accessible pour son haut taux de compression ou au Chart-based Compression pour ses grandes performances.

D. Dynamic Mesh Compression

Pour rappel, les Dynamic Mesh Compression ont pour rôle de compresser une suite d'objet 3D sur plusieurs instant t. Cela concerne principalement l'animation 3D.

Table IV. Summary of the main dynamic mesh compression algorithms.

Algorithm	Comp. rates (bpfv)	Local/global	Scalability	Remarks
PCA [Alexa and Müller 2000]	-	Global	No	First spatial PCA method
Dynapack [Ibarria and Rossignac 2003]	-	Local	No	First real-time algorithm
PCA + Linear Prediction [Karni and Gotsman 2004]	-	Global	No	Very efficient for long sequences of relatively coarse meshes
Anisotropic Wavelet Transform [Guskov and Khodakovsky 2004]	-	Local	Spatial	First wavelet method
Scalable Predictive Coding [Stefanoski and Ostermann 2010]	3.5 to 8	Local	Spatial and temporal	Focus on low latency streaming
MPEG-4 FAMC [Mamou et al. 2006] [Stefanoski and Liu 2007] [Mamou et al. 2008]	2.5 to 8	Global	Spatial and temporal	Different modes (dowloadable/streamable /scalable)
Fine Granular Scalable [Ahn et al. 2013]	2 to 6	Local	Spatial and temporal	Focus on low latency streaming
CoDDyAC + Optimizations [Váša and Skala 2007; 2009] [Váša and Skala 2010; Váša 2011]	0.5 to 5	Global	No	Trajectory space PCA

Note: compression rates in bit per vertex per frame for a KG error equal to 0.05%.

Ce type d'algorithme concernant non pas des objets 3D précisément, ces algorithmes ne nous intéressent pas dans notre sélection de l'algorithme que nous implémenterons.

E. Conclusion

En termes des différentes observations que nous avons pu faire, nous éliminerons les algorithmes Dynamic Mesh Compression pour la raison que nous avons déjà cité. Nous éliminerons aussi les algorithmes Single-Rate Mesh Compression pour leur tendance à être dépendant à la composition du modèle 3D. Notre choix se porte plus sur les Progressives Mesh Compression, comparé aux Random Accessible Mesh Compression particulièrement pour des termes de simplicité, surtout pour l'interface que nous aurons à implémenter.

Nous avons décidés de porter notre choix sur l'algorithme Kd-Tree coder pour plusieurs raison :

- Le Kd-Tree est un concept que nous avons déjà abordé dans le cadre du master Imagine ;
- Taux de compression intéressant ;
- Avantage des Progressive Mesh Compressions ;
- Peut être étendu et optimisé de plusieurs manière (par exemple : applications aux modèles 3D de polygones, ou des "triangle soups") ;

II. Choix et prévision d'implémentation

Notre choix étant donc l'algorithme du Kd-Tree coder, nous nous sommes intéressé à celui par l'article scientifique *Progressive Lossless Compression of Arbitrary Simplicial Complexes* [2].

Le fichier compressé sera un flux binaire, qui contiendra le model 3D décimé dans sa forme de moins bonne qualité sous la forme non pas des vertexs avec leurs coordonnées mais de la division récursive de la bounding box jusqu'à ce que chaque "case" ne soit composé que d'une seul vertex. Puis le fichier sera composé d'une suite "d'opération" de décimation effectuée sur le modèle pour inverser le processus de décimation et ainsi peu à peu retrouver un modèle 3D très similaire à celui qui a été compressé (un modèle jugé comme compressé sans perte).

On peut remarquer que la division de la boîte englobante du modèle s'effectue, d'une certaine manière, comme la phase de Split dans l'algorithme de *Split & Merge*, notre sujet de projet l'année dernière.
Des informations complémentaires en début de fichiers seront probablement ajoutées pour la lecture du fichier et son utilisation.

Concernant le chiffrement, il pourra potentiellement s'effectuer au cours de la compression, en chiffrant par exemple les informations nécessaires à la lecture du fichier, en chiffrant certaines parties du modèle 3D totalement décimé ou les opérations pour inverser la décimation.

III. Gestion du temps

Les parties du projet étant dépendantes avec d'autres pour la plupart pour pouvoir visualiser leur fonctionnement, la répartition des tâches ne pourra pas se

faire de manière précise dès le début du projet. Elle se fera au fur et à mesure de l'avancement des différentes parties.

Voici les différentes parties (les tâches d'une certaine manière) du projet :

- Lecture d'une ou plusieurs types de fichiers 3D ;
- Division récursive de la boîte englobante ;
- Opération de "Edge Collapse" ;
- Opération de "Vertex Unification" ;
- Opération de "Edge Expansion" ;
- Opération de "Vertex Split" ;
- Organisation du fichier binaire ;
- Choix d'utilisation des opérateurs de décimation ;
- Algorithme de compression ;
- Algorithme de décompression ;
- Chiffrement du fichier binaire ;
- Test liée aux chiffrement ;
- Logiciel visuel pour l'affichage du modèle décompressé sous les différents niveaux de détails ;

IV. Bibliographie

(Note : Ici ne sont cités que les documents utilisés dans le compte-rendu, beaucoup plus d'articles ont été lue pour choisir l'algorithme que nous avons choisi).

[1] A. MAGLO, G. LAVOUÉ, F. DUPONT, C. HUDELOT (2013) 3D mesh compression: survey, comparisons and emerging trends. *ACM Computing Surveys*, Vol. 9, No. 4, Article 39. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.687.74&rep=rep1&type=pdf

[2] PIERRE-MARIE GANDOIN, OLIVIER DEVILLERS. (2002) Progressive Lossless Compression of Arbitrary Simplicial Complexes. *ACM Transactions on Graphics, Association for Computing Machinery*, pp.372-379. https://hal.inria.fr/file/index/docid/167216/filename/hal.pdf