3D Mesh Simplification Techniques for Image-Page Clusters Detection

Costin-Anton Boiangiu, Bogdan Raducanu
Computer Science Department
"Politehnica" University of Bucharest
Splaiul Independentei 313, Bucharest
ROMANIA
Costin@cs.pub.ro, BRaducanu@student.cs.pub.ro

Abstract: Entity clustering is a vital feature of any automatic content conversion system. Such systems generate digital documents from hard copies of newspapers, books, etc. At application level, the system processes an image (usually in black and white color mode) and identifies the various content layout elements, such as paragraphs, tables, images, columns, etc. Here is where the entity clustering mechanism comes into play. Its role is to group atomic entities (characters, points, lines) into layout elements. To achieve this, the system takes on different approaches which rely on the geometrical properties of the enclosed items: their relative position, size, boundaries and alignment. This paper describes such an approach based on 3D mesh reduction.

Key-Words: automatic content conversion, document digitization, layout identification, entity clustering, mesh reduction, heightmaps, terrain, level of detail

1 Introduction

Automatic content conversion systems rely heavily on a good entity clustering module. The purpose of this module is to extract the intended form of the original document and present it to an OCR engine for the final processing. [2, 3] Once the original layout has been identified, the application can safely process texts, images or graphic regions and export the results in an electronic document.

Within this kind system, one of the most difficult tasks is to identify text paragraphs by grouping individual characters together. [21]There are several approaches used to solve this problem [14, 15, 16], most of them making use of geometrical estimators in order to group words or characters into clusters.

Every method is unique, as it takes into account different properties of the examined entities: size, relative position, shape, orientation [17, 18, 23].

Our approach is based on a model which tries to incorporate all these geometrical aspects into a graph-like structure – a triangular mesh. A graph naturally contains information about the relative position of the elements, their orientation and alignment. Graph algorithms can be used for grouping neighboring elements into desired clusters.

Considering an image as a collection of points, the triangulation of this collection has some interesting properties. For example, we have successfully used a Delaunay [20] triangulation to solve the entity clustering task. [6, 19] In the following we will present an alternative to the current solutions for this task, based on a triangulation of the original image. [1, 7, 8]

2 Problem Formulation

Given a black and white input document, obtained from a scanning device, we need to retrieve a collection of entity clusters that form the layout of it. An entity is a generic term and can be associated with both a character or a line. A cluster of character entities might represent a paragraph while a cluster of line entities might form a table, a column separator, a border, etc.

The primary logic used to group entities into clusters is based on their relative position. Entities that are close to each other have a greater probability of belonging to the same collection. Some of the current standard clustering algorithms are based on exactly this kind of observation. They generally use Delaunay triangulations to form clusters by growing them in each possible direction.

This paper focuses on a similar approach, based on forming a triangle mesh, using mesh reduction algorithms. This takes advantage of the fact that white spaces in the image act as a separators between entities. Apart from that, large whitespaces may be naturally detected by the mesh approximation techniques. Several mesh-simplification techniques will be evaluated in order to decide which ones are

the most suitable for this type of processing. All of them are generating output triangles that fall into two categories: small-sized and large-sized triangles.

3 Mesh Reduction

3D computer graphics work with polygonal models. Every 3D object is represented as a collection of vertices, edges and faces. In order to enhance performance of applications working with polygonal models, the collection of faces (polygons) of a model is often reduced to a subset which holds its basic topology (this is where mesh reduction algorithms come into play). [6, 7, 9]

Below, we will briefly review a few different approaches used for the simplification of polygonal surfaces. [10, 15]

3.1 Vertex Decimation

This method iteratively selects each vertex, removes all adjacent faces and then triangulates the remaining hole. The vertex is selected based on its distance to the plane formed by its neighbors.[10]

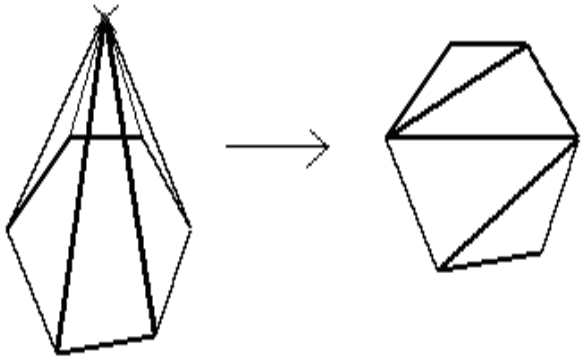

Fig. 1 – Vertex Decimation

3.2 Vertex Clustering

With this approach, a bounding box is formed around the 3D model and divided into a grid (small cubes). The vertices that lay within every cube are removed and a new vertex is added in their place.[10]

3.3 Edge Contraction

This is a general technique which basically selects two adjacent vertices and removes the edge between them. This way the faces that were adjacent to that edge will be reduced. There are several algorithms that use edge contraction. The essential difference between them is the way they choose which edges to contract.[10, 11, 16]

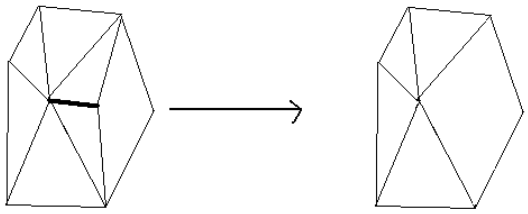

Fig. 2 – Edge Contraction

3.4 Pair Contraction

Pair Contraction is a generalization of Edge Contraction. Here, a set V of vertices is chosen and joined in a single vertex, instead of choosing just 2 adjacent vertices. [11, 12] In order to select a contraction to perform during a given iteration, we need to introduce a cost for this contraction. This cost is introduced under a quadratic form for each vertex.

Through a contraction several vertices are joined into one. The resulting vertex is determined by minimizing the quadratic form. This is an advanced mesh reduction method that can be used with any error metric. [10, 11, 13]

3.5 Real-time Optimally Adapting Meshes (ROAM)

A ROAM (Real Time Optimally Adapting Meshes) algorithm [4, 5] may also be used for building triangular meshes. Apart from generating the best results in both from the point of view of execution time and accuracy, this method is optimized in an intuitive and flexible way. The error obtained in the visualization space produces a guaranteed representation error-range window and results in a specific number of triangles at output. Frame coherence is used for operation at high frame rates, employing outputs of thousands of triangles per frame.

The ROAM method uses two priority queues for the 'Split' and 'Merge' operations. [4] As a result, the processing can be realized by a string of 'merge' / 'split' commands which maintain continuous triangulations based on a collection of binary trees (holding the preprocessed triangles). Two standard optimizations are commonly used: incremental triangle strip generation and use of runtime priority lists.

The execution time of a ROAM algorithm is proportional with the number of triangles modified in each frame. This is usually just a low percentage of the computed triangular space. This property makes the algorithm practically independent of the terrain space resolution. Rendering of the

dynamically modified terrain as well as performing node animation operations can also be carried out using this technique.

All algorithms belonging to this type allow the adaptive production of a triangular mesh in a viewpoint dependent way. They are based on representing the terrain as a multiresolution Height-Map which is used during the adaptive representation of the triangle strips inside an image frame.

Fig. 3 illustrate a terrain represented through a rendering method like the one previously described. Fig. 4 shows the edges of the resulting triangular mesh. Having in mind the above observations, we can notice the different resolutions of the two figures.

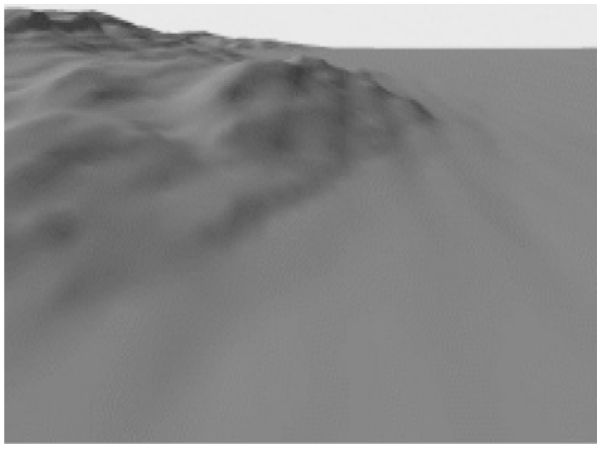

Fig. 3 – A terrain represented with a generation method of the type ROAM, without the triangle mesh edges highlighted

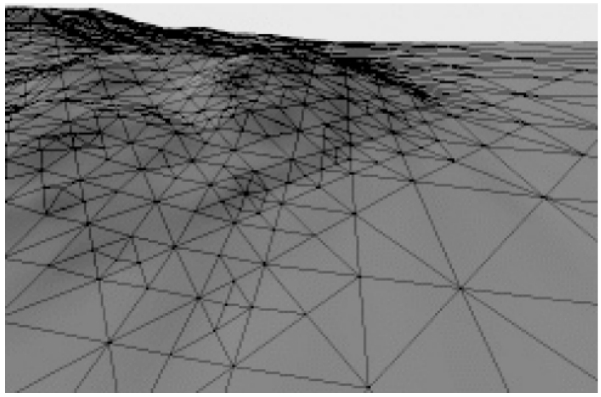

Fig. 4 – The terrain represented in Fig. 3 with the triangle mesh edges highlighted

Fig. 5 is a top view of the resulting triangulated terrain. This is a typical result of a ROAM algorithm: smooth neighborhoods (low frequency/ further apart) generate more imprecise triangulations than the neighborhoods closer to each other or with a rough aspect (high frequency) which are triangulated much slender.

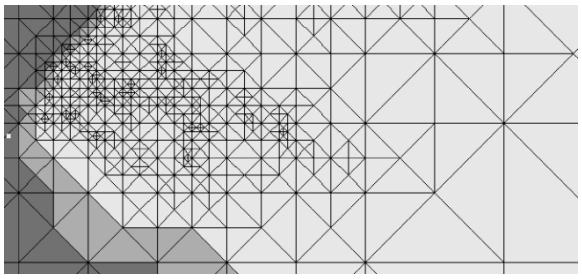

Fig. 5 – Top view of the triangular mesh

ROAM is composed of a preprocessing stage and multiple running stages. The preprocessing step uses a binary triangulation tree in order to produce nested error intervals in bottom-up order. At runtime the generation of each frame consists of 4 steps [4,5]:

1. Incremental and recursive update of the rejection of primitives from the visualization space;
2. Priority updates only for the triangles which may occur in 'split' or 'merge' operations from Step 3;
3. Triangulation update using a Greedy approach for the 'split' or 'merge' operations - two priority queues for each operation;
4. Triangle strip update for the strips affected by the rejection in Step 1 or by the split/merge operations in Step3;

3.5.1. Triangle Strip Representation - Triangulation of Binary Trees

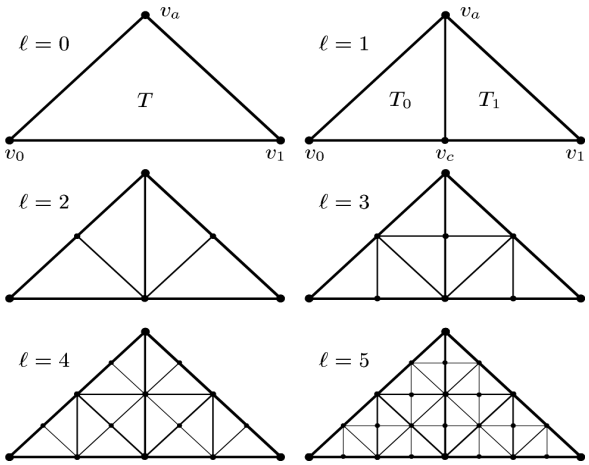

Fig. 6 – Recursive splitting of triangles in the ROAM approach

The root of the tree T = (V_a, V_0, V_1) is defined as being the isosceles right triangle from the smallest level of detail (l = 0).

At the next level of detail (l = 1) the children of the binary tree node are defined by splitting the triangle using the median of the right angle vertex. This results in two new isosceles right triangles. T's left child is T_0 = (V_c, V_a, V_0) and its right child is T1= (V_c, V_1, V_a).

The rest of the triangulation tree is defined through a recursive process which follows the same split pattern of the triangles contained in each node. Fig. 6 displays a tree for l = 0…5.

3.5.2. Triangle Strip Representation - Continuous and Dynamic Triangulation

The triangle strips in the scene space are formed by assigning scene coordinates W(v) to each node. A set of triangles taken from the binary tree may form a continuous strip if any two triangles either do not overlap at any point, or they share a common point or edge.

These triangle strip sets resulted from the triangulation binary tree are to be referred simply as triangulations. Fig. 7 shows a typical neighborhood of triangle T from the triangulation.

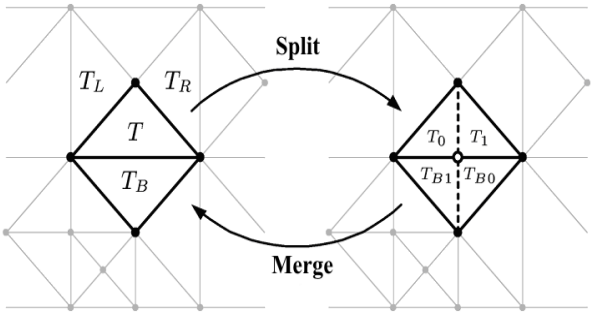

Fig. 7 – Split and Merge operations

Let T_b be T's base neighbor (the one which has the same hypotenuse as T: b_0-b_1). T_l is T's left neighbor as it has the same left edge (V_a-V_0) while T_r is T's right neighbor and has the same right edge (V_1-V_a).

A key observation is that all binary triangulations have T's neighbors on the same level l just like T or on l + 1 for the left/right ones and l - 1 for the base one. All these possibilities are shown in Fig. 7.

When T and T_b are on the same level the (T, T_b) pair will be referred as a diamond. Fig 7 shows a split/merge operation for a diamond triangulation. 'Split' replaces T with the children (T_0, T_1) and T_b with (T_{b0}, T_{b1}). This operation introduces a new node V_c in the middle of the diamond, which maintains a continuous triangulation.

If T does not have a single base T_b, it will be split into its children. 'Merge' can be applied to the (T, T_b) diamond when the children of T and T_b (if T_b exists) are all included in the triangulation. In this case (T, T_b) is called diamond and supports merging within the triangulation.

An important fact for the 'split' and 'merge' operations is that any triangulation can be obtained from another by a careful selection of a sequence of 'merge' / 'split' operations.

If T_b, the base neighbor of a triangle T, belongs to a less smoother level than T, than the initial triangle T cannot be a split-up immediately in the triangulation process. To force T to perform a split, it is necessary that first T_b is forced to execute the split. This approach, in turn, can generate recursive requirements.

A case which requires a total of 4 split operations is presented in Fig. 8. These forced splits are necessary for a good operation of the following algorithm.

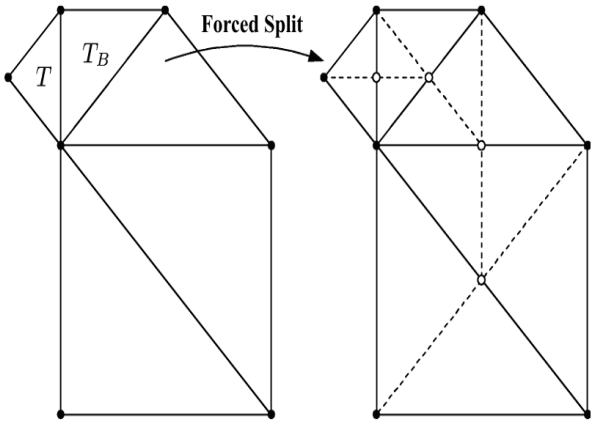

Fig. 8 – Forced Split

3.5.3 Performing optimizations using two waiting queues

1. *Split queue*: Let there exist an association between each triangle t and a monotone

priority p(t) in the range [0..1]. Because the triangulation is constructed top-down, a priority queue Q will be maintained in order to retain information about all triangles in T. The Greedy approach used in top-down fashion generates optimal triangulations each iteration.

2. *Merge queue*: Assume that we are dealing with time variable priorities $P_f(t)$ in the interval [0..1] for the frames f(0,1,..). The goal is to construct optimal triangulations (T_0, T_1, …). If these priorities are modified slowly and continuously, then the optimal triangulations of these successive frames tend to be very similar. In this case, the triangulation for the step $f(T_f)$ can be greatly improved if the step f - 1 is used as a starting point for its deduction. This can be performed by employing a second priority queue Q_m, which will contain all diamonds able to execute 'merge' in the current triangulation. The priority for a diamond that can execute 'merge' (T, T_b) is set to the maximum of the priorities of the included triangles: $\max\{P_f(T), Pf(T_b)\}$. The incremental Greedy algorithm produces an optimal triangulation T_f which has the same priority as the one obtained after applying the top-down not-incremental greedy algorithm on the base triangulation.

3.5.4 Criteria for evaluating the performance of the ROAM approach

A complete system for the visualization of large datasets, and at high frame rates, is generally composed from [4]:

1. A geometrical data disk paging system
2. A texture disk paging system
3. LOD ("Level of Detail") techniques for the pre-triangulated geometry
4. LOD techniques for texture management
5. The rejection of data outside the current volume of visualization
6. The construction of triangular bands

Given the considered target, only the last two aspects are of importance for the current discussion. Taking into consideration the operation execution applied to data that is stored in the main memory, we are interested only in the fast incremental generation of the triangulation meshes and the distribution of the triangle sizes.

The criteria by which the current method will be evaluated are [4]:

1. The necessary time required to reach a pre-established number of output triangles: the presented approach allows the cursive visualization of the scene. The rate is of at least 30 frames/second, for an output size of several thousand triangles. The running time of the algorithm is directly proportional with the number of modified triangles at the output. Normally, this only represents a small percentage of the total triangulation output, making the algorithm practically independent on the resolution and size of the terrain.
2. The flexibility in choosing the error measurement system, in a visualization-dependent way: ROAM uses the maximum geometrical representation error in the screen space as an error measure and a priority for the operation tails. This method can be improved in a number of ways: ensuring a certain degree of representation over a local horizon line, ensuring the correct positioning of the terrain under an auxiliary object from the scene or the elimination of "back-face" oriented details.
3. The representation of interconnected triangular collections (both to the ones found in the preprocessing stage, as well as the ones in the runtime stage): the presented method utilizes binary triangulation trees and continuously builds adaptive triangular bands based on fine "split"/"merge" steps. The triangles are always rectangular-isosceles. As a result, the apparition of degenerated triangles or very narrow triangles that pose great visualization problems is avoided.
4. The simplicity of the algorithm: ROAM class algorithms are relatively easy to understand and implement, because they are oriented towards "split"/"merge" operations with binary trees (this naturally avoids performing of a great deal of unnecessary operations).
5. The quality of the triangular approximation, with respect to the number of given triangles:

the current algorithm produces optimal triangular collections by minimizing the maximum representation error interval for "monotone" error limits (those that do not grow through a "split"). Practical results have shown an excellent quality of the representations obtained through this method.

6. Direct control over the number of output triangles: ROAM produces triangular bands with a directly specified number of triangles. The algorithms that follow only error tolerance can only indirectly control the number of resulted (output) triangles by means of tuning the maximal admitted error. This process is however slow and unproductive. ROAM can operate by aiming a certain tolerance, although in the majority of cases a number of resulted output triangles are desirable.
7. Strict frame rate: ROAM can be progressively optimized from frame to frame and can stop when the allocated time for a frame is about to expire. Even though, normally only a fraction of the allocated time for a frame is necessary to complete the execution of ROAM.
8. Guaranteed error limit: ROAM produces guaranteed errors in the screen space (geometrical distortions). These error limits are obtained either locally or globally through the Greedy approach of the optimization algorithm. In the local case a fast conversion from scene coordinates geometry (preprocessed in binary trees) to screen coordinates is used, depending on the visualization.
9. Memory requirements: one must make the distinction between preprocessing memory and runtime memory. The preprocessing memory is equal to the number of rows, the "finest" height map samples, plus a single "thickness" value for every triangle of the binary tree. The runtime memory occupies space proportionally to the number of triangles that have to be output, which generally represent only a small fraction of the necessary memory in the preprocessing stage.
10. The low production of visual artifacts. Given the screen nature of the metrics used in the error evaluation system in ROAM, the tendency to produce visual artifacts during the movement of the visualization point is minimal. This fact is also due to the relatively small number of changes per frame.
11. General input data: because the algorithm is based on the usage of terrains as a main visualization object (the terrain is described as a height map), the input triangulation is arbitrary and free of any imposed preconditions.

4 Mesh Creation and Reduction

In the attempt to separate entities into homogenous collections we will use one of the algorithms above in the following way: first, we treat the input binary image as a displacement map (or height-map) [12][13].

A heightmap is a grayscale image used to store 3D terrain-like information. Every pixel value is treated as an altitude of that point, as seen in the images below. Only grayscale images will be considered as valid HeightMaps because a multiple-component image does not naturally translate into a terrain-like HeightMap.

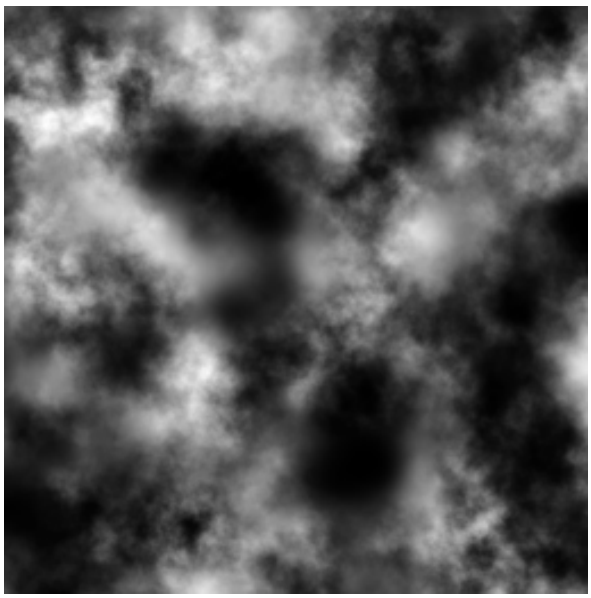

Fig. 9 – HeightMap

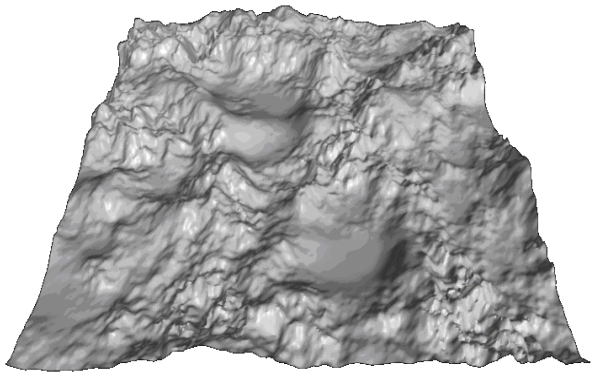

Fig. 10 – Terrain generated from the HeightMap in Fig. 9

ICE: Kacheln auf den Gleisen – war es nur ein böser Jux?

Der ICE-Zwischenfall auf der Strecke Hamburg–Berlin – ein Kinderstreich?

17.45 Uhr, der ICE 537 „Carl von Ossietzky" gleitet mit 150 km/h bei Boizenburg über die Strecke. Plötzlich spürt der Lokführer ein Rumpeln. Er meldet dem Leitstand: **„Ich fuhr über Schottersteine."**

Er bremst auf 120 km/h ab, kommt mit 20 Minuten Verspätung in Berlin an. Die Bahn erstattet Anzeige gegen Unbekannt und zieht den Zug aus dem Verkehr.

30 Grenzschutzbeamte suchen die Strecke nach Hinweisen auf einen Anschlag ab, ein Hubschrauber überfliegt das Gelände. **Beim zweiten Sucheinsatz entdecken die Beamten zertrümmerte Fliesen im Schotterbett.** BGS-Hauptkommissar Stührwohld: „Es waren etwa fünf Kacheln."

Bahnsprecher Hartmut Sommer: „Im ICE-Werk Hamburg wurden kleine Schaden an den Monoblockrädern – sie haben keine Radreifen – festgestellt."

Nachmittags fuhr der Zug leer nach Bremen, von dort als ICE 997 um 16.19 Uhr weiter nach Frankfurt.

In der Aufnahme der Unfallstation fragt der Arzt den Patienten routinemäßig: „Sind Sie verheiratet?" – „Nein, ich bin überfahren worden."

***

„Tolles Kleid, Mami. Hast du das von Papi?" – „Nein, wenn ich mich auf den verlassen müßte, hätten wir nicht einmal dich."

***

Das Känguruh-Baby hat einen Wunsch: „Mama, darf ich das Glühwürmchen mit hineinnehmen? Ich möchte noch ein bißchen lesen."

***

Im Urlaubsflieger wird die Mutter blaß: „Ich habe den Herd nicht abgestellt. Das Haus kann abbrennen." Ihr Mann beruhigt sie: „Kein Problem. Ich habe das Badewasser laufen lassen."

Fig. 11 Original Image

From the input image (Fig. 11) we construct the polygonal surface model of the underlying terrain.

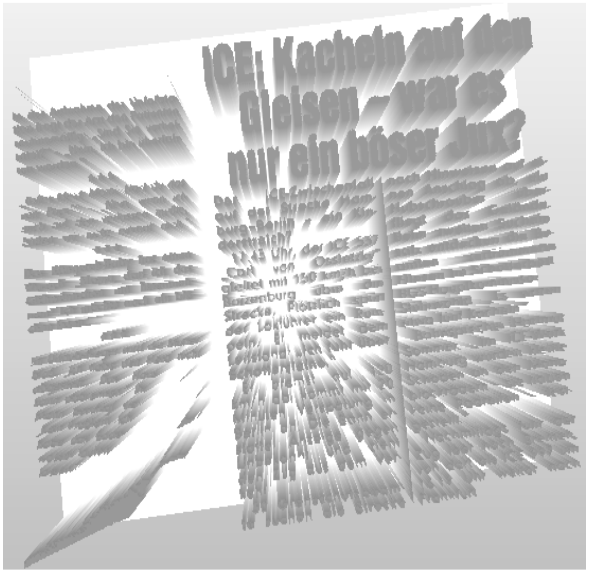

Fig. 12 - 3D Model generated from the Original Image presented in Fig. 11

Once the 3D model is born, which in fact is a *complete triangulation mesh* (at 4 adjacent 3D points will result in a grid-like disposition 2 opposite triangles) one mesh reduction algorithm is applied and at output will result a simplified polygonal skeleton.

Fig. 13 shows such a 3D skeleton obtained from the image in Fig. 11 as a Height-Map.

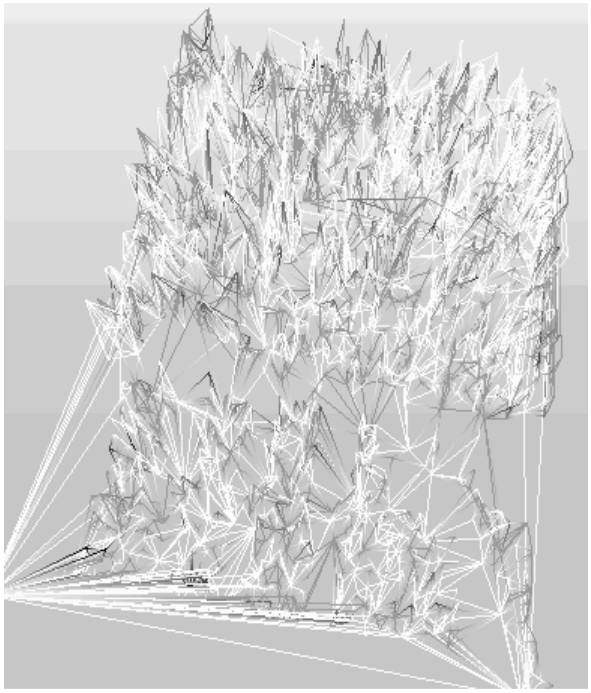

Fig. 13 – Simplified Polygonal Mesh obtained from the 3D model in Fig.12

Some *key-observations* can be made regarding this new mesh:

1. Space regions from the page tend to be represented by large area triangles;
2. Textual paragraphs have small area triangles;
3. Noises in the documents will be successfully suppressed by any mesh reduction algorithm that takes volumetric error as an error metric, because noise appearance inside large whitespaces will not greatly affect the volume of the mesh.

The resulted triangular mesh in then projected on the original image. Because of the way that mesh reduction algorithms work, each triangle, with high probability, now encloses either white space or character entities. Small, closely positioned triangles will form character clusters while large adjacent triangles will form white space regions.

Fig. 14 – Original image

Fig. 16 – Arcs after triangle size-based threshold

Fig. 15 – Image with all the Delaunay "Inter"-Arcs

Fig 17 – The resulted clusters

The following mechanism was successfully used to classify entities into clusters: first, a Delaunay triangulation on top of the original image (Fig. 14) is constructed. A dense triangular mesh will be formed (Fig. 15), were the figured arcs represents the lowest distance between two entities with respect to the Delaunay triangulation. In order to obtain this result, all the triangles that connect only one element ("Intra"-triangles) in the original image or those that connect three different elements are rejected. The resulted triangles are then grouped in "*proximity sets*" for every couple of distinct elements. Inside these "*proximities*", the minimum-distance edges are computed. These edges are called the "Inter"-Arcs (Fig. 15). The simplified 3D mesh will be afterwards projected onto it and all Delaunay triangle edges will be inspected. Every edge intersecting a projected triangle that exceeds a threshold size will be removed (Fig 16). Based on the properties of the 3D mesh triangles, the remaining edges, with high probability, will form connected clusters, thus solving our problem (Fig. 17). Please observe that the main clusters were detected correctly, without breaking further in more unnecessary parts.

5 Conclusions and Future Work

Even though it may seem that using such sophisticated algorithms is inappropriate for this task, our experience in this field has proven us that the entity clustering problem is of considerable difficulty and every promising approach should be investigated carefully [1, 22]. We have tested this new method on our collection of scanned newspaper pages and after analyzing the results we concluded that it is a decent solution for layout clustering that in most of the cases should be combined with classical approaches like separator detection [2] or like white-space retrieval.

Other techniques, like grouping base elements into first-level clusters based on some heuristically measurements for the resemblance [3] might also be considered in order to validate the formation of the clusters after the mesh simplification and triangle size-based threshold operation.

Also, some post-processing techniques [4] applied on the resulted mesh might be taken into account as well, because in most of the cases, an altered distribution between the "small"-size and "big"-size triangles might be needed.

As a follow-up research, a close study of 3D polygonal metrics would be helpful to identify the best suited metric to be used with one of the mesh reduction algorithms for our particular type of 3D models.

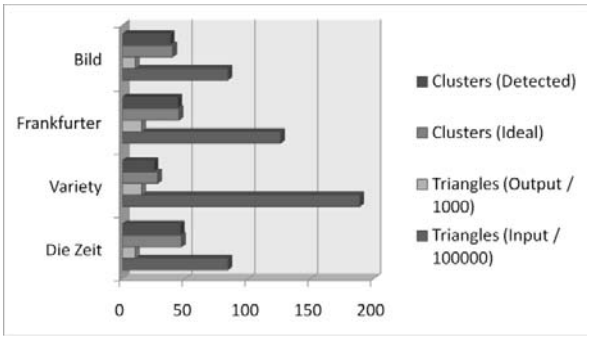

Fig. 18 – Results of the described approach

In order to validate the proposed approach, the following test scenario was selected: a set of representative documents from 4 different publications:

1. Bild
2. Frankfurter
3. Variety
4. Die Zeit

The results of the simplification process as well as the resulted number of clusters were plotted in the chart presented in Fig. 18. The conclusion is that with a massive simplification of the input data, the output clusters are very close (but slightly smaller in number) to the ideal number of clusters contained in the input documents and this means that most of the clusters were successfully detected, validating the proposed solution.

The test scenario was implemented using a ROAM-based triangulation simplification, with a fixed number of triangles to output. Since the ROAM approach is view dependent, the resulted mesh is in fact a combination of simplification meshes resulted by moving the camera throughout the HeightMap in a raster-like motion and making use of the incremental generation of the algorithm. For every pixel, the size of the triangle that is detected with the view camera placed directly above the pixel is the size that is afterwards used in the triangle threshold-based processing.

Another thing that should be mentioned is that the proposed approach is a "*Split-Based Top-Down clustering*", which is different than the most of the current approaches used in Automatic Content Conversion Systems [3] which are *"Merge-Based Bottom-Up clustering*". The Bottom-Up approaches are generally constructed in the process of adding basic layout elements together using a normalized resemblance function between those elements.

Other modern mesh simplification strategies will be evaluated in the near future, as well, in order to fully explore the qualities of the proposed approach.

References:

[1] C. A. Boiangiu, B. Raducanu "Entity Clustering Using 3D Mesh Simplification", *Proceedings Of The 9th WSEAS International Conference On Automation And Information*, Bucharest, Romania, June 2008, pp. 460-463.

[2] C.-A. Boiangiu, B. Raducanu "Robust Line Detection Methods", *Proceedings Of The 9th WSEAS International Conference On Automation And Information*, Bucharest, Romania, June 2008, pp. 464-467.

[3] C. A. Boiangiu, A. C. Spataru, D. C. Cananau, A. I. Dvornic, "Automatic Text Clustering and Classification Based on Font Geometrical Characteristics", *Proceedings Of The 9th WSEAS International Conference On Automation And Information,* Bucharest, Romania, June 2008, pp. 468-473.

[4] Irina Semenova, Ichiro Hagiwara, "Explicit Method to Optimize Surface Mesh Quality",*WSEAS Transactions On Information Science And Applications*, Issue 1, Volume 1, July 2004, pp. 517-523

[5] Duchaineau, M. Wolinsky, M. Sigeti, D.E. Miller, M.C. Aldrich, C. Mineev-Weinstein, M.B, "ROAMing Terrain: Real-time Optimally Adapting Meshes", *IEEE Visualization '97, Proceedings*. 19-24 Oct 1997, pp. 81-88

[6] Costin-Anton BOIANGIU, *Multimedia Techniques*, Macarie. 2002

[7] Costin-Anton Boiangiu, *Elements of Virtual Reality*, Macarie, 2002

[8] Costin-Anton Boiangiu, The "Beta-Shape" Algorithm for Polygonal Contour Reconstruction, *The 14th International Conference on Control System and Computer Science*, C.6. Vol. II, 2003

[9] Serban Petrescu, Zoea Racovita, Florica Moldoveanu, Costin-Anton Boiangiu, Alin Moldoveanu, Gabriel Hera, Neuron GIS Solutions for the Optimal Path Selection, *The 11th International Conference on Control System and Computer Science*, 11.10, Vol. II, 1997

[10] M. Garland and P. Heckbert, Surface simplification using quadric error metrics, *SIGGRAPH 97 Proceedings*, 1997, pp. 209–216

[11] P. Cignoni, C. Montani, and R. Scopigno, A Comparison of Mesh Simplification Algorithms, *Computers & Graphics*,Vol. 22, 1998, pp. 37-54

[12] M. Garland and E. Shaffer, A multiphase approach to efficient surface simplification, *Visualization'02 Proceedings*, 117–124.

[13] L. DeFlorani, E. Puppo, and R. Scopigno, Level-of-Detail in Surface and Volume Modeling, *IEEE Visualization 98 Tutorials*, Vol 6, IEEE CS Press. 1998

[14] F. Bernardini, F., Martin, I., Mittleman, J., Rushmeier, and G. Taubin, Building a digital model of Michelangelo's Florentine Pieta, *IEEE Computer Graphics and Applications* 22, pp. 59–67.

[15] Poh Kok Loo; Chew Lim Tan, Detection of word groups based on irregular pyramid, *Proceedings of the Sixth International Conference on Document Analysis and Recognition, ICDAR, IEEE Computer Society* , 2001, pp. 200 – 204

[16] Pietik¨Ainen, M. Okun, O., Text Extraction from Grey Scale Page Images by Simple Edge Detectors, *Proc. of the 12th Scandinavian Conference on Image Analysis*, SCIA, Bergen: Norway, 2001, 628–635

[17] Karatzas, D. Antonacopoulos, A., Two Approaches for Text Segmentation in Web Images, *Seventh International Conference on Document Analysis and Recognition, IEEE*, Scotland, Edinburgh, 2003, pp.1–131

[18] Clark, P., Mirmehdi, M., Finding Text Regions Using Localized Measures, *Proceedings of the 11th British Machine Vision Conference*, 2000

[19] Koivusaari, M. Sauvola, J. Pietik, Ainen, M., Automated Document Content Characterization for a Multimedia Document Retrieval System, *Proc. SPIE Multimedia Storage and Archiving Systems* II, TX, Dallas, 1997, pp.148–159

[20] Fortune, S., Voronoi *Diagrams and Delaunay triangulations, Handbook of discrete and computational geometry*, CRC Press, 1997, pp. 377-388

[21] B. Chen, and L. He, "Fuzzy template matching for printing character inspection", *WSEAS Transactions on Circuits and Systems*, Issue 3, Vol. 3, 2004

[22] Chen Zhuo, Francis Y. L. Chin, Ronald H. Y. Chung , Automated Hierarchical Image Segmentation Based on Merging of Quadrilaterals, *WSEAS Transactions on Signal Processing*, Issue 8, Volume 2, August 2006

[23] M.I. Rajab., "Feature Extraction of Epiluminescence Microscopic Images by Iterative Segmentation Algorithm", *WSEAS Transactions on Information Science and Applications*, Issue 8, Vol. 2, 2005

[24] Nadeem Abbas Zaidi, Noor Muhammad Shiekh , A Robust Object Recognition Technique using Statistical Parameters, *WSEAS Transactions On Signal Processing*, Issue 9, Volume 2, September 2006