Planar Grid Generation for Cloth Simulation

Stanislav Kostadinov, Tzvetomir Vassilev

Abstract: *.This paper presents two approaches for planar grid generation. One of them is algebraic and it is based on equality of quad areas and the other one is an improved direct optimization method. The idea of the paper is to provide a fast algorithm for mesh generation to do a more realistic simulation of mass-spring models. Results and conclusions are presented at the end of the paper.*

Key words: *Cloth Simulation, Grid Generation*

INTRODUCTION

Cloth simulation has become a major field of research in computer graphics in the last 20 years. Its main applications are computer games, garment design and electronic commerce. Most of the mass-spring based approaches [1, 2] require a rectangular topology grid to be generated on a cloth piece, so that the physical simulation of the masses and springs can run. One of the easiest solutions is to use an algebraic method [3], but as shown in the next sections, it does not always produce good results.

One of the approaches presented in this paper modifies an existing direct optimisation method by adding another term to the length functional. The other solution is an algebraic and it uses division into relatively equal areas.

The rest of the paper is organized as follows. The next section reviews previous work on grid generation. Section 3 describes the algebraic and direct optimisation approaches. Section 4 presents the algebraic and the modified optimisation approach using the diagonal term in the functional. Section 5 gives results of the experiments and the last section concludes the paper.

PREVIOUS WORK GRID GENERATION

As above mentioned the fastest and easiest method to implement is the algebraic mapping with two dimensional interpolation [3]. However, in many cases it does not produce good results and more complex methods were developed that generate smoother meshes. One of the best is the elliptic grid generator [4] borrowing ideas from fluid dynamics and solving partial differential equations. Its main disadvantage is that non-linear systems of many unknowns have to be solved, which can be done only by iterative numerical methods. As a result the method is slow.

Khamayseh and Hamann [5] modified the elliptic system using non-uniform rational B-splines (NURBS). In this approach the original geometry is given as analytically defined NURBS surfaces, but the method is designated mainly for generating meshes on 3D surfaces, not on a plane.

Castillo and Otto [6, 7] proposed a direct variational optimisation approach. They treat the grid generation problem as discrete from the very beginning. Several discrete functionals (length, area, orthogonal) are defined and their variation is minimised. This produces linear systems, which can be solved much faster than for the elliptic generator. If only the length functional is used, then the x and y coordinates are decoupled and two independent systems should be solved, one for x and one for y coordinates, which have the same matrix. This results in a very fast generation, but minimisation only of length variation does not always produce good results. That is why area has to be added too. However, x and y coordinates are coupled and the number of unknowns is increased twice.

The main idea of the first approach, proposed in this work, is to improve the length functional by adding the length of diagonals and in this way preserve the efficiency of the method, while improving its quality at the same time. The second approach provides a

mechanism for subdivision of the original figure into quads with almost equal areas that is fast and it is based on intersection of possible geometrical locus of points.

ALGEBRAIC AND DIRECT OPTIMISATION METHODS

The problem of rectangular grid generation is illustrated in Fig. 1. Given a rectangular logical domain, on which it is easy to generate a rectangular mesh, we have to find a mapping function R(u,v) to the physical domain. The variables u and v are in the interval from 0 to 1. The boundary curves R(u,0), R(u,1), R(0,v), R(1,v) can be easily generated using an equal length approach. So, the main problem is to find point distribution inside area of the physical domain.

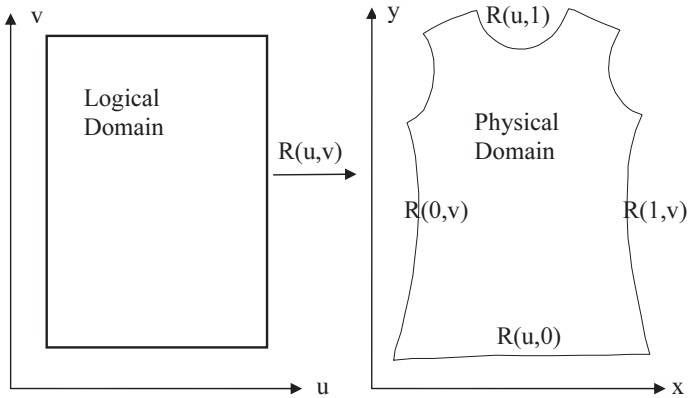

Fig. 1. The problem of rectangular topology grid generation

Algebraic interpolation

The main idea here is to apply a two-dimensional linear interpolation. Equation (1) gives the matrix form of the linear interpolant.

$$R(u,v) = \begin{bmatrix} 1-u & u \end{bmatrix} \begin{bmatrix} R(0,v) \\ R(1,v) \end{bmatrix} + \begin{bmatrix} R(u,0) & R(u,1) \end{bmatrix} \begin{bmatrix} 1-v \\ v \end{bmatrix} - \begin{bmatrix} 1-u & u \end{bmatrix} \begin{bmatrix} R(0,0) & R(0,1) \\ R(1,0) & R(1,1) \end{bmatrix} \begin{bmatrix} 1-v \\ v \end{bmatrix} \quad (1)$$

Direct optimisation

Castillo and Otto presented a discrete variational grid generation method. They define the following length functional:

$$F_L = \frac{1}{2} \sum_{u,v} \frac{\left(R(u+1,v) - R(u,v)\right)^2}{Lu^2} + \frac{\left(R(u,v+1) - R(u,v)\right)^2}{Lv^2} \quad (2)$$

and minimise it by setting the partial derivatives to zero. In their functional they assume different weights L_u, L_v for the horizontal and vertical edges of the grid. It is easy to be seen that if L_u is bigger than L_v then the vertical lines have greater influence than the horizontal ones. The minimisation of (2) leads to equation (3), which is applied to all R(u,v) where u is between 1 and u_{max} -1 and v is between 1 and v_{max} -1.

$$\begin{aligned} &R(u,v)*[2/(Lv*Lv)+2/(Lu*Lu)]- \\ &[R(u-1,v)+R(u+1,v)]/(Lu*Lu)- \\ &[R(u,v-1)+R(u,v+1)]/(Lv*Lv)=0 \end{aligned} \quad (3)$$

The result is a system of linear equations with $(v_{max}-2)*(u_{max}-2)$ unknown variables. If $L_u=L_v$, the system matrix elements M[i, j] can be computed in the following way. All main diagonal elements are equal to 4. If $i=u*v_{max}+v$ and $j=i\pm1$ then M[i, j]=-1. If $j = i\pm v_{max}$ then M[i, j]=-1.

A well-known method for solving linear equations is the LU decomposition. It is called that way, because of L for lower and U for upper triangle matrixes that are used for solving the equations. More details about the LU decomposition can be found in [9].

Another faster solution is a special case of LU decomposition which is called Cholecky decomposition [10]. It is used for square symmetric matrices. The difference between LU decomposition is that in case of real number solutions the L and U matrix are transposed ($L_{ij}=U_{ji}$). That speeds up the solution by removing half of the pre-generated solution data.

METHODS PROPOSED IN THIS PAPER

Algebraic method based on equality of inside quads area

This approach is iterative and it separates physical domain into V_{max} number of quad-strips that have relatively equal areas S/U_{max}, where S is the total area. It could be considered as recursive and after each iteration a new pass of the algorithm is performed with 1 iteration less than the previous. Each R(u,v) point is generated as an intersection of two lines. The first line is the bisector of the angle R(u-1,v-1), R(u,v-1), R(u+1,v-1). The second line is parallel to the line between R(u-1,v-1) and R(u,v-1) and passes through point R(u-1,v). The distance between the two parallel lines is equal to $1/V_{max}$ of the distance between R(u,0) and $R(u,v_{max})$. After all on the current row are generated, then the area of the quad-strip formed by the current and previous rows is computed. If the difference between that area and S/U_{max} is greater than a certain threshold (2%-3%), then that iteration is started again but the distance between the parallel lines is multiplied by the ratio between S/U_{max} and the last previously computed area. The threshold could be set to almost 0% if the algorithm should be more precise, but there is no visual difference for threshold less than 2.5%, however the number of iterations is increased. This is illustrated in Fig. 2.

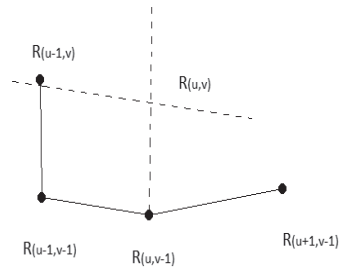

Fig. 2. Generation of a single point

If the difference is less than the threshold, then a function is called, which divides the currently generated row R(0,v) $R(u_{max},v)$ into segments with relatively equal lengths. This algorithm is used for generation of the contours of the initial grid. This functionality is illustrated in Fig. 3.

Another approach for the generation the row is: firstly the area of the R(u-1,v), R(u,v-1), R(u,v) is calculated by subtracting the ideal area for each quad $S/(u_{max}*v_{max})$ with the

area of the triangle R(u-1,v-1), R(u,v-1), R(u-1,v). Then the length of the segment R(u,v-1), R(u-1,v) is calculated. The formula for areas of triangles is $St=0.5*a.h_a=0.5*b.h_b=0.5*c.h_c$. So the formula for the height of the segment R(u,v-1) R(u-1,v) looks like this: $h=2*St/a$ where St is the area and a is the length of the segment. Finally new line is generated and it is parallel to the line R(u,v-1) R(u-1,v) and the distance between the two lines is h. The intersection point of this line and the line parallel to R(u-1,v-1) R(u,v-1) used in the above approach is R(u,v). The second approach generates grids with evenly distributed quads' areas, but it is slower and that's why the first one is more appropriate.

Newly generated quads have relatively equal areas. The ratio of the biggest and the smallest area in this polygon is 1.62. After each iteration new row is generated and all of the segments R(u-1,v) R(u,v) are equal.

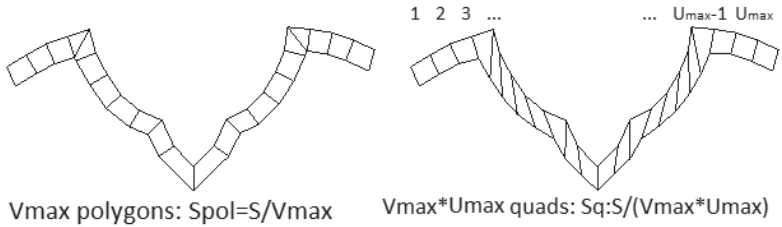

Fig. 3. Generation of row with algebraic method based on equality on quad areas

Improved length functional

The direct optimization method has one big disadvantage. Some of the quads (in this example they are located around the neck and arms) are thin and long and their area is almost equal to 0 despite the good situation and shape of the others. One solution of minimization of this problem is to involve the quad diagonals in the equation (3).

$$\begin{aligned} &R(u,v)*[2/(Lv*Lv)+2/Lu*Lu)+4/(Lv*Lv+Lu*Lu)]- \\ &[R(u-1,v)+R(u+1,v)]/(Lu*Lu)- \\ &[R(u,v-1)+R(u,v+1)]/(Lv*Lv)- \\ &[R(u-1,v-1)+R(u+1,v-1)+ \\ &R(u+1,v+1)+R(u-1,v+1)]/(Lv*Lv+Lu*Lu)=0 \end{aligned} \quad (4)$$

Equation (4) makes the grid look more realistic. Benchmark tests show that execution times of the original and the optimized methods vary less than 1%.

RESULTS

The algorithms were implemented in Java 1.6.0_26 under Windows 7. Results are shown in Figures 4, 5 and 6. For the tests presented below we assume that the ratio between L_u and L_v for the direct optimisation is 4.

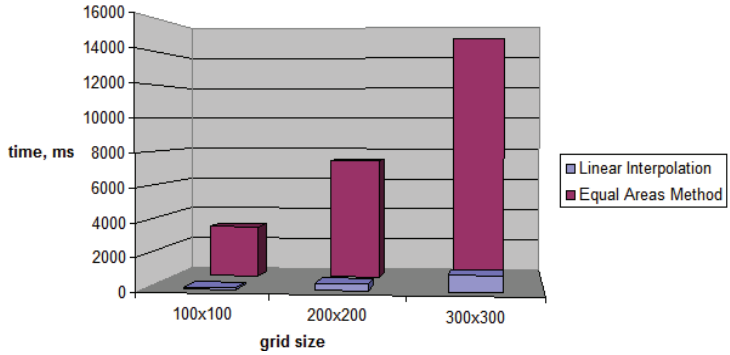

Fig. 4. Time for execution of 1000 iteration versus grid size. Algebraic algorithms

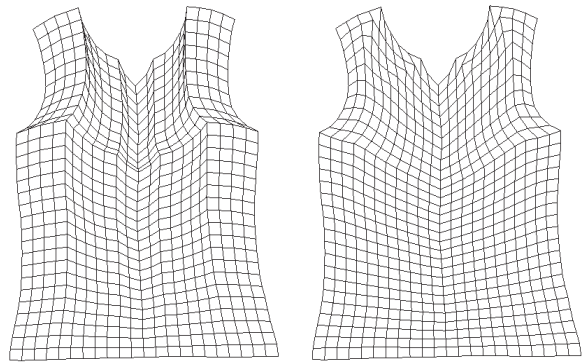

Fig. 5. Algebraic generation: left-linear interpolation, right-based on equality of areas

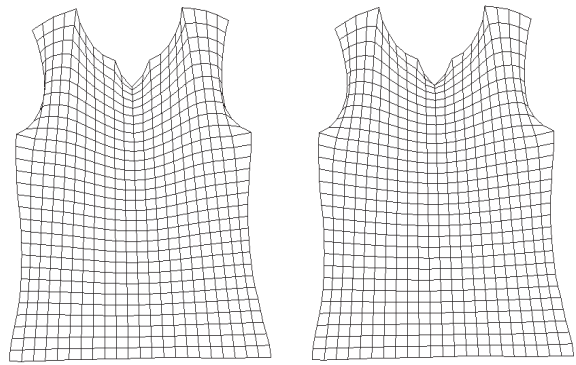

Fig. 6. Direct optimisation: left-original method, right-improved length functional

CONCLUSIONS AND FUTURE WORK

This paper presented two approaches for planar grid generation. The following conclusions can be drawn.

The approach based on discrete functional minimisation generates smooth rows and columns but there is huge difference between the areas of the quads. The execution time for grids bigger than 50 by 50 knots is very big. One possible optimization is caching of decomposition matrices. Once it is been used for generation, the matrix could be kept in the memory for further use. Of course, a lot of memory should be allocated for that process. One optimization is solving the equations using the Cholesky decomposition method.

The method based on the equal areas is faster if the difference between the smallest and biggest quad areas is approximately equal or at least the ratio is smaller than 2. It could be tuned and then the ratio will approach 1, but the execution time will become almost equal to the first algorithm. The advantage of that solution comes from using of geometric locus of points and integration of both x and y coordinates. One disadvantage is the rough look of the inner rows and columns.

A point for future work is implementation of a discrete functional that is based on equality of quad areas. The execution time should be kept in mind because these approaches have geometrical progression between the number of grid knots and the time for generation.

REFERENCES

[1] Provot X. Deformation constraints in a mass-spring model to describe rigid cloth behaviour. Proceedings of Graphics Interface 1995; 141-155.

[2] Vassilev, T., Spanlang, B., Chrysanthou Y. Fast Cloth Animation on Walking Avatars, Computer Graphics Forum, 2001, 3 (20), 260-267.

[3] Smith, R. E., Algebraic Grid Generation, Numerical Grid Generation, Ed. Joe F. Thompson, North Holland, 137, 1982.

[4] Thompson, J. F., Warsi, Z.U.A., Mastin, C. W., Numerical Grid Generation, Elsevier Science Publishers, 1985, http://www.erc.msstate.edu/publications/gridbook/

[5] Khamayseh, A. Hamann, B. Elliptic Grid Generation using NURBS surfaces, Computer Aided Geometric Design, 13, 1996, 369-386.

[6] Castillo, J. E., J.S. Otto. A Practical Guide to Direct Optimization for Planar Grid-Generation. Computers and Mathematics with Applications 37, 1999, 123-156.

[7] Castillo, J.E. A discrete variational grid generation method, SIAM Journal on Scientific and Statistical Computing, 12, 1991, 454-468.

[8] Egidi, N., P. Maponi. A linearly constrained optimization problem for planar grid generation, Applied Numerical Mathematics 55, 2005, 283–294.

[9] Press W H, Flannery B P, Teukolsky S A and Vetterling W T. Numerical Recipes in C: the Art of Scientific Computations. Cambridge University Press 1992

[10] Depeng Y., Gregory D.,Husheng L., High Performance Reconfigurable Computing for Cholesky Decomposition, Symposium on Application Accelerators in High Performance Computing (SAAHPC), UIUC, July, 2009

ABOUT THE AUTHORS

Stanislav Dimchev Kostadinov, MSc, PhD student at the Department of Informatics and Information Technologies, University of Ruse, Phone: +359 898 237753, E-mail: sdkostadinov@uni-ruse.bg.

Assoc. Prof. Dr. Tzvetomir Ivanov Vassilev, Department of Informatics and Information Technologies, University of Ruse, Phone: +359 82 888475, E-mail: TVassilev@uni-ruse.bg.

Докладът е рецензиран.