# The Shortest Path Finding between two points on a Polyhedral Surface

Eugene Vladimirovich Popov
Professor of NNSACEU
Ilyinskaya Street 65
603950, Nizhny Novgorod,
Russia
popov@sandy.ru

Tatyana Petrovna Popova
Assosiated Professor of
National Research University
Higher School of Economics,
25/12 Bolshaja Pecherskaja
Ulitsa, 603155, Nizhny Novgorod,
Russia,
tatpop@list.ru

Sergej Igorevich Rotkov
Professor of NNSACEU
Ilyinskaya Street 65
603950, Nizhny Novgorod,
Russia
rotkov@nngasu.ru

## ABSTRACT

The paper describes the approximate method of the shortest path finding between two points on a surface. This problem occurs when generating a cutting pattern after the form of the fabric tensile surface is found. The shortest path finding is reduced to the problem of finding the geodesic line on the surface. However, the numerical problem solution of the form finding of fabric tensile structure leads to the fact that the final surface is represented by an arbitrary polyhedron. There is no analytical problem solution of finding shortest paths in this case. The described method allows finding the shortest path on a surface of any regular polyhedron form.

**Keywords** Tensile fabric structures, geodesic line, cutting pattern, shortest paths, polyhedral surface

## 1. INTRODUCTION

Over the past few decades, there has been a rapid growth in the use of fabric tension structures. The fabric tension or fabric tensile structures are architecturally innovative forms of construction art that have double curved shapes, and are aesthetically pleasing. However, the design of a fabric tension structure is a very complex task. Three steps are usually required in the design process of tension structure, namely, form finding, load analysis and cutting pattern generation. This paper is dedicated to the third design step, i.e. cutting pattern generation [1], [2].

Cutting pattern generation problem for tensile fabric structure can be generally defined as follows. It involves finding the strips that will have the minimum area difference with the sum area of a number of plane strips. Here, the seam line between strips is determined by the width of the membrane material. However, the reference configuration for cutting pattern generation of membrane structures is the geometric information attained from form finding, and it is represented by nodal coordinates of three-dimensional discontinuous points. [3]

When cutting pattern generation is conducted on the basis of a connected line, the seam line of the strip reconstructed on a plane will have large curvature, and the quantity of membrane material consumption will also increase. For this reason, the economic method of generating cutting pattern is to use geodesic line in the process of patterning. Besides, cloths must be cut out from fabric rolls of relatively narrow width. For economic reasons it is desirable that each cloth should maximize the use of the available width. The use of geodesic seam lines is therefore particularly appropriate in almost all cases. It is sometimes economically advantageous for cloths to be patterned with one straight side. Seam lengths of adjacent cloths should be the same, and cloth distortion at structure borders must be avoided.

The geodesic problem is the problem of finding the shortest distance between two points on arbitrary surfaces. In addition, geodesic generates directions on the surface as well. The shortest distance can be found by generalizing the equation for the length of a curve, and then by minimizing this length using the calculus of variations. This has some minor technical problems, because there are different ways to parameterize shortest paths. While the case of finding the shortest path between two points on a plane is a straight line, the shortest path between two points over a surface will be geodesic. In many of the existing methods for making this generalization, such as using isometric maps to surfaces with known geodesics, one of the most useful methods is finding analytical solution by using the Euler equation. A geodesic can be represented by the solution of a second-order ordinary differential equation [4]

$$u''v' - v''u' + Av' - Bu' = 0, \qquad (1)$$

where

$(u, v)$ – parametric co-ordinate of a surface point;

$\left.\begin{aligned} u &= u(t) \\ v &= v(t) \end{aligned}\right\}$, - a curve equation near a given point.

Many researchers have studied the problem of finding the shortest path between two points on the surface. One of the interesting ways of doing this is presented in [5]. The authors of the work focus their attention on the problem of computing geodesics on smooth surfaces. First, the authors assume they are given an approximate path to start from when attempting to compute a geodesic between two points. Then they attempt to compute the geodesic between two points iteratively using the midpoints of an approximate path between them. Further the authors explore a similar method, gradient descent, to iteratively update the path approximating the geodesic.

## 2. THE SHORTEST PATH ON A SURFACE

In most cases the analytic solution of the equation (1) is impossible because of the lack of an analytic surface representation. However, it is well known that the geodesic line between two arbitrary points close enough to each other on a depressed shell is the shortest line between those points. Therefore, let us try to develop a more or less general approach for finding the shortest path between two points without equation (1) solution based on the mentioned fact. Assume that there is a straight line between two points ***1*** and ***2*** on an arbitrary surface (see Fig.1)

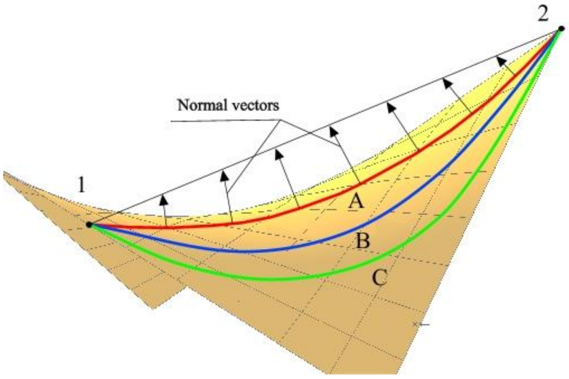


**Figure 1. Straight-line normal projection**

Further let us form line ◡***1A2*** on the surface according to the following rule: the normal vector to the surface at each point of line ◡***1A2*** intersects the straight line ***12*** (see Fig.1). It means that each point of line ◡***1A2*** is the normal projection of the corresponding point of the straight line ***12*** onto the surface. On this basis, we can enunciate the following theorem:

***Theorem***

If each point of a curved line between two points on a surface is formed by the orthogonal projection of the corresponding point of a straight line between those two points along normal vector to a surface, then, this curved line is the shortest path between two points.

***Proof***

Let us suppose there are two points ***1*** and ***2*** on a surface close enough to each other as it is shown in Fig.2. Let us bind the points by the straight line ***12***. Assume that this line has dimensionless parameter ***t*** (see Fig.2). Let us further form line ◡***1A2*** on the surface according to the rule illustrated on Fig.1. Then we can write the following equations for an arbitrary point ***p*** on the straight line and point ***k*** relative to ***p*** on line ◡***1A2***

$$\mathbf{r}_k(t) = \mathbf{r}_p(t) + \mathbf{r}_{pk}(t) = \mathbf{r}_p(t) - d(t)\cdot\mathbf{n}(t), \qquad (2)$$

where $\mathbf{r}_p(t)$ – radius-vector to the point ***p***;

$\mathbf{r}_k(t)$ – radius-vector to the point ***k***;

$\mathbf{r}_{pk}(t)$ –vector between the points ***p*** and ***k***;

$\mathbf{n}(t)$ – normal vector to the surface at the point ***k***;

$d(t)$ – distance between the points ***p*** and ***k***;

Now let us assume there is another line ◡***1B2*** on the surface between points ***1*** and ***2***. The equation for point ***f*** on this line relative to the given point ***p*** can have the following form

$$\mathbf{r}_f(t) = \mathbf{r}_p(t) + \mathbf{r}_{pf}(t) = \mathbf{r}_p(t) + \mathbf{r}_{pk}(t) + \mathbf{r}_{kf}(t), \qquad (3)$$

where $\mathbf{r}_{pf}(t)$ –vector between the points ***p*** and ***f***;

Note that obviously vector $\mathbf{r}_{pf}(t)$ is always longer than vector $\mathbf{r}_{pk}(t)$ because the last is the shortest distance between point ***p*** and the surface. Moreover, if we take into consideration that the set of vectors similar to vector $\mathbf{r}_{pk}(t)$ form the ruled surface bounded by lines ***12*** and ◡***1A2*** in total one can notice that this surface is a surface of minimum area. It can be shown in the following way. One can calculate the area of the surface bounded by lines ***12*** and $\cup$ ***1A2*** by the following obvious expression

$$S_{1A21} = L_{12}\int_0^1 \mathbf{r}_{pk}(t)dt, \qquad (4)$$

where $L_{12}$ - the length of the straight line ***12***.

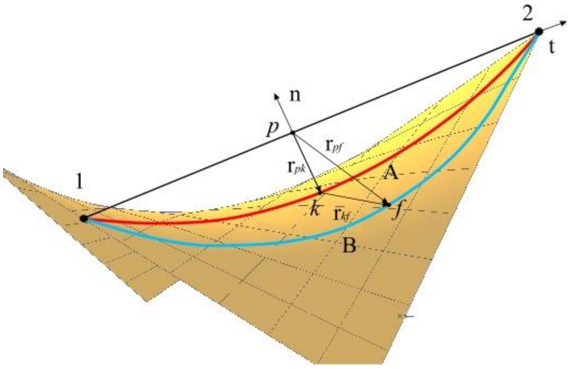


**Figure 2. To the Theorem proof**

On the other hand, the area of the surface bounded by line ***12*** and an arbitrary line $\cup$ ***1B2*** is equal to

$$S_{1B21} = L_{12}\int_0^1 \mathbf{r}_{pf}(t)dt. \qquad (5)$$

Since inequality $\mathbf{r}_{pk}(t) < \mathbf{r}_{pf}$ is always, true inequality $S_{1A21} < S_{1B21}$ is always true as well. Hence, the surface bounded by lines ***12*** and $\cup$ ***1A2*** is the minimum surface.

In its turn the length of line $\cup$ ***1A2*** can be expressed by the following integral

$$L_{1A2} = \int_0^1 \left|\dot{\mathbf{r}}_k(t)\right| dt, \qquad (6)$$

where derivation is done on the parameter $t$. Similarly, the length of line $\cup$ ***1B2*** is equal to

$$L_{1B2} = \int_0^1 \left|\dot{\mathbf{r}}_f(t)\right| dt. \qquad (7)$$

If we transform the expressions (6) and (7) taking into account equations (2) and (3)

$$\left|\dot{\mathbf{r}}_k(t)\right| = \left|\dot{\mathbf{r}}_p(t) + \dot{\mathbf{r}}_{pk}(t)\right|$$

$$\left|\dot{\mathbf{r}}_f(t)\right| = \left|\dot{\mathbf{r}}_p(t) + \dot{\mathbf{r}}_{pf}(t)\right|$$

we can indicate that if $\mathbf{r}_{pk}(t) < \mathbf{r}_{pf}$ it means that $\left|\dot{\mathbf{r}}_{pk}(t)\right|$ cannot exceed $\left|\dot{\mathbf{r}}_{pf}(t)\right|$ as well because of the derivation on the same parameter $t$. Moreover, the derivative $\left|\dot{\mathbf{r}}_{pk}(t)\right|$ can only always be less than $\left|\dot{\mathbf{r}}_{pf}(t)\right|$.

It means that $L_{1B2} > L_{1A2}$ and $\cup$ ***1A2*** is the shortest path between two points ***1*** and ***2*** on the surface because line $\cup$ ***1B2*** was arbitrary.

That establishes the Theorem.

## 3. THE SHORTEST PATH ON A POLYHEDRON

Unfortunately, the majority of numerical methods of fabric tensile structures form finding allow obtaining a surface only approximately. Usually a surface is represented by a facet model as it is shown in Fig. 3 where the triangle mesh of surface is shown. A surface given by a triangular mesh is not, in general, differentiable at triangle vertices or at points on the triangle edges. At these points, a curve on the surface is not differentiable, therefore its curvature is undefined and form (1) is not applicable in this case.

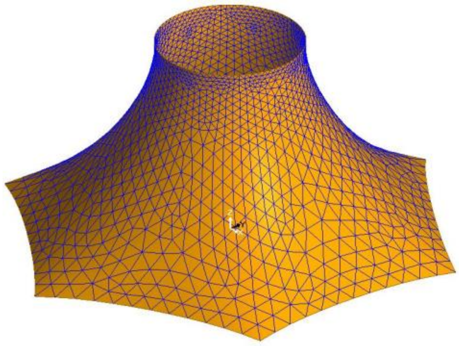


**Figure 3. Typical triangle mesh of a fabric structure**

The problem of finding the shortest path between two points lying on the surface of a polyhedron is a basic problem in computational geometry and is studied for instance in [6], [7], [8], [9]. Especially [8] presents a method for building a subdivision of the surface which can be used for finding shortest paths from a fixed source to a given query point efficiently.

Several forms of the problem solution can be defined when we change the properties of the polyhedron (e.g. considering faces to have weights, being non-convex etc.) or constrain the path with different restrictions. An example of the constrained versions is the problem of finding the shortest path, which does not go above some given height as studied in [10].

In work [11] the authors developed the algorithm that used a technique based on the continuous Dijskstra method. This simulates the continuous propagation of a wave front of points equidistant from the starting point across the surface, updating the wave front at discrete events.

In work [12] the authors describe algorithms to compute edge sequences, the shortest path map, and the Fréchet distance for a convex polyhedral surface. The length of a Euclidean shortest path measures distances on the surface. Their approach uses persistent trees, star unfoldings, and kinetic Voronoi diagrams. An implementation of the exact "single source, all destination" algorithm presented by Mitchell, Mount, and Papadimitriou (MMP) was described in [13]. The authors extend the algorithm with a merging operation to obtain computationally efficient and accurate approximations with bounded error. To compute the shortest path between two given points, they use a lower-bound property of their approximate geodesic algorithm to efficiently prune the frontier of the MMP algorithm, thereby obtaining an exact solution even more quickly. An efficient O(n) (where n is the number of points on the surface) numerical algorithm for first-order approximation of geodesic distances on geometry images is presented in [14]. The structure of this algorithm allows efficient implementation on parallel architectures.

In [15] a new algorithm for detecting self-collisions on highly discretized moving polygonal surfaces is presented. It is based on geometrical shape regularity properties that permit avoiding many useless collision tests. Nevertheless, it should be recognized that nowadays there is no universal method of finding the shortest path on the smooth surface represented by a polyhedron.

Here we present the algorithm for finding the approximate shortest path on (convex and non-convex) polyhedral surface based on a straight-line normal projection onto a surface. The algorithm computes a pseudo-geodesic line between every pair of points.

Let ***P*** be a (non-convex in general case) polyhedron in 3D space. We consider ***P*** to be specified by a set of faces, edges and vertices. Without loss of generality, we assume that all faces are triangles. We are given two special points on the surface, namely ***1*** and ***2***. The problem is to find the shortest path $\cup$***12*** between ***1*** and ***2*** lying on ***P*** (see Fig.4). As soon as the surface is represented by a faced model we can find the length of a path in $R^3$ by approximating it with piecewise linear path.

To find the shortest path on a polyhedral surface we will use the Theorem from the previous Section. However, the main problem in this case arises at once, e.g. a polyhedral surface has no continuous normal vector. Therefore, it is necessary to define the approximate normal vectors at vertices and edges.

To resolve this problem we present a simple Phong normal interpolation (see work [16]) that is used in computer graphics to shade polygonal models and create an appearance of a smooth surface. The basic principle behind the method is as follows: The estimation of the surface normal of each vertex in a 3D model is found by averaging the surface normal vectors of polygons, which meet at each vertex. Let us define some notation and formally describe Phong normal interpolation.

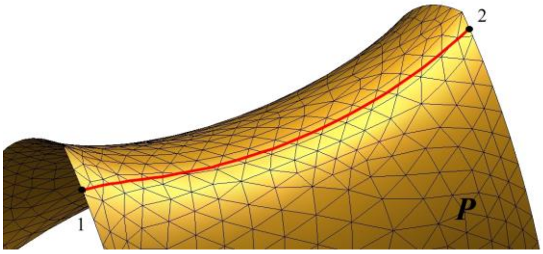


**Figure 4. The line between two points on a polyhedron surface**

Phong normal interpolation conceptually requires the following steps (see Fig.5):

1. For each vertex, compute the vertex normal vector. This normal is often computed by averaging the adjacent face normal vectors, but the algorithm does not have any dependence on how the normal is computed. Suppose that equation of plane polygonal faces are given, and then a normal to their common vertex can be defined by sum value of the normal vector to all polygons joining at this vertex.

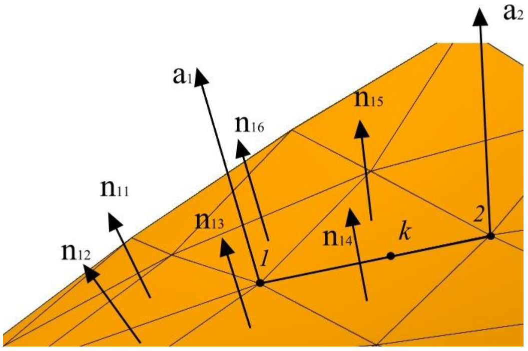


**Figure 5. Phong normal interpolation**

For example, in Fig.5 the approximate direction of the normal at point ***1*** is equal to

$$\mathbf{a}_1 = \mathbf{n}_{11} + \mathbf{n}_{12} + \mathbf{n}_{13} + \mathbf{n}_{14} + \mathbf{n}_{15} + \mathbf{n}_{16}. \quad (8)$$

where $\mathbf{n}_{11}; \mathbf{n}_{12}; \mathbf{n}_{13}; \mathbf{n}_{14}; \mathbf{n}_{15}; \mathbf{n}_{16}$ - normal vectors at vertex ***1*** of adjacent planes.

2. Normalize this vector to obtain the unit-length normal as follows

$$\mathbf{n}_1 = \frac{\mathbf{a}_1}{|\mathbf{a}_1|}. \tag{9}$$

3. Linearly interpolate the vertex normal for any point along the edges by the following equation

$$\begin{aligned}\mathbf{n}_k &= (\mathbf{n}_2 - \mathbf{n}_1)\cdot t_{1k} + \mathbf{n}_1 = \\ &= \Delta\mathbf{n}\cdot t_{1k} + \mathbf{n}_1.\end{aligned} \tag{10}$$

where $t_{1k}$ - dimensionless parameter of ***12*** edge (see Fig.5) at point ***k*** (see Fig.5).

Another problem is to find the set of points belonging to the shortest path and to the polyhedral surface at the same time. We will find such points as the normal projection of the initial straight line between points ***1*** and ***2*** to the polyhedral edges according to the scheme described below.

Let ***P*** be the polyhedron mentioned in Fig.4. Firstly, we define the straight line between points ***1*** and ***2*** (see Fig.6) and apply dimensionless parameter $t_1$ to it that runs from 0 at point ***1*** to 1.0 at point ***2***.

Thereby the Cartesian co-ordinates of an arbitrary point on line ***12*** can be calculated by the following system

$$\begin{aligned}X &= (X_2 - X_1)\cdot t_1 + X_1 = \Delta X\cdot t_1 + X_1;\\ Y &= (Y_2 - Y_1)\cdot t_1 + Y_1 = \Delta Y\cdot t_1 + Y_1;\\ Z &= (Z_2 - Z_1)\cdot t_1 + Z_1 = \Delta Z\cdot t_1 + Z_1;\end{aligned} \tag{11}$$

where $X_1; X_2; Y_1; Y_2; Z_1; Z_2$ - the Cartesian co-ordinates of points ***1*** and ***2***.

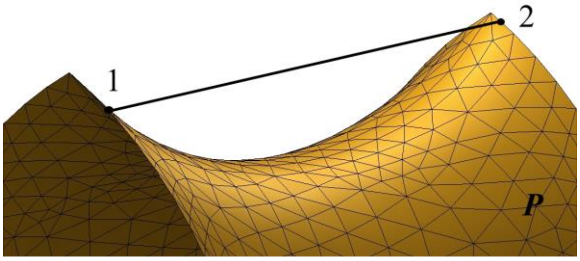


**Figure 6. Initial straight line between two points**

The next step is to sort out the set of all model edges with the aim to obtain the point on the edge, which corresponds to the point on the straight line. Let us consider an arbitrary edge ***1'2'*** of the polyhedron (see Fig.7). We apply dimensionless parameter $t_2$ to it that also runs from 0 at point ***1'*** to 1.0 at point ***2'***.

Assume that we have already obtained the desirable point ***k'*** on the edge ***1'2'*** (see Fig.7). Further, we can define new co-ordinate axis $\xi$ collinear to the normal vector $\mathbf{n}$ calculated by form (10) (see also Fig.7). Obviously, axis $\xi$ intersects straight line ***12*** at point ***k*** due to the proved Theorem because point ***k'*** is a normal projection of point ***k*** to edge ***1'2'***. Then we can calculate Cartesian co-ordinates of point ***k*** using the following expression

$$\begin{aligned}\mathbf{x}_k &= [(\mathbf{x}_2 - \mathbf{x}_1)\cdot t_2 + \mathbf{x}_1] + \xi\cdot[(\mathbf{n}_2 - \mathbf{n}_1)\cdot t_2 + \mathbf{n}_1] = \\ &= [\xi\cdot(\mathbf{n}_2 - \mathbf{n}_1) + (\mathbf{x}_2 - \mathbf{x}_1)t_2] + \xi\cdot\mathbf{n}_1 + \mathbf{x}_1 = \\ &= [\xi_k\cdot\Delta\mathbf{n} + \Delta\mathbf{x}]\cdot t_{2k} + \xi_k\cdot\mathbf{n}_1 + \mathbf{x}_1.\end{aligned} \tag{12}$$

where $\mathbf{x}_1; \mathbf{x}_2$ - the Cartesian co-ordinates of points ***1'*** and ***2'***;

$t_{2k}$ - the $t_2$ parameter value at point ***k'***;

$\xi_k$ – the value of $\xi$ parameter at point ***k***.

If parameter $t_1 = t_{1k}$ one can indicate the following obvious equality

$$\mathbf{x}_k = \mathbf{X}_k, \tag{13}$$

where vector $\mathbf{X}_k$ is the Cartesian co-ordinates of point ***k*** calculated by form (11).

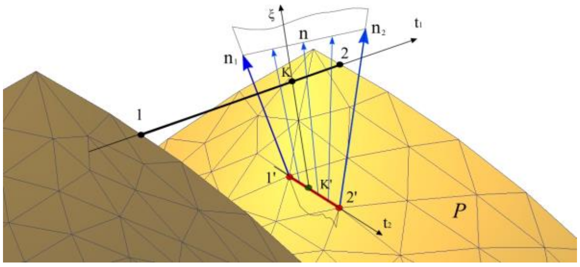


**Figure 7. Normal projection of point *k* to the edge**

Equating the left-hand parts of expressions (11) and (12) we can obtain the following expression

$$[\xi_k\cdot\Delta\mathbf{n} + \Delta\mathbf{x}]\cdot t_{1k} + \xi_k\cdot\mathbf{n}_1 + \mathbf{x}_1 = \Delta\mathbf{X}\cdot t_{2k} + \mathbf{X}_1, \tag{14}$$

where $\xi_k$ $t_{1k}$ and $t_{2k}$ are the values of respective parameters at points ***k*** and ***k'***.

Further, we can transform expn (14) to the form presented below that can be used for finding the triad of parameters $\xi_k$ $t_{1k}$ $t_{2k}$

$$\Delta n_x\cdot\xi_k\cdot t_{1k} + \Delta x\cdot t_{1k} + n_{1x}\cdot\xi_k - \Delta X\cdot t_{2k} = X_1 - x_1. \tag{15}$$

In practice, expn (15) has the form of a basic non-linear system of algebraic equations (see expn (16)) and can be resolved by any appropriate method.

$$\begin{cases}\Delta x\cdot t_{1k}+n_{1x}\cdot\xi_k-\Delta X\cdot t_{2k}=(X_1-x_1)-\Delta n_x\cdot\xi_k\cdot t_{1k};\\ \Delta y\cdot t_{1k}+n_{1y}\cdot\xi_k-\Delta Y\cdot t_{2k}=(Y_1-y_1)-\Delta n_y\cdot\xi_k\cdot t_{1k};\\ \Delta z\cdot t_{1k}+n_{1z}\cdot\xi_k-\Delta Z\cdot t_{2k}=(Z_1-z_1)-\Delta n_z\cdot\xi_k\cdot t_{1k}.\end{cases}\tag{16}$$

The solution of system (16) is used for calculating the Cartesian co-ordinates of point ***k'*** that is the point of normal projection of point ***k*** onto the edge ***1'2'***

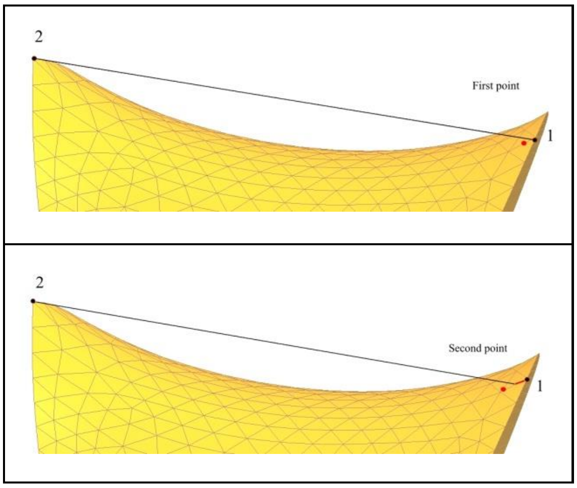
$$\begin{aligned}x&=(x_2-x_1)\cdot t_{2k}+x_1;\\ y&=(y_2-y_1)\cdot t_{2k}\ +y_1;\\ z&=(z_2-z_1)\cdot t_{2k}+z_1.\end{aligned}\tag{17}$$

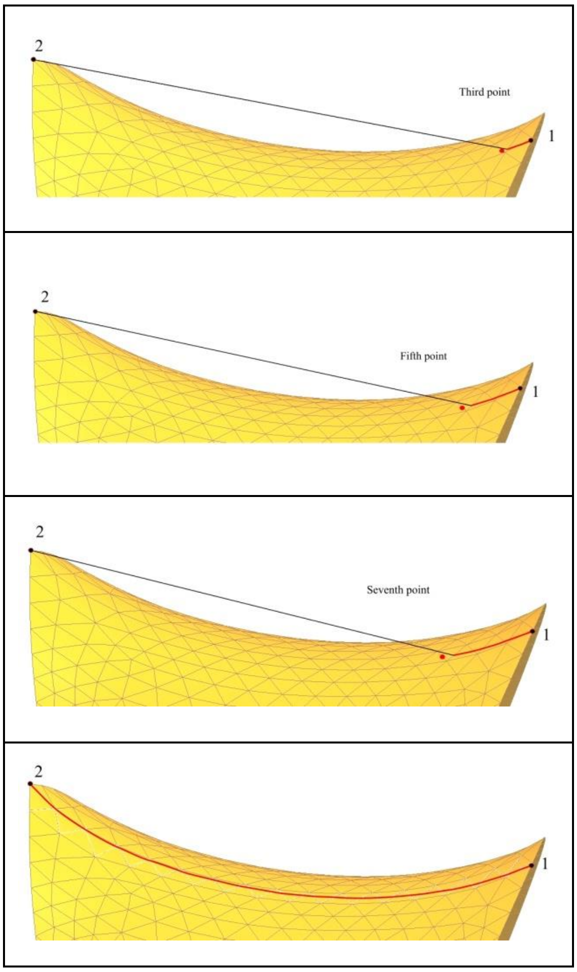
## 4. "SLIDING LASER RAY" METHOD

It stands to reason that the described approach is most efficient in application to depressive surfaces with weak curvature. The critical criterion of the approach applicability is the ratio of lines ***12*** and ***kk'*** length is more or equal to 5 (e.g. ***d*$_{12}$ / *d*$_{kk'}$** >≈ 5). It means that near ending points ***1*** and ***2*** the shortest path nodes are found with the best precision. But nodes near the middle of the shortest path in some cases may not be found at all because of high curvature deflection. However, when designing real structures the problems in tracing the shortest path may occur very often. Therefore, the less the surface curvature is, the more precise the shortest path is.

**Table 1**

**The illustration of "Sliding Laser Ray" method**





To avoid any problems related to large surface curvature a new method was developed in this work. It was called "***Sliding Laser Ray***" method due to its specifics. The main idea of this method consists in the mobility of one of the initial straight line ending point. The illustration of the method is presented in Table 1. Initially we should find the point on the polyhedral surface that is the nearest point to ending point ***1*** (the *First point* in Table 1) by the method described in the previous Section. Further, we shift the starting point of the straight line (point ***1*** in Table 1) to the *First point* and find the *Second point* (see Table 1) that is nearest to the *First point*. Then we find the *Third point* that is nearest to the *Second point* and do the same each time while the nearest point exists. The process stops when the nearest point is ***2*** of the initial straight line. Finally, all the found points are joined by the piecewise polyline that is the desirable shortest path on the polyhedral surface.

One can indicate that the process is very similar to laser ray sliding from point ***2*** to the point on the surface, tracing the shortest path on it. As it is shown in the following Sections, the "Sliding Laser Ray" method is very efficient and works very well for very complex polyhedral surfaces with large curvature.

## 5. THE GENERAL ALGORITHM

To sum up we can formulate the general algorithm of the shortest path finding on a polyhedral surface by taking the following steps:

*Step 1.* Define two ending points ***1*** and ***2*** on a surface and join them by a straight line.

*Step 2.* Define the set of polyhedron edges.

*Step 3.* Select the first edge from the set of edges and resolve system (14) in the first approximation with $\Delta\mathbf{n}\cdot\xi_k\cdot t_{1k}=0$. If $t_{1k}$ or $t_{2k}$ do not satisfy, the inequalities $0\le t_{1k}\le 1$ or $0\le t_{2k}\le 1$ we repeat *Step 3* for the next edge.

*Step 4*. Add the values of $\Delta\mathbf{n}\cdot\xi_k\cdot t_{1k}$ calculated with the previously defined parameters $\xi_k$ $t_{1k}$ and $t_{2k}$ to the right-hand side of the system (14) and resolve it once again. Do *Step 4* as many times as necessary to make parameters $\xi_k$ $t_{1k}$ and $t_{2k}$ invariable according to some predefined tolerance. Calculate by expn (15) the Cartesian co-ordinates of the found point and add it to the intermediate set of points. If the set of edges is not exhausted, repeat *Step 3* for the next edge. **Note:** The current edge should be deleted from the set of edges.

*Step 5*. Sort the intermediate set of the found points by bubble sorting and select the nearest one to the first straight line point. Add this point to the set of the shortest line points. Shift the first straight line point to this point and repeat the process from *Step 3* if the set of edges is not empty. If the set of edges is empty then do *Step 6*.

*Step 6*. Trace the shortest path by piecewise polyline .

## 6. THE ALGORITHM APPLICATION

We already know how to efficiently handle the problem of the shortest path finding. We will now see how the developed algorithm can be checked. In order to test this algorithm the simplest and the best known sample is used. The sample concerns the shortest path on a spherical surface.

The shortest path on a spherical surface problem was well studied by Leonard Euler in the XVIII century. As it was written by him in [17] "On a spherical surface, on which it is not possible to draw straight lines, it has been established by the geometers that the shortest path between two given points is the [shorter arc of the] great circle joining them." Therefore, we should compare the numerically developed great circle on a spherical surface with exact great circle to be convinced that our algorithm works well. In Fig. 8 the result of testing in the form of two arcs of the great circle namely ∪ ***1A2*** (red line) and ∪ ***1A*** (blue line) is shown. The spherical semi-surface was created by the Stretched grid method described in work [18]. The red arc ∪ ***1A2*** is the exact arc of the great circle. The blue arc ∪ ***1A*** is the line traced by the algorithm described in this work. Another thin red line crossing two previous lines plays a supplementary role.

The test has shown that both arcs ∪ ***1A2*** and ∪ ***1A*** between points ***1*** and ***A*** are quite identical. The relative residual here between lengths of two lines does not exceed 0.5%.

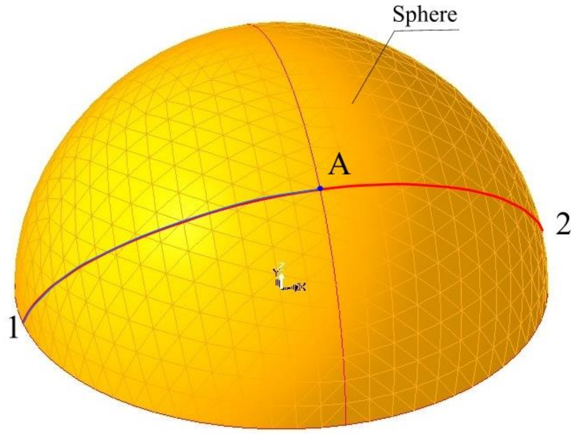


**Figure 8. The arc of great circle of sphere**

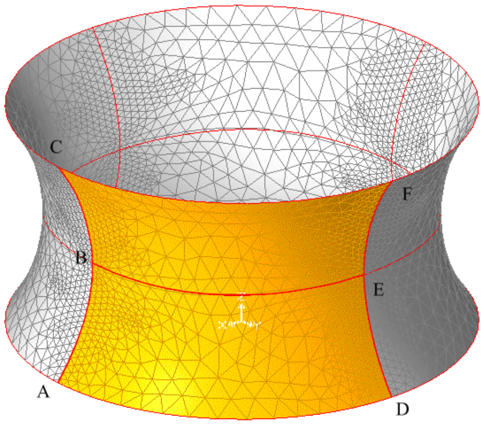


**Figure 9. The shortest paths on cathenoid**

The next sample is of the shortest lines on the cathenoidal surface. The solution is given for the case with two rings of radii equal to 1 and the distance equal to 1 between them. The length of two arcs ∪ ***ABC*** and ∪ ***BE*** (see Fig.9) were compared with their analytic values. The numerical length of arc ∪ ***ABC*** is 1.085081 (analytic value 1.087601) and arc ∪ ***BE*** 1.303446 (analytic value 1,332569). The tolerance between numerical and analytic arc lengths here is also acceptable. It is between 0.3% - 2.1%.

The running CPU (AMD Phenom II N 930 Quad-Core Processor) time was 1.1 sec to compute 40 intermediate points for $\smile \boldsymbol{BE}$ arc. It is acceptable in the majority of cases.

Another test was made on the basis of the surface of real fabric structure. Two arbitrary shortest paths have been traced between points ***1 – 2*** and ***3 – 4*** respectively (see Fig. 10). The test has shown that the developed algorithm allows tracing the shortest lines of very complex form.

It should be noted that the shortest path between two points sometimes can be ambiguous. For example, in Fig. 11 we can see the shortest path between points ***B*** and ***E*** that has the so called ***'branch point'***. The path has two equipollent branches in section between ***branch point*** and point ***E***. However, we call one of them ***Real branch*** and the other one – ***Imaginary branch***.

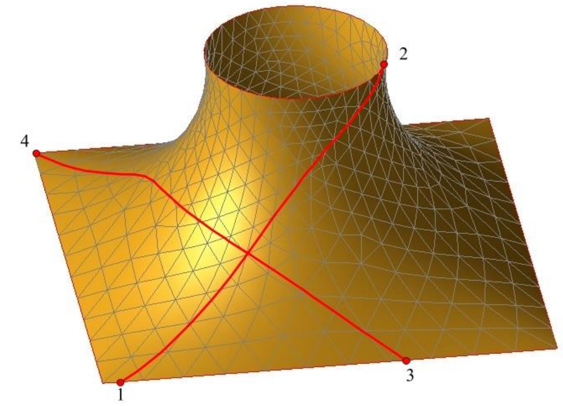


**Figure 10. The testing shortest paths on cone tent**

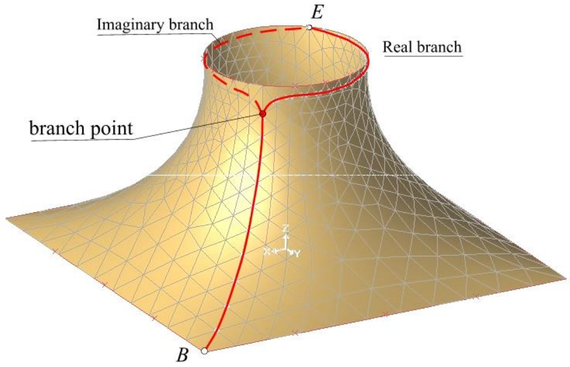


**Figure 11. The shortest path with two branches**

When a ***branch point*** is detected, the algorithm finds the next two points of both branches, selects one of them and traces the whole ***Real branch*** of the shortest path. The ***Imaginary branch*** is ignored by the algorithm. The ***branch point*** detection is made by topology specifics of triangle mesh. If all three edges of a triangle have only two points that belong to the shortest path crossing this triangle then there is no ***branch point*** inside this triangle and vice versa. It is obvious that the choice of the ***Real branch*** in this case is largely random.

Cutting out the cloth surface is the basic and most important stage of tensile structures design. It is a process of the cloth surface subdivision into separate patches and further unfolding them onto a plane to prepare a pattern.

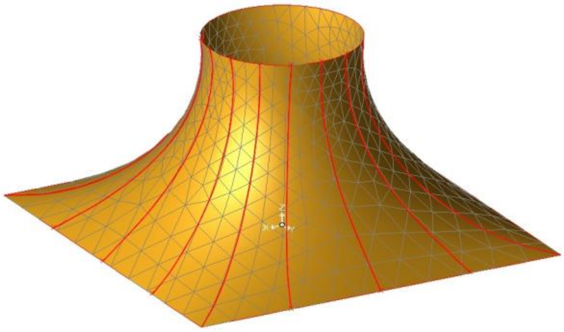


**Figure 12. Cutting pattern of cone tent**

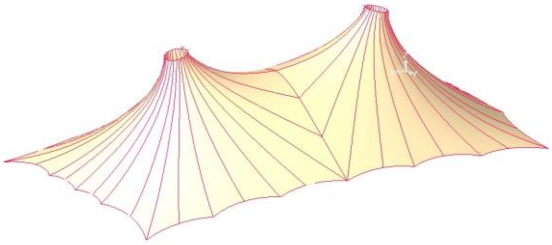


**Figure 13. Cutting pattern of twin-peak tent**

The designer traces the mark and cutting lines (prospective seams) on the modeled 3D tent surface. There are various algorithms of tracing lines on the cloth surface including the algorithm of tracing the shortest line between two arbitrary points described in this work.

As an example, in Fig. 12, the cutting pattern of a typical cone tent made by the set of shortest lines is presented and in Fig. 13, the cutting pattern of the so-called twin-peak tent is shown.

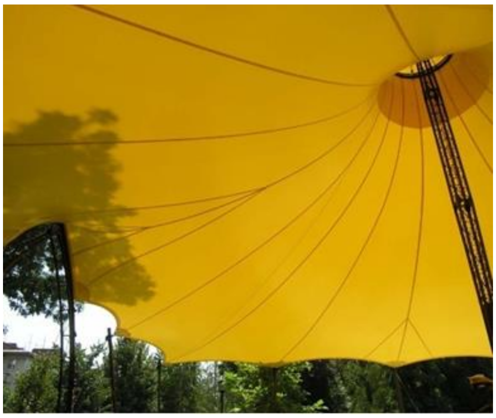


**Figure 14. The seams on the cone tent**

In Fig. 14 one can see the seams on the surface of the cone tent traced by the algorithm described in this paper. The outer view of the cone tent is represented in Fig. 15.

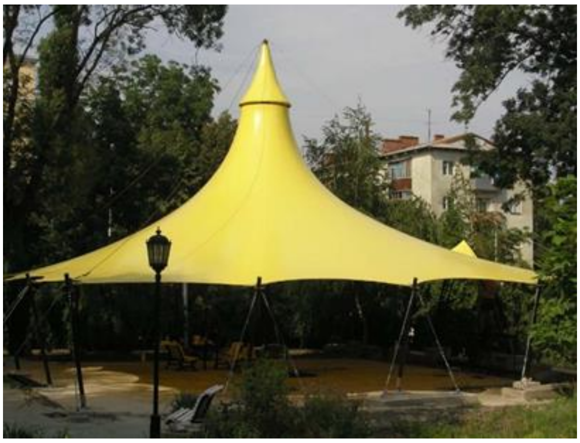


**Figure 15. The outer view of the cone tent**

## 7. CONCLUSION

In this paper the following contributions have been made:

1. The concept of straight line normal projection on a surface has been introduced and the theorem concerning the shortest distance between two points on a surface has been proved.
2. The algorithm of the point projection onto edges of triangular meshes where the vertices are equipped with a normal has been proposed.
3. The algorithm has been introduced for computing the shortest path along a manifold polyhedral surface based on a triangular mesh.
4. The *"Sliding Laser Ray"* method has been also proposed to avoid any problems concerning large surface curvature.
5. It has been shown that using the described approach to cut out architectural membrane structures automatically, the resulting program tool is very convenient, powerful and flexible. It is also applicable to other membrane design fields as diverse as clothing and sails.

The flexible line generation capability of geodesic seam lines is extremely comprehensive, and capable of dealing with problems of much greater complexity than conventional architectural membranes. It should be accentuated that the geodesic lines generation in the form of shortest paths on polyhedral surfaces is also in great demand in shipbuilding.

## 8. REFERENCES

[1] Lothar Gründig, Erik Moncrieff, Peter Singer, Dieter Ströbel. Hign Performance Cutting Pattern Generation of Architectural Textile Structures, IASS-IACM 2000 Fourth International Colloquium on Computation of Shell & Spatial Structures June 5-7, 2000.

[2] Popov E.V. Geometric Approach to Chebyshev Net Generation Along an Arbitrary Surface Represented by NURBS, Proceedings of the 12th Int. Conference on Computer Graphics & Vision GRAPHICON'2002, UNN, Nizhny Novgorod, 2002.

[3] SHON Su-dcok, LEE Scung-jac, LEE Kang-guk. Smooth cutting pattern generation technique for membrane structures using geodesic line on subplane and spline interpolation. J. Cent. South Univ. 20: 3131-3141 DOI: I0.1007/S11771-013-1836-9 Springer, 2013.

[4] Nathaphon Boonnam, Pakkinee Chitsaku. The shortest path between two points on some surface by using the application of Euler equation. Proceedings of the 6th IMT-GT Conference on Mathematics, Statistics and its Applications (ICMSA2010) Universiti Tunku Abdul Rahman, Kuala Lumpur, Malaysia, 2010.

[5] Jongmin Baek, Anand Deopurkar, Katherine Redfield. Finding Geodetics on Surface, 18.821: Project Lab in Mathematics, Massachusets Institute of Technology, 2007.

[6]. M. Sharir and A. Schorr. Shortest Paths in Polyhedral Spaces. *SIAM J. Comput*. 15:193-215, 1986.

[7] J. S. H. Mitchell. 1). M. Mount and C. H. Papadimitriou, The Discrete Geodesic Problem, *SIAM J. Comput*. 16:647-668, 1987.

[8] J. Chen and Y. Han. Shortest Paths on a Polyhedron. *Iternat. J. Compu/. Geotn. AppL*, 6:127-144, 1996.

[9] S. Kapoor, Efficient Computation of Geodesic Shortest Paths, In *Proc. 32nd Annu. ACM Sympos. Theory Comput.*, 1999.

[10] M. de Berg and M. van Kreveld, Trekking in the Alps Without Freezing or Getting Tired, *Algorithmica*, 18:306-323, 1997.

[11] J. O'Rourke, Computational Geometry Column 35, SIGACT News, 30(2) Issue #111 (1999) 31-32.

[12] Atlas F. Cook IV, Carola Wenk. Shortest Path Problems on a Polyhedral Surface, Algorithmica (2014), May 2014, Volume 69, Issue 1, pp 58-77

[13] Surazhsky V, Surazhsky T, Kirsanov D, Gortler S, Hoppe H. Fast exact and approximate geodesics on meshes, ACM Transactions on Graphics (TOG) – Proceedings of ACM

SIGGRAPH 2005 Volume 24 Issue 3, July 2005 pp. 553-560

[14] Ofir Weber, Yohai S. Devir, Alexander M. Bronstein, Michael M. Bronstein, and Ron Kimmel, Parallel algorithms for approximation of distance maps on parametric surfaces, ACM Transactions on Graphics, 27 (2008).

[15] Pascal Volino, Nadia Magnenat Thalmann. Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity. Computer Graphics Forum Volume 13, Issue 3, 1994, pp. 155–166.

[16] Phong, B.-T. Illumination for Computer Generated Pictures. In Communications of the ACM, vol. 18 (6), 1975.

[17] L. Euler. Concerning the shortest line on any surface by which any two points can be joined together. Comm: Ac.Scient.Petr.Tom.III p.110; Nov. 1728.

[18] Popov, E.V. Geometrical Modeling of Tent Fabric Structures with the Stretched Grid Method, Proceedings of the 11th International Conference on Computer Graphics & Vision GRAPHICON'2001, UNN, Nizhny Novgorod, 2001.