FITTING A POINT CLOUD TO A 3D POLYHEDRAL SURFACE

Eugene Vladimirovich Popov[a,*], Serge Igorevich Rotkov[a]
[a] Engineering Geometry and Computer Graphics Chair
Nizhegorodsky State Architectural and Civil Engineering University,
603950, 65, Ilyinskaya Street, Nizhny Novgorod, Russia
popov_eugene@list.ru

Commission II, WG II/10

KEYWORDS: Contactless measurement, point clouds fitting, Stretched Grid method, Principal Component Analysis

ABSTRACT

The ability to measure parameters of large-scale objects in a contactless fashion has a tremendous potential in a number of industrial applications. However, this problem is usually associated with an ambiguous task to compare two data sets specified in two different co-ordinate systems. This paper deals with the study of fitting a set of unorganized points to a polyhedral surface. The developed approach uses Principal Component Analysis (PCA) and Stretched grid method (SGM) to substitute a non-linear problem solution with several linear steps. The squared distance (SD) is a general criterion to control the process of convergence of a set of points to a target surface. The described numerical experiment concerns the remote measurement of a large-scale aerial in the form of a frame with a parabolic shape. The experiment shows that the fitting process of a point cloud to a target surface converges in several linear steps. The method is applicable to the geometry remote measurement of large-scale objects in a contactless fashion.

1. INTRODUCTION

The geometry measurement of large-scale objects in any industry is a very acute problem. It reduces to the comparison of a 3D point set given by remote measurement to a continuous theoretical surface. We can classify it as a point-to-surface (PTS) problem. Usually one can treat such comparison as superposition that requires not less than three reference points. However, reference points can be either unknown or meaningless for some classes of product. In this case, the problem comes to a comparison of two geometric objects in 3D space according to a given criterion of optimality. That is, the unknown parameters of the 3D transformation such as translation and rotation are a subject to be found according to objects optimal matching.

All the algorithms of two 3D sets comparison can be classified in the following way:

1. ICP- algorithm (iterative closest point algorithm). The basis of ICP-algorithm is the assumption that two objects have common area where they coincide well enough. It means that in the common area, for both of them, each point of one object has a corresponding point of another object. Vaillant and Glaunes (Vaillant at al., 2005.) described the basics of the ICP-algorithm. Though some disadvantages of ICP-algorithm take place:

- the computational complexity of the closest points finding is $O(mN_1 N_2)$ where m - number of iterations, N_1- number of the first object points, N_2 - number of the second object points (Friedman , at al. 1977);

- strong dependence on the given initial approximation;

- strong dependence on the density of point clouds;

- the method requires the existence of a large overlap region where the points of one cloud correspond to the points of another cloud.

Recently, many variants of the original ICP approach have been proposed, such as:

- Brunnstrom and Stoddart (Brunnstrom, at al. 1996) describe the genetic algorithm of finding the most successful initial approximation which is input data to ICP-algorithm;

- the research of Dyshkant (Dyshkant, 2010) is also dedicated to the ICP-algorithm modification based on the k-d trees, which allows minimization of the computational complexity to $O(mN_1 \log N_2)$;

- in works of Liu, Li, and Wang (Y. Liu, at al. 2004) algorithms to improve the accuracy and reliability of the ICP-algorithm by imposing certain restrictions of the input data are proposed.

2. Methods based on curvature maps.

This class of methods requires knowledge of the curvature of the surface given by the point cloud. The algorithm was described by Gatzke at al. (Gatzke, at al., 2005). The disadvantage of this method is a strong dependence on the point cloud density because it affects the accuracy of the curvature calculation.

3. Other methods.

The authors of (Bergevin, at al. 1995) describe the algorithm that does not require the approach of initial data. This algorithm can use a free-form surface; however, it has a very low speed.

The authors of work (Sitnik, at al. 2002) improved the method of the steepest descent optimization. The disadvantage of the approach is the quadratic computational complexity.

Delaunay triangulation algorithm in combination with Nelder-Mead method are described in works (Dyshkant, 2010) and (Nelder, at al.1965). The algorithm assumes that the surfaces are single-valued. This algorithm as well as ICP-algorithm depends on the given initial approximation.

The authors of (Gruen, at al. 2005) proposed the algorithm based on the least square method. The algorithm requirement is that the point clouds have a significant overlap area.

In work (Popov, at al. 2013) the algorithm based on step by step geometric transformation of the point cloud is formulated. The disadvantage of this algorithm is the lack of mathematical rigorousness.

* Corresponding author

Nowadays there are two trends in the surface superpose problem solution. The first group of methods limits the initial data therefore they work fast. The second group is more general but has a large computational complexity. Hence, we need more algorithms for the comparison of the two data sets.

2. INITIAL BACKGROUND

The initial assumption is that there are two sets: $P{:}P_i(x_i,y_i,z_i)$ – the cloud of source points obtained by measuring and $\bar{P}{:}\bar{P}_i(x_i,y_i,z_i)$ – the cloud of target points (see Fig. 1). We should fit point cloud P to a target point cloud $\bar{P}$.

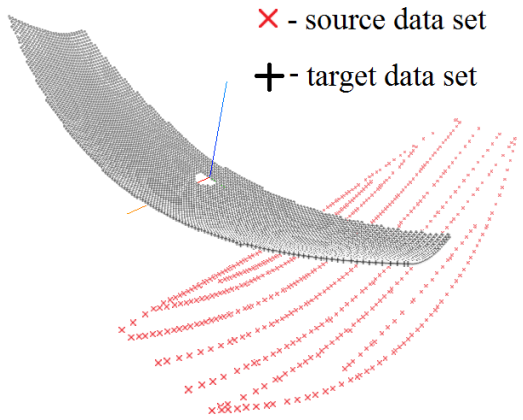

Figure 1: Two data sets

However, we cannot make it directly because of the lack of definite and understandable base points. Besides, both data sets have different structure so there are no corresponding points to compare. Therefore, we triangulate a target cloud and turn it into a continuous 3D polyhedral surface Σ (see Fig.2) specified on the same bounded domain $D \subset R^2$ as a target data set. It is required to find such Ω_{opt} transformation amongst all possible Ω 3D transformations so that the source set $\Omega_{opt}(P)$ could be in closest state to the surface Σ according to the given distance function ρ. That is

$$\sum_{i=1}^{N} \rho\left(\Sigma, \Omega_{opt}(P_i)\right) = \min_{\Omega} \sum_{i=1}^{N} \rho\left(\Sigma, \Omega(P_i)\right) \qquad (1)$$

where $\rho(\Sigma,X)$ - the distance function from point X to surface Σ.

Once the two sets are specified with respect to two different origins we need such transformation of one of them as 'rigid body' so that to ensure the satisfaction of the eqn (1). Such 3D transformation is defined by six parameters: the components of the translation vector $\Delta x_c, \Delta y_c, \Delta z_c$ (here C is the geometry center of relocatable set) and three rotation angles φ_x, φ_y, φ_z,.

The numerical solution of this non-linear problem by the usual optimization approaches is very complicated for various reasons, namely:

- In general, it is difficult to fit two sets even approximately. Hence, it is impossible to find the initial values of $\Delta x_c, \Delta y_c, \Delta z_c$, φ_x, φ_y, φ_z. It forces us to take them with maximum values that makes the computing process slow down.
- The surface cannot be simply single coherent that increases the number of constraints in the optimization problem.
- Often the surface does not have analytical representation, so its derivatives are unknown or do not exist. That makes it impossible to use efficient numerical algorithms based on the function derivatives.
- The computation time depends on value N (the number of points in P set). Therefore, the computing process becomes very slow when the dense of the point set grows significantly.

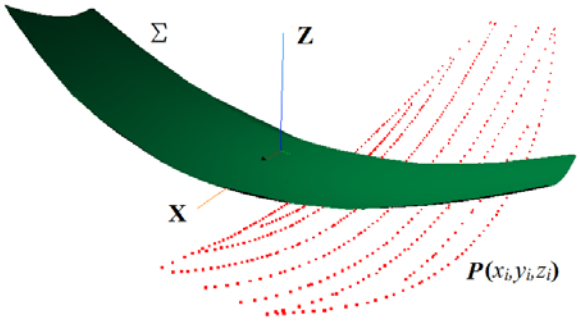

Figure 2: The source data set and the 3D surface

Taking it into account we propose a new approach that consists of two stages and the first of them is Principal Component Analysis (PCA). We apply PCA to the so-called 'rough fit' that actually is the approximate initial fit of two sets. The second stage is the precise fit based on Stretched grid method (SGM) that allows accurate fitting of two sets according to minimum SD criterion in 1-4 linear steps. We demonstrate this approach with the help of the parabolic aerial where Σ –analytic aerial surface, P – source point cloud obtained by measuring the aerial with the standard electronic tacheometer «Trimble-M3» (Survey of Trimble M3 Mechanical Total Station). Thus, the target data set was obtained on the basis of available documentation.

3. ROUGH FIT

It should be noted, that PCA is often used to map data on a new orthonormal basis in the direction of the largest variance (Draper, at al., 2002). The largest eigenvector of the covariance matrix always points to the direction of the largest variance of the data.

In our case, the first data set is the point cloud and the second is the continuous surface, therefore, we should represent the surface by another point cloud as well. The further procedure follows the scheme described in work (Bellekens, at al. 2014). If the covariance matrix of two point clouds differs from the identity matrix, a rough fit can be obtained by simply aligning the eigenvectors of their covariance matrices. This alignment is obtained in the following way: at the first step the two point clouds are centered in such a way that the origins of their final bases coincide. The centering of the point cloud simply corresponds to subtracting the centroid coordinates from each of the point coordinates. The centroid of the point cloud corresponds to the average coordinate and is thus, obtained by dividing the sum of all point-coordinates by the number of points in the point cloud. Since the rough fit based on PCA simply aligns the directions in which the point clouds vary the most, the second step consists of calculating the covariance matrix of each point cloud. The covariance matrix is an orthogonal 3 × 3 matrix, the diagonal values of which represent the variances while the off-diagonal values represent the covariance. As the third step, the eigenvectors of both

covariance matrices are calculated. The largest eigenvector is a vector in the direction of the largest variance of the 3D point cloud, and therefore, represents the point cloud's rotation. Further, let A be the covariance matrix, let v be an eigenvector of this matrix, and let λ be the corresponding eigenvalue. The problem of eigenvalues decomposition is then defined as

$$Ax = \lambda x, \tag{2}$$

and further reduces to

$$x(A - \lambda I) = 0. \tag{3}$$

It is clear that (3) only has a non-zero solution if $A - \lambda I$ is singular, and consequently if its determinant equals to zero

$$det(A - \lambda I) = 0. \tag{4}$$

The eigenvalues can simply be obtained by solving (4), whereas the corresponding eigenvectors are obtained by substituting the eigenvalues into (2). Once the eigenvectors are known for each point cloud, the fit is achieved by aligning these vectors. Then, let us assume that matrix T_Σ represents the transformation that would align the largest eigenvector of the target point cloud related to surface Σ with the *X*-axis. Now, let us suppose that matrix T_P represents the transformation that would align the largest eigenvector of the source point cloud *P* with the *X*-axis as well. Finally, we can align the source point cloud with the target point cloud easily if we take into account coincidence of both principal component systems (X_{pr}, Y_{pr}, Z_{pr}) of source and target point clouds (see Fig.3).

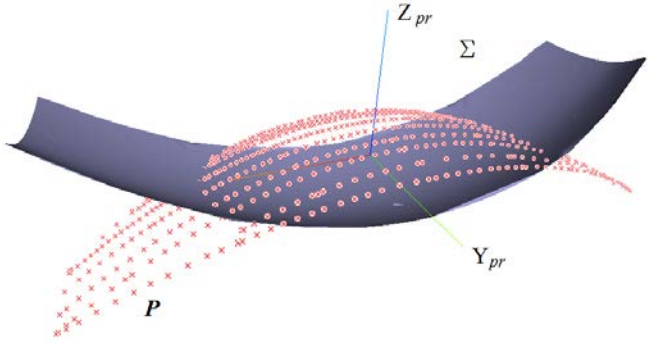

Figure 3: Two data sets in common principal component system

Here we face some disadvantage, as we cannot always determine the direction of collinear principal component axes uniquely with PCA (see Fig.3). Therefore, we correct their directions in this issue manually by rotating the source point cloud about axes X_{pr}, Y_{pr}, Z_{pr} consequently to meet the minimum of SD criterion. In our sample, we rotate the point cloud about X_{pr} (see Fig. 4.)

The rough fit cannot obtain real minimum solution according to the SD criterion; therefore, the next stage is the precise fit.

4. PRECISE FIT

The precise fit stage is based on SGM. SGM described in work (Popov E.V., 1997) is a numerical technique for finding approximate solutions of various mathematical and engineering problems that can be related to an elastic grid behavior. In our case, we apply SGM to drag in the source point cloud as a 'rigid body' to the target surface by the set of elastic springs (Fig.5).

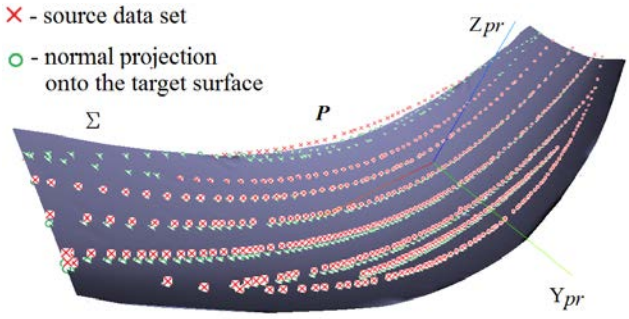

Figure 4: The rough fit of two data sets in common principal system

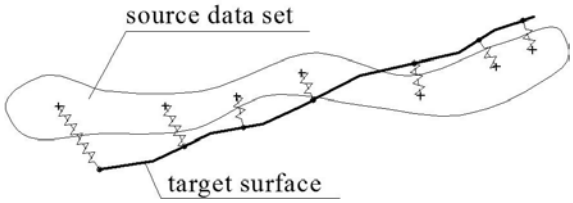

Figure 5. The precise fit scheme

Each elastic spring for our cloud connects the nearest neighbor point on the target surface q_i of each point p_i in the source point cloud (see Fig. 5). We find the neighbor point on the target surface by normal projection of the source point onto the target surface. This approach is similar to ICP point-to-point technique described in (Ben Bellekens, at al. 2014) but is much easier and has another physical meaning.

The aim of the precise fit is to find functions Δx_c,Δy_c,Δz_c, φ_x, φ_y, φ_z that obtain the minimum to exp (1). If we apply classical motion equation, we should further resolve non-linear equation system consisted of transcendental functions. The advantage is that we can consider linear dependence of displacement of points on the cloud rotation as a rigid body (see Fig. 6).

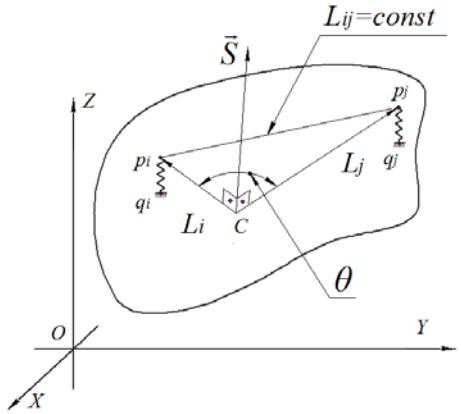

Figure 6: To rigid body rotation and transformation

Taking into consideration the 'rigid body' rotation of point cloud due to precise fit, we can write the displacement of an arbitrary point p_i (Fig.6) as follows

$$\begin{Bmatrix} \Delta x_i \\ \Delta y_i \\ \Delta z_i \end{Bmatrix} = \begin{bmatrix} B_{11}^{(i)} & B_{12}^{(i)} & B_{13}^{(i)} \\ B_{21}^{(i)} & B_{22}^{(i)} & B_{23}^{(i)} \\ B_{31}^{(i)} & B_{32}^{(i)} & B_{33}^{(i)} \end{bmatrix} \begin{Bmatrix} \Delta x_j \\ \Delta y_j \\ \Delta z_j \end{Bmatrix} + \begin{Bmatrix} \Delta x_c \\ \Delta y_c \\ \Delta z_c \end{Bmatrix}, \tag{5}$$

where $\Delta x_i,\Delta y_i,\Delta z_i$ – displacements of an arbitrary point i;

$\Delta x_j,\Delta y_j,\Delta z_j$ – displacements of point j;

$\Delta x_c,\Delta y_c,\Delta z_c$ - displacements of the point cloud centroid.

We can calculate the components of the normalized matrix B for an arbitrary point of the cloud as a rotation matrix about the unit vector $\vec{s}(u,v,w)$ at the angle θ (see Fig. 6) by the following expressions

$$\begin{aligned}
B_{11}^{(i)} &= (u^2 + (1-u^2)\cos\theta)\cdot F,\\
B_{12}^{(i)} &= (uv(1-\cos\theta) - w\sin\theta)\cdot F,\\
B_{13}^{(i)} &= (uw(1-\cos\theta) + v\sin\theta)\cdot F,\\
B_{21}^{(i)} &= (uv(1-\cos\theta) + w\sin\theta)\cdot F,\\
B_{22}^{(i)} &= (v^2 + (1-v^2)\cos\theta)\cdot F,\\
B_{23}^{(i)} &= (vw(1-\cos\theta) - u\sin\theta)\cdot F,\\
B_{31}^{(i)} &= (uw(1-\cos\theta) - v\sin\theta)\cdot F,\\
B_{32}^{(i)} &= (vw(1-\cos\theta) + u\sin\theta)\cdot F,\\
B_{33}^{(i)} &= (w^2 + (1-w^2)\cos\theta)\cdot F.
\end{aligned} \tag{6}$$

Here $F = L_i / L_j$, where L_i, L_j – vectors to points i and j respectively from the point cloud centroid.

Due to exp (5) we can calculate the displacement of each point in the point cloud if we know the displacement of single point number j only.

The further step is to write the expression for the potential energy of entire connecting lines between the cloud points and the springs including (Fig. 4) that takes the following form

$$\Pi = D\sum_{m=1}^{n} R_m^2, \tag{7}$$

here n - total number of springs,

R_m - the length of spring number m,

D - an arbitrary constant (D = 1 in our case).

Then, let us assume that co-ordinate vector $\{X\}$ of all the points of the cloud is associated with a final cloud position, when the source cloud is fit to the target surface and the vector $\{X\}'$ is associated with the initial point cloud position. Thus, vector $\{X\}$ will look in the following way

$$\{X\} = \{X\}' + \{\Delta X\}, \tag{8}$$

where $\{\Delta X\}$ - vector of the co-ordinate increment of entire points.

To determine vector $\{\Delta X\}$ we should derive function (7) by incrementing vector $\{\Delta X\}$ with form (8) taken into account, i.e.

$$\frac{\partial \Pi}{\partial \Delta X_{kt}} = 0, \tag{9}$$

where k – number of the current point,

t - number of the current co-ordinate.

After transformations using exps (5), (7), (8), (9) and keeping all lengths L_{ij} constant (see Fig.6) we can obtain the following linear equation system 6×6

$$K\cdot\Delta x = Q, \tag{10}$$

where vector Δx has only 6 unknown components to be found, namely Δx_j, Δy_j, Δz_j, Δx_c, Δy_c, Δz_c;

K – the matrix of solution with the following components

$$K_{11} = \sum_{i=1}^{n}\left\{\left(B_{11}^{(i)}\right)^2 + \left(B_{21}^{(i)}\right)^2 + \left(B_{31}^{(i)}\right)^2\right\},$$

$$K_{12} = \sum_{i=1}^{n}\left\{B_{11}^{(i)}\cdot B_{21}^{(i)} + B_{21}^{(i)}\cdot B_{22}^{(i)} + B_{31}^{(i)}\cdot B_{32}^{(i)}\right\},$$

$$K_{13} = \sum_{i=1}^{n}\left\{B_{11}^{(i)}\cdot B_{13}^{(i)} + B_{21}^{(i)}\cdot B_{23}^{(i)} + B_{31}^{(i)}\cdot B_{33}^{(i)}\right\},$$

$$K_{14} = \sum_{i=1}^{n} B_{11}^{(i)}, \quad K_{15} = \sum_{i=1}^{n} B_{21}^{(i)}, \quad K_{16} = \sum_{i=1}^{n} B_{31}^{(i)}$$

$$K_{21} = \sum_{i=1}^{n}\left\{B_{11}^{(i)}\cdot B_{12}^{(i)} + B_{21}^{(i)}\cdot B_{22}^{(i)} + B_{31}^{(i)}\cdot B_{32}^{(i)}\right\},$$

$$K_{22} = \sum_{i=1}^{n}\left\{\left(B_{12}^{(i)}\right)^2 + \left(B_{22}^{(i)}\right)^2 + \left(B_{32}^{(i)}\right)^2\right\},$$

$$K_{23} = \sum_{i=1}^{n}\left\{B_{12}^{(i)}\cdot B_{13}^{(i)} + B_{22}^{(i)}\cdot B_{23}^{(i)} + B_{32}^{(i)}\cdot B_{33}^{(i)}\right\},$$

$$K_{24} = \sum_{i=1}^{n} B_{12}^{(i)}, \quad K_{25} = \sum_{i=1}^{n} B_{22}^{(i)}, \quad K_{26} = \sum_{i=1}^{n} B_{32}^{(i)},$$

$$K_{31} = \sum_{i=1}^{n}\left\{B_{11}^{(i)}\cdot B_{13}^{(i)} + B_{21}^{(i)}\cdot B_{23}^{(i)} + B_{31}^{(i)}\cdot B_{32}^{(i)}\right\},$$

$$K_{32} = \sum_{i=1}^{n}\left\{B_{12}^{(i)}\cdot B_{13}^{(i)} + B_{22}^{(i)}\cdot B_{23}^{(i)} + B_{32}^{(i)}\cdot B_{33}^{(i)}\right\},$$

$$K_{33} = \sum_{i=1}^{n}\left\{\left(B_{13}^{(i)}\right)^2 + \left(B_{23}^{(i)}\right)^2 + \left(B_{33}^{(i)}\right)^2\right\},$$

$$K_{34} = \sum_{i=1}^{n} B_{13}^{(i)}, \quad K_{35} = \sum_{i=1}^{n} B_{23}^{(i)}, \quad K_{36} = \sum_{i=1}^{n} B_{33}^{(i)}$$

$$K_{41} = \sum_{i=1}^{n} B_{11}^{(i)}, \quad K_{42} = \sum_{i=1}^{n} B_{12}^{(i)}, \quad K_{43} = \sum_{i=1}^{n} B_{13}^{(i)},$$
$$K_{44} = n, \quad K_{45} = 0, \quad K_{46} = 0,$$

$$K_{51} = \sum_{i=1}^{n} B_{21}^{(i)}, \quad K_{52} = \sum_{i=1}^{n} B_{22}^{(i)}, \quad K_{53} = \sum_{i=1}^{n} B_{23}^{(i)},$$
$$K_{55} = n, \quad K_{54} = 0, \quad K_{56} = 0,$$

$$K_{61} = \sum_{i=1}^{n} B_{31}^{(i)}, \quad K_{62} = \sum_{i=1}^{n} B_{32}^{(i)}, \quad K_{63} = \sum_{i=1}^{n} B_{33}^{(i)},$$
$$K_{66} = n, \quad K_{64} = 0, \quad K_{65} = 0;$$

Q – the right hand vector with the following components

$$Q_1 = \sum_{i=1}^{n}\left\{B_{11}^{(i)}(x_j - x_i) + B_{21}^{(i)}(y_j - y_i) + B_{11}^{(i)}(z_j - z_i)\right\},$$

$$Q_2 = \sum_{i=1}^{n}\left\{B_{12}^{(i)}(x_j - x_i) + B_{22}^{(i)}(y_j - y_i) + B_{32}^{(i)}(z_j - z_i)\right\},$$

$$Q_3 = \sum_{i=1}^{n} \left\{ B_{13}^{(i)}(x_j - x_i) + B_{23}^{(i)}(y_j - y_i) + B_{33}^{(i)}(z_j - z_i) \right\},$$

$$Q_4 = x_j, \; Q_5 = y_j, \; Q_6 = z_j,$$

When all six unknown functions Δx_j, Δy_j, Δz_j, Δx_c, Δy_c, Δz_c are found we can calculate displacement of each source point due to precise fit using exp (5).

5. "SURFACE FITTING" PROGRAM

We developed the program named "Surface Fitting" based on the above described algorithm (Popov, 2016). To make the program extremely accessible and mobile we developed it as a web-based open source application using JavaScript language (Flanagan, 2011) in couple with THREE.JS library (Dirksen, 2013). As it is known, JavaScript is an object-oriented language designed in 1995 to allow non-programmers to extend web sites with client-side executable code. The language is also becoming a general-purpose computing platform with office applications, browsers and development environments being developed in JavaScript. Unlike other traditional languages such as Java, C++ or C#, JavaScript strives to maximize programming flexibility and is ideal for programming accessible applications including portable measuring systems. THREE.JS is a high-level, scene graph framework for 3D graphics built on top of WebGL. THREE.JS programs are written in JavaScript because there is no alternative for WebGL. The main idea when creating the program was to provide the user with a simple, accessible tool that depends neither on the hardware nor on the software platform. In Fig.7 one can see the user interface of the "Surface Fitting" program.

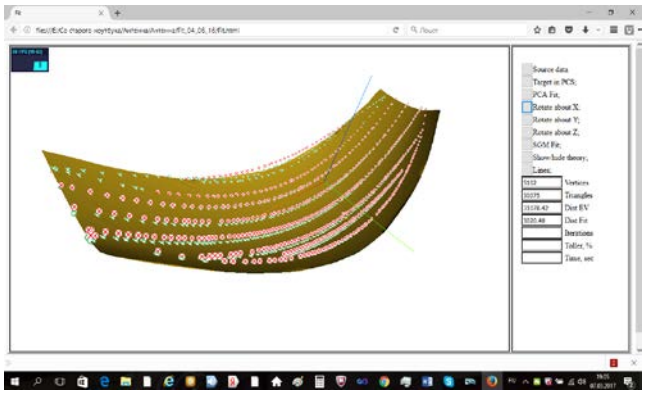

Figure 7: User interface of "Surface Fitting" program.

Using "Surface Fitting" program we can calculate the displacements of entire source points to fit it to the target surface and visualize the whole scene. Input data in the form of source and target point clouds are loaded into the program automatically. The process involves four consecutive steps

- target cloud triangulation,
- rough fit,
- source cloud rotation about principal axes manually at π-angle if necessary
- and the final precise fit.

In spite of linear nature of the precise fit, the process needs some iterations to converge because of some disparity of two sets after rough fit. Table 1 shows the matching error of SD against the number of iterations.

Iterations	Relative Error of SD,%	Time, sec
2	In spite of linear nature	4.64
8	1.5257	18.30
17	0.0984	46.72
20	0.0107	55.00
29	0.0064	85.55

Table 1: The process convergence

The final fit of two sets can be seen in Fig.8.

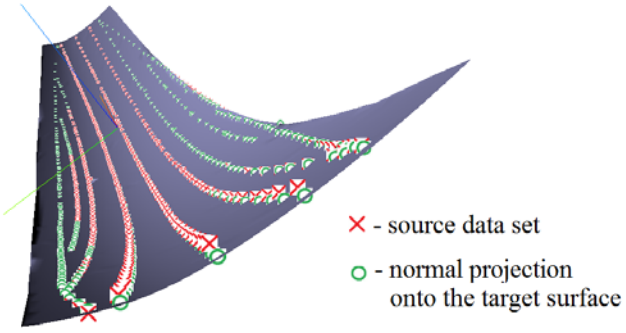

Figure 8: The final fit of two data sets.

As we can see, the process meets minimum SD criterion very quickly. The final error (about 0.01%) means that the fit precision is about 1-2 mm for the aerial with about 15m of overall dimension.

6. CONCLUSION

This paper has introduced a new two-stage method based on rough fit with PCA and precise fit by SGM for direct point cloud and polyhedral surface fit in 3D. The algorithm has been designed for accurate distance minimizing between source and target data sets. It differs from techniques based on the ICP-algorithm and it differs from any Least Square approaches due to clear physical meaning. It does not depend on the configuration of data sets and their connectivity. The method consists of several linear steps and converges very fast. By vary the residual criterion, we can achieve any level of the fitting relative error.

The "Surface Fitting" allows calculation of both data sets at the initial and fitting position. It also provides the user with comprehensive and detailed tools of the whole scene visualization.

The only temporary inconvenience in using this method for 3D data sets comparison is the ambiguity of transformations with a rough fitting when applying PCA. We are currently resolving this problem by simple manual correction consequentially rotating the source data set at π-angle about principal axes to obtain local SD minimum. In the future, we plan to make it automatic. Besides, we intend to improve PCA algorithm by making it less sensitive to the data set singularities.

Finally, it should be noted that one could successfully apply algorithms and "Surface Fitting" program described in the paper in healthcare, where a soft-body model often needs to

be aligned accurately with a set of 3D measurements. Such applications are cancer-tissue detections, hole detection, dental occlusion modelling, artefact recognition, etc.

7. ACKNOWLEDGMENTS

Russian Foundation for Basic Research has supported this work under grant # 15-07-01962 and grant # 15-07-05110.

REFERENCES

Vaillant M., Glaunes J., 2005. Surface matching via currents. *Lecture Notes in Computer Science: Information Processing in Medical Imaging* — Vol. 3565. - Pp. 1-5

Friedman J., Bentley J.L., Finkel R.A., 1977. An Algorithm for Finding Best Matches in Logarithmic Expect.ed Time. *ACM Transactions on Mathematical Software*. - Vol.3, no.3.-pp. 209-226

Brunnstrom K., Stoddart A. J., 1996. Genetic algorithms for free-form surface matching Proc. *ICPR*. - pp. 689-693.

Dyshkant N., 2010 An algorithm for calculation the similarity measures of surfaces represented as point clouds. *Pattern Recognition and Image Analysis: Advances in Mathematical Theory and Applications* -Vol.20, no.4-Pp. 495-504.

Y. Liu, L. Li, and Y. Wang, 2004. Free Form Shape Matching Using Deterministic Annealing and Softassign, in *17th International Conference on Pattern Recognition*, Cambridge, United Kingdom, August 23-26.

Y. Liu, L. Li, and B. Wei, 2004. 3D Shape Matching Using Collinearity of Constraint, in *IEEE International Conference on Robotics and Automation*, New Orleans, LA, April 26 - May 1.

Gatzke T., Zelinka S., Grimm C., Garland M., 2005. Curvature Maps for Local Shape Comparison. In: *Shape Modeling International*. — Pp. 244-256

Bergevin, R., Laurendeau, D., Poussart D., 1995. Registering Range Views of Multi-Part Objects, Computer Vision and Image Understanding. *IEEE Trans. Pattern Analysis and Machine Intell.*, 61(1):1–16, Jan.

Sitnik R., Kujawinska M., 2002. Creating true 3d-shape representation merging methodologies. *Three-Dimensional Image Capture and Applications V, Proceedings of SPIE* Vol. 4661, pp. 92-99.

Nelder J.A., Mead R., 1965. A simplex method for function minimization. *Computer Journal*. - Vol. 7. Pp. 308-313

Gruen A., Akca D., 2005. Least Squares 3D Surface and Curve Matching. *ISPRS Journal of Photogrammetry and Remote Sensing*. — Vol. 59. — Pp. 151-174.

Popov E.V., Rotkov S.I. , 2013. The Optimal Superpose of a Finite Point Set With a 3D Surface, *Proceedings of the International Conference on Physics and Technology CPT2013*, 12-19 of May 2013, Larnaca, Ciprus

Survey of Trimble M3 Mechanical Total Station, http://www.trimble.com/Survey/trimblem3.aspx

B. Draper, W. Yambor, and J. Beveridge, 2002. Analyzing PCA-based face recognition algorithms: Eigenvector selection and distance measures, *Empirical Evaluation Methods in Computer Vision*, pp. 1–14.

Ben Bellekens, Vincent Spruyt, Rafael Berkvens, and Maarten Weyn, 2014. A Survey of Rigid 3D Point cloud Registration Algorithms, AMBIENT 2014: *The Fourth International Conference on Ambient Computing, Applications, Services and Technologies*, IARIA, pp. 8–13.

Popov E.V., 1997. On Some Variational Formulations for Minimum Surface. *Transactions of Canadian Society of Mechanics for Engineering*, Univ. of Alberta, vol.20, N 4, pp. 391–400.

Popov E.V., 2016. “Surface Fitting”, The State Certificate # 2016661441 of computer program, Federal Service of intellectual property, Russian Federation.

Flanagan, David, 2011. JavaScript: The Definitive Guide (6th ed.). O'Reilly & Associates. ISBN 978-0-596-80552-4.

Dirksen, Jos, 2013. Learning Three.js: The JavaScript 3D Library for WebGL. UK: Packt Publishing. ISBN 9781782166283.