An efficient algorithm for non-rigid object registration

A. Makovetskii [1], S. Voronin [1], V. Kober [1], A. Voronin [1]
[1] Chelyabinsk State University, ul. Bratiev Kashirinykh, 129, 454001, Chelyabinsk, Russia

Abstract

An efficient algorithm for registration of two non-rigid objects based on geometrical transformation of the template object to target object is proposed. The transformation is considered as warping of the template onto the target. To choose the most suitable transformation from all possible warps, a registration algorithm should satisfy deformation constraints referred to as regularization of non-rigid objects. In this work, we use variational functionals for affine transformations. With the help of computer simulation, the proposed method for searching the optimal geometrical transformation is compared with that of common algorithms.

Keywords: iterative closest points, nonrigid ICP, shape registration, affine transformation, orthogonal transformation, point-to-point, point-to-plane, deformable surfaces.

Citation: Makovetskii A, Voronin S, Kober V, Voronin A. An efficient algorithm for non-rigid object registration. Computer Optics 2020; 44(1): 67-73. DOI: 10.18287/2412-6179-CO-586.

Acknowledgments: The work was supported by the Ministry of Education and Science of Russian Federation (grant No. 2.1743.2017) and by the RFBR (grant No. 18-07-00963).

Introduction

Non-rigid registration of two point clouds can be considered as warping of the template cloud onto the target cloud. To find the correct deformation from all possible warps, the proposed registration algorithm imposes a penalty for deformation. We minimize difference between transformations acting on neighboring vertices of the mesh. Basically, the Iterative Closest Point (ICP) method [1, 2] looks for a transformation for all template point clouds. So, ICP based methods compute a preliminary set of correspondences by searching the closest points to template vertices on the target surface, then seek a transformation which aligns the template with the correspondences. The procedure is repeated with new correspondences obtained by searching the closest points to the vertices of the displaced template. ICP methods can be classified by the type of deformation as well as by the way in which these deformations are found. A global rigid transformation was independently proposed [1, 2]. In these methods, each step of the iteration is optimal with respect to fixed correspondences.

The difference between these two methods is as follows: in [2] preliminary correspondences are computed only along the surface normal, while [1] uses the closest points. In contrary, we consider separate geometrical transformations for every point of the template cloud. Originally, this approach was introduced in [3, 4]. Registration is also used for building a morphable model [5]. It can be done by registering the same template onto multiple targets yielding consistent parametrization over all scans. For noiseless and complete data, a correct registration should be one-to-one. In practice, surfaces contain holes and artefacts owing to scanning process. Therefore, a registration method should be robust to outliers and has to fill missing data. Numerous registration algorithms exist; each of them is applicable to different scenarios and has advantages and drawbacks. The proposed approach is different to hole-filling methods [6], which are able to close holes by interpolating border elements. Methods [7, 8] extend the deformation field into regions without correspondences. When registering face parts, which may move independently, like the lower and upper lips, smoothing isotropically throughout the volume [7, 8], unwanted effects of tying together unconnected parts appear. This paper addresses anisotropic regularization, which propagates along the surface [10]. To choose a mapping from all possible registrations, allowable deformations are constrained by regularization of the deformation field. A common approach is to smooth the deformation field by minimizing the squared norm of its gradient, effectively allowing locally smooth translations. A typical deformation of faces is the mixture of rotation and translation; for instance, by lowering the jaw, we look for a regularizer that favors rigid deformations. Curvature based registration [10] minimizes the Laplacian of deformation field, allowing locally smooth affine transformation. Regularization [11] uses locally different affine transformations for surfaces. Our algorithm uses the regularization similar to [9]. We choose spherical subsets of the template volume and determine affine transformation within this region. To get a smooth deformation field, the resulting affine transformations are linearly combined around the sphere centers. Since affine transformations are determined independently per sphere, the resulting deformation is no longer optimal with respect to fixed correspondences. Note, that if affine transformation is defined for a set of vertices [11], then the transformation is uniquely determined by the correspondences. In [9] each vertex has its own unique transformation. The regularizer can be interpreted as a stiffness term. It forces neighboring vertices to undergo similar transformations. The above-mentioned methods are widely used in human face recognition [12 – 19].

The distance term [3, 4] is based on the affine point-to-point approach. The affine point-to-plane approach is described in [20 – 23]. In this paper we propose an algorithm to solve the variational problem [3] using the functional with distance and stiffness terms. Computer simulation results are provided to illustrate the performance of the proposed method.

1. Functional for non-rigid affine point-to-point ICP

Let $P=\{p_1,...,p_s\}$ be a template point cloud, and $Q=\{q_1,...,q_s\}$ be a target point cloud in $\mathbb{R}^3$. Suppose that relationship between points in P and Q is given in such a manner that for each point p_i exists the corresponding point q_i. The ICP algorithm is geometrical transformation for rigid objects mapping P to Q:

$$Rp_i + t, \tag{1}$$

where R is the rotation matrix, t is the translation vector, $i=1,...,s$.

The group of affine transformations in dimension of three has 12 generators. It means that the affine transformation in dimension of three is a function of 12 variables.

The ICP variational problem for affine transformations in the non-rigid point-to-point case is formulated [3, 4]. We consider the same variational problem as in [3] but without the landmark term. Denote by $S(P)$ and $S(Q)$ surfaces constructed from the clouds P and Q, respectively. Let $J(A_1,...,A_s)$ be the following functional:

$$J(A_1,...,A_s) = J_1(A_1,...,A_s) + \lambda J_2(A_1,...,A_s), \tag{2}$$

where

$$J_1(A_1,...,A_s) = \sum_{i=1}^{s} ||Ap_i - q_i||_{L_2}^2 = \\ = \sum_{i=1}^{s} \langle Ap_i - q_i, Ap_i - q_i \rangle, \tag{3}$$

$$J_2(A_1,...,A_s) = \sum_{\{i,j\}\in E} ||A_i - A_j||_{L_2}^2, \tag{4}$$

where $\langle\ ,\ \rangle$ denotes the inner product, A_i is the matrix of affine transformation in the homogenous coordinates:

$$A_i = \begin{pmatrix} a_i^{11} & a_i^{12} & a_i^{13} & t_i^1 \\ a_i^{21} & a_i^{22} & a_i^{23} & t_i^2 \\ a_i^{31} & a_i^{32} & a_i^{33} & t_i^3 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{5}$$

p_i and q_i are points from the cloud P and Q respectively,

$$p_i = (p_i^1, p_i^2, p_i^3, 1)^t,\ q_i = (q_i^1, q_i^2, q_i^3, 1)^t. \tag{6}$$

E is the set of edges of the triangulated surface $S(P)$, λ is the regularization parameter. In the formulas (5), (6) are used superscripts for vector and matrix elements. Also, subscripts for the numbers of vectors or matrices are used.

The non-rigid affine point-to-point ICP variational problem can be stated as follows:

$$(A_{1*},...,A_{s*}) = \arg\min_{A_1,...,A_s} J(A_1,...,A_s). \tag{7}$$

2. Computation of gradient of functional J

Let us compute the gradient of the functional J_1 as

$$\nabla J_1 = ((\nabla J_1)_1,...,(\nabla J_1)_s). \tag{8}$$

Compute the component $(\nabla J_1)_s$ of the functional J_1 gradient as

$$\begin{aligned} &J_1(\ldots, A_k + h, \ldots) - J_1(\ldots, A_k, \ldots) = \\ &= \langle (A_k + h)p_k - q_k, (A_k + h)p_k - q_k \rangle = \\ &= \langle A_k p_k, A_k p_k \rangle + 2\langle A_k p_k, hp_k \rangle + \langle hp_k, hp_k \rangle - \\ &-2\langle A_k p_k, q_k \rangle - 2\langle hp_k, q_k \rangle + \langle q_k, q_k \rangle - \\ &-\langle A_k p_k, A_k p_k \rangle + 2\langle A_k p_k, q_k \rangle - \\ &-\langle q_k, q_k \rangle + \langle hp_k, hp_k \rangle = \\ &= \langle 2A_k p_k, hp_k \rangle - \langle 2q_k, hp_k \rangle + \langle hp_k, hp_k \rangle = \\ &= \langle 2(A_k p_k - q_k), hp_k \rangle + \langle hp_k, hp_k \rangle = \\ &= \langle 2(A_k p_k - q_k)p_k{}^t, h \rangle + o(h), \end{aligned} \tag{9}$$

where h is a small variation of the matrix A_k.

Therefore, the gradient component is defined by the following way:

$$(\nabla J_1)_k = 2(A_k p_k - q_k)p_k{}^t. \tag{10}$$

Compute the component $(\nabla J_2)_k$ of the functional J_2 gradient as

$$\begin{aligned} &J_2(\ldots, A_k + h, \ldots) - J_2(\ldots, A_k, \ldots) = \\ &= \sum_{j=1}^{s_k} \langle (A_k + h) - A_{k_j}, (A_k + h) - A_{k_j} \rangle - \\ &-\sum_{j=1}^{s_k} \langle A_k - A_{k_j}, A_k - A_{k_j} \rangle = \\ &= \sum_{j=1}^{s_k} \langle (A_k + h) - A_{k_j}, (A_k + h) - A_{k_j} \rangle - \\ &-\langle A_k - A_{k_j}, A_k - A_{k_j} \rangle = \\ &= \sum_{j=1}^{s_k} \langle A_k, A_k \rangle + 2\langle A_k, h \rangle + \langle h, h \rangle - \\ &-2\langle A_k, A_{k_j} \rangle - 2\langle h, A_j \rangle + \langle A_{k_j}, A_{k_j} \rangle - \\ &-\langle A_k, A_k \rangle + 2\langle A_k, A_{k_j} \rangle - \langle A_{k_j}, A_{k_j} \rangle = \\ &= \sum_{j=1}^{s_k} \langle 2(A_k - A_{k_j}), h \rangle + \langle h, h \rangle = \\ &= \langle 2\sum_{j=1}^{s_k} (A_k - A_{k_j}), h \rangle + o(h), \end{aligned} \tag{11}$$

where s_k is the number of neighbors of the k-th point of the triangulated surface $S(P)$, k_j is the j-th neighbor of the k-th point of the triangulated surface $S(P)$.

We get the following expression for the component $(\nabla J)_k$ of the functional gradient:

$$\begin{aligned}\left(\nabla J\right)_k &= \left(\nabla J_1\right)_k + \lambda\left(\nabla J_2\right)_k = \\ &= 2\left(A_k p_k - q_k\right) p_k{}^t + 2\lambda\sum\nolimits_{j=1}^{s_k}\left(A_k - A_{k_j}\right).\end{aligned} \quad (12)$$

Denote by P_k and Q_k, $k=1,...,s$ the following matrices:

$$P_k = p_k p_k{}^t,\ Q_k = q_k p_k{}^t. \quad (13)$$

Consider $\left(\nabla J\right)_k = 0$, $k=1,...,s$,

$$A_k P_k - Q_k + \lambda s_k A_k - \lambda\sum\nolimits_{j=1}^{s_k} A_{k_j} = 0. \quad (14)$$

Rewrite (14) as

$$A_k P_k + \lambda s_k A_k - \lambda\sum\nolimits_{j=1}^{s_k} A_{k_j} = Q_k. \quad (15)$$

3. Solution to non-rigid affine point-to-point ICP with linearization

We search the solution $\{A_k\}$, $k=1,...,s$ as a set of linear functions regarding to λ^{-1}. Also we consider for auxiliary purposes the coefficient of λ^{-2}.

$$A_k = A_k^0 + A_k^1\lambda^{-1} + A_k^2\lambda^{-2}. \quad (16)$$

Substitute the expansion of A_k (16) in (15),

$$\begin{aligned}&\left(A_k^0 + A_k^1\lambda^{-1} + A_k^2\lambda^{-2}\right)P_k + \\ &+\lambda s_k\left(A_k^0 + A_k^1\lambda^{-1} + A_k^2\lambda^{-2}\right) - \\ &-\lambda\sum\nolimits_{j=1}^{s_k}\left(A_{k_j}^0 + A_{k_j}^1\lambda^{-1} + A_k^2\lambda^{-2}\right) = Q_k.\end{aligned} \quad (17)$$

We solve (15) by the uncertain coefficients method. Coefficient of λ^1 is

$$s_k A_k^0 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^0 = 0. \quad (18)$$

Coefficient of λ^0 is

$$A_k^0 P_k + s_k A_k^1 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^1 = Q_k. \quad (19)$$

Coefficient of λ^{-1} is

$$A_k^1 P_k + s_k A_k^2 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^2 = 0. \quad (20)$$

It follows from (18) that

$$A_k^0 = \frac{1}{s_k}\sum\nolimits_{j=1}^{s_k} A_{k_j}^0. \quad (21)$$

Suppose that the graph consisting of vertices and edges of the triangulated surface $S(P)$ is connected. Then it follows from (21) that there is a such matrix A^0 that

$$A^0 = A_1^0 = A_2^0 = \ldots = A_s^0. \quad (22)$$

We search the matrix A^0 from (19). Consider the sum of equations in (19), $k=1,...,s$

$$\sum\nolimits_{k=1}^{s}\left(A^0 P_k + s_k A_k^1 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^1\right) = \sum\nolimits_{k=1}^{s} Q_k, \quad (23)$$

$$A^0\sum\nolimits_{k=1}^{s} P_k + \sum\nolimits_{k=1}^{s}\left(s_k A_k^1 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^1\right) = \sum\nolimits_{k=1}^{s} Q_k. \quad (24)$$

Proposition. The following condition holds:

$$\sum\nolimits_{k=1}^{s}\left(s_k A_k^1 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^1\right) = \sum\nolimits_{k=1}^{s} s_k A_k^1 - \sum\nolimits_{k=1}^{s}\sum\nolimits_{j=1}^{s_k} A_{k_j}^1 = 0. \quad (25)$$

From (25) we get that

$$A^0\sum\nolimits_{k=1}^{s} P_k = \sum\nolimits_{k=1}^{s} Q_k, \quad (26)$$

$$A^0 = \left(\sum\nolimits_{k=1}^{s} Q_k\right)\left(\sum\nolimits_{k=1}^{s} P_k\right)^{-1}. \quad (27)$$

Remark. The obtained (27) is the closed form solution of the common affine point-to-point ICP variational problem [24]. Matrix P_k for any fixed k is not invertible, but the sum of all matrices is generally invertible.

Rewrite (19) as

$$s_k A_k^1 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^1 = Q_k - A_k^0 P_k. \quad (28)$$

The expression (28) is the nonhomogeneous system of linear equations. Consider the corresponding homogeneous system,

$$s_k A_k^1 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^1 = 0. \quad (29)$$

Denote by $(A_k^1)_g$ a general solution of the homogeneous system (29), by $(A_k^1)_p$ a partial solution of the nonhomogeneous system (28). Then the general solution A_k^1, $k=1,...,s$, of the nonhomogeneous system (28) is given as

$$A_k^1 = \left(A_k^1\right)_g + \left(A_k^1\right)_p. \quad (30)$$

Note that

$$s_k\left(A_k^1\right)_g - \sum\nolimits_{j=1}^{s_k}\left(A_{k_j}^1\right)_g = 0, \quad (31)$$

$$\left(A_k^1\right)_g = \frac{1}{s_k}\sum\nolimits_{j=1}^{s_k}\left(A_{k_j}^1\right)_g, \quad (32)$$

and

$$\left(A^1\right)_g = \left(A_1^1\right)_g = \left(A_2^1\right)_g = \ldots = \left(A_s^1\right)_g. \quad (33)$$

The partial solution $(A_k^1)_p$ can be obtained with the following iterative process:

$$\begin{cases}\left(A_k^1\right)_p^0 = 0, \\ \left(A_k^1\right)_p^{n+1} = \dfrac{1}{s_k}\left(\sum\nolimits_{j=1}^{s_k}\left(A_k^1\right)_p^n + Q_k - A^0 P_k\right).\end{cases} \quad (34)$$

Remark. From results of our experiments it follows that the iterative process (31) converges to the partial solution $(A_k^1)_p$.

The general solution of the homogeneous system can be found with the help of (20). Consider the sum of equations in (20), $k=1,...,s$

$$\sum\nolimits_{k=1}^{s}\left(A_k^1 P_k + s_k A_k^2 - \sum\nolimits_{j=1}^{s_k} A_{k_j}^2\right) = 0, \quad (35)$$

$$\sum\nolimits_{k=1}^{s} A_k^1 P_k + \sum\nolimits_{k=1}^{s} s_k A_k^2 - \sum\nolimits_{k=1}^{s}\sum\nolimits_{j=1}^{s_k} A_{k_j}^2 = 0, \quad (36)$$

$$\sum_{k=1}^{s} A_k^1 P_k = 0. \quad (37)$$

Substitute (30) to (37)

$$\sum_{k=1}^{s} \left(\left(A_k^1\right)_g + \left(A_k^1\right)_p \right) P_k = 0, \quad (38)$$

$$\sum_{k=1}^{s} \left(A_k^1\right)_g P_k + \sum_{k=1}^{s} \left(A_k^1\right)_p P_k = 0, \quad (39)$$

$$\left(A^1\right)_g \sum_{k=1}^{s} P_k = -\sum_{k=1}^{s} \left(A_k^1\right)_p P_k, \quad (40)$$

$$\left(A^1\right)_g = -\left(\sum_{k=1}^{s} \left(A_k^1\right)_p P_k\right) \left(\sum_{k=1}^{s} P_k\right)^{-1}. \quad (41)$$

So, we get the expression for A_k^1

$$A_k^1 = \left(A_k^1\right)_p - \left(\sum_{k=1}^{s} \left(A_k^1\right)_p P_k\right) \left(\sum_{k=1}^{s} P_k\right)^{-1}. \quad (42)$$

Therefore the linearized representation of A_k can be written as

$$A_k = \left(\sum_{k=1}^{s} Q_k\right) \left(\sum_{k=1}^{s} P_k\right)^{-1} + \left(\left(A_k^1\right)_p - \right.$$
$$\left. - \left(\sum_{k=1}^{s} \left(A_k^1\right)_p P_k \right) \left(\sum_{k=1}^{s} P_k\right)^{-1} \right) \lambda^{-1}. \quad (43)$$

4. Computer simulation

The performance of the proposed method is compared with that of known methods. The method [3] solves variational problem (7) by sequence of matrix transformations. Denote this algorithm as "M-PPt". The most computationally complex part of the M-PPt is searching of the inverse matrix of the size of $4s \times 4s$, where s is the number of points of the cloud P. The computation of the inverse matrix in program realization of the M-PPt is made with the function "PartialPivLU" of the open source linear algebra library "Eigen" for C++. We chose this function because it has the best speed among considered program functions.

The proposed algorithm is referred to as "L-PPt". The algorithms M-PPt and L-PPt solve the same variational problem (7). Both algorithms should ideally return the same results if the values of regularization parameter λ coincide. However, the proposed method is approximation to the exact solution. Therefore, M-PPt ICP and L-PPt ICP algorithms can utilize different values of λ. We apply such values of the regularization parameter λ to obtain the closest results.

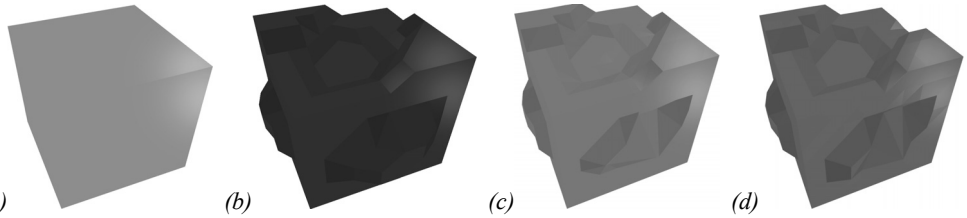
Let the result of the M-PPt algorithm be a set of matrices $A_1^M, \ldots, A_s^M$, the result of the L-PPt algorithm be a set of matrices $A_1^L, \ldots, A_s^L$, s be the number of points in the cloud P. Denote by D the following matrix:

$$D = \frac{1}{s} \sum_{i=1}^{s} \left(A_i^M - A_i^L \right). \quad (44)$$

If the absolute values of the matrix D elements are sufficiently small, then the results of the M-PPt and L-PPt algorithms are geometrically close.

1. Let P be the template cloud (see fig. 1*a*), Q be the target cloud (see fig. 1*b*). The cloud P consists of 386 points. We use for the M-PPt $\lambda = 2$, for L-PPt $\lambda = 20$. The result P' of the M-PPt is shown in fig. 1*c*. The result P'' of the L-PPt is shown in fig. 1*d*.

$$D_1 = \begin{pmatrix} 0.00732275 & -0.00133542 & -0.00117996 & -0.00198606 \\ -0.0015366 & 0.0449268 & 0.0167424 & -0.012183 \\ -0.00476054 & 0.00945598 & 0.0127938 & -0.00354531 \\ 2.36207e-16 & -2.23461e-17 & 1.57164e-16 & 0 \end{pmatrix}.$$



(a) (b) (c) (d)

Fig. 1. (a) The template cloud P; (b) the target cloud Q; (c) cloud P ′for M-PPt; (d) cloud ′ P ″for L-PPt

The processing time of M-PPt is 128 439 ms, while the processing time of L-PPt is 2 090 ms.

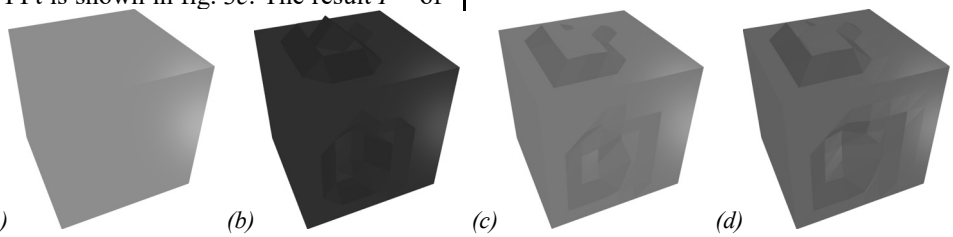
2. Let P be the template cloud (see fig. 1*a*), Q be the target cloud (see fig. 2*b*). The cloud P consists of 386 points. We use for the M-PPt $\lambda = 2$, for L-PPt $\lambda = 20$. The result P' of the M-PPt is shown in fig. 2*c*. The result P'' of the L-PPt is shown in fig. 2*d*.

$$D_2 = \begin{pmatrix} 0.0110707 & -0.00236798 & -0.00262535 & -0.0105562 \\ 0.00271782 & 0.00722583 & -0.00481619 & 2.63567e-05 \\ -0.0048493 & 0.000506319 & 0.0136053 & -0.00121706 \\ 3.1641e-18 & -1.38203e-16 & -1.72059e-16 & 0 \end{pmatrix}.$$

The processing time of M-PPt is 126 170 ms, the processing time of L-PPt is 1 830 ms.

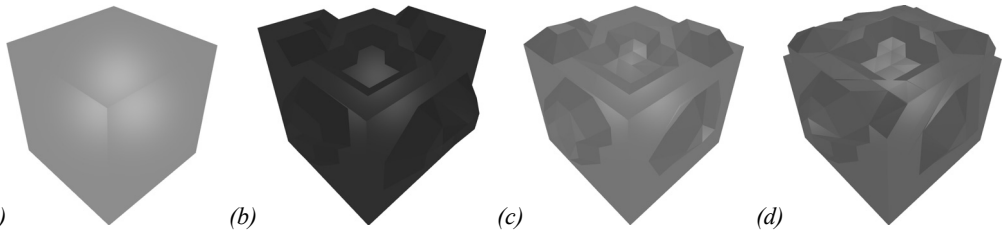
3. Let P be the template cloud (see fig. 3*a*), Q be the target cloud (see fig. 3*b*). The cloud P consists of 386

points. We use for M-PPt $\lambda = 4$, for L-PPt $\lambda = 20$. The result P' of the M-PPt is shown in fig. 3*c*. The result P'' of the L-PPt is shown in fig. 3*d*.



(a) (b) (c) (d)

Fig. 2. (a) The template cloud P; (b) the target cloud Q; (c) cloud P ′for M-PPt; (d) cloud ' P ″for L-PPt



(a) (b) (c) (d)

Fig. 3. (a) The template cloud P; (b) the target cloud Q; (c) cloud P ′for M-PPt; (d) cloud ' P ″for L-PPt

$$D_3 = \begin{pmatrix} 0.0209787 & 0.0251011 & -0.0138976 & 0.0197959 \\ 0.0233503 & 0.0840137 & 0.0182364 & 0.0195154 \\ -0.00716925 & 0.00344465 & 0.016215 & 0.00889768 \\ -8.12852e-18 & 4.85318e-16 & 1.72304e-17 & 0 \end{pmatrix}.$$

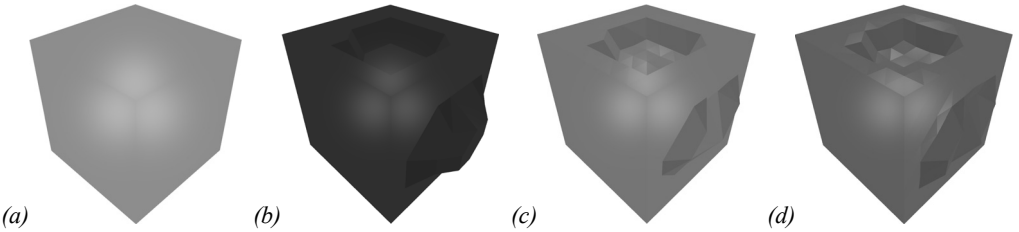
The processing time of M-PPt is 129 305 ms, the processing time of L-PPt is 1 814 ms.

4. Let P be the template cloud (see fig. 4*a*), Q be the target cloud (see fig. 4*b*). The cloud P consists of 386 points. We use for M-PPt $\lambda = 4$, for L-PPt $\lambda = 20$. The result P' of the M-PPt is shown in fig. 4*c*. The result P'' of the L-PPt is shown in fig. 4*d*.

$$D_4 = \begin{pmatrix} 0.0110124 & -0.00226438 & 3.04287e-05 & -0.00146024 \\ -0.00589484 & 0.0297209 & 0.00994416 & -0.0353524 \\ 0.00276987 & 0.000417316 & 0.00106658 & 0.00151188 \\ -1.89313e-16 & -2.88554e-17 & -8.309e-17 & 0 \end{pmatrix}.$$

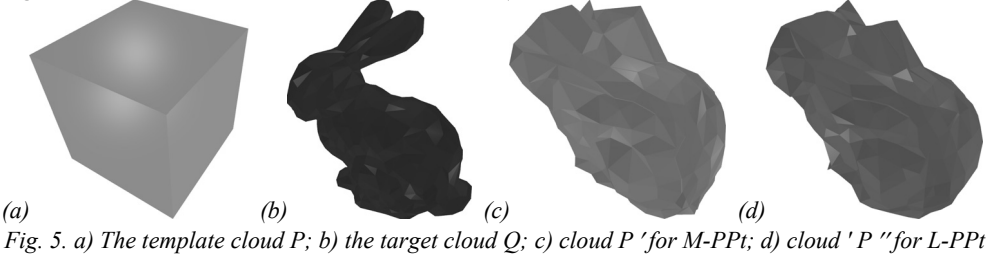
The processing time of M-PPt is 216 509 ms, the processing time of L-PPt is 1 899 ms.

5. Let P be the template cloud (see fig. 5*a*), Q be the target cloud (see fig. 5*b*). The cloud P consists of 386 points. We use for M-PPt $\lambda = 4$, for L-PPt $\lambda = 20$. The result P' of the M-PPt is shown in fig. 5*c*. The result P'' of the L-PPt is shown in fig. 5*d*.



(a) (b) (c) (d)

Fig. 4. a) The template cloud P; b) the target cloud Q; c) cloud P ′for M-PPt; d) cloud ' P ″for L-PPt



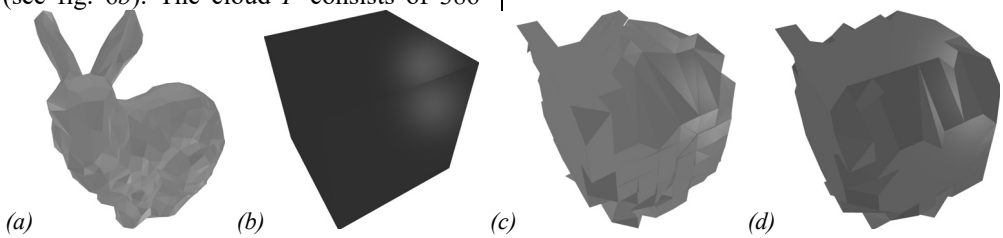
(a) (b) (c) (d)

Fig. 5. a) The template cloud P; b) the target cloud Q; c) cloud P ′for M-PPt; d) cloud ' P ″for L-PPt

$$D_5 = \begin{pmatrix} -0.0174952 & 0.0539372 & -0.0109088 & 0.0850687 \\ 0.0349777 & 0.125974 & -0.00577287 & -0.0262244 \\ -0.0287937 & -0.00240358 & 0.109681 & -0.029737 \\ 9.27819e-17 & -1.93867e-16 & -3.89343e-16 & 0 \end{pmatrix}.$$

The processing time of M-PPt is 263 698 ms, the processing time of L-PPt is 2 157 ms.

6. Let P be the template cloud (see fig. 6*a*), Q be the target cloud (see fig. 6*b*). The cloud P consists of 386 points. We use for M-PPt $\lambda = 4$, for L-PPt $\lambda = 40$. The result P' of the M-PPt is shown in fig. 6*c*. The result P'' of the L-PPt is shown in fig. 6*d*.



(a) *(b)* *(c)* *(d)*

Fig. 6. a) The template cloud P; b) the target cloud Q; c) cloud P ′for M-PPt; d) cloud ' P ″for L-PPt

$$D_6 = \begin{pmatrix} 0.198248 & -0.0202579 & -0.074494 & 0.191726 \\ -0.0707778 & 0.232688 & -0.0775521 & -0.119818 \\ 0.0993572 & -0.0766943 & 0.0147339 & 0.107009 \\ 1.34144e-15 & -2.6923e-15 & -1.09982e-15 & 0 \end{pmatrix}.$$

The processing time of M-PPt is 1 075 841 ms, the processing time of L-PPt is 4 725 ms.

Remark. The experiments show that it is possible to find a such value of λ that approximates well the algorithms to each other. On the other hand, the L-PPt is much faster than the M-PPt. The computational complexity of L-PPt is $O(s)$, while the computational complexity of M-PPt is $O(s^2)$, where s is the number of cloud points.

Conclusion

In this paper, we proposed a computationally efficient algorithm for the affine non-rigid point-to-point ICP. At each iterative step of the algorithm, approximation of the closed-form solution is utilized. Variational functional based on the point-to-point metric for affine transformation is exploited. The proposed algorithm is much faster than known methods based on minimization of the corresponded functional. For instance, the processing time of the M-PPt algorithm program realization utilizes the inverse matrix and based on the linear algebra library "Eigen" for C++ is 263 698 milliseconds, the processing time of the proposed L-PPt program realization is 2 157 milliseconds, for the considered point clouds. With the help of computer simulation, the performance of the proposed method is presented and discussed.

References

[1] Besl PJ, McKay ND. A method for registration of 3-D shapes. IEEE Trans Patt Anal Machine Intell 1992; 14(2): 239-256.

[2] Chen Y, Medioni G. Object modeling by registration of multiple range images. Proc IEEE Conf Robot Automat 1991; 3: 2724-2729.

[3] Amberg B, Romdhani S, Vetter T. Optimal step nonrigid icp algorithms for surface registration. Proc IEEE Conf Comput Vis Patt Recogn 2007: 1-8.

[4] Cheng S, Marras I, Zafeiriou S, Pantic M. Active nonrigid ICP algorithm. IEEE International Conference and Workshops on Automatic Face and Gesture Recognition 2015: 1-8.

[5] Blanz V, Vetter T. A morphable model for the synthesis of 3D faces. Proc SIGGRAPH 1999: 187-194.

[6] Davis J, Marschner S, Garr M, Levoy M. Filling holes in complex surfaces using volumetric diffusion. Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission 2002: 428-441.

[7] Kahler K, Haber J, Yamauchi H, Seidel H-P. Head shop: Generating animated head models with anatomical structure. Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2002: 55-63.

[8] Szeliski R, Lavalle S. Matching 3-D anatomical surfaces with non-rigid deformations using octree-splines. Int J Comp Vis 1996; 18(2): 171-186.

[9] Allen B, Curless B, Popovic Z. The space of human body shapes: Reconstruction and parameterization from range scans. ACM Trans on Graph 2003; 22(3): 587-594.

[10] Fischer B, Modersitzki J. Curvature based image registration. J Math Imaging Vis 2003; 18(1): 81-85.

[11] Dekker M, Feldmar J, Ayache N. Rigid, affine and locally affine registration of free-form surfaces. Int J Comp Vis 1996; 18(2): 99-119.

[12] Picos K, Diaz-Ramirez V, Kober V, Montemayor A, Pantrigo J. Accurate three-dimensional pose recognition from monocular images using template matched filtering. Opt Eng 2016; 55(6): 063102.

[13] Echeagaray-Patron B, Kober V, Karnaukhov V, Kuznetsov V. A method of face recognition using 3D facial surfaces. J Commun Technol Electron 2017; 62: 648-652.

[14] Echeagaray-Patron B, Miramontes-Jaramillo D, Kober V. Conformal parameterization and curvature analysis for 3D facial recognition. Proc IEEE Int Conf Comput Sci Comput Intelligence 2015: 843-844.

[15] Ruchay A, Dorofeev K, Kolpakov V. Fusion of information from multiple kinect sensors for 3D object reconstruction. Computer Optics 2018; 42(5): 898-903.

[16] Ruchay A, Dorofeev K, Kober A. 3D object reconstruction using multiple Kinect sensors and initial estimation of sensor parameters. Proc SPIE 2018; 10752: 1075222.
[17] Ruchay A, Dorofeev K, Kober A. Accurate reconstruction of the 3D indoor environment map with a RGB-D camera based on multiple ICP. CEUR Workshop Proceedings 2018; 2210: 300-308.
[18] Labunets V, Kokh E, Ostheimer E. Algebraic models and methods of computer image processing. Part 1. Multiplet models of multichannel images. Computer Optics 2018; 42(1): 84-95.
[19] Labunets V, Chasovskikh V, Smetanin J, Ostheimer E. Many-parameter mcomplementary Golay sequences and transforms. Computer Optics 2018; 42(6): 1074-1082.
[20] Tihonkih D, Makovetskii A, Kuznetsov V. A modified iterative closest point algorithm for shape registration. Proc SPIE 2016; 9971: 99712D.
[21] Tihonkih D, Makovetskii A, Kuznetsov V. The iterative closest points algorithm and affine transformations. CEUR Workshop Proceedings 2016; 1710: 349-356.
[22] Makovetskii A, Voronin S, Kober V, Tihonkih D. An efficient point-to-plane registration algorithm for affine transformations. Proc SPIE 2017; 10396: 103962J.
[23] Makovetskii A, Voronin S, Kober V, Tihonkih D. Affine registration of point clouds based on point-to-plane approach. Procedia Engineering 2017; 201: 322-330.
[24] Du S, Zheng N, Meng G, Yuan Z. Affine registration of point sets using ICP and ICA. IEEE Signal Proces Lett 2008; 15: 689-692.

Authors' information

Artyom Yurievch Makovetskii – born in 1967. A graduate of Chelyabinsk State University (1992). Candidate of Physical and Mathematical Sciences (2000). Since 2004, he has been a lecturer at the Chelyabinsk State University. The sphere of scientific interests is digital image processing. E-mail: *artemmac@csu.ru* .

Sergei Mikhailovich Voronin – born in 1955. A graduate of the Moscow State University (1977). Doctor of Physical and Mathematical Sciences (2012). Since 2012, he has been a professor of the Chelyabinsk State University. The sphere of scientific interests are theory of functions of a complex variable and digital image processing.
E-mail: *voron@csu.ru* .

Vitalii Ivanovich Kober – born in 1961. Graduated from the Kuibushev Institute of Aviation in 1984. Received Candidate's degree in 1992 and Doctor's in 2004. At present he is a leading researcher at the Institute for Information Transmission Problems, Russian Academy of Sciences, and a professor at the Chelysbinsk State University. Scientific interests: processing of signals and images, pattern recognition. E-mail: *vkober@hotmail.com* .

Aleksei Vyacheslavovich Voronin – born in 1995. Since 2014, a student of the Mathematical department of the Chelysbinsk State University. Scope of scientific interests – programming, digital image processing.
E-mail: *ununus@mail.ru* .

Code of State Categories Scientific and Technical Information (in Russian – GRNTI)): 28.23.15
Received June 20, 2019. The final version – October 31, 2019.