Analytical algorithm of weighted 3D datum transformation using the constraint of orthonormal matrix

Huainen Zeng

Abstract

Based on the Lagrangian extremum law with the constraint that rotation matrix is an orthonormal matrix, the paper presents a new analytical algorithm of weighted 3D datum transformation. It is a stepwise algorithm. Firstly, the rotation matrix is computed using eigenvalue-eigenvector decomposition. Then, the scale parameter is computed with computed rotation matrix. Lastly, the translation parameters are computed with computed rotation matrix and scale parameter. The paper investigates the stability of the presented algorithm in the cases that the common points are distributed in 3D, 2D, and 1D spaces including the approximate 2D and 1D spaces, and gives the corresponding modified formula of rotation matrix. The comparison of the presented algorithm and classic Procrustes algorithm is investigated, and an improved Procrustes algorithm is presented since that the classic Procrustes algorithm may yield a reflection rather than a rotation in the cases that the common points are distributed in 2D space. A simulative numerical case and a practical case are illustrated.

Keywords: Weighted 3D datum transformation; Analytical algorithm; Lagrangian extremum; Constraint of orthonormal matrix; Procrustes algorithm

Background

Three-dimensional datum transformation is a frequently used work in geodesy, engineering surveying, photogrammetry, mapping, geographical information science (GIS), machine vision, etc., e.g., Aktuğ (2009), Akyilmaz (2007), El-Mowafy et al (2009), Ge et al (2013), Han and Van Gelder (2006), Horn (1986), Kashani (2006), Neitzel (2010), Palánész et al (2013), Soler (1998), Soler and Sny (2004), Soycan and Soycan (2008), Zeng (2014). Usually, in order to compute the transformed coordinate, the transformation parameters in the transformation model (e.g., seven-parameter similarity transformation, see Aktuğ 2012, Leick 2004, Leick and van Gelder 1975,) need to be solved with several control points in advance. So far, a large number of algorithms for recovering the parameters have been presented, which can be divided into two classes. One is the numerical iterative algorithm, and the other one is analytical algorithm. The former needs the initial parameter values, linearization, and iterative computation, e.g., Zeng and Tao (2003), Chen et al. (2004), Zeng and Huang (2008), El-Habiby et al. (2009), Zeng and Yi (2011), etc. In the case that the rotation angle is large, the initial values are difficult even never to be obtained in advance, and consequently, it leads to the failure of solution (see Zeng and Yi 2011). We should note that if global optimization algorithms are used, then no initial values are required (see, e.g., Xu 2003a, Xu 2003b, Xu 2002). In contrast, the latter does not involve the initial parameter values, linearization, and iterative computation, and can give the exact solution quickly. However, because of the complexity of mathematical derivation, only several analytical algorithms have been put forward. Grafarend and Awange (2003) presented the Procrustes algorithm which utilized the singular value decomposition technique. Shen et al. (2006) presented a quaternion-based algorithm which utilized the quaternion property and eigenvalue-eigenvector decomposition. Han (2010) presented a stepwise approach to individually calculate the transformation parameters by...
the physical interpretation of similarity transformation. Zeng and Yi (2010) presented a new analytical algorithm based on the good properties of Rodrigues matrix and Gibbs vector.

The present study is organized as follows. In the Methods section, a new analytical algorithm to weighted 3D datum transformation is derived in detail, based on the Lagrangian extremum law with the constraint that the rotation matrix is an orthonormal matrix. In the meanwhile, its stability is discussed when the distribution of 3D control points degenerates into 2D (planar) or even 1D (collinear). The presented algorithm and classic Procrustes algorithm are compared, and an improved Procrustes algorithm is presented since that the classic Procrustes algorithm may yield a reflection rather than a rotation in the cases that the common points are distributed in 2D space. In the Results and discussion section, a simulative numerical case and a practical case are given to demonstrate the presented algorithm, classic Procrustes algorithm, and improved Procrustes algorithm. Lastly, conclusions are made in the Conclusions section.

Methods
Presentation of the basic algorithm
The seven-parameter similarity transformation model can be expressed as

\[ a_i = \lambda R b_i + t, \]  

subject to

\[ R^T R = I_3, \quad \det(R) = +1, \]  

where \( a_i = [X_i \ Y_i \ Z_i]^T \) and \( b_i = [x_i \ y_i \ z_i]^T \ i = 1, 2, \cdots, n \) are the 3D coordinates of a common point in target and source coordinate systems of transformation, tagged as system A and system B, respectively. Superscript \( T \) stands for transpose, \( I_3 \) denotes a \( 3 \times 3 \) identity matrix, and \( \det \) means computing the determinant of matrix. \( \lambda \) denotes the scale parameter, \( t = [\Delta X \ \Delta Y \ \Delta Z]^T \) denotes the three translation parameters, and \( R \) denotes the \( 3 \times 3 \) rotation matrix, which contains the three rotation angles. Supposing \( R \) is formed by rotating angles \( \alpha, \beta, \gamma \) counterclockwise around the Cartesian \( X, Y, Z \) axes, respectively, then \( R \) can be expressed by rotation angles as

\[ R = \begin{bmatrix} \cos \gamma \cos \beta & \cos \gamma \sin \beta & \sin \gamma \\ -\sin \gamma \cos \beta & \sin \gamma \sin \beta & \cos \gamma \\ \sin \beta & -\cos \beta & 0 \end{bmatrix} \]  

Using (3), the rotation angles \( \alpha, \beta, \gamma \) can be computed if \( R \) is recovered as

\[ a = -\tan^{-1} \frac{R_{32}}{R_{33}}, \quad \beta = \sin^{-1}(R_{31}), \quad \gamma = -\tan^{-1} \frac{R_{31}}{R_{11}}, \]  

where \( R_{ij} \) is the element of \( R \) in the \( i \)th row and \( j \)th column.

Introducing the following matrix form of the coordinates as

\[ A = [a_1 \ a_2 \ \cdots \ a_n], \quad B = [b_1 \ b_2 \ \cdots \ b_n], \]  

then Eq. (1) is rewritten as

\[ A = \lambda RB + t1_n, \]  

where \( 1_n = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \), i.e., a row vector with \( n \) elements and all elements are 1. It is obvious that in order to determine the seven parameters, the number \( n \) of common points must be greater than or equal to 3.

Considering the coordinates include errors, Eq. (7) is transformed as

\[ A = \lambda RB + t1_n + E, \]  

where \( E \) is the transformation error matrix. The criterion of the least squares can be constructed by the Lagrangian extremum law with the constraint of Eq. (2), i.e., orthonormal matrix as follows. It is worthy of note that the constraint \( \det(R) = +1 \) is not imposed, since it can be separately treated at some extra computation as in the Stability of the basic algorithm and its modification section.

\[ L(\lambda, t, R, \Lambda) = \text{tr}(EPE^T) + \text{tr}(\Lambda(R^TR-I_3)) = \min, \]  

where \( \text{tr} \) denotes trace operation of matrix, \( \Lambda \) is a symmetric Lagrangian multiplier matrix, and \( P \) represents the weight matrix that every point has an isotropic weight and is independent of each other. Substituting the expression of \( E \) easily obtained from Eq. (7) into Eq. (8), one can obtain

\[ L(\lambda, t, R, \Lambda) = \text{tr}(EPE^T) + \text{tr}(\Lambda(R^TR-I_3)) \]
\[ = \text{tr}((A-\lambda RB-t1_n)(A-\lambda RB-t1_n)^T) + \text{tr}(\Lambda(R^TR-I_3)) \]
\[ = \min. \]  

If and only if the following conditions are satisfied, the Lagrangian extremum exists.

\[ \frac{\partial L}{\partial t} = 0, \]
\[ \frac{\partial L}{\partial \lambda} = 0, \]
\[ \frac{\partial L}{\partial R} = 0. \]  
(12)

\[ \frac{\partial L}{\partial \Lambda} = 0. \]  
(13)

By Eqs. (9) and (10), one gets

\[ L(\lambda, t, R, \Lambda) = \text{tr} \left( (A - \lambda RB - t_1 p)(A^T - \lambda B^T R t^T - 1_n^T t^T) \right) \]
\[ + \text{tr}(\Lambda(R^T R - I_3)) \]
\[ = \text{tr} \left( (A - \lambda RB) P(A^T - \lambda B^T R t^T) - t_1 P(A^T - \lambda B^T R t^T) \right) \]
\[ + \text{tr}(\Lambda(R^T R - I_3)). \]
(14)

\[ \frac{\partial L}{\partial t} = -\text{tr} \left( (A - \lambda RB) P \right) - \text{tr}(\lambda B^T t^T) \]
\[ = -2(\lambda - \lambda RB) P t + 2 \lambda P t \]
\[ = 0, \]
(15)

and thus,

\[ t = \left( 1_n P \right)^{-1} (A - \lambda RB) P 1_n^T. \]
(16)

Obviously, \( t \) is the function form of \( \lambda \) and \( R \).

Substituting Eq. (16) into Eq. (7), one gets

\[ E = A - \lambda RB - \left( 1_n P \right)^{-1} (A - \lambda RB) P 1_n^T 1_n \]
\[ = (A - \lambda RB) \left( I_n - \left( 1_n P \right)^{-1} (1_n^T P) 1_n \right), \]
(17)

where \( I_n = \left( 1_n P \right)^{-1} (1_n^T P) 1_n \) is the centering matrix.

Let \( \Delta A = A(I_n - \left( 1_n P \right)^{-1} (1_n^T P) 1_n) \), \( \Delta B = B(I_n - \left( 1_n P \right)^{-1} (1_n^T P) 1_n) \), and thus, they are the centralized coordinate matrix, and then, Eq. (17) is written as

\[ E = \Delta A - \lambda R \Delta B. \]
(18)

Substituting Eq. (18) into Eq. (8), one gets

\[ L(\lambda, R, \Lambda) = \text{tr} \left( (\Delta A - \lambda R \Delta B) P(\Delta A - \lambda R \Delta B)^T \right) + \text{tr}(\Lambda(R^T R - I_3)) \]
\[ = \text{tr}(\Delta A - \lambda R \Delta B) P(\Delta A^T - \lambda R \Delta B^T R t^T) + \text{tr}(\Lambda(R^T R - I_3)) \]
\[ = \text{tr}(\Delta A^T \Delta A - \lambda R \Delta B^T R t^T + \lambda^2 R \Delta B^T R t^T + \lambda R \Delta B^T R t^T) \]
\[ + \text{tr}(\Lambda(R^T R - I_3)). \]
(19)

Derivation of Eq. (19) makes use of the properties of trace operation, i.e.,

\[ \text{tr}(\Delta A^T \Delta A - \lambda R \Delta B^T R t^T + \lambda^2 R \Delta B^T R t^T + \lambda R \Delta B^T R t^T) \]
\[ = \text{tr}\left( (\Delta A^T \Delta A - 2\lambda R \Delta B^T R t^T + \lambda^2 R \Delta B^T R t^T) \right) \]
\[ + \text{tr}(\Lambda(R^T R - I_3)). \]
Table 1 Simulative true coordinates of control points in system B

| Point number | Set 1 (m) | Set 2 (m) | Set 3 (m) | Set 4(m) | Set 5 (m) | Set 6 (m) |
|--------------|-----------|-----------|-----------|-----------|-----------|-----------|
|              | x         | y         | z         | x         | y         | z         |
| 1            | 10.000    | 30.000    | 5.000     | 10.000    | 30.000    | 5.000     |
| 2            | 20.000    | 30.000    | 12.500    | 20.000    | 30.000    | 12.500    |
| 3            | 30.000    | 30.000    | 15.000    | 30.000    | 30.000    | 15.000    |
| 4            | 10.000    | 20.000    | 9.500     | 10.000    | 20.000    | 9.500     |
| 5            | 20.000    | 20.000    | 11.000    | 20.000    | 20.000    | 11.000    |
| 6            | 30.000    | 20.000    | 10.000    | 30.000    | 20.000    | 10.000    |
| 7            | 10.000    | 10.000    | 14.500    | 10.000    | 10.000    | 14.500    |
| 8            | 20.000    | 10.000    | 4.500     | 20.000    | 10.000    | 4.500     |
| 9            | 30.000    | 10.000    | 4.000     | 30.000    | 10.000    | 4.000     |

Fig. 1 The distribution of control point in system B. a Set 1. b Set 2. c Set 3. d Set 4. e Set 5. f Set 6
where \( d_i \) and \( v_i \) for \( i = 1, 2, 3 \) are the eigenvalues and corresponding eigenvectors of the matrix \( D^T D \). So, Eq. (30) can be written as

\[
R = D \left( \frac{v_1 v_1^T}{\sqrt{d_1}} + \frac{v_2 v_2^T}{\sqrt{d_2}} + \frac{v_3 v_3^T}{\sqrt{d_3}} \right)
\]

Stability of the basic algorithm and its modification

Obviously, the construction of the inverse of the square root of \( D^T D \), i.e., Eq. (31), fails if one or two of \( d_i \) for \( i = 1, 2, 3 \) equals to 0. Assume that the eigenvalues of the matrix \( D^T D \) satisfies the following condition.

\[
d_1 \leq d_2 \leq d_3.
\]

When the common points are distributed in a plane, i.e., 2D space, the matrix \( D^T D \) is singular and of rank 2, and thus,

\[
d_1 = 0, \ d_3 \geq d_2 > 0.
\]

One can compute the rotation matrix as follows.

\[
D_1 = \left( \frac{v_2 v_2^T}{\sqrt{d_2}} + \frac{v_3 v_3^T}{\sqrt{d_3}} \right),
\]

\[
D_2 = \left( D D_1 (D D_1)^T - I_3 \right) \frac{v_1 v_1^T}{\sqrt{\text{tr}(D_2)}},
\]

\[
R = D D_1 \pm \frac{D_2}{\sqrt{\text{tr}(D_2)}},
\]

where the sign \( \pm \) is chosen so that \( \det(R) = +1 \) is satisfied.

The above case is the ideal one, but in the case that the common points are distributed in an approximate plane, although

\[
d_3 \geq d_2 > d_1 > 0,
\]

and the order of magnitude of \( d_i \) for \( i = 1, 2, 3 \) differs greatly from each other. In other words, the matrix \( D^T D \) is approximately singular, and the condition number of matrix \( D^T D \) is very big (usually the threshold is set to 100). The Eq. (31) is ill-conditioned and leads to a biased solution. At this time, one can also obtain the exact solution by Eqs. (35)–(37).

| Point number | X (m) | Y (m) | Z (m) | X (m) | Y (m) | Z (m) | X (m) | Y (m) | Z (m) |
|--------------|-------|------|------|-------|------|------|-------|------|------|
| 1            | 52.116| 7.239| 14.222| 52.116| 7.239| 14.222| 94.294| 37.450| 17.742|
| 2            | 58.807| 9.608| 24.512| 58.807| 9.608| 24.512| 103.013| 41.272| 28.201|
| 3            | 61.443| 9.072| 34.463| 61.443| 9.072| 34.463| 111.732| 45.093| 38.659|
| 4            | 49.949| 17.746| 16.493|       |       |       | 76.310| 36.628| 18.693|
| 5            | 51.773| 16.629| 26.376|       |       |       | 85.029| 40.450| 29.151|
| 6            | 51.570| 14.060| 36.090|       |       |       | 93.748| 44.271| 39.610|
| 7            | 48.187| 28.543| 18.797|       |       |       | 58.325| 35.806| 19.643|
| 8            | 40.683| 20.745| 27.902|       |       |       | 67.044| 39.627| 30.102|
| 9            | 40.886| 18.466| 37.650|       |       |       | 75.763| 43.449| 40.560|

| Point number | X (m) | Y (m) | Z (m) | X (m) | Y (m) | Z (m) | X (m) | Y (m) | Z (m) |
|--------------|-------|------|------|-------|------|------|-------|------|------|
| 1            | 60.227| 13.048| 14.899| 44.537| 25.929| 18.493| 30.608| 28.012| 19.782|
| 2            | 60.835| 11.060| 24.681| 59.073| 21.858| 26.985| 31.216| 26.023| 29.563|
| 3            | 61.443| 9.072| 34.463| 73.610| 17.787| 35.478| 31.824| 24.035| 39.345|
| 4            | 54.410| 20.941| 16.865| 88.146| 13.716| 43.971|       |       |       |
| 5            | 55.017| 18.953| 26.647| 102.683| 9.645| 52.463|       |       |       |
| 6            | 55.625| 16.965| 36.429| 117.219| 5.573| 60.956|       |       |       |
| 7            | 48.592| 28.834| 18.831| 131.756| 1.502| 69.449|       |       |       |
| 8            | 49.200| 26.846| 28.613| 146.292| −2.569| 77.941|       |       |       |
| 9            | 49.808| 24.857| 38.394| 160.829| −6.640| 86.434|       |       |       |
When the common points are distributed in a line, i.e., 1D space, the matrix $D^T D$ is singular and of rank 1, and thus,

$$d_2 = d_1 = 0, \quad d_3 > 0.$$  \hfill (39)

The rotation matrix is impossible to recover in a whole; however, one can recover at most two rotation angles by the following formula.

$$R = \frac{v_3 v_3^T}{\sqrt{d_3}}.$$  \hfill (40)

and the utilization of Eq. (40) make it feasible to compute the translation parameter and scale parameter.

The above case is the ideal one, but in the case that the common points are distributed in an approximate line, although

$$d_3 > d_2 > d_1 = 0,$$  \hfill (41)

and the order of magnitude of $d_i$ for $i = 2, 3$ differs greatly from each other (e.g., $d_3 > 100d_2$). The computation by Eqs. (35)–(37) is biased, and no exact solution can be found like the case that the common points are distributed in a line space. At this time, one can carry out the computation by Eq. (40).

**Comparison to classic Procrustes algorithm and improvement of the classic Procrustes algorithm**

The classic Procrustes algorithm presented by Grafarend and Awange (2003) is a well-known analytical algorithm of 3D datum transformation. It is also based on the Lagrangian extremum law, similarly to the presented algorithm in this paper. But differently, it does not constrain the orthonormal matrix condition. For the Procrustes algorithm, due to utilization of the singular value decomposition technique, the computed rotation matrix always satisfies the constraint condition $R^T R = I_3$; however, in the cases that the common points are

| Table 4 Calculated transformation parameters and mean error |
|---------------------------------|-----------------|-----------------|-----------------|-----------------|
| Parameters | Set 1 | Set 2 | Set 3 | Set 4 | Set 5 | Set 6 |
| PA | CPA | IPA | PA | CPA | IPA | PA | CPA | IPA | PA | CPA | IPA |
| ΔX (m) | 30.000215 | 30.000215 | 30.000215 | 29.997125 | 64.907810 | 29.997125 | 29.999564 | 29.884451 | 29.999564 |
| ΔY (m) | 30.000104 | 30.000104 | 30.000104 | 29.994918 | 64.907810 | 29.994918 | 30.000156 | 31.848392 | 30.000156 |
| ΔZ (m) | 9.999992 | 9.999992 | 9.999992 | 10.000804 | 64.907810 | 10.000804 | 9.999562 | 9.420190 | 9.999562 |
| α (°) | 70.998025 | 70.998025 | 70.998025 | 70.994443 | 64.907810 | 70.994443 | 60.478813 | 70.994949 | 60.478813 |
| β (°) | 73.001648 | 73.001648 | 73.001648 | 73.000253 | 73.000253 | 73.000253 | 73.000571 | 73.000571 | 73.000571 |
| γ (°) | 4.999992 | 4.999992 | 4.999992 | 10.000804 | 1.797635 | 10.000804 | 9.999562 | 9.420190 | 9.999562 |
| λ | 0.000315 | 0.000315 | 0.000315 | 0.000197 | 0.000197 | 0.000197 | 0.000313 | 0.000313 | 0.000313 |
| ME (m) | 0.000315 | 0.000315 | 0.000315 | 0.000197 | 0.000197 | 0.000197 | 0.000313 | 0.000313 | 0.000313 |

| Table 5 Number of solvable transformation parameters |
|---------------------------------|-----------------|-----------------|-----------------|
| Space | Translation | Angle | Scale | Total | Translation | Angle | Scale | Total | Translation | Angle | Scale | Total |
| PA | CPA | IPA | PA | CPA | IPA | PA | CPA | IPA | PA | CPA | IPA |
| 3D | 3 | 3 | 1 | 7 | 3 | 3 | 1 | 7 | 3 | 3 | 1 | 7 |
| 2D | 3 | 3 | 1 | 7 | 0–3 | 0–3 | 1 | 1–7 | 3 | 3 | 1 | 7 |
| 1D | 3 | 0–2 | 1 | 4–6 | 3 | 0–2 | 1 | 4–6 | 3 | 0–2 | 1 | 4–6 |
distributed in a rigid or approximate plane, this situation that \( \det(R) = -1 \) rather than \( \det(R) = +1 \) usually happens. This means the computed \( R \) is a reflection instead of a rotation. For the presented algorithm in this paper, the constraint condition \( R^T R = I_3 \) is imposed in the computation, and thus, it is always satisfied. And in the cases that the common points are distributed in a rigid or approximate plane, the sign \( \pm \) in Eq. (36) is properly chosen so that \( \det(R) = +1 \) is satisfied.

To recover the exact rotation matrix by Procrustes algorithm when \( \det(R) = -1 \), the computation formula of \( R \), i.e., Eq. (22) in Grafarend and Awange (2003) should be improved as

\[
X_M = U \tilde{V}^T,
\]

(42)

where \( \tilde{V} = [V_1 \ V_2 \ -V_3] \), \( V_1 \), \( V_2 \), and \( V_3 \) are the column matrix of \( V \), and \( V_3 \) is the column matrix that corresponds to the singular value that is 0.

Therefore, the general improved computation formula of \( R \), i.e., Eq. (22) in Grafarend and Awange (2003) is

\[
X_M = U \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \left( \det(UV^T) \right)^{-1} V^T,
\]

(43)

which is stable for the cases that the common points are distributed in 3D and 2D spaces including approximate 2D space. In the case that the common points are distributed in 1D space, Eq. (43) can recover at most two rotation angles and be used to compute the translation parameter and scale parameter.

**Results and discussion**

**Simulative case**

The case data is simulated as follows. In order to investigate the stability performance of the presented algorithm (PA) in this paper, the classic Procrustes algorithm (CPA) and improved Procrustes algorithm (IPA) in the cases that the control points are distributed in 3D, 2D, and 1D spaces, six sets of control point in system B is first given in Table 1, of which set 1 is distributed in 3D space, sets 2, 3, and 4 are distributed in 2D space, and sets 5 and 6 are distributed in 1D space. The distribution of six sets of control point is depicted intuitively in Fig. 1. Set 2 has only three control points, and it is necessary for the least point number to solve the seven parameters. Secondly, the theoretical seven parameters are given in Table 2. For the sake of an efficient test of the algorithms, the rotation angles are designed to be big angles. Thirdly, the coordinates of control points in system A are computed by Eq. (1), and the result is listed in Table 3. In this case, the stability of the three algorithms is focused, so the weight matrix is designated to identity matrix for easy demonstration.

Next, the transformation seven parameters are recovered by the three algorithms, and the result is listed in Table 4. ME in Table 4 is the mean error and computed by

\[
ME = \sqrt{\frac{\text{tr}(E^T PE)}{3n-7}}.
\]

(44)

![Fig. 2 The distribution of seven control points in local system](image)
It is seen from Table 4 that the results of seven parameters are identical and accurate for the three algorithms in the case of set 1, i.e., the case that the control points are distributed in 3D space. For the cases that the control points are distributed in 2D space, PA and IPA have the identical and accurate results of seven parameters, but CPA has one correct result of seven parameters for set 4 and two wrong results of rotation angles for sets 2 and 3, because reflections rather than rotations are found, and all translation and scale parameters are correct. For the cases that the control points are distributed in 1D space, i.e., sets 5 and 6, the three algorithms all recover the correct translation and scale parameters and at most recover two rotation angles. The number of solvable transformation parameters for the three algorithms is counted and listed in Table 5.

Actual case
The case data is from Grafarend and Awange (2003). The coordinates of control points in system B (local system) and A (WGS-84 system) is listed in Table 6. The distribution of seven control points in local system is depicted in Fig. 2. From this figure, it is seen that the distribution of control points are in an approximate plane. In the process of PA computation, condition number of matrix \( D^T D \) is \( 2.5 \times 10^{11} \), so Eq. (32) is seriously ill-conditioned and yields a biased solution if not processed. When the weight matrix is an identity matrix, the computed results of seven parameters with PA and CPA are listed in Table 7. For the situation that the weight matrix is a point-wise matrix, i.e., every point has isotropic weight and is independent of each other, the point-wise matrix is generated by the way introduced in Grafarend and Awange (2003) and is listed in Table 8. The computed results of seven parameters with PA and CPA are listed in Table 9.

It is seen from Tables 7 and 9 that the results of PA and CPA are identical if the bias caused by decimal rounding is ignored. Hence, the PA is comparable with CPA.

Conclusions
The numerical case study shows that the presented new algorithm and improved Procrustes algorithm are both stable and reliable for the cases that the control points are distributed in 3D, and 2D including approximate 2D space, and can recover at most two angles as well as all translation and scale parameters for the cases that the control points are distributed in 1D space. The classic Procrustes algorithm also can compute all translation and scale parameters for all the cases that the control

|          | PA          | CPA          |
|----------|-------------|--------------|
| Rotation matrix |
| \( R \) | 1.0000000000 | 0.0000048146 |
|          | -0.0000048146 | 1.0000000000 |
|          | 0.0000043327  | 0.0000048408  |
| Rotation angles (\(^\circ\)) |
| \( \alpha \) | -0.998496121 |
| \( \beta \) | 0.893693325  |
| \( \gamma \) | 0.993086229  |
| Translation (m) |
| \( \Delta X \) | 641.8805 |
| \( \Delta Y \) | 68.6551 |
| \( \Delta Z \) | 416.3982 |
| Scale |
| \( \lambda \) | 1.000005583 |
| ME (m) | 0.0773 |

Table 8 Point-wise weight matrix

| Values   | 2.170137 | 0 | 0 | 0 | 0 | 0 | 0 |
|----------|----------|---|---|---|---|---|---|
|          | 0 | 2.097755 | 0 | 0 | 0 | 0 | 0 |
|          | 0 | 0 | 2.208968 | 0 | 0 | 0 | 0 |
|          | 0 | 0 | 0 | 2.201671 | 0 | 0 | 0 |
|          | 0 | 0 | 0 | 0 | 2.182928 | 0 | 0 |
|          | 0 | 0 | 0 | 0 | 0 | 2.268808 | 0 |
|          | 0 | 0 | 0 | 0 | 0 | 0 | 2.643404 |
points are distributed in 3D, 2D, and 1D spaces; how-
ever, it may yield a reflection rather than a rotation in
the cases that the common points are distributed in 2D
space. The numerical case study also shows the pre-
sented algorithm in this paper, and improved Procrustes
algorithm is both stable and reliable when the rotation
angles are big. And the presented algorithm in this paper
is comparable with the classic Procrustes algorithm
when the point-wise weight matrix is involved.

### Appendix

Let

\[ K = R^T R - I_3, \]

(45)

\[ K^T = R^T R - I_3 = K, \]

(46)

then \( K \) is a symmetric matrix. Since that \( \Lambda \) is a symmet-
ric Lagrangian multiplier matrix, \( \Lambda \) and \( K \) can be de-
described as follows.

\[ \Lambda = \begin{bmatrix}
\lambda_1 & \lambda_2 & \lambda_3 \\
\lambda_2 & \lambda_4 & \lambda_5 \\
\lambda_3 & \lambda_5 & \lambda_6
\end{bmatrix}, \]

(47)

\[ K = \begin{bmatrix}
\kappa_1 & \kappa_2 & \kappa_3 \\
\kappa_2 & \kappa_4 & \kappa_5 \\
\kappa_3 & \kappa_5 & \kappa_6
\end{bmatrix}. \]

(48)

\[ \text{tr}(\Lambda K) = \lambda_1 \kappa_1 + \lambda_2 \kappa_2 + \lambda_3 \kappa_3 + \lambda_4 \kappa_4 + \lambda_5 \kappa_5 + \lambda_6 \kappa_6, \]

\[ = \lambda_1 \kappa_1 + 2 \lambda_2 \kappa_2 + 2 \lambda_3 \kappa_3 + \lambda_4 \kappa_4 + 2 \lambda_5 \kappa_5 + \lambda_6 \kappa_6, \]

(49)

\[ \frac{\delta \text{tr}(\Lambda K)}{\delta \Lambda} = \begin{bmatrix}
\delta \lambda_1 & \delta \lambda_2 & \delta \lambda_3 \\
\delta \lambda_2 & \delta \lambda_4 & \delta \lambda_5 \\
\delta \lambda_3 & \delta \lambda_5 & \delta \lambda_6
\end{bmatrix} = \begin{bmatrix}
\kappa_1 & 2 \kappa_2 & 2 \kappa_3 \\
2 \kappa_2 & \kappa_4 & 2 \kappa_5 \\
2 \kappa_3 & 2 \kappa_5 & \kappa_6
\end{bmatrix}. \]

(50)

By Eqs. (19) and (13), one gets

\[ \frac{\delta L}{\delta \Lambda} = \frac{\delta \text{tr}(\Lambda K)}{\delta \Lambda} = 0, \]

(51)

and then,

\[ \begin{bmatrix}
\kappa_1 & 2 \kappa_2 & 2 \kappa_3 \\
2 \kappa_2 & \kappa_4 & 2 \kappa_5 \\
2 \kappa_3 & 2 \kappa_5 & \kappa_6
\end{bmatrix} = 0. \]

(52)

Further,

\[ \kappa_i = 0 \ (i = 1, 2, \cdots, 6). \]

(53)

So,

\[ K = R^T R - I_3 = 0. \]

(54)
Competing interests
The author declares that he has no competing interests.

Acknowledgments
The study is supported by the Open Foundation of Hubei Key Laboratory of Construction and Management in Hydropower Engineering, China Three Gorges University (Grant No. 2014KED13), the National Natural Science Foundation of China (Grant No. 41104009), the Open Research Fund Program of the Key Laboratory of Geospace Environment and Geodesy, Ministry of Education, China (Grant No. 11-01-04), and the 2015 Open Foundation of the Key Laboratory of Precise Engineering and Industry Surveying, National Administration of Surveying, Mapping, and Geoinformation of China. The author is grateful for the support and good working atmosphere provided by his research team in China Three Gorges University. Last but not the least, the author’s special thanks are given to Dr. Koji Matsumoto and two anonymous reviewers for their valuable comments and suggestions, which enhance the quality of this manuscript.

Received: 3 March 2015 Accepted: 4 June 2015
Published online: 30 June 2015

References
Akyilmaz O (2007) Total least squares solution of coordinate transformation. Surv Rev 39(302):69–80

Xu PL (2002) A hybrid global optimization method: the one-dimensional case. J Comput Appl Math 155:423–446

Zeng HE (2014) Planar coordinate transformation and its parameter estimation in the complex number field. Acta Geod Geophys 49(1):79–94

Zeng HE, Huang SX (2008) A kind of direct search method adopted to solve 3D coordinate transformation parameters. Geomat Info Sci Wuhan Uni 33(11):1118–1121

Zeng HE, Yi QL (2010) A new analytical solution of nonlinear geodetic datum transformation. In: Proceedings of the 18th International Conference on Geoinformatics

Zeng HE, Yi QL (2011) Quaternion-based iterative solution of three-dimensional coordinate transformation problem. J of Computers 6(7):1361–1368

Zeng WX, Tao BZ (2003) Non-linear adjustment model of three-dimensional coordinate transformation. Geomat Info Sci Wuhan Uni 28(5):566–568

Submit your manuscript to a SpringerOpen journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Immediate publication on acceptance
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com