applied sciences

MDPI

Article

An Automatic 3D Point Cloud Registration Method Based on Biological Vision

Jinbo Liu [1,*], Pengyu Guo [2,*] and Xiaoliang Sun [3,*]

[1] Hypervelocity Aerodynamics Institute, China Aerodynamics Research and Development Center, Mianyang 621000, China
[2] National Innovation Institute of Defense Technology, Beijing 100071, China
[3] College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, China
* Correspondence: liujinbo@cardc.cn (J.L.); pengyu.guo@nudt.edu.cn (P.G.); alexander_sxl@nudt.edu.cn (X.S.)

Abstract: When measuring surface deformation, because the overlap of point clouds before and after deformation is small and the accuracy of the initial value of point cloud registration cannot be guaranteed, traditional point cloud registration methods cannot be applied. In order to solve this problem, a complete solution is proposed, first, by fixing at least three cones to the target. Then, through cone vertices, initial values of the transformation matrix can be calculated. On the basis of this, the point cloud registration can be performed accurately through the iterative closest point (ICP) algorithm using the neighboring point clouds of cone vertices. To improve the automation of this solution, an accurate and automatic point cloud registration method based on biological vision is proposed. First, the three-dimensional (3D) coordinates of cone vertices are obtained through multi-view observation, feature detection, data fusion, and shape fitting. In shape fitting, a closed-form solution of cone vertices is derived on the basis of the quadratic form. Second, a random strategy is designed to calculate the initial values of the transformation matrix between two point clouds. Then, combined with ICP, point cloud registration is realized automatically and precisely. The simulation results showed that, when the intensity of Gaussian noise ranged from 0 to 1 mr (where mr denotes the average mesh resolution of the models), the rotation and translation errors of point cloud registration were less than 0.1° and 1 mr, respectively. Lastly, a camera-projector system to dynamically measure the surface deformation during ablation tests in an arc-heated wind tunnel was developed, and the experimental results showed that the measuring precision for surface deformation exceeded 0.05 mm when surface deformation was smaller than 4 mm.

Keywords: point cloud registration; biological vision; automatic registration; cone vertices; deformation measurement

Citation: Liu, J.; Guo, P.; Sun, X. An Automatic 3D Point Cloud Registration Method Based on Biological Vision. *Appl. Sci.* **2021**, *11*, 4538. https://doi.org/10.3390/app11104538

Academic Editors: José Luis Rojo-Álvarez and Athanasios Nikolaidis

Received: 20 April 2021
Accepted: 14 May 2021
Published: 16 May 2021

1. Introduction

Near-space is a connected region of traditional aeronautics and space. Near-space supersonic vehicles have great potential, but if flown for a long time in an aerothermal environment, the surface of vehicles can be deformed, which causes functional failure. Thus, deformation properties of materials in an aerothermal environment need to be urgently explored. An arc-heated wind tunnel is the main device to simulate an aerothermal environment. There are many methods to reconstruct 3D shape data at different times during ablation tests, but the alignment of point clouds is difficult, because the overlap is too small.

Point cloud registration involves calculating a rigid transformation matrix, consisting of a rotation matrix and a translation vector, to minimize the alignment error between two point clouds, and has been widely used for simultaneous localization and mapping (SLAM) [1–3], multi-view point cloud registration [4,5], object recognition [6,7], etc. The classical iterative closest point (ICP) algorithm is the most widely used in point cloud

registration [8]. It was proposed by Besel, in 1992, and has been used to solve the registration problem of free-form surfaces. The basic idea is to search a corresponding closest point in a test point cloud for each point in a reference point cloud. According to the set consisting of all the closest points, a transformation matrix is calculated between two point clouds, resulting in a registration error. If the registration error cannot satisfy the stopping criterion, the transformed test point cloud is taken as a new test point cloud, and the above steps are repeated until the stopping criterion is satisfied. In the classical ICP algorithm, the registration error is presented in two forms, i.e., point-to-point and point-to-plane. In general, the ICP algorithm based on the point-to-plane registration error converges at a faster rate [9]. Using the classical ICP algorithm as a foundation, there have been many variants proposed by researchers [10–14]. The key step of these methods involves searching the set of closest points. However, in a real scenario of surface deformation, it is very difficult to accurately determine the closest points without any prior information. Moreover, these methods are very sensitive to the initial values of the transformation matrix. If the quality of initial values is not good enough, these methods easily suffer from local minima. Another way to solve the point cloud registration problem is by using probabilistic methods. The core idea is to transform the registration problem of point clouds into an optimization problem of probability model parameters [15–20]. As compared with ICP and its variants, these methods perform more robustly with respect to noise and outliers, but their computational complexity is high. Similar to ICP methods, these methods can only be used to align point clouds with large overlaps, whereas an initial value of the transformation matrix with high quality is also required. In the application of surface deformation measurement, the overlap between point clouds is small and the quality of initial values cannot be guaranteed. Thus, if the above methods are directly used to align point clouds, the registration error is large.

In order to solve the problem of surface deformation measurement, a complete solution is proposed. Figure 1 shows the flowchart. Firstly, fix the target to a mounting bracket with at least three cones. Secondly, a cone vertex detection algorithm is deduced to extract feature points. Thirdly, accurately align the point cloud before deformation to the point cloud after deformation using neighboring point clouds of each cone vertices which are not deformed during ablation tests through an automatic registration algorithm. Then, surface deformation can be obtained. This paper is organized as follows: In Section 2, we introduce the detection principles of cone vertices in detail; in Section 3, we derive an automatic registration algorithm for point clouds; in Section 4, we introduce the research methodology, including cone vertex detection, automatic registration, and surface deformation measurement; in Section 5, we present the research results, and in Sections 5.1 and 5.2 we present the accuracy and robustness of the cone vertex detection algorithm and automatic registration algorithm, respectively; in Section 5.3, we provide the results of surface deformation measurement; in Section 6, we discuss the research results; and, in Section 7, we conclude with the contributions of this study and next steps in research.

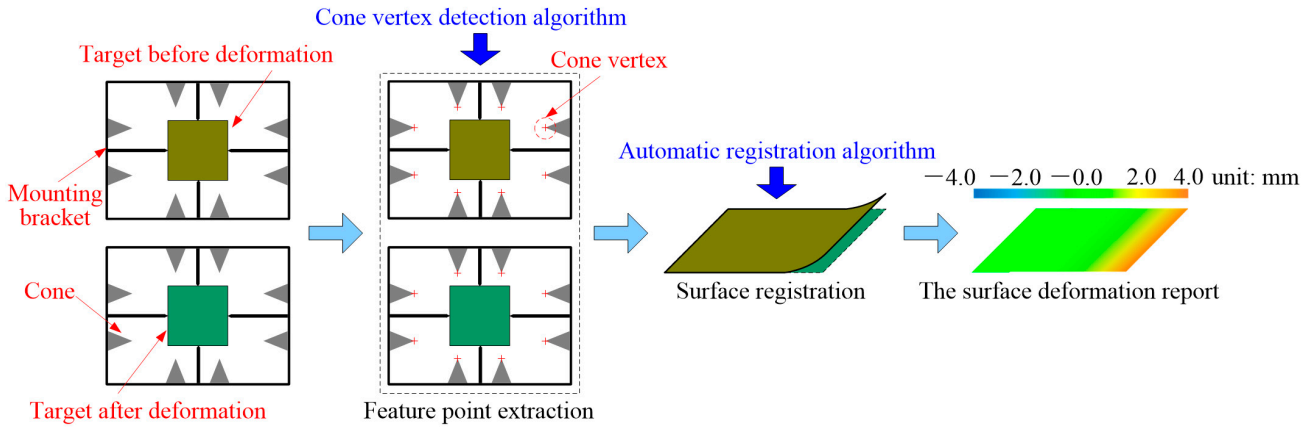


Figure 1. A flowchart of surface deformation measurement using the proposed method.

2. Cone Vertex Detection

As shown in Figure 2, the perception of geometric features in a 3D scene for human beings relies on multi-view observation and data fusion, which confer the advantages of high detection accuracy and robustness to noise. The detection process is executed in several steps. Firstly, a target is independently observed from different viewpoints. Its 3D point clouds are projected onto the retina, and corresponding 2D images are generated. Secondly, the features of interest in each 2D image are extracted. All 2D feature points are reprojected onto the surface of the target, and the corresponding 3D feature points are obtained. Thirdly, all 3D feature points observed from different viewpoints are fused, and the fake 3D feature points (e.g., fake vertices) are deleted according to the analysis made by the brain. Lastly, the target's corresponding geometric shape is fitted using the neighboring point cloud of each 3D feature point on the surface of the target in order to improve the precision of the detected 3D feature points.

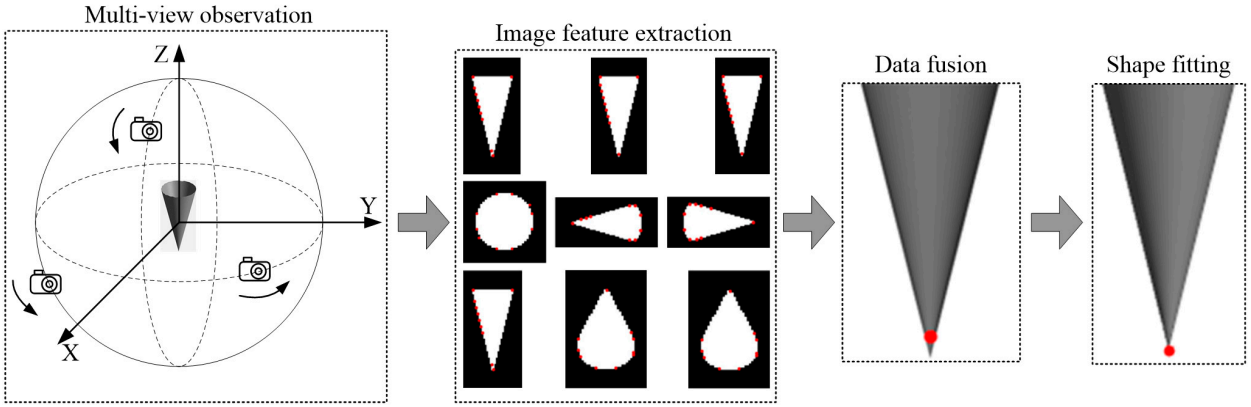


Figure 2. The principles of cone vertex detection based on biological vision.

2.1. Multi-View Observation

Since the measured space is three-dimensional, in order to carry out the all-round observation of a target, three base planes need to be selected as observation planes, for example, XOY, YOZ, and ZOX planes. In each observation plane, the target can be observed from different viewpoints. Without loss of generality, the XOY observation plane can be taken as an example, as shown in Figure 3a.

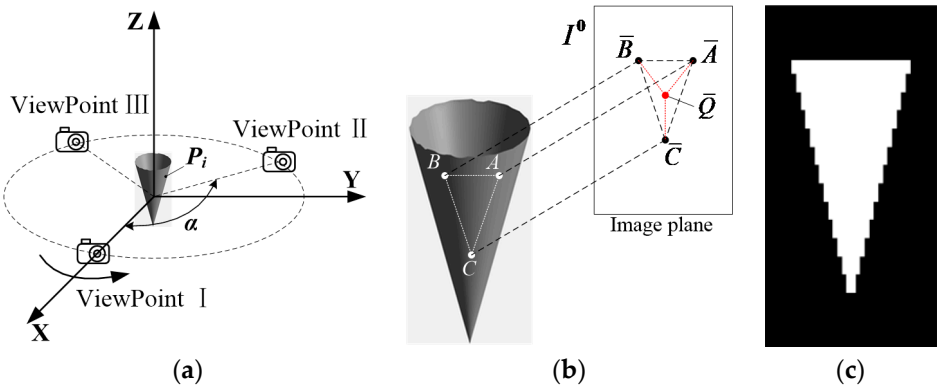


Figure 3. Multi-view observation. (**a**) Observations from different viewpoints; (**b**) image binarization under constraints; (**c**) binary image.

At viewpoint I, the observed 3D point is notated as $P_i^1 = (x_i^1, y_i^1, z_i^1)(i = 1, 2, \ldots, n-1, n)$, where n is the number of points. Its corresponding 2D projection point on the YOZ plane is notated as $\widetilde{p}_i^1 = (\widetilde{x}_i^1, \widetilde{y}_i^1)$.

$$\begin{cases} \widetilde{x}_i^1 = y_i^1 \\ \widetilde{y}_i^1 = z_i^1 \end{cases} \tag{1}$$

The center point of the 2D point cloud is notated as $\widetilde{p}_c^1 = (\widetilde{x}_c^1, \widetilde{y}_c^1)$.

$$\widetilde{p}_c^1 = \frac{\sum_{i=1}^{n} \widetilde{p}_i^1}{n}. \tag{2}$$

The width w_1 and height h_1 of its corresponding minimum enclosing rectangle (MER) are calculated as follows:

$$\begin{cases} w_1 = \max\limits_{i=1,2,\ldots,n-1,n} (\widetilde{x}_i^1) - \min\limits_{i=1,2,\ldots,n-1,n} (\widetilde{x}_i^1) \\ h_1 = \max\limits_{i=1,2,\ldots,n-1,n} (\widetilde{y}_i^1) - \min\limits_{i=1,2,\ldots,n-1,n} (\widetilde{y}_i^1) \end{cases}. \tag{3}$$

In the YOZ plane, the imaging process of the retina can be simulated to generate a binary image I^1 with width W_1 and height H_1 as:

$$\mathrm{v}\begin{cases} W_1 = kw_1 \\ H_1 = kh_1 \end{cases}, k \geq 1, \tag{4}$$

where k is a zoom coefficient.

Then, the pixel coordinate of the center point $I_c^1 = \left(\overline{x}_c^1, \overline{y}_c^1\right)$ of the image can be calculated as follows:

$$\begin{cases} \overline{x}_c^1 = (W_1 - 1)/2 \\ \overline{y}_c^1 = (H_1 - 1)/2 \end{cases}. \tag{5}$$

Then, the 2D point cloud $\widetilde{p}_i^1$ can be shifted to the image pixel coordinate system as:

$$\overline{p}_i^1 = \widetilde{p}_i^1 - \widetilde{p}_c^1 + I_c^1, \tag{6}$$

where $\overline{p}_i^1$ is the corresponding point of $\widetilde{p}_i^1$ in image I^1. If the grayscale of $\overline{p}_i^1$ is directly set to 255 and that of the other pixels is set to 0, the target area in the binary image becomes only a set of discrete pixels but not a connected domain, which cannot be used to detect corners.

As is shown in Figure 3b, A, B, and C are three vertices of an arbitrary triangle patch $T_j(j = 1, 2, \ldots, t-1, t)$ on the target surface, and its corresponding triangle in the image is notated as $\overline{T}_j$, where t is the number of triangle patches. Both T_j and $\overline{T}_j$ are connected domains. The coordinates of A, B, and C are notated as P_A^1, P_B^1, and P_C^1, respectively. The coordinates of $\overline{A}$, $\overline{B}$, and $\overline{C}$ in the image are notated as $\overline{p}_A^1$, $\overline{p}_B^1$, and $\overline{p}_C^1$, respectively. $\overline{Q}_k$ is an arbitrary pixel in image I^1, and its coordinates are notated as $\overline{q}_k^1$, where $(k = 1, 2, \ldots, m-1, m)$, and m is the number of pixels in image I^1. Then, the following vectors can be obtained:

$$\begin{cases} \overline{Q}_k\overline{A} = \overline{p}_A^1 - \overline{q}_k^1 \\ \overline{Q}_k\overline{B} = \overline{p}_B^1 - \overline{q}_k^1 \\ \overline{Q}_k\overline{C} = \overline{p}_C^1 - \overline{q}_k^1 \end{cases}. \tag{7}$$

The sum of angles between vectors is notated as β_j, as:

$$\begin{aligned} \beta_j &= \angle\overline{A}\overline{Q}_k\overline{B} + \angle\overline{B}\overline{Q}_k\overline{C} + \angle\overline{C}\overline{Q}_k\overline{A} \\ &= \arccos\left(\frac{\overline{Q}_k\overline{A} \times \overline{Q}_k\overline{B}}{|\overline{Q}_k\overline{A}||\overline{Q}_k\overline{B}|}\right) + \arccos\left(\frac{\overline{Q}_k\overline{B} \times \overline{Q}_k\overline{C}}{|\overline{Q}_k\overline{B}||\overline{Q}_k\overline{C}|}\right) + \arccos\left(\frac{\overline{Q}_k\overline{C} \times \overline{Q}_k\overline{A}}{|\overline{Q}_k\overline{C}||\overline{Q}_k\overline{A}|}\right) \end{aligned} \tag{8}$$

If $\overline{Q}_k$ is inside $\overline{T}_j$ or lying exactly on the border of $\overline{T}_j$, then β_j must be equal to 360°. On the basis of this, the image binarization under geometric constraints can be executed according to the following equation:

$$I^1(\overline{Q}_k) = \begin{cases} 255, \ if \ \min\limits_{j=1,2,\ldots,t-1,t}\left(\left|\beta_j - 360\right|\right) = 0 \\ 0, \ if \ \min\limits_{j=1,2,\ldots,t-1,t}\left(\left|\beta_j - 360\right|\right) \neq 0 \end{cases}. \quad (9)$$

The function $I^1(\overline{Q}_k) = g$ means that the grayscale of the pixel $\overline{Q}_k$ in image I^1 is set to g. Figure 3c shows the result of image binarization under geometric constraints, where the first observation has been completed.

As shown in Figure 3a, the point cloud $P_i^1 = \left(x_i^1, y_i^1, z_i^1\right)$ can be rotated by α degrees around the Z-axis; then, the point cloud $P_i^2 = \left(x_i^2, y_i^2, z_i^2\right)(i = 1, 2, \ldots, n-1, n)$ observed at position II can be obtained according to Equation (10) as:

$$P_i^2 = R_Z(\alpha)P_i^1. \quad (10)$$

From Equations (1)–(9), the simulated image I^2 at position II can be generated. In the same way, all simulated images $I^u(u = 1, 2, \ldots, s-1, s)$ can be obtained, where s is an integer obtained as:

$$s = \frac{360}{\alpha}. \quad (11)$$

When the XOY, YOZ, and ZOX planes are taken as the observation planes, then $3s$ simulated binary images $I^u(u = 1, 2, \ldots, 3s-1, 3s)$ can be achieved in total.

2.2. Cone Vertex Recognition

The Harris operator can be used to detect corners in all simulated images $I^u(u = 1, 2, \ldots, 3s-1, 3s)$. $\overline{Q}_v^u(u = 1, 2, \ldots, 3s-1, 3s)(v = 1, 2, \ldots, r_u-1, r_u)$ represents the detected v-th corner in image I^u. Its pixel coordinates are notated as $\overline{q}_v^u$, where r_u is the number of all detected corners in image I^u. Then, its corresponding point $\widetilde{q}_v^u$ in 2D space can be calculated according to Equation (12) as:

$$\widetilde{q}_v^u = \overline{q}_v^u - I_c^u + \widetilde{p}_c^u. \quad (12)$$

The closest point $\widetilde{p}_{h_v^u}^u$ can be searched for $\widetilde{q}_v^u$ in the 2D point cloud.

$$h_v^u = \min_{i=1,2,\ldots,n-1,n} dis(\widetilde{q}_v^u, \widetilde{p}_i^u), \quad (13)$$

where h_v^u is the index of $\widetilde{p}_{h_v^u}^u$, in 2D point clouds.

Since the corresponding relationship of the points remains unchanged during the projection process from a 3D point cloud to a 2D point cloud, the corresponding point in the 3D point cloud of $\widetilde{q}_v^u$ can be written as $P_{h_v^u}^1$.

The clustering of $\left\{P_{h_v^u}^1\right\}$ is executed on the basis of Euclidean distance according to the following steps:

(a) η is notated as the number of categories.

$$\eta = \sum_{u=1}^{3s} r_u + 1 \quad (14)$$

r_u is the number of all detected corners in image I^u, $3s$ is the number of all simulated images.

(b) d_{ij} represents the distance between the centers of C_i and C_j. C_i and C_j is the i-th and j-th category. If the minimum of $\{d_{ij}\}$ is smaller than the distance threshold λ, cluster $\left\{P_{h_v^u}^1\right\}$ into η-1 categories and $\eta = \eta$-1.

(c) Repeat step (b) until the minimum of $\{d_{ij}\}$ is equal to or greater than λ.
(d) The coordinate of each clustering center can be obtained by calculating the mean value of members in its corresponding category.

ψ_i represents the number of members of C_i. Since the cone vertex is a robust feature point, it should be observed from the most viewpoints. Thus, all values of C_i that satisfy $\psi_i \leq \kappa$ should be deleted, as shown in Figure 4c, where κ is a threshold set for the number of observations. Figure 4d shows the rough localization result of a cone vertex. The detected cone vertex is notated as $\chi_i (i = 1, 2, \ldots, \rho - 1, \rho)$, where ρ is the number of all detected cone vertices.

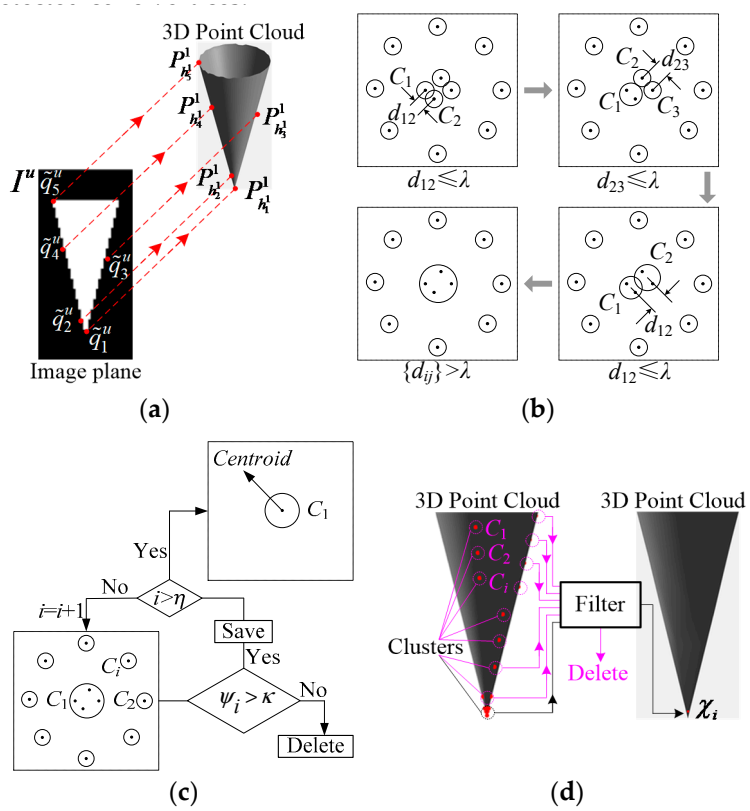


Figure 4. The principles of cone vertex recognition. (**a**) Corner detection; (**b**) clustering of feature points; (**c**) data filtering; (**d**) rough localization.

2.3. Shape Fitting

Without loss of generality, χ_1 can be taken as an example. In this case, $P_i^1 = (x_i^1, y_i^1, z_i^1)(i = 1, 2, \ldots, \xi - 1, \xi)$ represents the neighboring point cloud of χ_1, where ξ is the number of neighboring points. The quadratic form of the conic surface can be written as:

$$a_1\left(x_i^1\right)^2 + a_2\left(y_i^1\right)^2 + a_3\left(z_i^1\right)^2 + a_4 x_i^1 y_i^1 + a_5 x_i^1 z_i^1 + a_6 y_i^1 z_i^1 + a_7 x_i^1 + a_8 y_i^1 + a_9 z_i^1 + a_{10} = 0 \quad (15)$$

Then, Equation (14) can be rewritten in matrix form, where

$$E\varphi = 0. \quad (16)$$

The matrices of E and φ are as follows:

$$E = \begin{pmatrix} (x_i^1)^2 & (y_i^1)^2 & (z_i^1)^2 & x_i^1 y_i^1 & x_i^1 z_i^1 & y_i^1 z_i^1 & x_i^1 & y_i^1 & z_i^1 & 1 \\ (x_i^1)^2 & (y_i^1)^2 & (z_i^1)^2 & x_i^1 y_i^1 & x_i^1 z_i^1 & y_i^1 z_i^1 & x_i^1 & y_i^1 & z_i^1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (x_i^1)^2 & (y_i^1)^2 & (z_i^1)^2 & x_i^1 y_i^1 & x_i^1 z_i^1 & y_i^1 z_i^1 & x_i^1 & y_i^1 & z_i^1 & 1 \\ (x_i^1)^2 & (y_i^1)^2 & (z_i^1)^2 & x_i^1 y_i^1 & x_i^1 z_i^1 & y_i^1 z_i^1 & x_i^1 & y_i^1 & z_i^1 & 1 \end{pmatrix}$$

$$\varphi = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} \end{pmatrix}^T$$

E can be decomposed using a singular value decomposition (SVD)

$$E = U_E D_E V_E^{\mathrm{T}}. \tag{17}$$

Then, φ becomes the last column of V_E. The quadratic form of the conic surface can be rewritten in matrix form as:

$$\begin{pmatrix} x_i^1 & y_i^1 & z_i^1 & 1 \end{pmatrix} F \begin{pmatrix} x_i^1 & y_i^1 & z_i^1 & 1 \end{pmatrix}^{\mathrm{T}} = 0, \tag{18}$$

where

$$F = \begin{pmatrix} a_1 & a_4/2 & a_5/2 & a_7/2 \\ a_4/2 & a_2 & a_6/2 & a_8/2 \\ a_5/2 & a_6/2 & a_3 & a_9/2 \\ a_7/2 & a_8/2 & a_9/2 & a_{10} \end{pmatrix}$$

F can be decomposed using an SVD.

$$F = U_F D_F V_F^{\mathrm{T}}. \tag{19}$$

The homogeneous coordinates of the cone vertex are represented by the last column of V_F (see proof in Appendix A) and notated as $\begin{bmatrix} v_1 & v_2 & v_3 & v_4 \end{bmatrix}^{\mathrm{T}}$. Thus, the χ_1' coordinates of the cone vertex are as follows:

$$\chi_1' = \left(\frac{v_1}{v_4}, \frac{v_2}{v_4}, \frac{v_3}{v_4} \right). \tag{20}$$

In the same way, all coordinates of cone vertices $\chi_1' = (x_i, y_i, z_i)(i = 1, 2, \ldots, \rho - 1, \rho)$ can be obtained.

3. Automatic Registration Algorithm

$\{{}^1\chi_i'\}(i = 1, 2, \ldots, \rho_1 - 1, \rho_1)$ and $\{{}^2\chi_i'\}(i = 1, 2, \ldots, \rho_2 - 1, \rho_2)$ are notated as the cone vertices in the reference and test point clouds, respectively. Since the corresponding relationship is unknown, $\{{}^1\chi_i'\}$ and $\{{}^2\chi_i'\}$ cannot be directly used to calculate the transformation matrix between the reference and test point clouds. The transformation matrix consists of a 3×3 rotation matrix R and a 3×1 translation vector T. To solve the corresponding relationship and improve the robustness of the algorithm, a random strategy is used to calculate the corresponding relationship between $\{{}^1\chi_i'\}$ and $\{{}^2\chi_i'\}$.

Two sequences of numbers are constructed in which the probability of each element obeys a mean distribution.

$${}^1\Gamma = [1, 2, \ldots, \rho_1 - 1, \rho_1]$$
$${}^2\Gamma = [1, 2, \ldots, \rho_2 - 1, \rho_2]$$

Three different elements ${}^1\tau_j(j=1,2,3)$ and ${}^2\tau_j(j=1,2,3)$ are taken from ${}^1\Gamma$ and ${}^2\Gamma$, respectively. The corresponding cone vertices are as follows:

$$\begin{cases} {}^1\chi'_{{}^1\tau_j} = \left(x_{{}^1\tau_j}, y_{{}^1\tau_j}, z_{{}^1\tau_j}\right) \\ {}^2\chi'_{{}^2\tau_j} = \left(x_{{}^2\tau_j}, y_{{}^2\tau_j}, z_{{}^2\tau_j}\right) \end{cases}.$$

The transformation between ${}^1\chi'_{{}^1\tau_j}$ and ${}^2\chi'_{{}^2\tau_j}$ can be written as Equation (21) as:

$$\left({}^2\chi'_{{}^2\tau_j}\right)^{\mathrm{T}} = R \times \left({}^1\chi'_{{}^1\tau_j}\right)^{\mathrm{T}} + T\,. \tag{21}$$

The matrices ${}^1M$ and ${}^2M$ can be constructed using $\left\{{}^1\chi'_{{}^1\tau_j}\right\}$ and $\left\{{}^2\chi'_{{}^2\tau_j}\right\}$, respectively, as:

$${}^1M = \begin{pmatrix} {}^1\chi'_{{}^1\tau_1} \\ {}^1\chi'_{{}^1\tau_2} \\ {}^1\chi'_{{}^1\tau_3} \end{pmatrix} = \begin{pmatrix} x_{{}^1\tau_1} & y_{{}^1\tau_1} & z_{{}^1\tau_1} \\ x_{{}^1\tau_2} & y_{{}^1\tau_2} & z_{{}^1\tau_2} \\ x_{{}^1\tau_3} & y_{{}^1\tau_3} & z_{{}^1\tau_3} \end{pmatrix}, {}^2M = \begin{pmatrix} {}^2\chi'_{{}^2\tau_1} \\ {}^2\chi'_{{}^2\tau_2} \\ {}^2\chi'_{{}^2\tau_3} \end{pmatrix} = \begin{pmatrix} x_{{}^2\tau_1} & y_{{}^2\tau_1} & z_{{}^2\tau_1} \\ x_{{}^2\tau_2} & y_{{}^2\tau_2} & z_{{}^2\tau_2} \\ x_{{}^2\tau_3} & y_{{}^2\tau_3} & z_{{}^2\tau_3} \end{pmatrix}. \tag{22}$$

Their center points can be calculated using Equation (23) as follows:

$$\begin{cases} {}^1M_c = \dfrac{\sum_{j=1}^{3} {}^1\chi'_{{}^1\tau_j}}{3} \\ {}^2M_c = \dfrac{\sum_{j=1}^{3} {}^2\chi'_{{}^1\tau_j}}{3} \end{cases}. \tag{23}$$

The origin of the coordinate system can be shifted to the center points of ${}^1M$ and ${}^2M$ as:

$$\begin{cases} {}^1\overline{M} = {}^1M - {}^1M_c \\ {}^2\overline{M} = {}^2M - {}^2M_c \end{cases}. \tag{24}$$

Then, there only exists rotation transformation between ${}^1\overline{M}$ and ${}^2\overline{M}$, expressed as:

$${}^2\overline{M} = R \times \left({}^1\overline{M}\right). \tag{25}$$

Matrix $\Omega = \left({}^2\overline{M}\right)^T \left({}^1\overline{M}\right)$ can be decomposed using an SVD as follows:

$$\Omega = U_\Omega D_\Omega V_\Omega^{\mathrm{T}}. \tag{26}$$

$U_\Omega(i)(i=1,2,3)$ represents the i-th column of U_Ω; thus, R must be one of the following eight forms:

$$\begin{cases} R_1 = \begin{bmatrix} U_\Omega(1) & U_\Omega(2) & U_\Omega(3) \end{bmatrix} V_\Omega^{\mathrm{T}} \\ R_2 = \begin{bmatrix} U_\Omega(1) & U_\Omega(2) & -U_\Omega(3) \end{bmatrix} V_\Omega^{\mathrm{T}} \\ R_3 = \begin{bmatrix} U_\Omega(1) & -U_\Omega(2) & U_\Omega(3) \end{bmatrix} V_\Omega^{\mathrm{T}} \\ R_4 = \begin{bmatrix} -U_\Omega(1) & U_\Omega(2) & U_\Omega(3) \end{bmatrix} V_\Omega^{\mathrm{T}} \end{cases}, \begin{cases} R_5 = \begin{bmatrix} -U_\Omega(1) & -U_\Omega(2) & U_\Omega(3) \end{bmatrix} V_\Omega^{\mathrm{T}} \\ R_6 = \begin{bmatrix} -U_\Omega(1) & U_\Omega(2) & -U_\Omega(3) \end{bmatrix} V_\Omega^{\mathrm{T}} \\ R_7 = \begin{bmatrix} U_\Omega(1) & -U_\Omega(2) & -U_\Omega(3) \end{bmatrix} V_\Omega^{\mathrm{T}} \\ R_8 = \begin{bmatrix} -U_\Omega(1) & -U_\Omega(2) & -U_\Omega(3) \end{bmatrix} V_\Omega^{\mathrm{T}} \end{cases}.$$

The rotation error is defined as follows:

$$e_i = \frac{\sqrt{trace\left(\left[{}^2\overline{M} - R_i \times \left({}^1\overline{M}\right)\right] \times \left[{}^2\overline{M} - R_i \times \left({}^1\overline{M}\right)\right]^{\mathrm{T}}\right)}}{3} + \zeta_p \times \left(\left|R_i^{\mathrm{T}} R_i\right| - 1\right). \tag{27}$$

The first item in Equation (27) is the data term used to represent the geometric error of rotation estimation; the second item in Equation (27) is the constraint term used to limit the coordinate system to being a right-handed coordinate system; $trace(\cdot)$ is a function used to calculate the sum of the diagonal elements of the matrix; whereas ζ_p is a penalty coefficient. Therefore,

$$\begin{cases} R = \min\limits_{R_i}(e_i) \\ T = {}^2M_c - R \times {}^1M_c \\ e_{\min} = \min\limits_{i=1,2,\ldots,7,8}(e_i) \end{cases}. \tag{28}$$

The threshold of the rotation error is notated as ε_e. If $e_{\min} < \varepsilon_e$, it indicates that the cone vertices in ${}^1M$ and ${}^2M$ are corresponding points and the solution of rotation and translation is credible. Otherwise, the following steps are repeated until $e_{\min} < \varepsilon_e$ or the number of iterations is greater than a threshold N_{max}: (a) take three different elements from ${}^1\Gamma$ and ${}^2\Gamma$, respectively; (b) calculate R, T, and e_{min} according to Equations (22)–(28). Since the probability of each element obeys a mean distribution, the value of N_{max} can be calculated using Equation (29) as:

$$N_{\max} = \frac{A_{\rho_1}^3 A_{\rho_2}^3}{A_3^3} = \frac{\rho_1\rho_2(\rho_1-1)(\rho_1-2)(\rho_2-1)(\rho_2-2)}{6}. \tag{29}$$

The corresponding point between $\{{}^1\chi_i'\}$ and $\{{}^2\chi_i'\}$ is notated as $\{{}^1\hat{\chi}_i', {}^2\hat{\chi}_i'\}(i = 1, 2, \ldots, n_c - 1, n_c)$; thus, $\{{}^1\hat{\chi}_i', {}^2\hat{\chi}_i'\}$ should satisfy Equation (29) as:

$$\left\| R \times \left({}^1\hat{\chi}_i'\right)^{\mathrm{T}} + T - {}^2\hat{\chi}_i' \right\|_2 \le \delta_d, \tag{30}$$

where δ_d is the distance threshold.

$\left\{{}^1P_j\right\}(j = 1, 2, \ldots, n_1 - 1, n_1)$ represents the neighboring point cloud of $\{{}^1\hat{\chi}_i'\}$ in the reference point cloud. $\left\{{}^2P_j\right\}(j = 1, 2, \ldots, n_2 - 1, n_2)$ represents the neighboring point cloud of $\{{}^2\hat{\chi}_i'\}$ in the test point cloud. On the above basis, taking the solution of Equation (28) as the initial value, the rotation matrix $\widetilde{R}$ and translation vector $\widetilde{T}$ can be solved using the ICP algorithm. Figure 5 shows the flowchart of the automatic registration algorithm.

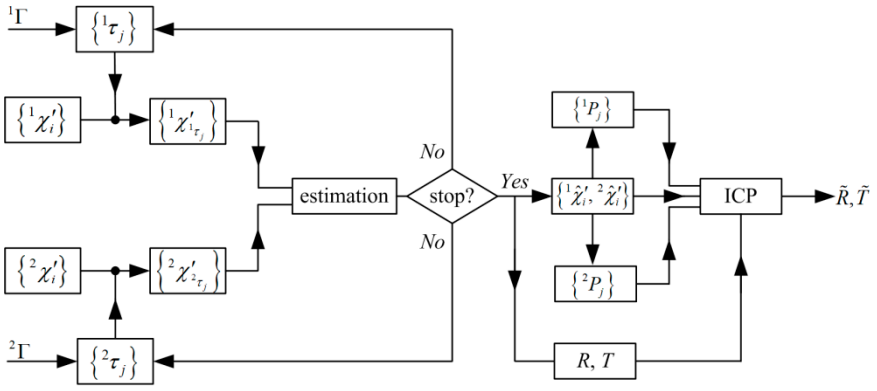


Figure 5. Flowchart of the automatic registration algorithm.

4. Research Methodology

4.1. Cone Vertex Detection

The algorithm's robustness and accuracy can be evaluated by the detection rate S_D and location error E_D under Gaussian noise with different intensities. S_D is defined as follows:

$$S_D = \frac{N_D}{N_T}, \tag{31}$$

where N_D is the number of detected cone vertices and N_T is the total number of cone vertice. E_D is defined as follows:

$$E_D = \sqrt{\frac{\sum_{i=1}^{N_D} E_i^2}{N_D}}, \tag{32}$$

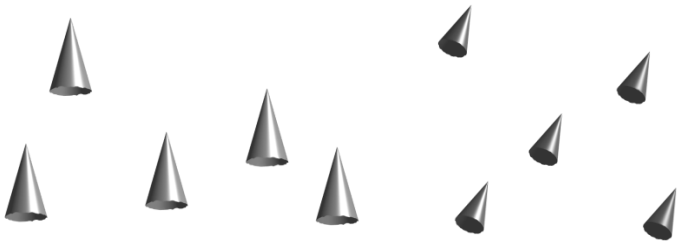
where E_i represents the location error of the i-th cone vertex.

A cone model provided by [21] was adopted to test the algorithm performance. To test the influence of noise intensity on the detection of cone vertices, Gaussian noise was independently added to the X-, Y-, and Z-axes of each scene point. The standard deviation σ_{GN} of Gaussian noise ranged from 0 to 1 mr with a step size of 0.1 mr, where mr denotes the average mesh resolution of the models.

4.2. Automatic Registration

As shown in Figure 6, the reference and test point clouds consisted of five cones. There exists a rigid transformation between the reference and test point clouds, as expressed in Equation (21). The ideal rotation matrix and translation vector are notated as $\hat{R}$ and $\hat{T}$, respectively. The calculated rotation matrix and translation vector are notated as $\widetilde{R}$ and $\widetilde{T}$, respectively. Rotation and translation errors under Gaussian noise with different intensities were used to evaluate the algorithm's robustness and accuracy. The rotation error ε_r is defined as follows:

$$\varepsilon_r = \arccos\left(\frac{trace\left(\widetilde{R}\hat{R}^{\mathrm{T}}\right) - 1}{2}\right)\frac{180}{\pi}. \tag{33}$$



(**a**) (**b**)

Figure 6. Point cloud registration. (**a**) Reference point cloud; (**b**) test point cloud.

The translation error ε_T is defined as follows:

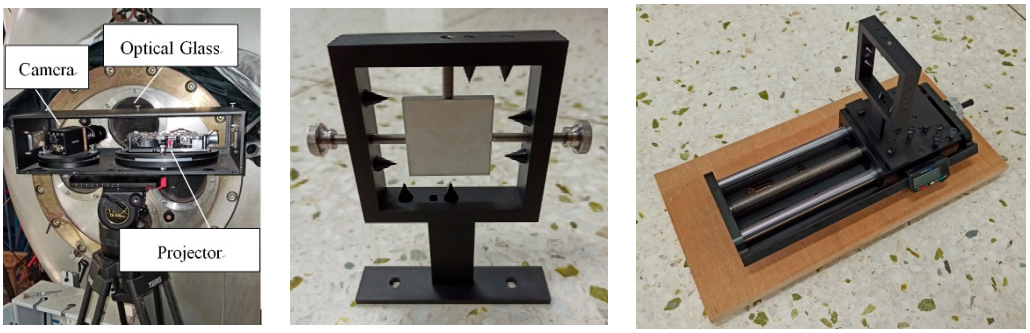
$$\varepsilon_T = \frac{\|\widetilde{T} - \hat{T}\|}{d_{res}}, \tag{34}$$

where d_{res} is equal to the average mesh resolution (d_{res} = 1 mr). To test the influence of Gaussian noise with different intensities on registration performance, Gaussian noise was independently added to the X-, Y-, and Z-axes of each scene point. The standard deviation and step size were the same as in Section 4.1. In the experiment, the corresponding Euler angle of $\hat{R}$ was [40°, 40°, 40°], and the rotation sequence was "XYZ". $\hat{T}$ was [237 mr, 166 mr, −144 mr].

4.3. Surface Deformation Measurement

A camera-projector system based on the proposed method, shown in Figure 7a, was developed to measure the surface deformation dynamically during ablation tests in an arc-heated wind tunnel. The resolutions of the projector and camera images were 1280 × 800 px and 2456 × 2058 px, respectively. The sizes of CCD and CCD pixel were 2/3" and 3.45 × 3.45 μm, respectively. The focal length was 75 mm. The data processing

platform comprised a Dell laptop with an Intel(R) Core(TM) i7-6700HQ CPU @ 2.6 GHz and 16 GB RAM. The aim was to evaluate the system's precision. Firstly, a model with the size of 40 × 40 × 5 mm was fixed to a special device which had six cones, as shown in Figure 7b. Secondly, the device was placed on a translation stage, as shown in Figure 7c. The translation stage's precision was 0.01 mm. Thirdly, surface ablation could be simulated by moving the model on the translation stage. The camera-projector system was used to reconstruct the model surface and the six cones at time 0 and time k, denoting the reference point cloud and test point cloud, respectively. Fourthly, the reference and test point clouds were aligned using the proposed registration method, and then the model's surface deformation could be measured. Lastly, the measurement result was compared with the reading from the translation stage, and the measurement error of the camera-projector system was calculated. Root-mean-square (RMS) was introduced to evaluate the above measurement error.



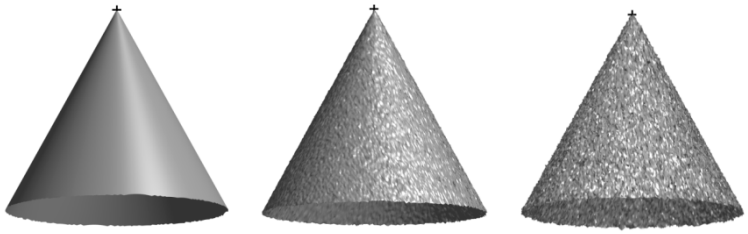
(**a**) (**b**) (**c**)

Figure 7. Experimental equipment. (**a**) Camera-projector system; (**b**) Model and special device; (**c**) Translation stage.

5. Research Results

5.1. Cone Vertex Detection

Figure 8 shows the point clouds of cone models where Gaussian noise was added with different intensities. Here, "+" represents the detected position of the cone vertex. Table 1 shows the statistical results of cone vertex detection.



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

Figure 8. The detection results. (**a**) σ_{GN} = 0 mr; (**b**) σ_{GN} = 0.5 mr; (**c**) σ_{GN} = 1 mr.

Table 1. The statistical results of cone vertex detection (mr denotes the average mesh resolution of the models).

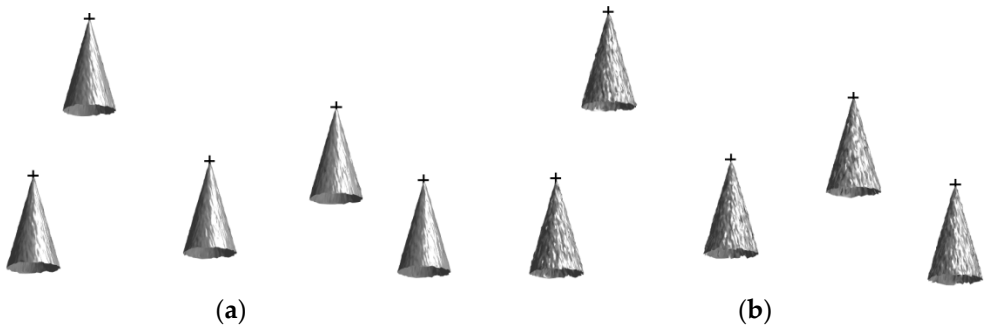
Noise Intensity (mr)	Location Error (mr)	Detection Rate
0.0	0.00	100%
0.1	0.16	100%
0.2	0.27	100%
0.3	0.45	100%

Table 1. *Cont.*

Noise Intensity (mr)	Location Error (mr)	Detection Rate
0.4	0.75	100%
0.5	1.24	100%
0.6	1.56	100%
0.7	1.85	100%
0.8	2.29	100%
0.9	2.58	100%
1.0	3.13	97%

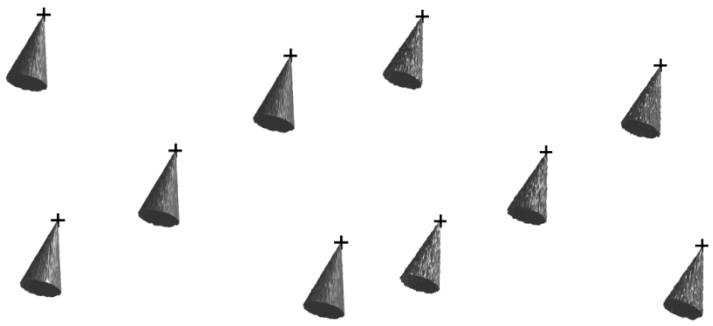
5.2. Automatic Registration

Figures 9 and 10 show the reference and test point clouds, respectively, where Gaussian noise was added with different intensities. Here, "+" represents the detected position of the cone vertex. Figure 11 shows the relationship between registration error and noise intensity.



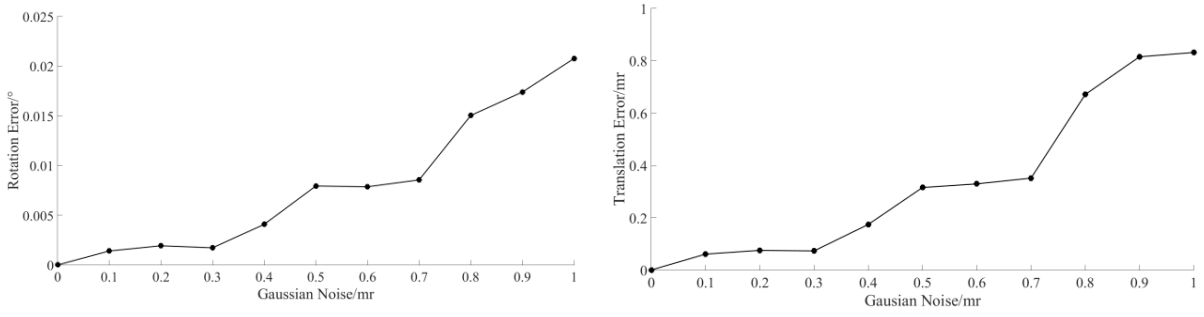
(**a**) (**b**)

Figure 9. Reference point cloud. (**a**) σ_{GN} = 0.5 mr; (**b**) σ_{GN} = 1 mr.



(**a**) (**b**)

Figure 10. Test point cloud. (**a**) σ_{GN} = 0.5 mr; (**b**) σ_{GN} = 1 mr.

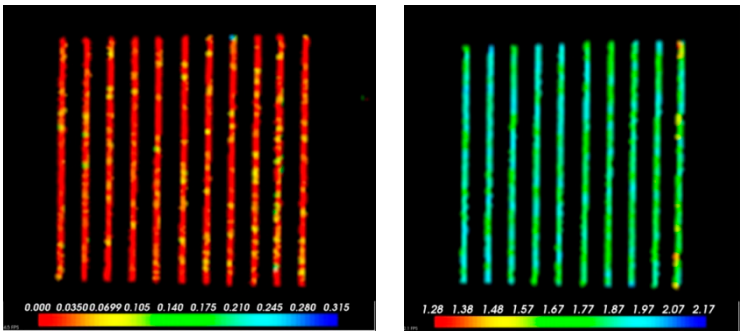


(**a**) (**b**)

Figure 11. The relationship between registration error and noise intensity. (**a**) Rotation error; (**b**) translation error. Note, mr denotes the average mesh resolution of the models.

5.3. Surface Deformation Measurement

Figure 12 shows the model's surface deformation at different times. Table 2 shows the comparison results of readings and measurement results.



(**a**) (**b**)

Figure 12. Surface deformation measurement results. (**a**) 0s, reading = 0.000 mm; (**b**) 9s, reading = 1.810 mm.

Table 2. Statistical results.

Time (s)	Reading (mm)	Measurement Result (mm)	Error (mm)
0.0	0.000	0.035	0.035
0.5	0.100	0.109	0.009
1.0	0.200	0.196	0.004
1.5	0.300	0.300	0.000
2.0	0.400	0.402	0.002
2.5	0.500	0.492	0.008
3.0	0.600	0.605	0.005
3.5	0.700	0.703	0.003
4.0	0.800	0.795	0.005
4.5	0.900	0.903	0.003
5.0	1.000	1.008	0.008
5.5	1.100	1.106	0.006
6.0	1.200	1.210	0.010
6.5	1.300	1.320	0.020
7.0	1.400	1.433	0.033
7.5	1.500	1.522	0.022
8.0	1.600	1.620	0.020
8.5	1.710	1.742	0.032
9.0	1.810	1.843	0.033
9.5	1.910	1.952	0.042
10.0	2.010	2.056	0.046
12.5	2.500	2.537	0.037
15.0	3.000	3.029	0.029
20.0	4.000	4.035	0.035

6. Discussion

(1) As shown in Figure 8c, when σ_{GN} = 1 mr, the conic surface was very rough, but its vertex could still be detected accurately, which fully proves the robustness of the algorithm to Gaussian noise. Table 1 shows the statistical results of the detection rate and location error under Gaussian noise with different intensities. It could be seen that the location error increased with increasing noise intensity. In general, detection of the cone vertex was successful if the location error was lower than 5 mr. Thus, the algorithm can maintain a high detection rate under Gaussian noise with intensity ranging from 0 to 1 mr.

(2) Figures 9–11, indicate that the rotation and translation errors increase with increasing noise intensity. When σ_{GN} = 1 mr, most details on the conic surfaces were lost, but the

point clouds could still be aligned accurately, which proves the robustness of the algorithm to Gaussian noise. In general, the point cloud registration was successful if the rotation error was lower than 5° and the translation error was lower than 5 mr. Thus, the algorithm performs well under Gaussian noise with an intensity ranging from 0 to 1 mr.

(3) In general, surface deformation of a near-space supersonic vehicle is no more than 4 mm during ablation tests in an arc-heated wind tunnel. Table 2 shows the statistical results, where it can be seen that the precision of surface deformation measurement exceeds 0.05 mm when surface deformation is smaller than 4 mm.

7. Conclusions

In order to solve the problem of surface deformation measurement during ablation tests in an arc-heated wind tunnel, in this study, we proposed an automatic point cloud registration method. As compared with other registration methods, we provided a complete solution, including two parts: (1) Guarantee high-quality initial values and overlaps for aligning point clouds which have deformed much. (2) A strategy to automatically compute rigid transformations between point clouds. Inspired by 2D artificial targets used to improve precision of camera calibration or photogrammetry, we introduced 3D artificial targets to obtain accurate registration results, which was the key idea for solving the problem of surface deformation measurement; most state-of-art approaches only considered how to align point clouds more accurately and robustly using a big enough overlap. Simulations and experiments were conducted, and the research results indicated the following: (1) The proposed method performed well under Gaussian noise with an intensity ranging from 0 to 1 mr. When σ_{GN} = 1 mr, rotation and translation error were smaller than 0.025° and 1 mr, respectively. (2) The error of surface deformation measurement was smaller than 0.05 mm when deformation was no more than 4 mm. In addition to surface deformation measurement, the proposed method can also be applied for experimental studying of soft matter.

However, there are still some aspects that need to be studied: (1) The procedure of cone vertex detection should be more efficient. As compared with the Harris operator used in cone vertex detection, Radon or Hough transform may be more simplified [22,23], but the robustness needs to be evaluated. (2) The 3D artificial target should be more multiple. An artificial neutral network (ANN) can be adopted to train a classifier. And rotational projection statistics (RoPS) can be taken as the inputs of the classifier [6]. The classifier can be used to recognize different geometric features and delete fake features, which can improve the efficiency of data fusion and decrease the time cost.

Author Contributions: Conceptualization, J.L.; Data curation, P.G.; Formal analysis, J.L. and X.S.; Funding acquisition, J.L.; Investigation, J.L. and P.G.; Methodology, J.L.; Supervision, X.S.; Validation, X.S.; Writing—original draft, J.L.; Writing—review and editing, P.G. and X.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Natural Science Foundation of China with grant no. 11802321.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Lemma A1. *The quadratic form of the conic surface can be written as:*

$$P^{\mathrm{T}} A P = 0, \tag{A1}$$

and:

$$A = \begin{pmatrix} a_1 & a_4/2 & a_5/2 & a_7/2 \\ a_4/2 & a_2 & a_6/2 & a_8/2 \\ a_5/2 & a_6/2 & a_3 & a_9/2 \\ a_7/2 & a_8/2 & a_9/2 & a_{10} \end{pmatrix}, P = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^{\mathrm{T}}.$$

A can be decomposed using SVD.

$$A = U_A D_A V_A^{\mathrm{T}}. \tag{A2}$$

Then, the last column of V_A represents the homogeneous coordinate of the cone vertex.

Proof. Matrix A can be diagonalized on the basis of eigenvalues and eigenvectors.

$$A = RDR^{\mathrm{T}}, \tag{A3}$$

where D is a diagonal matrix and R is a transformation matrix. Substituting Equation (A3) into Equation (34) yields:

$$P^{\mathrm{T}}AP = \left(P^{\mathrm{T}}R\right)D\left(P^{\mathrm{T}}R\right)^{\mathrm{T}} = \left(P'\right)^{\mathrm{T}}DP' = 0, \tag{A4}$$

where $\left(P'\right)^{\mathrm{T}}DP' = 0$ is the standard quadratic form of the conic surface. Equation (A4) is essentially a transformation between coordinate systems. Through this transformation, the cone vertex can be shifted to the origin of the new coordinate system, and the axis of the cone can be made parallel to the Z-axis of new coordinate system. The homogeneous coordinates of the cone vertex in the new coordinate system can be notated as:

$$P'_v = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^{\mathrm{T}}. \tag{A5}$$

Because $P' = R^{\mathrm{T}}P$, the homogeneous coordinates of the cone vertex in the original coordinate system can be calculated using Equation (A6) as follows:

$$P_v = \left(R^{\mathrm{T}}\right)^{-1}{P'}_v. \tag{A6}$$

A is a real symmetric matrix; thus, $R^{\mathrm{T}}R = \lambda I$. I is a 4×4 unit matrix; therefore,

$$A^{\mathrm{T}}A = RDR^{\mathrm{T}}RDR^{\mathrm{T}} = \lambda RD^2R^{\mathrm{T}}. \tag{A7}$$

R is a matrix consisting of eigenvectors of $A^{\mathrm{T}}A$. A can be decomposed using SVD.

$$A = U_A D_A V_A^{\mathrm{T}}. \tag{A8}$$

Thus,

$$R = V_A. \tag{A9}$$

Substituting Equation (A9) into Equation (A6) yields

$$P_v = \left(R^{\mathrm{T}}\right)^{-1}{P'}_v = V_A \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. \tag{A10}$$

Therefore, the homogeneous coordinates of the cone vertex become the last column of V_A. The lemma has been proven. □

References

1. Nguyen, T.H.; Nguyen, T.M.; Xie, L. Tightly-coupled ultra-wide band-aided monocular visual SLAM with degenerate anchor configurations. *Auton. Robot.* **2020**, *44*, 1519–1534. [CrossRef]
2. Shane, G.W.; Voorhies, R.C.; Laurent, I. Efficient velodyne SLAM with point and plane features. *Auton. Robot.* **2018**, *43*, 1207–1224.
3. Wang, F.R.; Lu, E.; Wang, Y.; Qiu, G.J.; Lu, H.Z. Efficient stereo visual simultaneous localization and mapping for an autonomous unmanned forklift in an unstructured warehouse. *Appl. Sci.* **2020**, *10*, 698. [CrossRef]
4. Lei, H.; Jiang, G.; Quan, L. Fast descriptors and correspondence propagation for robust global point cloud registration. *IEEE Trans. Image Process.* **2017**, *26*, 1. [CrossRef] [PubMed]
5. Dong, Z.; Liang, F.; Yang, B. Registration of large-scale terrestrical laser scanner point clouds: A review and benchmark. *ISPRS J. Photogramm. Remote Sens.* **2020**, *163*, 327–342. [CrossRef]
6. Guo, Y.; Sohel, F.; Bennamoun, M.; Lu, M.; Wan, J. Rotational projection statistics for 3D local surface description and object recognition. *Int. J. Comput. Vis.* **2013**, *105*, 63–86. [CrossRef]
7. Tran, D.-S.; Ho, N.-H.; Yang, H.-J.; Baek, E.-T.; Kim, S.-H.; Lee, G. Real-time hand gesture spotting and recognition using RGB-D Camera and 3D convolutional neural network. *Appl. Sci.* **2020**, *10*, 722. [CrossRef]
8. Besl, P.; McKay, N.D. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 239–256. [CrossRef]
9. Servos, J.; Waslander, S.L. Multi-Channel Generalized-ICP: A robust framework for multi-channel scan registration. *Robot. Auton. Syst.* **2017**, *87*, 247–257. [CrossRef]
10. Serafin, J.; Grisetti, G. NICP: Dense normal based point cloud registration. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–3 October 2015; pp. 742–749.
11. Attia, M.; Slama, Y. Efficient initial guess determination based on 3D point cloud projection for ICP algorithms. In Proceedings of the 2017 International Conference on High Performance Computing & Simulation (HPCS) 2017, Genoa, Italy, 17–21 July 2017; pp. 807–814.
12. Makovetskii, A.; Voronin, S.; Kober, V.; Tihonkih, D. An efficient point-to-plane registration algorithm for affine transformation. In *Applications of Digital Image Processing XL*; SPIE: Bellingham, WA, USA, 2017.
13. Li, P.; Wang, R.; Wang, Y.; Tao, W. Evaluation of the ICP algorithm in 3D point cloud registration. *IEEE Access* **2020**, *8*, 68030–68048. [CrossRef]
14. Wu, P.; Li, W.; Yan, M. 3D scene reconstruction based on improved ICP algorithm. Microprocess. *Microsystems* **2020**, *75*, 103064. [CrossRef]
15. Kang, Z.; Yang, J. A probabilistic graphical model for the classification of mobile LiDAR point clouds. *ISPRS J. Photogramm. Remote Sens.* **2018**, *143*, 108–123. [CrossRef]
16. Hong, H.; Lee, B.H. Probabilistic normal distributions transform representation for accurate 3D point cloud registration. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2017, Vancouver, BC, Canada, 24–28 September 2017; pp. 3333–3338. [CrossRef]
17. Zhu, H.; Guo, B.; Zou, K.; Li, Y.; Yuen, K.-V.; Mihaylova, L.; Leung, H. A review of point set registration: From pairwise registration to groupwise registration. *Sensors* **2019**, *19*, 1191. [CrossRef] [PubMed]
18. Lu, M.; Zhao, J.; Guo, Y.; Ma, Y. Accelerated coherent point drift for automatic 3D point cloud registration. *IEEE GRSL* **2016**, *13*, 162–166.
19. Wang, G.; Chen, Y. Fuzzy correspondences guided Gaussian mixture model for point set registration. *Knowl. Based Syst.* **2017**, *136*, 200–209. [CrossRef]
20. Ma, J.; Jiang, J.; Liu, C.; Li, Y. Feature guided Gaussian mixture model with semi-supervised EM and local geometric constraint for retinal image registration. *Inf. Sci.* **2017**, *417*, 128–142. [CrossRef]
21. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3D ShapeNets: A deep representation for volumetric shapes. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.
22. Polyakova, A.P.; Svetov, I.E. On a singular value decomposition of the normal Radon transform operator acting on 3D 2-tensor fields. *J. Phys. Conf. Ser.* **2021**, *1715*, 012041. [CrossRef]
23. Wang, Y.S.; Qi, Y.; Man, Y. An improved hough transform method for detecting forward vehicle and lane in road. *J. Phys. Conf. Ser.* **2021**, *1757*, 012082. [CrossRef]