FITEE

Three-dimensional face point cloud hole-filling algorithm based on binocular stereo matching and a B-spline[*]

Yuan HUANG[†‡1,2], Feipeng DA[1,2]

1School of Automation, Southeast University, Nanjing 210096, China
2Shenzhen Research Institute of Southeast University, Shenzhen 518000, China
[†]E-mail: whhbb@163.com

Abstract: When obtaining three-dimensional (3D) face point cloud data based on structured light, factors related to the environment, occlusion, and illumination intensity lead to holes in the collected data, which affect subsequent recognition. In this study, we propose a hole-filling method based on stereo-matching technology combined with a B-spline. The algorithm uses phase information acquired during raster projection to locate holes in the point cloud, simultaneously extracting boundary point cloud sets. By registering the face point cloud data using the stereo-matching algorithm and the data collected using the raster projection method, some supplementary information points can be obtained at the holes. The shape of the B-spline curve can then be roughly described by a few key points, and the control points are put into the hole area as key points for iterative calculation of surface reconstruction. Simulations using smooth ceramic cups and human face models showed that our model can accurately reproduce details and accurately restore complex shapes on the test surfaces. Simulation results indicated the robustness of the method, which is able to fill holes on complex areas such as the inner side of the nose without a prior model. This approach also effectively supplements the hole information, and the patched point cloud is closer to the original data. This method could be used across a wide range of applications requiring accurate facial recognition.

1 Introduction

Computer vision technology is one of the key core technologies of artificial intelligence (Russell et al., 2010). Meanwhile, visual knowledge (Pan, 2019) represents knowledge in a new form, which can be applied in the fields of security, finance, hardware, marketing, automobile design, and medicine.

Within the field of computer vision, facial recognition algorithms based on fundamental issues of visual recognition (Pan, 2021) have received extensive attention in recent years, and many two-dimensional (2D) and three-dimensional (3D) facial recognition algorithms have been produced over a short period (O'Toole et al., 2012; Black et al., 2016; Gilani and Mian, 2016). At present, 2D facial recognition technology is present in a large number of commercial products. However, compared with 2D recognition, 3D facial recognition technology, which involves more extracted information and higher accuracy, is still in the research and development stage. Collection of data and vulnerability to ambient light are two important factors that constrain further development. Three-dimensional face data can usually be acquired by laser

---

3D measurement or raster projection measurement (Schaffer et al., 2011; Liu K et al., 2014; Stone and Skubic, 2015). However, if a recognition system uses raster projection measurement for data collection, holes in the cloud point data may appear on the inner side of the nose and at the corners of the eyes in the early stages of data storage. This is due to the need to project auxiliary light during measurement, as well as the degree of cooperation from the user. Holes seriously affect automatic data storage and create issues related to subsequent identification. Therefore, filling point cloud holes is a crucial step in the pre-processing of data.

There are two major difficulties related to filling holes in point cloud data. The first is rapid location and extraction of hole boundaries. Its accuracy directly affects effective filling. Existing hole-boundary extraction methods are based mainly on 3D data. These are based on a point cloud grid (Orriols and Binefa, 2003; Jun, 2005) or on the geometric features of boundary points (Floater and Reimers, 2001; Kurlin, 2014; Nguyen et al., 2015). The former method usually extracts smoother point cloud boundaries, but requires a large amount of computation. The latter is less computationally intensive, but is susceptible to noise and outliers.

The second difficulty, and also the more critical, concerns producing fill-in data that is close to the original data. It is well known that filling a hole in a flat-panel point cloud data set is different from filling a point cloud data set describing a table that contains a bump (the hole is located at the bump). Existing hole-filling algorithms directly employ information related to the neighborhood around the hole, directly using methods such as a spline curve (Chui and Lai, 2000; Bendels et al., 2006) or a neural network (Carr et al., 2001) for interpolation. This type of algorithm is simple, easy to use, and does not require any additional information. However, for a hole area with large curvature variation or complicated shape, the filling may not be ideal because of excessive information loss. Pernot et al. (2007) proposed a method of filling holes that involved inserting a topological mesh, which was then deformed. In this approach, the shape of the inserted mesh is the result of minimizing the quadratic function based on the linear mechanical model. This model is used to approximate the curvature variation between the inner mesh and surrounding mesh. It is

also possible to specify other geometric constraints to further shape the inserted mesh. Another method has been proposed based on moving least squares to fill point cloud holes (Wang and Oliveira, 2007); it can be used to adaptively interpolate geometric and shadow information.

In recent years, Liu YJ et al. (2016) proposed a classification and repairing strategy based on a fuzzy inference system. This approach involves an algorithm that divides holes into five types, each with corresponding filling strategies. It also considers the recovery of sharp features of the model, based on different hole types. At the same time, researchers have suggested that data can be processed in combination with additional information and surrounding structures. Quinsat and Lartigue (2015) proposed a digital hole-filling method based on mesh deformation. This method relies on a numerical model as a priori knowledge of the nominal mesh. It uses an a priori computer aided design (CAD) model and the continuity of the neighborhood around the hole as a constraint for recovering data. Panchetti et al. (2010) used 2D information related to the corresponding picture and the curvature of the point cloud mesh to repair holes. The employed algorithm relates the position of the mesh vertices to the light intensity of the image pixels. The objective function based on a mechanical model of a bar network is used to simulate the evolution of curvature over the mesh. These methods can satisfactorily fill complex areas. However, no matter whether these establish an a priori CAD model or provide registration between 2D pixels and 3D point clouds, the overall complexity of these algorithms is high. Moreover, the conditions for obtaining the prior model are demanding and not universal.

Based on the above reasons and methodological extension, and considering the measurement principle of a raster projection measurement system, in this study we propose a 3D face point cloud hole-filling algorithm, using binocular stereo matching and a B-spline. The cross-source point cloud data is obtained using the structure from motion (SFM) algorithm of stereo matching technology (Jeong et al., 2011; Chen et al., 2016). The supplementary information points in the hole area are extracted to assist surface iterative computation of the B-spline curve. Pure SFM data cannot be compared with raster data in accuracy, but when the missing information is too much, certain

known information is definitely better than the repair of completely unknown information. Combining spline iteration by extracting observation points, our final repair data can be close to the data of pure raster projection. Experimental findings indicate high reliability and potential for practical applications of our approach.

2 Method

The setup of the proposed facial recognition system is shown in Fig. 1.
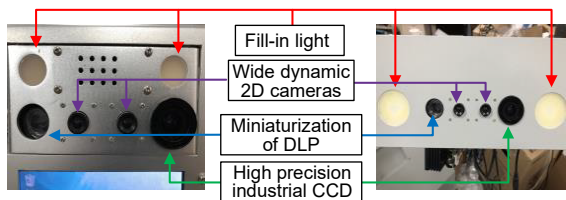

Fig. 1 Hardware setup for the facial recognition system

While acquiring point cloud data using a monocular measurement system consisting of a digital light projector (DLP) and charged-coupled device (CCD), two wide dynamic 2D cameras are added for ranging. Simultaneous collection and backup of the 2D picture information can also be achieved. Since the person who is being photographed is required to turn his or her head slightly to the left and right during acquisition, sufficient 2D picture information can be obtained.

A flow chart describing the facial recognition algorithm proposed in this study is shown in Fig. 2.

The algorithm described in Fig. 2 may appear rather complicated because it converts the point cloud data back and forth between 2D and 3D representations. However, for the repair of data related to a complex facet without a priori models, this step cannot be avoided. The 3D face data is quickly reconstructed using a standard SFM algorithm and registered with the measured data. The 2D phase information is then used to extract the boundary point sets on the hole areas of the point cloud. The required supplementary information points are extracted according to the boundary information of the registration results. Finally, the hole is repaired using a B-spline on the boundary point data sets with added supplementary points.
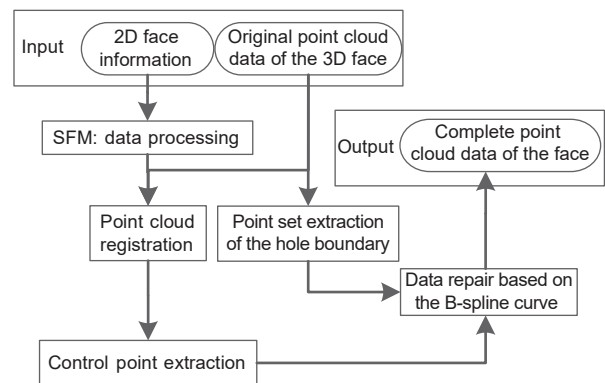

Fig. 2 Flow diagram describing the proposed algorithm for facial recognition

The SFM data sets are obtained using the Visual-SFM (Zheng and Wu, 2015) toolkit. There are two main advantages in employing this toolkit. First, it uses the patch-based multi-view stereo (PMVS) (Furukawa and Ponce, 2010) algorithm for dense reconstruction. A point cloud that has been densely reconstructed by PMVS has a relatively high reconstruction accuracy and integrity on the test site (Goesele et al., 2006). Some results are even comparable to laser-scanned data (Shi et al., 2011). In previous experimental findings, the reconstruction integrity of the method was above 90%. The second advantage of using this toolkit is that the reconstruction efficiency is high, and the time complexity of the main algorithm scales as $O(n)$.

3 Data preprocessing

3.1 Acquisition of phase information

We use a raster projection measurement method based on light mechanism. It processes the collected sinusoidal phase shift fringe image sequence to calculate the wrapped phase. We assume that the gray values of the four phase shift images are as follows:

$$\begin{cases} I_1(u,v) = I' + I'' \cos[\theta(u,v)], \\ I_2(u,v) = I' + I'' \cos[\theta(u,v) + \pi/2], \\ I_3(u,v) = I' + I'' \cos[\theta(u,v) + \pi], \\ I_4(u,v) = I' + I'' \cos[\theta(u,v) + 3\pi/2], \end{cases} \quad (1)$$

where $I_i(u, v)$ is the gray value of the ith image, I' is the background value of the fringe light intensity, I''

is the modulation intensity, and θ is the phase field to be obtained. The wrap phase value is

$$\varphi(u,v) = \arctan\left(\frac{I_4 - I_2}{I_1 - I_3}\right). \qquad (2)$$

The value range of the wrapped phase $\varphi(u, v)$ is $[-\pi, \pi]$, as shown in Fig. 3. The true phase value should be

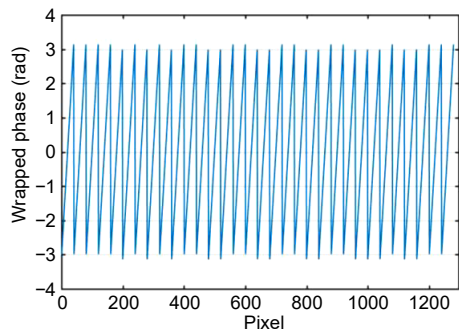$$\theta(u,v) = \varphi(u,v) + 2k(u,v)\pi. \qquad (3)$$

Fig. 3 The wrapped phase of a line in an image

In Eq. (3), $k(u, v)$ is an integer, representing the number of fringe cycles where the image point (u, v) is located. We determine the value of k and expand the wrapped phase into a continuously distributed absolute phase, called the unwrapped phase. The gray code pattern is a set of binary fringe pattern sequences with the number of cycles multiplied, and the maximum number of cycles is consistent with the sinusoidal fringe pattern. We arrange the acquired gray code fringe images in time series, decode the pixel points according to the gray value to obtain a binary number, and uniquely determine the cycle number k. The gray code encoding principle is shown in Fig. 4.
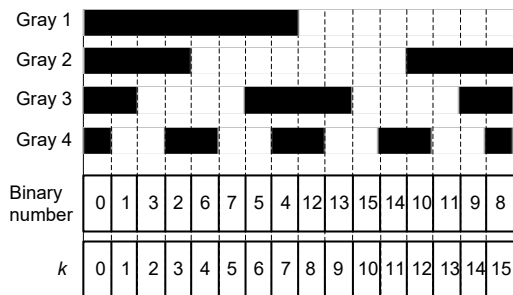

Fig. 4 Gray code pattern coding principle

The above calculation process of obtaining the absolute phase is independent of the processing of different pixels, and only the phase value of the key point can be calculated, reducing the time consumed by the algorithm. We take a set of horizontally coded fringe images of Venus' head, and obtain the phase field results as Fig. 5.
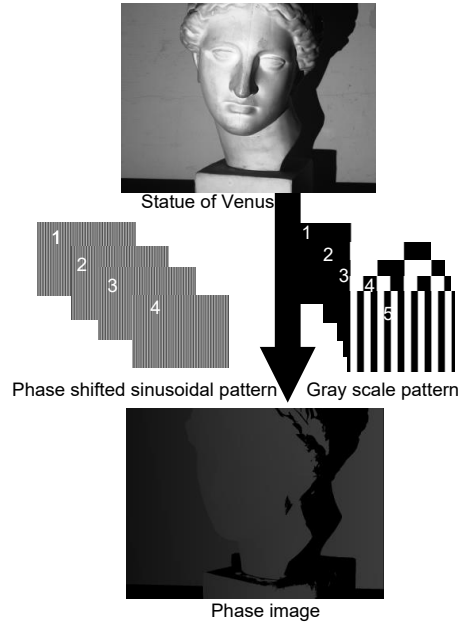

Fig. 5 Horizontal absolute phase of the Venus surface

3.2 Hole boundary extraction of the point cloud

In order for point cloud data to be repaired, the 2D phase information obtained is used to extract the hole boundary. Our method uses mainly the feature that hole boundary points are easy to identify and extract in a 2D phase map.

For non-closed point cloud data, the boundary points are composed of outer contour points and inner hole area boundary points. In a 2D phase diagram, the point cloud boundary points appear as phase jumps. To eliminate the influence of noise points and remove outliers, the boundary points of the point cloud are defined as phase jumps. The phase of the point itself is not 0, and the phases of the points in the neighborhood are also not equal to 0. In the experiment, the number of neighborhood points is set to eight.

Let the size of the 2D phase map be $M \times N$, and $p[m][n]$ is a point on the phase map. The set of all boundary points of the point cloud is B. The set of

outer contour points is B_E. The set of boundary points in the inner hole area is H_p. When $p[m][n]$ satisfies Eq. (4), $p[m][n] \in B$:

$$\begin{cases} |p[m][n] - p[m \pm 1][n \pm 1]| > \partial, \\ p[m][n] \neq 0, \\ \sum_{i=1}^{8} p[m \pm 1][n \pm 1] \neq 0, \\ \prod p[m \pm 1][n \pm 1] = 0, \end{cases} \quad (4)$$

where ∂ is the threshold for a phase jump, set to 5 in the experiment.

After extracting all the boundary points, further processing of the data is needed. On a 2D phase image, the outer contour points are characterized by the initial and final non-zero points of each row in the point cloud boundary point set B. When $p[m][n]$ satisfies Eq. (5), $p[m][n] \in B_E$:

$$\sum_{j=0}^{m-1} p[m][j] = 0 \,\|\, \sum_{j=m+1}^{N} p[m][j] = 0, \quad (5)$$

where $m=1, 2, \cdots, M$, and hole boundary points $H_p = \{p[m][n]|p[m][n] \in B - B_E\}$, $\|$ is the logical judgement for or. It can be seen from Shi et al. (2011) that the 3D coordinates $[X_W, Y_W, Z_W]^T$ of the point cloud data measured using raster projection have the following relationship with corresponding point $p[m][n] = \theta$ on the phase map:

$$\theta = \frac{a_1 X_W + a_2 Y_W + a_3 Z_W + a_4}{a_5 X_W + a_6 Y_W + a_7 Z_W + a_8}, \quad (6)$$

$$\rho \begin{bmatrix} m \\ n \\ 1 \end{bmatrix} = A_c \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}, \quad (7)$$

where θ is the phase value, $a_1 - a_8$ are system parameters, ρ is the scale factor, and A_c is the internal parameter matrix of the 3×3 camera. $a_1 - a_8$ and A_c can be obtained through the calibration of the system.

Therefore, the 2D phase coordinates of the boundary points of the hole are substituted. The corresponding 3D coordinate point $H = \{h_i|i=1, 2, \cdots, k\}$ of the hole boundary can then be calculated using Eqs. (6) and (7).

Fig. 6 is a schematic describing the extraction of

hole boundary points in a face point cloud. The green points are the extracted boundary points. Fig. 6a shows the original face point cloud map, Fig. 6b is the schematic of the boundary points, and Fig. 6c is the schematic after the removal of outer boundary points.
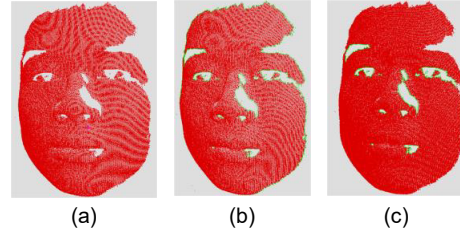

(a)   (b)   (c)

Fig. 6 Hole boundary extraction in a face point cloud: (a) initial point cloud; (b) boundary points (green); (c) point cloud without the outer boundary points
References to color refer to the online version of this figure

3.3 Point cloud registration

To obtain the supplementary points for the point cloud data set acquired using SFM in the hole area, it is necessary to register point cloud sets acquired by raster projection and SFM. In this paper, the initial registration is achieved using random sample consensus (RANSAC) (Buchin et al., 2009) based on a coplanar four-point set. The improved iterative closest point (ICP) provided in our previous work (Huang Y et al., 2015; Huang Y and Da, 2019) is then employed to accurately register the point cloud. For two cross-source point clouds (Huang XS et al., 2017) of the same object, the data can be converted using

$$\begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} = s \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_N \\ Y_N \\ Z_N \\ 1 \end{bmatrix} = M \begin{bmatrix} X_N \\ Y_N \\ Z_N \\ 1 \end{bmatrix}, \quad (8)$$

where R is the rotation matrix and the orthogonal unit matrix is a 3×3 one, $T = [T_X, T_Y, T_Z]^T$ is the 3D translation vector, s is the scaling factor, $[X_W, Y_W, Z_W]$ is a point in the data set P_W obtained by raster projection, $[X_N, Y_N, Z_N]$ is a point in the data set P_N obtained by SFM, and M is a 4×4 matrix. In the initial registration, the corresponding regions with distinct features are first marked in P_W and P_N, respectively, to increase the registration efficiency. The RANSAC method is used to randomly select four points of any non-collinear

line, find the corresponding points in P_N, and calculate the corresponding transformation matrix M. After r iterations, a transformation matrix M that minimizes the matching errors between the two point clouds is selected, and M is used to obtain the initial matching result. In the experiment, $r=10$. In precise registration, first we find the four points closest to the Euclidean distance P_N, for each point p in P_W. Next, we find the center of gravity of the smallest neighborhood formed by these four points, thus forming pairs of points. The parameters R and T are then calculated using the least squares method. The final result is applied to P_N to obtain the point cloud set P_N'. After obtaining P_N', the supplementary information points of the hole area required by the algorithm can be extracted to guide the repair of the point cloud hole. The method of screening supplementary information points is as follows: $H=\{h_i|i=1, 2, \cdots, k\}$ are the hole boundary points of the point cloud. For each point P_N in P_N', it must be determined whether it is in the bounding box of H. If so, $p\in P_I$; otherwise, p is not an element of P_I, where P_I is a set of supplementary information points. At this point, the point cloud set $P_s=P_W\cup P_I$ to which supplementary information points are added can be obtained.

4 Point cloud hole filling

After obtaining point set P_s consisting of hole boundary points and supplementary information points, a B-spline surface is employed to initialize hole filling.

4.1 B-spline surface

The number of control vertices b_{ij} ($i=0, 1, \cdots, m$; $j=0, 1, \cdots, n$) is set to $(m+1)(n+1)$, while the values of the two columns of surface nodes u and v satisfy $u_0\leqslant u_1\leqslant\cdots\leqslant u_{m+k+1}$ and $v_0\leqslant v_1\leqslant\cdots\leqslant v_{m+l+1}$. A $k\times l^{\text{th}}$-order B-spline surface is then defined using Eq. (9):

$$S(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} N_{ik}(u)N_{jl}(v)b_{ij}, \qquad (9)$$

where

$$u \in [u_k,u_{m+1}], v \in [v_l,v_{n+1}],$$
$$N_{ik}(u) = \frac{u-u_i}{u_{i+k-1}-u_i} N_{i,k-1}(u)$$
$$+ \frac{u_{i+k}-u}{u_{i+k}-u_{i+1}} N_{i+1,k-1}(u), \qquad (10)$$

$$N_{jl}(v) = \frac{v-v_j}{v_{j+l-1}-v_i} N_{j,l-1}(v)$$
$$+ \frac{v_{j+l}-v}{v_{j+l}-v_{j+1}} N_{j+1,l-1}(v), \qquad (11)$$

$$N_{ik}(u) = \begin{cases} 1, & k = 1, u_i \leqslant u \leqslant u_{i+1}, \\ 0, & k = 1, u_i \notin [u_i,u_{i+1}], \end{cases}$$
$$N_{jl}(v) = \begin{cases} 1, & l = 1, v_j \leqslant v \leqslant v_{j+1}, \\ 0, & l = 1, v_j \notin [v_j,v_{j+1}]. \end{cases} \qquad (12)$$

Here $N_{ik}(u)$ and $N_{jl}(v)$ are B-spline basis functions of the kth-order and l^{th}-order B-spline surfaces, defined by two columns of nodes u and v, respectively. The polyhedron composed of $(m+1)(n+1)$ control vertices b_{ij} is usually referred to as the control mesh of the B-spline surface, or the control polyhedron. The shape of the control polyhedron generally reflects the shape of the B-spline surface.

4.2 Parameterization of the point set

In the process of B-spline surface fitting for the selected scattered point set P_s, the parameterization of scattered points is indispensable. Commonly used parameterization methods are uniform parameterization, centripetal parameterization, and cumulative chord length parameterization. However, these methods are employed mainly for data points in an array of topological matrices. The base surface projection method is used mainly for the parameterization of scattered points.

A 3D point cloud contains information in three spatial directions: X, Y, and Z. However, in research and practical applications, we find that X and Y often carry more detailed information because of the way in which subjects are used to having facial images taken. Therefore, for the point cloud data set P_s, the four extreme points $(X_{\min}, Y_{\min})$, $(X_{\min}, Y_{\max})$, $(X_{\max}, Y_{\min})$, and $(X_{\max}, Y_{\max})$ of the 2D (X, Y) coordinates are separately selected to form a quadrilateral plane. The basic surface is then constructed using this as a plane. However, since a small number of data points may fall outside the plane after projection, it is necessary to adaptively enlarge the area of the quadrilateral $ABCD$ to ensure that all points of the point set P_s can fall in the plane after being projected. The basic curved surface B_s required for the parameterization process is constructed, as shown in Fig. 7.
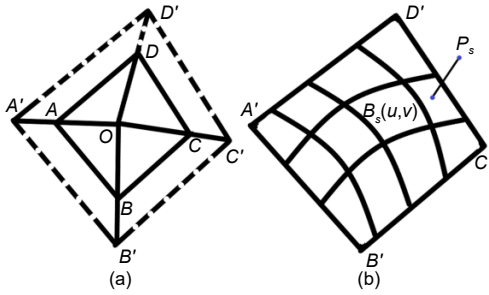
Fig. 7 Construction process for a basic curved surface B_s: (a) the quadrilateral plane is enlarged to ensure that all data points can fall within the plane; (b) the curved surface is constructed

The parameterized coordinates (u_s, v_s) of P_s can be obtained through iteration using the base projection method. Adding P_s into Eq. (9), we obtain

$$\sum_{i=0}^{m}\sum_{j=0}^{n} N_{ik}(u) N_{jl}(v) b_{ij} = P_s. \qquad (13)$$

Finally, the least squares method is used to solve the over-constrained equations. The control vertices of the surface are determined to obtain surface B_s of the preliminary fitting.

However, after initial fitting, the distance error between point set P_s and surface B_s is still relatively large. Therefore, it is necessary to further modify the control vertices, change the shape of the control polyhedron, and change the shape of the surface so that surface B_s approaches point set P_s. An iterative approximation algorithm based on the Newton iteration method used to improve the fitting accuracy is described in Section 4.3.

4.3 Iterative approximation to fitting surfaces

Given a surface $C(u, v)$ and a point F of an arbitrary parameter, the distance vector between any point on surface $C(u, v)$ and point F can be expressed as a function, with (u, v) as a parameter, as shown in Eq. (14):

$$V(u,v) = C(u,v) - F. \qquad (14)$$

To determine parameter (u_0, v_0) when the value of $\|V(u, v)\|$ is the lowest, we may set the line where the vector $V(u, v)$ is located to be l. If line l intersects with surface $C(u, v)$ at point $C(u_0, v_0)$, the vector $V(u_0, v_0)$ needs to be perpendicular to the tangent

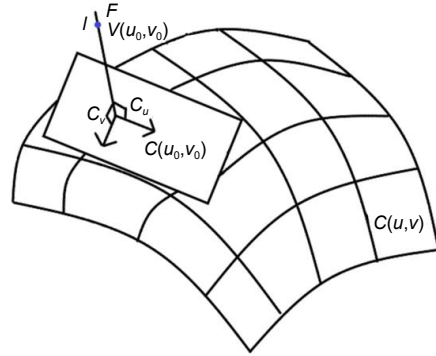plane of the surface $C(u, v)$ tangent to the point $C(u_0, v_0)$, as shown in Fig. 8.

Fig. 8 The condition for the shortest distance between point F and arbitrary parameter surface $C(u, v)$

Then $V(u_0, v_0)$ must be perpendicular to the partial differentials $C_u(u_0, v_0)$ and $C_v(u_0, v_0)$ of surface $C(u, v)$ at (u_0, v_0) in each direction:

$$\begin{cases} V(u_0,v_0) \cdot C_u(u_0,v_0) = 0, \\ V(u_0,v_0) \cdot C_v(u_0,v_0) = 0. \end{cases} \qquad (15)$$

The point selected in the point cloud to fit the surface is P_s, and the initially fitted surface is $S(u, v)$. The distance vector of any point on P_s and $S(u, v)$ can be expressed as a function with (u, v) as a parameter:

$$d(u,v) = S(u,v) - P_s, \qquad (16)$$

and

$$\begin{cases} f(u,v) = d(u,v) \cdot S_u(u,v) = 0, \\ g(u,v) = d(u,v) \cdot S_v(u,v) = 0. \end{cases} \qquad (17)$$

Here $S_u(u, v)$ and $S_v(u, v)$ are the first-order partial derivatives of surface $S(u, v)$ for u and v, respectively. The barycentric coordinates (u_s, v_s) of P_s are calculated and used as the initial estimate. Eq. (17) is solved according to the Newton iteration method:

$$H\sigma^{\mathrm{T}} = k^{\mathrm{T}}, \qquad (18)$$

where

$$\sigma = (\delta u, \delta v), \qquad (19)$$

$$k = -(f(u_s,v_s), g(u_s,v_s)), \qquad (20)$$

$$H = \begin{bmatrix} f_u & f_v \\ g_u & g_v \end{bmatrix} = \begin{bmatrix} \|S_u\|^2 + dS_{uu} & S_v S_u + dS_{uv} \\ S_u S_v + dS_{vu} & \|S_v\|^2 + dS_{vv} \end{bmatrix}, \qquad (21)$$

where δu and δv are the iterative steps in both directions, $d = d(u_s, v_s)$, f_u, f_v, g_u, and g_v represent the first-order partial derivatives of the corresponding vector pair u and v at point (u_s, v_s), and S_{uu}, S_{uv}, S_{vv}, and S_{vu} represent the second-order partial derivatives of surface $S(u, v)$ at point (u_s, v_s) for u and v.

According to Eq. (18), we obtain

$$\begin{bmatrix} \left\| S_u \right\|^2 + dS_{uu} & S_v S_u + dS_{uv} \\ S_u S_v + dS_{vu} & \left\| S_v \right\|^2 + dS_{vv} \end{bmatrix} \begin{bmatrix} \delta u \\ \delta v \end{bmatrix} = - \begin{bmatrix} f(u_s, v_s) \\ g(u_s, v_s) \end{bmatrix}. \tag{22}$$

After solving Eq. (22), iterative step lengths δu and δv can be determined, and the new parameter $(u_s + \delta u, v_s + \delta v)$ is substituted into Eq. (13) to obtain the surface control point, and the parameters of the surface are re-determined. The final value is set to $\gamma > 0$ (according to the needs of the given system), and Eq. (22) is iteratively solved until $\left[t \sum_{s=0}^{t} \left\| d(u_s + \delta u, v_s + \delta v) - d(u_s, v_s) \right\| \right]^{-1} \leq \gamma$, and surface $S'(u, v)$ is determined.

To fill the holes in the final fitted B-spline surface $S'(u, v)$, we consider a line on surface $S'(u, v)$, and then points on the line (Fig. 9).
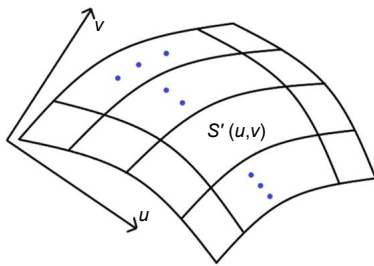

Fig. 9 Point collection on a surface based on its curvature

In a point cloud, curvature changes are usually relatively great when the point cloud density is higher. Therefore, the hole can be filled smoothly by selecting points based on such a parameter as curvature. In the process of obtaining the curvature, the original point cloud is first divided into grids, and then the k_n neighboring points of a particular point are obtained. Based on the relationship between each point and its k_n neighboring points, the average curvature $\bar{\rho}_0$ and the average point distance $\bar{d}$ of the original point cloud are then obtained. The average curvature $\bar{\rho}_s$ of

P_s $(s = 0, 1, \cdots, t)$ is similarly obtained, and the interval between collected points is set to $\Delta \omega = \bar{\rho}_0 \bar{d} / \bar{\rho}_s$. When collecting points on the B-spline surface $S'(u, v)$, the equal-parameter curve is obtained at an equal interval of $\Delta \omega$ in the direction of u on the surface. Points are then collected at an equal interval in the direction of v on each equal-parameter curve, to obtain discrete points for filling the holes.

5 Simulation results

To verify the effectiveness of the proposed algorithm, simulations were conducted on two models: a smooth ceramic cup and a human face. The simulations were performed on a computer with Core i5, 4 GB memory, and the Windows 10 operating system. Matlab 2015b was used as the software environment.

First, the algorithm was tested on ordinary smooth surface objects. Fig. 10a shows the ceramic cup point cloud data obtained by raster projection measurement. It can be seen that since the reflectivity of the ceramic cup was too high, a hole was formed in the middle of the point cloud. Fig. 10b shows the point cloud data measured using VisualSFM. Since the smooth ceramic cup itself lacked texture, speckle information was projected to improve the reconstruction accuracy when acquiring point cloud data. Fig. 10c shows the point cloud map after the holes were repaired by the proposed algorithm. The data set generated by VisualSFM provided 96 supplementary information points for hole repair, and the number of added patch points was 404. During ceramic cup hole repairing, when the B-spline curve was used to compute the surface equation of the corresponding points, the supplementary points extracted from the point cloud data obtained by VisualSFM will provide this process with additional information.
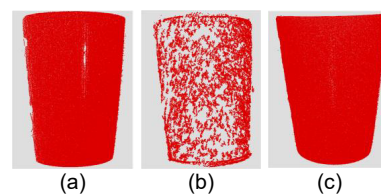

Fig. 10 Point cloud of a ceramic cup repaired using raster projection (a), VisualSFM (b), and the proposed algorithm (c)

As a result, the surface equations more closely matched the original data, and the hole repair results were more accurate.

For the more complex human face, the human face point cloud data measured by raster projection is shown in Fig. 11a. The eye and eyebrow areas contained holes due to low reflectance. There were also holes in the nose area due to its own occlusion. The point cloud data of the face measured by VisualSFM is shown in Fig. 11b, and the point cloud map after the holes were repaired by the proposed algorithm is shown in Fig. 11c. It can be seen that the repairing result of a complex area, such as the nose and eyes, accurately reflected the shape of its original form.
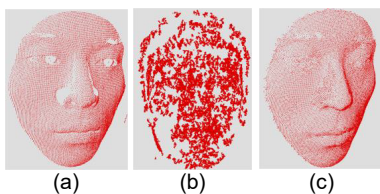

(a)   (b)   (c)

Fig. 11 Point cloud of a human face repaired using raster projection (a), VisualSFM (b), and the proposed algorithm (c)

To further verify the effectiveness of the method, two facial areas where there were typically data holes were selected for further analysis (Fig. 12). The hole shown in Fig. 12b is near one side of the nose. The hole in Fig. 12c is between the eyebrows.

Fig. 13a shows the complete point cloud data, while Figs. 13b and 13c are point cloud maps repaired by the point cloud hole repairing algorithm, based on the hole in Fig. 12b. The green part is a fixed point cloud with a total of 1612 points. A graph obtained using ICP to register added points after repairing and the original hole area points is shown in Fig. 13d. It can be seen that for a large hole with a large variation
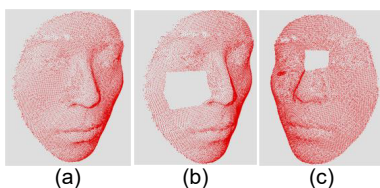

(a)   (b)   (c)

Fig. 12 Positions of point cloud holes: (a) complete point cloud; (b) hole near one side of the nose; (c) hole between the eyebrows

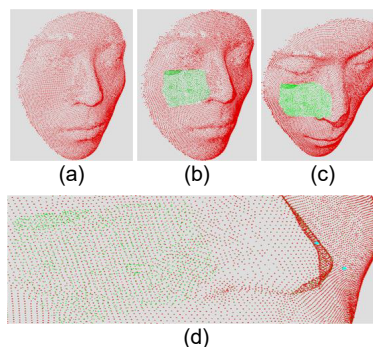(a)   (b)   (c)

(d)

Fig. 13 Simulation results of point cloud hole repairing in cheek: (a) complete point cloud; (b) left-hand view of the repaired point cloud; (c) a top view of the repaired point cloud; (d) registration of the repaired point cloud
References to color refer to the online version of this figure

in shape, the point cloud data obtained by the algorithm fitted the original shape completely. The original facial details were accurately described.

Fig. 14a shows the complete point cloud data for the space between the eyebrows on the test face, Figs. 14b and 14c show the patched point clouds, and Fig. 14d shows the result after registration with the complete point cloud. It can be seen that the added point cloud was consistent with the original point cloud in terms of curvature variation. The repaired points clearly described the detailed shape between the eyebrows. These results indicated that the proposed algorithm can effectively repair holes in areas with complex shapes.

Table 1 is a comparison between the implemented and initial points for the holes shown in Figs. 12b and 12c. The number of ICP iterations was 10. The
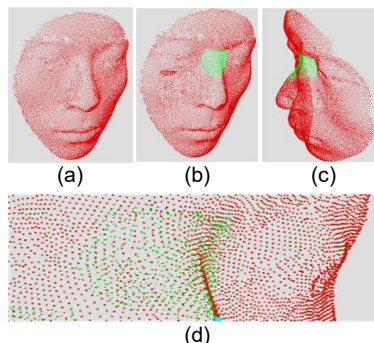

(a)   (b)   (c)

(d)

Fig. 14 Simulation results of point cloud hole repairing in wing of nose: (a) complete point cloud; (b) left-hand view of the repaired point cloud; (c) a top view of the repaired point cloud; (d) registration of the repaired point cloud
References to color refer to the online version of this figure

Table 1 Comparison between the implemented and initial points shown

Hole position	Number of initial points	Number of implemented points	Number of ICP iterations	RMSE	Runtime (s)
Nose & face	1518	1612	10	0.019	123
Between eyebrows	643	651	10	0.072	74

ICP: iterative closest points

root mean square errors (RMSE) of the point cloud registrations in Figs. 13d and 14d were 0.019 and 0.072, respectively. The implemented and initial points were consistent in terms of both their numerical values and curvature variation, both in large areas and in areas where the curvature varied greatly. For objects with complex shapes, the proposed algorithm can effectively restore the original shape. It should be mentioned that due to the large number of steps in the overall algorithm, it took a little more time to repair a hole alone, which is a problem to be solved in the future.

6 Conclusions

In practical 3D face recognition systems, the stored point cloud data often contains holes. In this paper, a hole-filling algorithm based on stereo matching and a B-spline is proposed, using a laboratory-developed identification system combined with a consideration of measurement principles. The algorithm first extracts a hole boundary using features of the 3D hole boundary points in the 2D phase. The SFM data set and the raster point cloud data set are then registered, and the supplementary information points are extracted. Finally, the B-spline curve surface is used to further repair the holes, making the repaired point cloud be more consistent with the original data. Simulations on a smooth ceramic cup and a human face show that the proposed algorithm can effectively reproduce surface details and accurately restore complex surface shapes. Since the algorithm requires a high-precision SFM data set, although the accuracy and completeness of the VisualSFM-reconstructed point cloud is high for most objects, this method is not particularly effective for objects that lack texture. In this case, how to improve the reconstruction accuracy is an issue to be addressed. In addition, we will explore adaptive data repairing of complex objects without a priori models to develop the proposed algorithm for use across a wide range of facial recognition applications.

Contributors

Yuan HUANG designed the research, processed the data, and drafted the paper. Feipeng DA helped organize the paper. Yuan HUANG and Feipeng DA revised and finalized the paper.

Compliance with ethics guidelines

Yuan HUANG and Feipeng DA declare that they have no conflict of interest.

References

Bendels GH, Schnabel R, Klein R, 2006. Detecting holes in point set surfaces. *J WSCG*, 14:89-96.

Black JA Jr., Gargesha M, Kahol K, et al., 2016. Framework for performance evaluation of face recognition algorithms. SPIE 4862:163-174.
https://doi.org/10.1117/12.473032

Buchin K, van Kreveld M, Meijer H, et al., 2009. On planar supports for hypergraphs. Proc 17th Int Conf on Graph Drawing, p.345-356.
https://doi.org/10.1007/978-3-642-11805-0_33

Carr JC, Beatson RK, Cherrie JB, et al., 2001. Reconstruction and representation of 3D objects with radial basis functions. Proc 28th Annual Conf on Computer Graphics and Interactive Techniques, p.67-76.
https://doi.org/10.1145/383259.383266

Chen H, Ma SW, Nuechter A, 2016. Non-synchronous point cloud algorithm for 3D reconstruction based on laser scanning and SFM. *Chin J Sci Instrum*, 37(5):1148-1157 (in Chinese).
https://doi.org/10.3969/j.issn.0254-3087.2016.05.024

Chui CK, Lai MJ, 2000. Filling polygonal holes using C^1 cubic triangular spline patches. *Comput Aided Geom Des*, 17(4):297-307. https://doi.org/10.1016/S0167-8396(00)00005-4

Floater MS, Reimers M, 2001. Meshless parameterization and surface reconstruction. *Comput Aided Geom Des*, 18(2):77-92. https://doi.org/10.1016/S0167-8396(01)00013-9

Furukawa Y, Ponce J, 2010. Accurate, dense, and robust multi-view stereopsis. *IEEE Trans Patt Anal Mach Intell*, 32(8):1362-1376. https://doi.org/10.1109/TPAMI.2009.161

Gilani ZS, Mian A, 2016. Towards large-scale 3D face recognition. Proc Int Conf on Digital Image Computing: Techniques and Applications, p.1-8.
https://doi.org/10.1109/DICTA.2016.7797090

Goesele M, Curless B, Seitz SM, 2006. Multi-view stereo revisited. Proc IEEE Computer Society Conf on Computer Vision and Pattern Recognition, p.2402-2409. https://doi.org/10.1109/CVPR.2006.199

Huang XS, Zhang J, Fan LX, et al., 2017. A systematic approach for cross-source point cloud registration by preserving macro and micro structures. *IEEE Trans Image Process*, 26(7):3261-3276. https://doi.org/10.1109/TIP.2017.2695888

Huang Y, Da FP, 2019. Registration algorithm for point cloud based on normalized cross-correlation. *IEEE Access*, 7: 137136-137146. https://doi.org/10.1109/ACCESS.2019.2942127

Huang Y, Da FP, Tao HJ, 2015. An automatic registration algorithm for point cloud based on feature extraction. *Chin J Lasers*, 42(3):308002 (in Chinese). https://doi.org/10.3788/CJL201542.0308002

Jeong Y, Bok Y, Kim JS, et al., 2011. Complementation of cameras and lasers for accurate 6D SLAM: from correspondences to bundle adjustment. Proc IEEE Int Conf on Robotics and Automation, p.3581-3588. https://doi.org/10.1109/ICRA.2011.5979568

Jun Y, 2005. A piecewise hole filling algorithm in reverse engineering. *Comput-Aided Des*, 37(2):263-270. https://doi.org/10.1016/j.cad.2004.06.012

Kurlin V, 2014. A fast and robust algorithm to count topologically persistent holes in noisy clouds. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.1458-1463. https://doi.org/10.1109/CVPR.2014.189

Liu K, Zhou CH, Wei SB, et al., 2014. Optimized stereo matching in binocular three-dimensional measurement system using structured light. *Appl Opt*, 53(26):6083-6090. https://doi.org/10.1364/AO.53.006083

Liu YJ, Wang M, Zhang HL, et al., 2016. Strategy of classification and repairing for hole of incomplete point clouds based on fuzzy inference. *J Comput Theor Nanosci*, 13 (11): 8227-8233. https://doi.org/10.1166/jctn.2016.5961

Nguyen VS, Trinh TH, Tran MH, 2015. Hole boundary detection of a surface of 3D point clouds. Proc Int Conf on Advanced Computing and Applications, p.124-129. https://doi.org/10.1109/ACOMP.2015.12

Orriols X, Binefa X, 2003. Finding breaking curves in 3D surface. Proc 1st Iberian Conf on Pattern Recognition and Image Analysis, p.681-688.

https://doi.org/10.1007/978-3-540-44871-6_79

O'Toole AJ, An XB, Dunlop J, et al., 2012. Comparing face recognition algorithms to humans on challenging tasks. *ACM Trans Appl Percept*, 9(4):16. https://doi.org/10.1145/2355598.2355599

Pan YH, 2019. On visual knowledge. *Front Inform Technol Electron Eng*, 20(8):1021-1025. https://doi.org/10.1631/FITEE.1910001

Pan YH, 2021. Miniaturized five fundamental issues about visual knowledge. *Front Inform Technol Electron Eng*, 22(5):615-618. https://doi.org/10.1631/FITEE.2040000

Panchetti M, Pernot JP, Véron P, 2010. Towards recovery of complex shapes in meshes using digital images for reverse engineering applications. *Comput-Aided Des*, 42(8):693-707. https://doi.org/10.1016/j.cad.2010.01.004

Pernot JP, Moraru G, Véron P, 2007. Repairing triangle meshes built from scanned point cloud. *J Eng Des*, 18(5): 459-473. https://doi.org/10.1080/09544820701403797

Quinsat Y, Lartigue C, 2015. Filling holes in digitized point cloud using a morphing-based approach to preserve volume characteristics. *Int J Adv Manuf Technol*, 81(1-4): 411-421. https://doi.org/10.1007/s00170-015-7185-0

Russell, Stuart J, Norvig P, 2010. Artificial intelligence: a modern approach. *Appl Mech Mater*, 263(5):2829-2833.

Schaffer M, Grosse M, Harendt B, et al., 2011. High-speed three-dimensional shape measurements of objects with laser speckles and acousto-optical deflection. *Opt Lett*, 36(16):3097-3099. https://doi.org/10.1364/OL.36.003097

Shi LM, Guo FS, Hu ZY, 2011. An improved PMVS through scene geometric information. *Acta Autom Sin*, 37(5):560-568 (in Chinese). https://doi.org/10.3724/SP.J.1004.2011.00560

Stone EE, Skubic M, 2015. Fall detection in homes of older adults using the Microsoft Kinect. *IEEE J Biomed Health Inform*, 19(1):290-301. https://doi.org/10.1109/JBHI.2014.2312180

Wang JN, Oliveira MM, 2007. Filling holes on locally smooth surfaces reconstructed from point clouds. *Image Vis Comput*, 25(1):103-113. https://doi.org/10.1016/j.imavis.2005.12.006

Zheng EL, Wu CC, 2015. Structure from motion using structure-less resection. Proc IEEE Int Conf on Computer Vision, p.2075-2083. https://doi.org/10.1109/ICCV.2015.240