Computer-Aided Design 81 (2016) 14–23

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

A novel method for 3D reconstruction: Division and merging of overlapping B-spline surfaces☆

Rui-Jun Yan [a], Jing Wu [a,*], Ji Yeong Lee [b,*], Abdul Manan Khan [b], Chang-Soo Han [b], Erdal Kayacan [a], I-Ming Chen [a]

[a] School of Mechanical and Aerospace Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798, Singapore
[b] Department of Robot Engineering, Hanyang University, Ansan, Gyeonggi-do, 426791, Republic of Korea

ARTICLE INFO

Article history:
Received 29 February 2016
Accepted 27 August 2016

Keywords:
B-spline surface
Merging and division
Overlapping
Scanned data
3D laser scanner
3D reconstruction

ABSTRACT

B-spline surfaces, extracted from scanned sensor data, are usually required to represent objects in inspection, surveying technology, metrology and reverse engineering tasks. In order to express a large object with a satisfactory accuracy, multiple scans, which generally lead to overlapping patches, are always needed due to, inter-alia, practical limitations and accuracy of measurements, uncertainties in measurement devices, calibration problems as well as skills of the experimenter. In this paper, we propose an action sequence consisting of division and merging. While the former divides a B-spline surface into many patches with corresponding scanned data, the latter merges the scanned data and its overlapping B-spline surface patch. Firstly, all possible overlapping cases of two B-spline surfaces are enumerated and analyzed from a view of the locations of the projection points of four corners of one surface in the interior of its overlapping surface. Next, the general division and merging methods are developed to deal with all overlapping cases, and a simulated example is used to illustrate aforementioned detailed procedures. In the sequel, two scans obtained from a three-dimensional laser scanner are simulated to express a large house with B-spline surfaces. The simulation results show the efficiency and efficacy of the proposed method. In this whole process, storage space of data points is not increased with new obtained overlapping scans, and none of the overlapping points are discarded which increases the representation accuracy. We believe the proposed method has a number of potential applications in the representation and expression of large objects with three-dimensional laser scanner data.

1. Introduction

The use of three-dimensional (3D) laser scanner measurements is indisputably one of the most convenient way of obtaining 3D point clouds of any objects with complex or unconventional shape. Undoubtedly, methods to accurately represent objects with point clouds have made tremendous impact on industrial and academic applications such as segmentation [1], expression [2], reconstruction [3], and classification [4]. However, point clouds need large storage spaces resulting in a fact that they cannot clearly express the object shapes because of their limited numbers. As a solution, one can propose to extract planar surfaces from point clouds to express the objects precisely yet with less number of geometric parameters [5]. Even though planar surfaces work well in the representation of objects with multiple planes, they cannot be used to accurately express the objects having complex shapes.

Request for increased, and almost perfect, accuracy for representing objects with sharp features by using a small number of control points urges the use of B-spline surface that can be extracted from 3D point clouds [6,7]. Whereas B-spline surface can be simply extracted by using approximation or interpolation method for ordered or organized points [8–10], sequencing 3D point cloud becomes a difficult issue for unordered or unorganized points [11]. What is more, triangulation [12–14] has recently been an effective and necessary method to extract a B-spline surface from organized corner points of triangles. On the contrary, there are two

☆ This paper has been recommended for acceptance by Nickolas S. Sapidis.

* Corresponding authors.
E-mail addresses: yrj@ntu.edu.sg (R.-J. Yan), wujing@ntu.edu.sg (J. Wu), jiyeongl@hanyang.ac.kr (J.Y. Lee), kam@hanyang.ac.kr (A.M. Khan), cshan@hanyang.ac.kr (C.-S. Han), erdal@ntu.edu.sg (E. Kayacan), michen@ntu.edu.sg (I-M. Chen).

http://dx.doi.org/10.1016/j.cad.2016.08.007
0010-4485/

handicaps for the triangulation–extraction method. First, the triangulation process brings extra error sources into the final surface extraction. In other words, the extracted B-spline surface error consists of three types of components: measurement error itself, triangulation error and extraction error. Secondly, this method is more appropriate to express small objects that can be covered with a small number of scans. The reason is that even though the triangulation itself takes short time, it only can be done after finishing the whole scan. It is obvious that multiple scans are needed to express a large object or an environment, e.g. a city, which is not practical to implement the triangulation for such huge datasets. In order to show the representation result with the B-spline surface simultaneously while the scanning is going on, B-spline surface must be extracted after each new scan is obtained, and the new surface must be merged with the previous surfaces. Many methods have been developed to merge non-overlapping surfaces. Pungotra et al. [15,16] proposed a pre-calculated blending matrix to generate discrete points on a B-spline surface. The discrete point matrices from two surfaces are combined to form a single matrix that represents the final shape. Chen et al. [17] introduced a surface merging method by applying the distance function between the two B-spline surfaces with respect to the L_2 norm as the approximate error. For two overlapping range scans, their overlapping geometry [18,19] is aligned by minimizing the distance of the corresponding scanned points of two scans to register these scans and the paired points are directly merged to update the triangle meshes. But it is quite difficult to guarantee that these paired points are discretely scanned from the exactly same location of an object. Marton et al. [20] proposed a fast surface reconstruction method by using incremental triangulation to remove the points of a new scan which are close to existing triangle faces. Zha et al. [21] reconstructed the 3D model by discarding a new triangle if it partially overlaps with some old ones. In summary, these aforementioned methods have roughly merged the close points, stored all the points, and discarded the overlapping points of a new scan while the newly obtained points overlap with previous obtained points. These operations probably lead to an incorrect merging result, large data storage problem, and information missing issue. However, B-spline surfaces with overlapping patches cannot be correctly merged with these methods especially without storing the previously obtained points, which is the main motivation of this paper. Comparing the previous methods, our proposed method can accurately merge and update the overlapping surface patches while no obtained informations are discarded and the storage space is not significantly increased.

This paper proposes a division and merging method for overlapping B-spline surfaces extracted from scanned data points. Our previous research [22] proposes a merging method by dealing with generated sample points of two surfaces. It has only analyzed two cases of two overlapping surfaces, in which at most one projection point of four corner points of one surface locates in the interior of the second surface. In this paper, as a further extension to our previous research paper, we consider all possible overlapping cases, and we propose a general division algorithm to deal with all possible cases of two overlapping surfaces. In our novel method, after a group of new scanned data points are available, they are firstly extracted as a B-spline surface. Afterwards, its overlapping surface, which was selected from the previously extracted surfaces, is found. Then, both surfaces are divided into overlapping patches and non-overlapping patches by projecting the generated sample points from the boundaries of one surface onto the other one. According to these separated patches of new surface, these scanned data points are divided into corresponding groups. The data points belonging to overlapping patch of new surface project onto overlapping patch of previous surface, and probabilistically merge with these projection points as a group of updated points having the same size of original data points. By replacing the previous data points with these updated data points, a merged B-spline surface can be extracted. Comparing with the previous surface reconstruction techniques dealing with overlapping scans, probabilistic merging of a point and its projection point is more accurate than the simple merging of the close points of two scans. In addition, storage space does not increase when an overlapping scan is obtained, and none of the overlapping points are discarded in our proposed method.

The rest of this paper is organized as follows: In Section 2, extraction of B-spline surface from scanned data points is introduced, and overlapping cases of two surfaces are summarized. Then, the division and merging algorithm is presented in detail in Section 3. The validations of the proposed algorithm with the simulated scanning points are given in Section 4. Finally, some conclusions are drawn from this paper in Section 5.

2. Overlapping B-spline surfaces

2.1. Extraction of B-spline surface

Definition. A B-spline surface of the order $p \times q$ with a bidirectional net of control points $P_{i,j}$ $(i = 0, \ldots, n;\ j = 0, \ldots, m)$, two knot vectors U, V and the product of the univariate B-spline basis functions $N_{i,p}(u)\, N_{j,q}(v)$ can be expressed as

$$S(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u)\, N_{j,q}(v)\, P_{i,j}. \tag{1}$$

Calculation of control points. Given a dataset $\{Q_{k,l}\}$ $(l = 0, \ldots, N)$ with $N + 1$ groups, and each group includes different number of raw data points. For the lth group, the least square approximation method is used to find the control points of the B-spline curve by minimizing the error between the raw sensor data and the extracted curve. This process is executed by fixing the parameter l in following equation.

$$\sum_{l=0}^{N} \sum_{k=0}^{M} \left| Q_{k,l} - S(\overline{u}_k, \overline{v}_l) \right|^2. \tag{2}$$

The control points extracted from each group can be calculated as follows:

$$T_l = \left(N_{\overline{u}}^T N_{\overline{u}}\right)^{-1} N_{\overline{u}}^T Q_l \tag{3}$$

where

$$N_{\overline{u}} = \begin{bmatrix} N_{0,p}(\overline{u}_0) & \cdots & N_{m,p}(\overline{u}_0) \\ \vdots & \ddots & \vdots \\ N_{0,p}(\overline{u}_M) & \cdots & N_{m,p}(\overline{u}_M) \end{bmatrix}$$

$$T_l = \left[T_{0,l}, \ldots, T_{m,l}\right]^T, \qquad Q_l = \left[Q_{0,l}, \ldots, Q_{M,l}\right]^T.$$

By adding new knot in an initial knot vector [22,23], the number of control points for all these groups would be the same. To find the control points of the B-spline surface, the previously extracted control points of the curves can be rearranged as

$$T = \begin{bmatrix} T_{0,0} & \cdots & T_{m,0} \\ \vdots & \ddots & \vdots \\ T_{0,N} & \cdots & T_{m,N} \end{bmatrix}. \tag{4}$$

Finally, the control points of the B-spline surface are obtained by considering the arranged control points as raw data points.

$$P_{:,i} = \left(N_{\overline{v}}^T N_{\overline{v}}\right)^{-1} N_{\overline{v}}^T T_{:,i}, \quad i = 0, \ldots, m \tag{5}$$

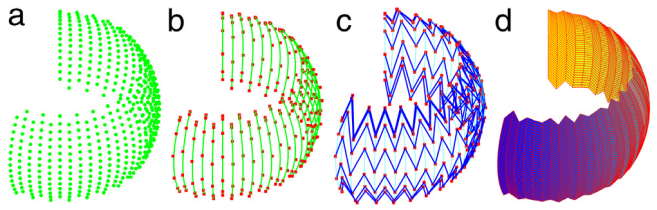


Fig. 1. An example of B-spline surface extraction: (a) Simulated sensor data; (b) Extracted B-spline curves with the control points; (c) Control points of the extracted B-spline surface; (d) Extracted B-spline surface. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where $P_{:,i}$ is the ith column of control points of the extracted B-spline surface with the dimension of $3n \times 1$ in three dimensional space.

Derivation of covariance matrix. To express the uncertainty of the extracted surface, the covariance matrix C_P of the control points is propagated from the covariance matrix C_Q of the raw data points by using first-order Taylor expansion.

$$\begin{aligned} C_{T_l} &= \left[\left(N_{\bar{u}}^T N_{\bar{u}}\right)^{-1} N_{\bar{u}}^T\right] C_{Q_l} \left[\left(N_{\bar{u}}^T N_{\bar{u}}\right)^{-1} N_{\bar{u}}^T\right]^T \\ C_{P_{:,i}} &= \left[\left(N_{\bar{v}}^T N_{\bar{v}}\right)^{-1} N_{\bar{v}}^T\right] C_{T_{:,i}} \left[\left(N_{\bar{v}}^T N_{\bar{v}}\right)^{-1} N_{\bar{v}}^T\right]^T. \end{aligned} \tag{6}$$

Based on the covariance matrix of the control points, the covariance matrix of the generated sample points from B-spline surface can be easily obtained by calculating the partial derivative of S in (1) with respect to $\{P_{i,j}\}$. An example is used to explicitly show the details of the extraction process of a B-spline surface in Fig. 1. In Fig. 1(a), a set of 3D sensor data is simulated. In each vertical group of this dataset, a B-spline curve is extracted by minimizing the error between sensor data and the extracted curve. The extracted B-spline curves and their control points are shown in Fig. 1(b) with green curve and red square, respectively. By horizontally arranging the control points, new B-spline curves can be extracted by minimizing the error between the previous control points and the new curve. The control points of these new curves are the control points of the extracted B-spline surface, which are plotted in Fig. 1(c) with red squares. To have a better view, these control points are connected in vertical direction with green lines and in horizontal direction with blue lines. Based on these control points, a B-spline surface extracted from the simulated sensor data in Fig. 1(a) is shown in Fig. 1(d). By comparing Fig. 1(a) and Fig. 1(c), it can be seen that the number of control points of the extracted B-spline surface is much smaller than the number of sensor data. It means that a smaller storage space is needed to store the B-spline surface instead of the sensor data. This example validates that this surface extraction method is capable of extracting a B-spline surface from a dataset with multiple groups while the number of sensor data in each group is different.

2.2. Overlapping cases of two B-spline surfaces

In a surface reconstruction of a large object or environment, a large number of scans are needed. During this reconstruction process, assume that N scans are obtained and the corresponding extracted B-spline surfaces are extracted from these scans. These B-spline surfaces are stored to represent the currently reconstructed part of the object without storing the point clouds because of its large storage space. In a following reconstruction step, new B-spline surfaces are extracted from the newly obtained $N + 1$th scan. Comparing with the previously stored B-spline surfaces, the point clouds of these new extracted surfaces are not discarded yet until obtaining the $N + 2$th scan. In the following presentation, a standard B-spline surface is assumed having four corners which are connected with four continuous boundaries. In this section, the possible overlapping cases of one previously stored B-spline surface and one new extracted B-spline surface are listed from a view of the locations of the projection points for the four corners of one surface relative to the other one. Here we assumed that the overlapping patch of these two surfaces extracted from different scans is scanned from the same part of the object.

There are many possible overlapping cases for two B-spline surfaces if an overlapping surface patch exists. However, a B-spline surface has only four corners. And there are two location cases for each corner relative to the other overlapping surface, which are interior and exterior. In Fig. 2, these overlapping cases are illustrated from a view of the locations of the four corner points of a B-spline surface. To have a clear description, the surface keeping a same pose in all these six cases is defined as surface **A**, and the other one having different poses in each case is defined as surface **B**. In all these cases, the locations of four corners of the surface **B** relative to the surface A are analyzed as follows:

(a) None of the four corners of surface **B** locate in the interior of surface **A**. To merge the overlapping patch of two surfaces, surface **A** needs to be divided into two patches, one overlapping patch and one non-overlapping patch.
(b) Although the locations of the four corners of surface **B** is similar to case (a), the surface **A** needs to be divided into three patches. This case is used to show that the division result of a B-spline surface may be different even with the same locations of the corners.
(c) One corner of the surface **B** locates in the interior of the surface **A** leading an additional division although two non-overlapping patches are connected. If only one surface is extracted from these two patches, the extracted B-spline surface is not accurate because a corner point always leads a sharp division result.
(d) Two corners of the surface **B** locate in the interior of the surface **A** leading to sharp corners for the non-overlapping surface patch. In this example, surface **A** is divided into four patches.
(e) Only one corner of the surface **B** locates in the exterior of the surface **A**. Surface **A** is divided into five patches which are plotted with different colors.
(f) All the corners of surface **B** locate in the interior of the surface **A**. Surface **A** is divided into four non-overlapping patches and one overlapping patch.

While even having the same locations of the four corners for each overlapping case in Fig. 2, more cases exist if the lengths of the boundaries of both these two surfaces are extended. It is very difficult to develop one division method for each case because a large number of cases need to be analyzed. However, by considering the locations of the projection points for the four corners of one surface relative to the other one, a general division method can be proposed to divide all these possible overlapping cases.

3. Division and merging algorithms

3.1. Surface division

Based on the above analysis, a general surface division algorithm is proposed in this section. An example of general overlapping case having two B-spline surfaces with complex shape in Fig. 3 is used to illuminate the pseudo-codes in Algorithm 1. In Fig. 3, the blue surface is considered as a surface extracted from previous scans (surface **A**), and red surface is considered as a new surface (surface **B**). It means that the raw sensor data for surface **B** are not discarded yet. First of all, the sample points of

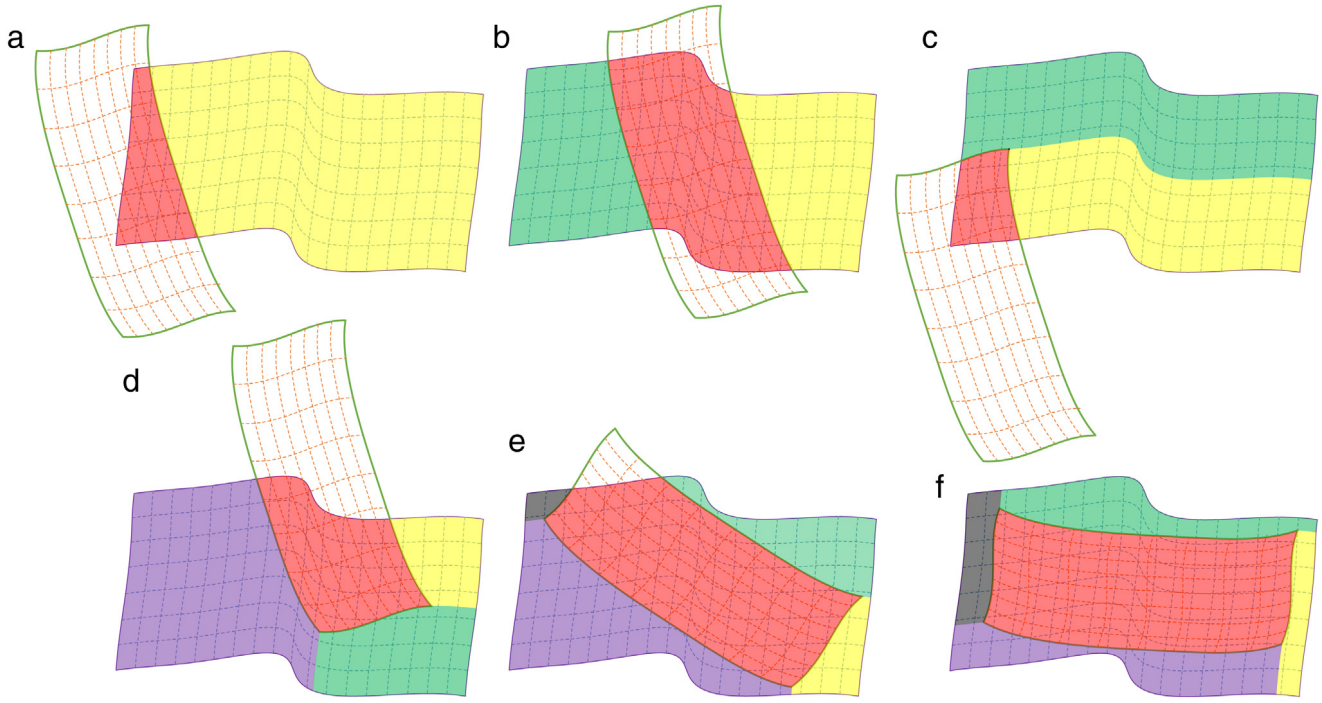


Fig. 2. Overlapping cases of two B-spline surfaces. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

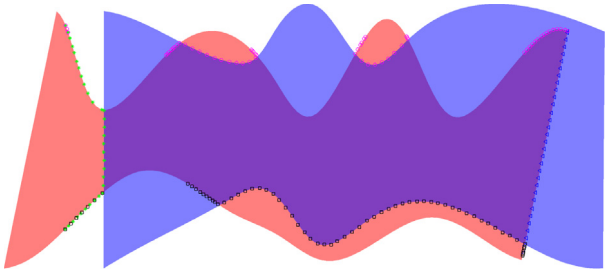


Fig. 3. An example of general overlapping case. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

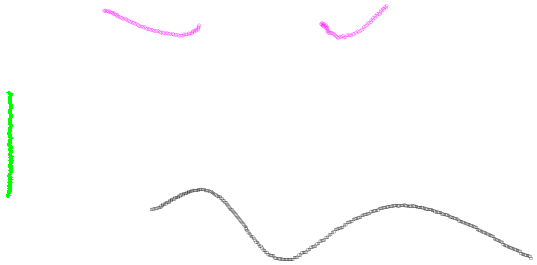


Fig. 4. Projected boundary points locating in the interior of surface **B**.

Algorithm 1: SurfaceDivision

```
Input  : two surfaces SurfA, SurfB.
Output : Number of patch NumOfPatch, PatchSet.
NumOfPatch = 1;
ProjSet = ∅;
//PointSetBorder is the generated sample points from the i-th
boundary of surface B;
for PointSetBorder ∈ SurfB do
    ProjPoint ← ProjSurf(SurfA, PointSetBorder);
    ProjSet.push(ProjPoint);
    N ⟵ InnerGroupNum(ProjPoint);
    NumOfPatch += N;
OverlapBorder = ∅;
PatchSet = ∅;
for ProjPoint ∈ projection points of ProjSet do
    if IsOnBorder(ProjPoint.begin()) and
    IsOnBorder(ProjPoint.end()) then
        TempPatchSet ← NonOverlapFunc(ProjPoint);
        for TempPatch ∈ TempPatchSet do
            PatchSet.push(TempPatch);
TempPatch ← OverlapFunc(ProjSet);
PatchSet.push(TempPatch);
```

the four boundaries of surface **A** are generated and projected on surface **B**. The projection points of the four boundaries located on surface **B** are plotted with different types of points. To decrease the computation time for the projection process, the generated sample points from the boundaries have bigger interval than the inner part. In other words, the number of sample points from the boundaries is smaller than that from the interior.

According to the previous projection points located in the interior of the surface **B**, a connective boundary point set can be constructed. During this construction, the sample points of surface **B** are added to the adjacent projection points by checking their knot values. The first and the last point of this point set must exactly locate on the boundaries of surface **B**. Based on this principle, all the boundary point sets of this general example are shown in Fig. 4. If the number of group of interior boundary sets are defined as n, the number of divided surface patches must be $n + 1$, because each boundary point set located in the interior of surface **B** leads to a non-overlapping surface patch. By dealing with these interior boundary points, non-overlapping surface patches and overlapping surface patch are obtained by using the following algorithms.

3.1.1. Non-overlapping patch formation

In Algorithm 2, there are three cases for each group of projection points located in the interior of the goal surface. These cases are distinguished by checking the locations of the first point, defined as **PA**, and the last point, defined as **PB**, of each group. When these

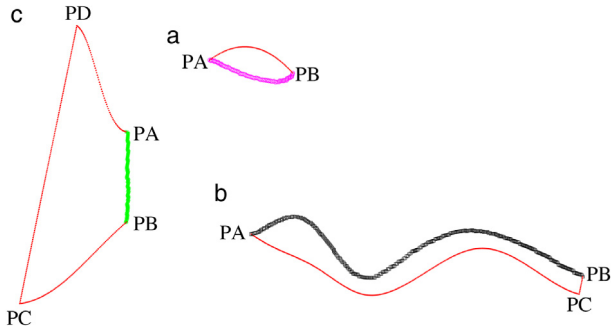


Fig. 5. Three location cases for the first and the last point of an interior boundary point set relative to the boundaries of surface **B**. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

two points are located on the same boundary of a surface, the sample points of this boundary can be easily obtained. The new obtained group of boundary points and the group of projection points form the close-loop boundary points of a surface patch, and its interior points can be found by considering the knot value of these boundary points. In the case (a) of Fig. 5, **PA** and **PB** of the interior boundary points plotted with pink circles locate on a same boundary of surface **B**. By selecting the boundary points of surface **B** between **PA** and **PB**, the close-loop boundary points of this non-overlapping surface patch is shown with red dots.

For the case of **PA** and **PB** of the projected boundary points are located on the two adjacent boundaries of a surface, the intersected

Algorithm 2: NonOverlapFunc

Input : Projection points *ProjPoint*, sample points *SampSetA* of surface A.
Output : Surface patches *TempPatchSet*.

```
//PC, PD are the corner points of surface B;
//TBA, TBB, TBC are a set of boundary points between two given points;
//TIA is a set of points located in the interior of surface A;
//TBP, TIP are two point sets storing all the boundary points of a non-overlapping surface patch;
PA = ProjPoint.begin(); PB = ProjPoint.end();
TempPatchSet = ∅;
if IsSameBorder(PA, PB) then
    InnerSet ← GetInnerGroups(ProjPoint);
    for TempInner ∈ InnerSet do
        TBP ← GetBorderPoints(TempInner);
        TIP ← GetInsidePoints(TempInner);
        PatchPoints ← GetPatchPoints(TBP, TIP, SampleSetA);
        NewSurf ← SurfExt(PatchPoints);
        TempPatchSet.push(NewSurf);
    return;
if IsAdjacentBorder(PA, PB) then
    PC ← FindCornerPoint(ProjPoint);
    TBA ← GetBorderPoints(PA, PC);
    TBB ← GetBorderPoints(PC, PB);
    TBP ← CombineBorderPoints(TBA, TBB);
    TIP ← GetInsidePoints(ProjPoint);
if IsOppositeBorder(PA, PB) then
    (PC, PD) ← FindTwoCorners(ProjPoint);
    TBA ← GetBorderPoints(PA, PD);
    TBB ← GetBorderPoints(PB, PC);
    TBC ← GetBorderPoints(PC, PD);
    TIA ← GetInsidePoints(ProjPoint);
    TBP ← CombineBorderPoints(TBA, TBC);
    TIP ← CombineBorderPoints(TBB, TIA);
DataSet ← GetPatchPoints(TBP, TIP, SampSetA);
NewSurf ← SurfExt(DataSet);
TempPatchSet.push(NewSurf);
```

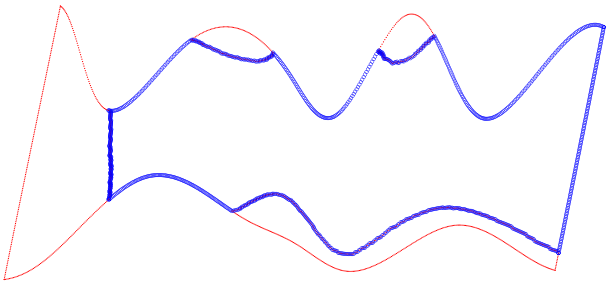


Fig. 6. Boundary points of the non-overlapping patches (red dot) and the overlapping patch (blue circle) for surface **B**. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

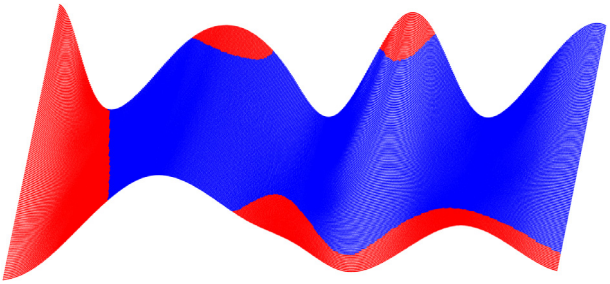


Fig. 7. Sample points of the non-overlapping patches (red dot) and the overlapping patch (blue point) for surface **B**. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

corner points, defined as **PC**, of these two boundaries should be found. In case (b) of Fig. 5, a group of interior boundary points plotted with black squares whose **PA** and **PB** are located on the two adjacent boundaries of surface **B**. The intersection point **PC** of these two boundaries is one corner point of surface **B**. To construct a close-loop boundary points for this non-overlapping surface patch, the generated boundary points of surface **B** between **PA** and **PC** and between **PB** and **PC** are selected and plotted with red dots.

If **PA** and **PB** of the projected boundary points are located on two opposite boundaries of a surface, two more corner points are needed to obtain the remaining boundary points of a non-overlapping surface patch. In case (c) of Fig. 5, a group of interior boundary points are plotted with green stars, its **PA** and **PB** are located on the top boundary and the bottom boundary of surface **B**, respectively. By checking the knot value of the green projected points located on the boundaries of surface **B** in Fig. 3, the two corner points on the left of surface **B** are selected, which are defined as **PC** and **PD** in following description. The sample points of the boundaries of surface **B** between the three pairs of points [**PB**, **PC**], [**PC**, **PD**], and [**PD**, **PA**] are selected, and the constructed close-loop boundary points are plotted with red dots.

3.1.2. Overlapping patch formation

By using the above methods, the close-loop boundary points for all the non-overlapping surface patch of the example in Fig. 3 are shown in Fig. 6. In case (a), (b), and (c) of Fig. 5, the number of the boundary point sets is two, three, and four, respectively. If the number of the point sets of a non-overlapping surface patch is bigger than two, these points need to be rearranged and combined as two. Then, the sample points located in the interior these two point sets can be selected by checking the knot values of these boundary points. The selected sample points for all the non-overlapping surface patch are shown in Fig. 7. Based on these sample points, the corresponding B-spline surfaces can be extracted.

When the boundary points of all non-overlapping surface patches for surface **B** are obtained, the boundary points of the

Algorithm 3: OverlapFunc

Input : All border points *ProjSet*, sample points of surface A *SampSetA*.
Output : Surface patch *TempPatch*.

```
BorderSet = ∅;
//SP is the sample point with u = 0 or v = 0;
//EP is the sample point with u = 1 or v = 1;
//BO, BS are two point sets storing all the boundary points of an overlapping surface patch;
for ProjPoint ∈ projection points of ProjSet do
    TempBor = ∅;
    if IsAllInside(ProjPoint) then
        TempBor ← GetInsidePoints(ProjPoint);
    else
        InnerSet ← GetInnerGroups(ProjPoint);
        for temp ∈ InnerSet do
            if temp == InnerSet.begin() then
                TA ← GetBorderPoints(SP, temp.begin());
                TempBor.connect(TA);
            TIP ← GetInsidePoints(temp);
            TempBor.connect(TIP);
            PA = temp.end();
            if temp == InnerSet.end() then
                TB ← GetBorderPoints(PA, EP);
            else
                PB = (temp + 1).begin();
                TB ← GetBorderPoints(PA, PB);
            TempBor.connect(TB);
    BorderSet.push(TempBor);
(BO, BS) ← CombineBorderSet(BorderSet);
DataSet ← GetPatchPoints(BO, BS, SampSetA);
TempPatch ← SurfExt(DataSet);
```

overlapping surface patch is selected by using Algorithm 3. If a whole group of projected boundary points are located in the interior of the goal surface, the selected sample points for this whole boundary can be easily obtained. If some partial points of this group are located onto the boundaries of the goal surface, the whole projected boundary points must be obtained by combining sample interior sample points and the sample points of the boundaries of this surface. In Fig. 3, the top boundary of overlapping surface patch for surface **B** includes two groups of interior sample points and three groups of sample points of the boundaries. The two groups of interior sample points have already obtained in the previous analysis of non-overlapping surface patches. The remaining three groups of points can be selected by considering the knot values between the end points of the interior point set. By repeating this process, the following three point sets of the boundaries of the overlapping surface patch are obtained. All the boundary points of this overlapping surface patch for surface **B** are shown in Fig. 6 with blue circles.

Similar to the non-overlapping surface patch, the boundary point sets of the overlapping surface patch are combined as two sets. The interior sample points are obtained by checking their knot values, which are shown in Fig. 7. Based on the sample points of this overlapping patch, the selected sensor data for this patch from the sensor data of surface **B** are shown in Fig. 8 with blue pentagram. To merge these selected sensor data with the overlapping patch of surface **A**, generated sample points from the boundaries of surface **B** are projected onto surface **A** in Fig. 9.

3.1.3. Patch formation for surface *A*

By repeating the previous process of Algorithm 1, the projected boundary points located in the interior of surface **A** are presented in Fig. 10. A different point from the previous process is that the projection points plotted with the pink circles and blue triangles

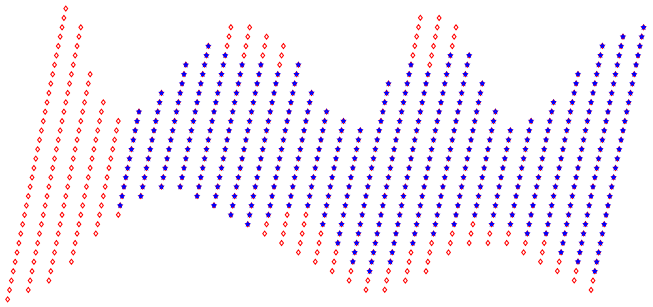


Fig. 8. Selected sensor data (blue pentagram) for the overlapping surface patch from all sensor data (red rhombus). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

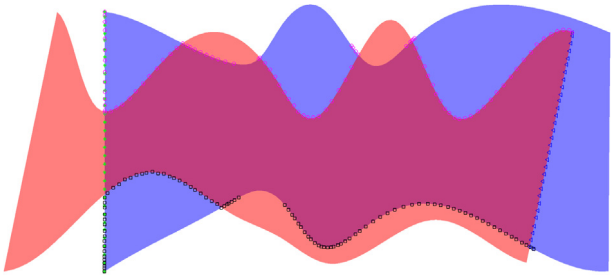


Fig. 9. Projection of generated sample points from the boundaries of surface **B** onto surface **A**.

Fig. 10. Projected boundary points located in the interior of surface **A**. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

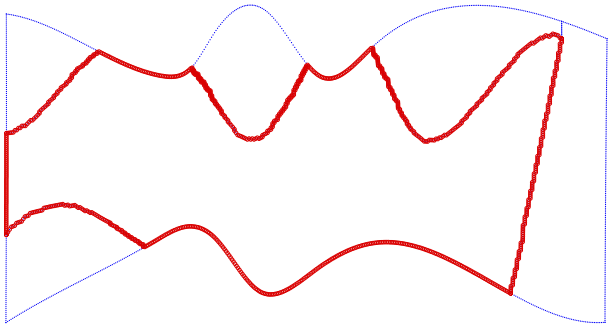


Fig. 11. Boundary points of the non-overlapping patches (blue dot) and the overlapping patch (red circle) for surface **A**. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

have a common intersection point. This is because one projected corner point of surface **B** is located in the interior of surface **A**. By using the Algorithm 2 and Algorithm 3, the boundary points of all the non-overlapping surface patches and the overlapping surface patch are obtained as in Fig. 11. It can be seen from the figure that the right-top surface patch is divided into two parts because of the projected corner point. In this division process, the common

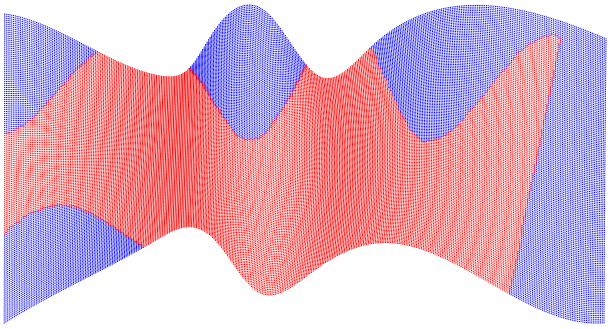


Fig. 12. Sample points of the non-overlapping patches (blue dot) and the overlapping patch (red dot) for surface **A**. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

boundary points of these two patches are the connection points of the projected corner point and a boundary point which have the same knot value in u or v direction referring to knot vectors U or V. To have a nice division result, the boundary point which has the smallest difference of the knot value with the projected corner point is selected. As can be seen from that figure, the sample point located on the top boundary of surface **A** is used to divide the non-overlapping surface patch instead of the right boundary, because the difference of the knot value between this point and the projected corner point is the smallest.

Based on these obtained close-loop boundary points of all the non-overlapping surface patches and the overlapping surface patches, the interior sample points for each patch are obtained as in Fig. 12. Then one B-spline surface can be extracted from these sample points for each non-overlapping patch. By comparing the points plotted with blue dot in Fig. 7 and the points plotted with red dot in Fig. 12, it can be seen that the selected points of the overlapping surface patch for both surface **A** and surface **B** construct an identical shape. It validates that the proposed surface division algorithms work well for dealing with the division of B-spline surface with complex shape.

3.2. Surface merging

To merge the selected sensor data (blue pentagram) of surface **B** in Fig. 8 and its overlapping surface patch of surface **A**, all these selected data points are projected onto the overlapping patch of surface **A**. The projection process leads a projection point for each selected sensor data. Therefore, merging of two overlapping surface patches is the merging of the selected sensor data and their projection points. To have a more accurate merging result, these two points are probabilistically merged based on their covariance matrix. The covariance matrix of the projection point can be derived from the covariance matrix of the control points of surface **A** [22]. The merging is realized by using the product of Gaussian probability density functions as follows:

$$\mu = \left(\Sigma_s^{-1} + \Sigma_p^{-1}\right)^{-1} \left(\Sigma_s^{-1}\mu_s + \Sigma_p^{-1}\mu_p\right) \tag{7a}$$

$$\Sigma = \left(\Sigma_s^{-1} + \Sigma_p^{-1}\right)^{-1} \tag{7b}$$

where μ_s, μ_p represent the coordinate value of a sensor point and a projection point, respectively, and Σ_s, Σ_P represent the corresponding covariance matrix.

After obtaining the updated data, previous sensor data belonging to the overlapping surface are replaced with these updated data. In Fig. 13, all the updated sensor data are plotted with red squares and selected sample data of non-overlapping surface patches of surface **A** are plotted with blue dots. It can be seen that the number of sensor data are smaller than that of sample points.

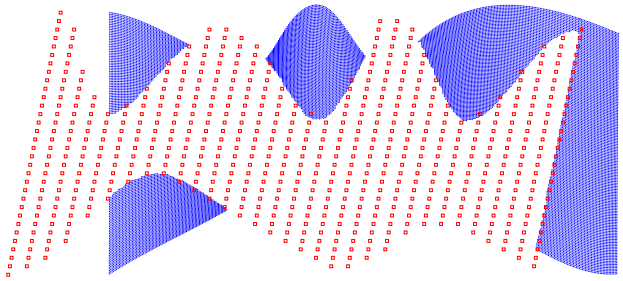


Fig. 13. Updated sensor data (red square) and the sample points (blue dot) of the non-overlapping patches. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

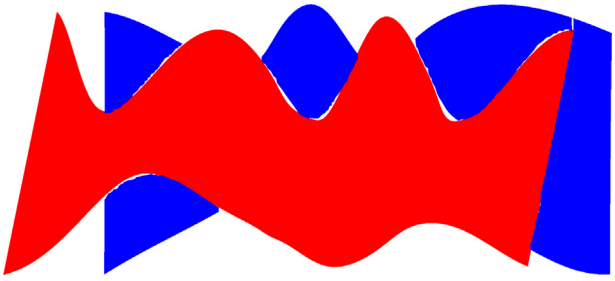


Fig. 14. All extracted surface patches from the updated sensor data and the sample data.

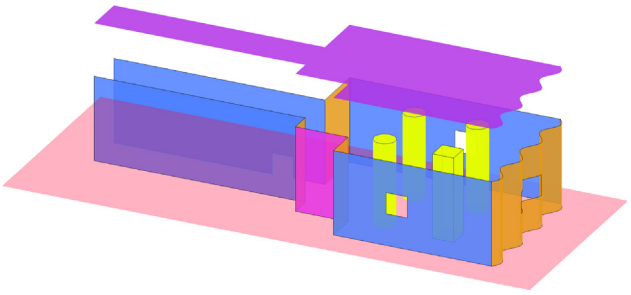


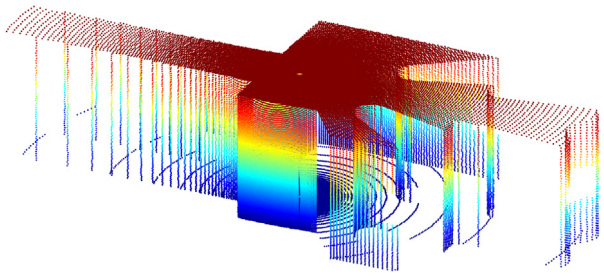
Fig. 15. A simulated house.

This leads small cracks between the extracted surface patches in Fig. 14. Some of these cracks can be stitched when they are overlapped with a B-spline surface extracted from a new obtained scan in a following step. For the remaining cracks, they can be stitched by merging its adjacent B-spline surfaces which are not overlapping with each other by using the methods of [15–17].

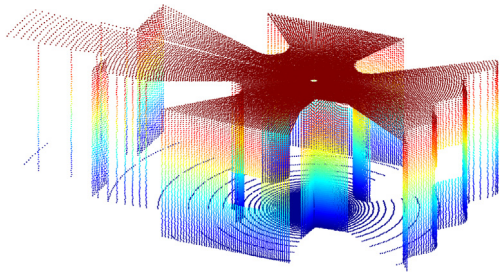
4. Simulation results

A simulated house is plotted in Fig. 15 to generate simulated sensor data. In the considered simulated environment, curved walls, windows, square columns and circular columns exist. Then, a 3D laser scanner including a vertical 2D laser scanner and a rotation motor is simulated to scan the simulated house at two different positions. Two 3D scans from the simulated house are shown in Fig. 16 in which analysis of 3D scan **A** are in the left column and that of 3D scan **B** are in the right column. In Fig. 16(a) and (b), the simulated raw sensor data with different heights for these two 3D scans are plotted with different colors. Before extracting surfaces from these two scans, 3D scans should be segmented as different groups.

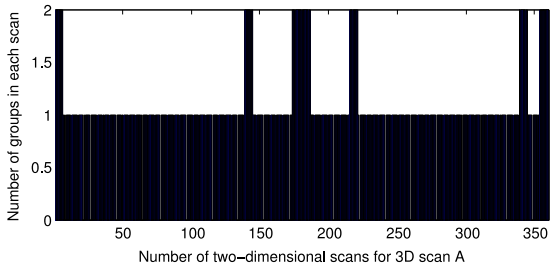
The number of all the 2D scans of the two 3D scans are shown in Fig. 16(c) and (d). It can be seen from these figures that each 2D scan at most can be divided into two groups, because the measurement value of the laser scanner at the locations of



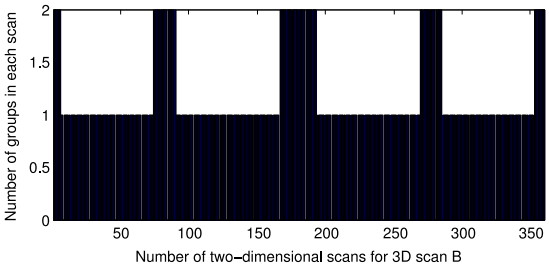
(a) Simulated sensor data of 3D scan **A**.



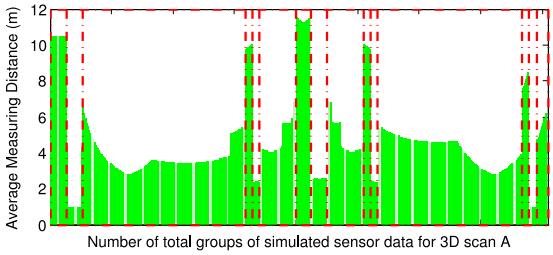
(b) Simulated sensor data of 3D scan **B**.



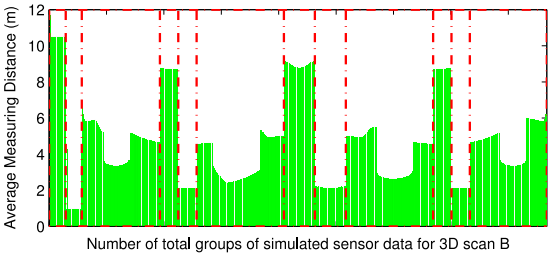
(c) Number of groups of 2D scans for scan **A**.



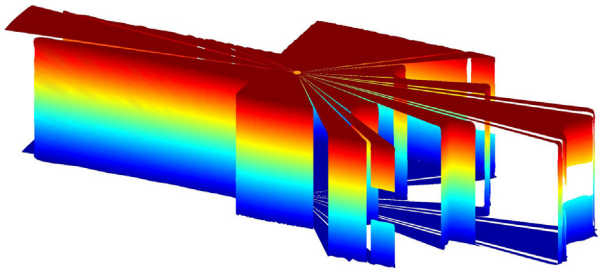
(d) Number of groups of 2D scans for scan **B**.



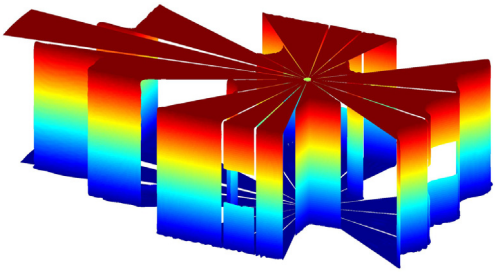
(e) Average measuring distance of raw sensor data in every group and separated data group with red dashed lines for Scan **A**.



(f) Average measuring distance of raw sensor data in every group and separated data group with red dashed lines for Scan **B**.



(g) Extracted surface patches from scan **A**.



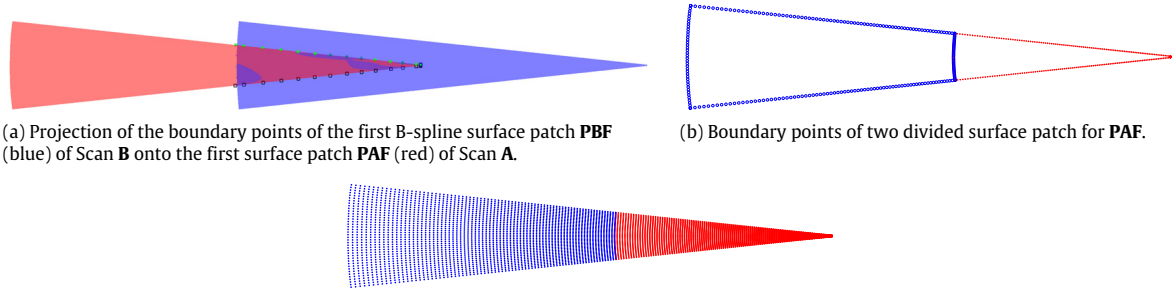
(h) Extracted surface patches from scan **B**.

Fig. 16. Division of two simulated 3D scans and the correspondingly extracted B-spline surfaces. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

windows are zeros. It means that the surface cannot be connected because of these gaps. Continuous 2D scans having only one group of data in each scan are considered as one dataset, and continuous 2D scans having two groups of data are merged as two dataset. For every group of each dataset, its average measuring distance is calculated in Fig. 16(e) and (f), and it is divided into two sets if the difference of average measuring distance for two adjacent groups is beyond a limit value.

Based on these datasets, the extracted surfaces from these two 3D scans are shown in Fig. 16(g) and (h). Most of the surface patches have at least one short boundary where two adjacent corner points are very close. This happens because the obtained raw data points close to the simulated laser scanner have higher density and smaller interval than the points far away from the scanner. It can be seen from the figure that B-spline surfaces can accurately represent the simulated house even in the joining part of two walls where sharp corners exist. There are many overlapping surface patches between these two sets of extracted B-spline surfaces. To show the division and merging process of 3D B-spline surfaces of these two simulated scans, one pair of surfaces from 3D Scan **A** and 3D Scan **B** is presented in Fig. 17. In Fig. 17(a), the boundary points of right blue surface defined as **PBF** are projected onto left red surface defined as **PAF**, both of which belong to the ceiling of simulated house. By using our proposed division algorithms, boundary points of separated surface patches for **PAF** and selected sample points for these patches are presented in Fig. 17(b) and (c), respectively.

To select the raw sensor data for the overlapping surface patch of **PBF**, boundary points of **PAF** are projected on **PBF** in Fig. 18(a). According to these projected points, surface **PBF** are divided into four patches, in which there are three non-overlapping surface patches because two corner points of surface **PAF** locate inside

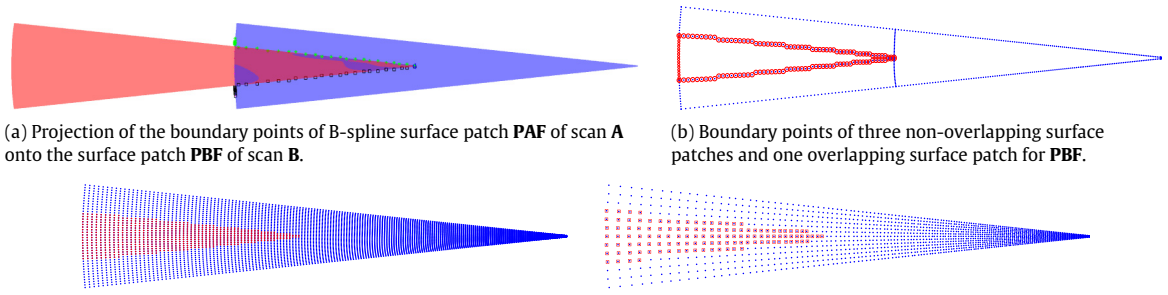


(a) Projection of the boundary points of the first B-spline surface patch **PBF** (blue) of Scan **B** onto the first surface patch **PAF** (red) of Scan **A**.

(b) Boundary points of two divided surface patch for **PAF**.

(c) Generated sample points for the two divided patches.

Fig. 17. Division of first surface patch **PAF** of Scan **A**. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) Projection of the boundary points of B-spline surface patch **PAF** of scan **A** onto the surface patch **PBF** of scan **B**.

(b) Boundary points of three non-overlapping surface patches and one overlapping surface patch for **PBF**.

(c) Generated sample points for the four divided patches.

(d) The selected raw data points for the overlapping surface patch.

Fig. 18. Division of the first surface patch **PBF** of Scan **B**.

(a) Sample points of the non-overlapping surface patch for **PAF**, and the updated raw data points for **PBF**.

(b) Final merged B-spline surface patches extracted from the sample points and the updated raw data points.

Fig. 19. All data points and the merged B-spline surface patches of **PAF** and **PBF**.

of surface **PBF**. Then boundary points of all four surface patches are shown in Fig. 18(b), and the corresponding sample points are selected as in Fig. 18(c). Finally, the raw data points of the overlapping surface patch is selected in Fig. 18(d).

Based on the sample points of non-overlapping surface patch of **PAF** and the updated raw sensor data of **PBF** in Fig. 19(a), finally merged surface patches are shown in Fig. 19(b). It can be seen from the figures that the overlapping part between two patches are merged as one patch. By repeating this division and merging process for all the B-spline surfaces from two 3D scans, all the overlapping patches between any two surfaces are merged and shown in Fig. 20. All these surface patches can accurately express the 3D simulated house in which different colors represent with different heights. There are discontinuities between adjacent B-spline surface patches, which are created in the division process of the 3D scan. This is a limitation of our proposed method because it is quite difficult to represent a large object with one continuous B-spline surface.

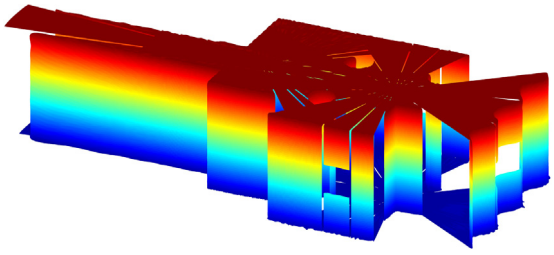


Fig. 20. All B-spline surface patches of 3D scan **A** and 3D scan **B** after the process of division and merging. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5. Conclusions

In this paper, we propose a division and merging method to divide and merge the overlapping parts between a B-spline surface and a new obtained raw sensor data of a 3D scan. In the beginning, all the possible overlapping cases of two overlapping surfaces are listed from a view of the locations of the projection points of four corner of one surface in the interior of its overlapping surface. Based on these analysis, the proposed method is illustrated with pseudo-codes and an example of two 2D B-spline surfaces having an overlapping patch. The detailed process of this example with final results show that the proposed method works well in merging the overlapping surface patch. To further validate proposed methods, two 3D scans obtained from a simulated house are separated into many groups, and a B-spline surface is extracted from each group. By merging the overlaps between any two surface from the two 3D scans, it has been shown that all the B-spline surface patches can accurately express the 3D simulated house without existing overlapping surface patch.

References

[1] Sampath A, Shan J. Segmentation and reconstruction of polyhedral building roofs from aerial lidar point clouds. IEEE Trans Geosci Remote Sens 2010;48(3): 1554–67. http://dx.doi.org/10.1109/TGRS.2009.2030180.

[2] Bae K-H, Belton D, Lichti D. A closed-form expression of the positional uncertainty for 3d point clouds. IEEE Trans Pattern Anal Mach Intell 2009; 31(4):577–90. http://dx.doi.org/10.1109/TPAMI.2008.116.

[3] de Medeiros Brito A, Doria Neto A, Dantas de Melo J, Garcia Goncalves L. An adaptive learning approach for 3-d surface reconstruction from point clouds. IEEE Trans Neural Netw 2008;19(6):1130–40. http://dx.doi.org/10.1109/TNN.2008.2000390.

[4] Wang Z, Zhang L, Fang T, Mathiopoulos P, Tong X, Qu H, Xiao Z, Li F, Chen D. A multiscale and hierarchical feature extraction method for terrestrial laser scanning point cloud classification. IEEE Trans Geosci Remote Sens 2015; 53(5):2409–25. http://dx.doi.org/10.1109/TGRS.2014.2359951.

[5] Jakovljevic Z, Puzovic R, Pajic M. Recognition of planar segments in point cloud based on wavelet transform. IEEE Trans Ind Inf 2015;11(2):342–52. http://dx.doi.org/10.1109/TII.2015.2389195.

[6] Cui Y, Feng J. Gpu-based smooth free-form deformation with sharp feature awareness. Comput Aided Geom Design 2015;3536:69–81. http://dx.doi.org/10.1016/j.cagd.2015.03.002, geometric Modeling and Processing 2015.

[7] Yamaura Y, Nanya T, Imoto H, Maekawa T. Shape reconstruction from a normal map in terms of uniform bi-quadratic b-spline surfaces. Comput-Aided Des 2015;63:129–40. http://dx.doi.org/10.1016/j.cad.2015.01.005.

[8] Burla RK, Kumar AV. Implicit boundary method for analysis using uniform b-spline basis and structured grid. Internat J Numer Methods Engrg 2008;76(13): http://dx.doi.org/10.1002/nme.2390, 1993.

[9] Douros I, Dekker L, Buxton B. An improved algorithm for reconstruction of the surface of the human body from 3d scanner data using local b-spline patches. In: IEEE international workshop on modelling people, 1999. proceedings. 1999, p. 29–36. http://dx.doi.org/10.1109/PEOPLE.1999.798343.

[10] Cohen F, Ibrahim W, Pintavirooj C. Ordering and parameterizing scattered 3d data for b-spline surface approximation. IEEE Trans Pattern Anal Mach Intell 2000;22(6):642–8. http://dx.doi.org/10.1109/34.862203.

[11] He X, Li C, Hu Y, Zhang R, Yang SX, Mittal GS. Automatic sequence of 3d point data for surface fitting using neural networks. Comput Ind Eng 2009; 57(1):408–18. http://dx.doi.org/10.1016/j.cie.2009.01.003, collaborative e-Work Networks in Industrial Engineering.

[12] Ristic M, Brujic D, Handayani S. Cad-based triangulation of unordered data using trimmed {NURBS} models. J Mater Process Technol 2000;107(13):60–70. http://dx.doi.org/10.1016/S0924-0136(00)00691-9.

[13] Eck M, Hoppe H. Automatic reconstruction of b-spline surfaces of arbitrary topological type. In: Proceedings of the 23rd annual conference on computer graphics and interactive techniques. SIGGRAPH '96, New York, NY, USA: ACM; 1996. p. 325–34. http://dx.doi.org/10.1145/237170.237271.

[14] Yin Z. Direct generation of extended {STL} file from unorganized point data. Comput-Aided Des 2011;43(6):699–706. http://dx.doi.org/10.1016/j.cad.2011.02.010.

[15] Pungotra H, Knopf GK, Canas R. Merging multiple -spline surface patches in a virtual reality environment. Comput-Aided Des 2010;42(10):847–59. http://dx.doi.org/10.1016/j.cad.2010.05.006.

[16] Pungotra H, Knopf GK, Canas R. An alternative methodology to represent b-spline surface for applications in virtual reality environment. Comput-Aided Des Appl 2013;10(4):711–26. http://dx.doi.org/10.3722/cadaps.2013.711-726.

[17] Chen J, Wang G-j. Approximate merging of b-spline curves and surfaces. Appl Math J Chinese Univ 2010;25(4):429–36. http://dx.doi.org/10.1007/s11766-010-2169-1.

[18] Pulli K, Shapiro LG. Surface reconstruction and display from range and color data. Graph Models 2000;62(3):165–201. http://dx.doi.org/10.1006/gmod.1999.0519.

[19] Bernardini F, Martin IM, Rushmeier H. High-quality texture reconstruction from multiple scans. IEEE Trans Vis Comput Graphics 2001;7(4):318–32. http://dx.doi.org/10.1109/2945.965346.

[20] Marton ZC, Rusu RB, Beetz M. On fast surface reconstruction methods for large and noisy point clouds. In: IEEE international conference on robotics and automation, 2009. ICRA'09. 2009, p. 3218–23. http://dx.doi.org/10.1109/ROBOT.2009.5152628.

[21] Zha H, Malkimoto Y, Hasegawa T. Surface reconstruction in overlapping range images for generating close-surface 3-d object models. In: 1998 IEEE international conference on systems, man, and cybernetics, 1998. vol. 5, 1998, p. 4530–5. http://dx.doi.org/10.1109/ICSMC.1998.727564.

[22] Yan RJ, Wu J, Lee JY, Han C-S. Representation of 3d environment map using b-spline surface with two mutually perpendicular lrfs. Math Probl Eng 2015; 2015:14. http://dx.doi.org/10.1155/2015/690310, (Article ID 690310).

[23] Piegl L, Tiller W. The NURBS book. 2nd ed. New York, NY, USA: Springer-Verlag New York, Inc.; 1997.