DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-398

Truncated Signed Distance Function Volume Integration Based on Voxel-Level Optimization for 3D Reconstruction

Fei Li, Yunfan Du, and Rujie Liu; Fujitsu Research & Development Center Co., Ltd.; Beijing, China

Abstract

3D reconstruction has been an active research topic with the popularity of consumer-grade range cameras, and the whole process mainly consists of registration and integration. Most recent methods pay their attention to making depth maps aligned with each other, but the step of integration is simply conducted by weighted average for the volumes of truncated signed distance function (TSDF), thus the relationship between individual and integrated TSDF representations is not well explored. In this paper, under the framework of voxel-level optimization, a novel method is proposed for TSDF volume integration. Considering camera distortions, each individual TSDF volume is corrected by a non-rigid transformation. Based on the consistency of TSDF values of individual and integrated volumes, both the final global TSDF representation and the transformation parameters are calculated by solving the optimization problem. Experimental results demonstrate that more satisfactory reconstruction performance can be obtained by our proposal.

Introduction

As one of the most important goals in computer vision and graphics, obtaining digital representations of real-world objects has been a hot research issue in the last decades. Among different kinds of methods, 3D reconstruction based on depth maps is a promising way. With the development of depth acquisition technology, consumer-grade range cameras have appeared. Due to their low cost, easy portability and high frame rate for streaming depth maps, consumer-grade range cameras have been widely used in various applications, such as computer games, augmented reality, and 3D printing. However, these cameras are often with inevitable distortions, and the obtained depth maps are not accurate enough for high-precision modeling [1]. Therefore, in recent years, how to get satisfactory 3D reconstruction performance with inaccurate capture devices has attracted more and more attention [2], [3], [4], [5], [6], [7], [8], [9], [10].

Generally speaking, there are two main steps in the whole process of 3D reconstruction based on depth maps: registration and integration [11], [12]. Since the available depth maps are captured from different positions and directions, they are first aligned into the same coordinate space in the step of registration. Then in the step of integration, the aligned depth maps are combined for the final reconstruction results.

Although much work has been dedicated to exploring 3D reconstruction based on depth maps, most recent methods mainly focus on the step of registration. The basic idea of some common methods is to estimate the camera pose of each depth map as accurate as possible, and the two frequently adopted approaches are frame-to-frame matching [13] and frame-to-model matching [2], [3]. The former approach estimates the camera pose of each new depth map by registering it to its last frame, while the latter one aligns the incoming depth map to the growing model constructed by all the frames coming before. Since more useful information is effectively involved in frame-to-model matching, the second approach significantly outperforms the first one [3]. To take the frames after the incoming depth map into consideration and to achieve more accurate results, the approach of two-pass registration is proposed [4]. A whole model is constructed by all the available depth maps in the first pass, and then each frame is aligned to the obtained whole model in the second pass. For performance improvement, some methods further deal with camera distortions in the step of registration. Two ways are mainly adopted for this purpose. One utilizes elaborate calibration and attempts to estimate a specific distortion function for the given camera, although the function is usually irregular and complicated [14], [15]. The other tries to correct the distortions by introducing non-rigid deformation to the acquired data [5]. Although the second way needs neither specialized calibration sequences nor additional assumption, it always leads to unnecessary warping to the final reconstruction results due to the lack of prior knowledge. Moreover, its computational cost is quite high. By factorizing the non-rigid deformation into a rigid localization component and a latent non-rigid calibration component, a method conducts localization and calibration simultaneously [7]. It achieves better reconstruction results. Meanwhile, the total computational load is greatly reduced.

In contrast to many research achievements on registration, the approach used for the step of integration is always quite simple. For a typical 3D reconstruction system, it often maintains a model represented by volumetric truncated signed distance function (TSDF). When a new depth map comes, after camera pose estimation, its corresponding TSDF volume is calculated, and the volume is integrated with the global TSDF volume by weighted average. This approach implies that every individual TSDF volume has already been well aligned, thus the integration performance directly depends on the registration results. However, even much related work has been developed for registration, due to the complexity of unknown distortions, perfect alignment results for multiple TSDF volumes cannot be obtained in practice. If camera distortions are also taken into account for the step of integration, it is hoped that the final global TSDF representation will become more accurate, and better reconstruction for real-world objects can be obtained.

In this paper, in the framework of voxel-level optimization, a novel method for TSDF volume integration is proposed. Without the assumption that all the individual TSDF volumes have already been aligned, we try to explore the relationship between individual and integrated TSDF representations. In order to make them aligned with each other, we introduce a suitable transformation for each individual TSDF volume before it is integrated into the final global one. Considering the complexity of unknown camera distortions, non-rigid transformation is adopted in our proposal. A problem involving both the final global TSDF representation and the transformation parameters is defined, and all the variables are optimized to maximize the consistency of TSDF values of

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-398

individual and integrated volumes. By combining the transformed individual TSDF volumes together, a more accurate global TSDF representation is acquired, which makes the final reconstruction more satisfactory. Furthermore, voxel-level transformation does not lead to unreasonable local displacements, which often result from point-level range camera calibration.

The rest of the paper is organized as follows. The next section describes our proposed TSDF volume integration method based on voxel-level optimization in detail. Then our experimental results are illustrated, and it is followed by some conclusions and analysis of future work in the last section.

TSDF Volume Integration Based on Voxel-Level Optimization

In this section, first we present our proposed optimization framework for TSDF volume integration. Since the problem is quite complicated, we discuss how to solve it in the second part. At last, we talk about some implementation issues to speed up the solving process.

Optimization Framework

Suppose there are altogether N individual TSDF volumes, and they are discretized into voxels with a predefined resolution. Similar to the existing methods, each volume is represented by a TSDF value $F_n(\mathbf{p})$ and a corresponding weight $W_n(\mathbf{p})$, where $n = 1, 2, \cdots, N$, and $\mathbf{p} \in \mathbb{R}^3$ is a point in the volume, usually the central point of one voxel. Let the global TSDF value for the point $\mathbf{p}$ be denoted as $F(\mathbf{p})$, which needs to be calculated in the step of integration.

Since multiple individual TSDF volumes may not be well registered, we introduce suitable non-rigid transformations for each of them, and try to make the transformed volumes aligned with each other. For the n-th $(n = 1, 2, \cdots, N)$ individual TSDF volume, let its corresponding non-rigid transformation be T_n. Considering the relationship between individual and integrated TSDF representations, the point $\mathbf{p}$ in the integrated volume corresponds to the point $T_n(\mathbf{p})$ in the n-th individual TSDF volume.

The basic idea for our proposed optimization framework is to maximize the consistency of TSDF values of individual and integrated volumes. That is to say, it is expected that the value of $F(\mathbf{p})$ in the integrated volume and its corresponding values $F_n(T_n(\mathbf{p}))$ $(n = 1, 2, \cdots, N)$ in the individual volumes should be as close as possible. By summing over all the points and all the individual volumes, the cost term is defined as

$$E_c = \sum_n \sum_{\mathbf{p}} \left(F(\mathbf{p}) - F_n(T_n(\mathbf{p})) \right)^2 \tag{1}$$

Usually each point in the TSDF volume is assigned with a weight, which indicates the certainty of the corresponding TSDF value. Therefore, the importance of each TSDF value is different. For a given point, its global TSDF value should be closer to the individual TSDF value with higher certainty, namely, with larger weight. Therefore, the weights should also be involved into the optimization problem. Using the aforementioned notations, the cost term is modified as

$$E_c = \sum_n \sum_{\mathbf{p}} W_n(T_n(\mathbf{p})) \left(F(\mathbf{p}) - F_n(T_n(\mathbf{p})) \right)^2 \tag{2}$$

Next we give the mathematical description of the non-rigid transformation T_n $(n = 1, 2, \cdots, N)$. Since the type of T_n may be complicated and cannot be determined in advance, the transformation is simply defined as a mapping from 3D space to 3D space. It is obvious that due to the large number, the mapping results cannot be explicitly defined for all the involved points. Therefore, we just directly define the results for some pre-given points, and calculate other results by interpolation. In order to determine the pre-given points, a uniform lattice $D = \{\mathbf{d}_m\} \subset \mathbb{R}^3$ is constructed, and the transformation T_n for the lattice points is defined as

$$T_n(\mathbf{d}_m) = \mathbf{d}_m + \mathbf{s}_{n,m} \tag{3}$$

where $\mathbf{s}_{n,m}$ is the parameter for describing the transformation, and it indicates the displacement for the point $\mathbf{d}_m$ by the n-th transformation. It should be noted that although the introduced transformations for each individual TSDF volume are usually different, the lattice is the same for all the TSDF volumes to simplify the following calculation. Based on the mapping results on the lattice points, the transformation is extended to other points by interpolation

$$T_n(\mathbf{p}) = \mathbf{p} + \sum_m \mu_m(\mathbf{p})\, \mathbf{s}_{n,m} \tag{4}$$

where $\mu_m(\mathbf{p})$ is the interpolation coefficient. The summation can be conducted over all the points in the lattice, but in our implementation, only 8 nearest neighboring lattice points are assigned with non-zero coefficients, and all the coefficients for other lattice points are set to zero. In this case, the mapping results for the point $\mathbf{p}$ can be calculated by trilinear interpolation, and the coefficients can be easily determined by the relative position between the point $\mathbf{p}$ and its corresponding nearest neighboring lattice points.

In the above definition for the non-rigid transformation, if the parameter, namely, the displacement $\mathbf{s}_{n,m}$, is not given any restriction, all the points may be mapped to the same point. If this case happens, although the cost term can get its minimum value, the optimal solution for the integrated TSDF volume is meaningless. Therefore, each lattice point should be mapped to its nearby point, and the magnitude of the displacement should be restricted. To make the optimization problem easy to solve, a quadratic regularization term is added as

$$E_r = \sum_n \sum_m \|\mathbf{s}_{n,m}\|^2 \tag{5}$$

At last, the final cost function is defined by combining the cost term and the regularization term together

$$\begin{aligned} E &= E_c + \lambda E_r \\ &= \sum_n \sum_{\mathbf{p}} W_n(T_n(\mathbf{p})) \left(F(\mathbf{p}) - F_n(T_n(\mathbf{p})) \right)^2 \\ &\quad + \lambda \sum_n \sum_m \|\mathbf{s}_{n,m}\|^2 \end{aligned} \tag{6}$$

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-398

where λ is a balanced coefficient for the two terms. From their definitions it can be seen that the value of E_c is related to all the involved points, and the value of E_r grows with number of lattice points. Generally speaking, if a sparser lattice is adopted, λ should be set to a larger value. In our implementation, each TSDF volume includes 512^3 voxels, the adopted lattice is with 9^3 control points, and λ is to 10 in all the experiments. By minimizing the overall cost function, the final global TSDF representation, as well as all the transformation parameters, can be calculated.

Solution to Optimization Problem

Due to the complicated description for our introduced non-rigid transformations, as well as the uncommon form of the cost function, it is difficult to directly solve the optimization problem in Eqn. (6). Based on the idea of approximately replacing $W_n\big(T_n(\mathbf{p})\big)$ by some already known values, an iterative solution framework is proposed in this paper.

First we analyze the property of the weight associated with TSDF value. Generally speaking, it is calculated as follows. For a TSDF volume constructed by one depth map, if a point $\mathbf{p}$ appears near the object surface and is assigned with a meaningful TSDF value, its weight is set to 1; otherwise, its weight is set to 0. Further, if a TSDF volume is constructed by multiple depth maps, the weight for a point $\mathbf{p}$ is defined as the summation of its weights in the TSDF volumes corresponding to each depth map. From the calculation process, it can be inferred that the value of the weight $W_n(\mathbf{p})$ changes slowly over 3D space.

Then, we talk about the introduced transformation. Since a regularization term as Eqn. (5) is added in the cost function, smaller displacement $\mathbf{s}_{n,m}$ is preferred for the optimal solution. That is to say, it is probable that the points $\mathbf{p}$ and $T_n(\mathbf{p})$ are close to each other.

Considering the aforementioned two factors, we propose the following iterative solution for the optimization problem. In the beginning, we roughly take the place of $W_n\big(T_n(\mathbf{p})\big)$ by $W_n(\mathbf{p})$, and deal with the cost function as

$$E^{(1)} = \sum_n \sum_{\mathbf{p}} W_n(\mathbf{p}) \left(F^{(1)}(\mathbf{p}) - F_n\left(T_n^{(1)}(\mathbf{p})\right)\right)^2 + \lambda \sum_n \sum_m \left\|\mathbf{s}_{n,m}^{(1)}\right\|^2 \quad (7)$$

In the $(k+1)$-th $(k = 1, 2, \cdots)$ round of iteration, we proximately replace $W_n\left(T_n^{(k+1)}(\mathbf{p})\right)$ by $W_n\left(T_n^{(k)}(\mathbf{p})\right)$, thus the overall cost function is modified as

$$E^{(k+1)} = \sum_n \sum_{\mathbf{p}} W_n\left(T_n^{(k)}(\mathbf{p})\right) \left(F^{(k+1)}(\mathbf{p}) - F_n\left(T_n^{(k+1)}(\mathbf{p})\right)\right)^2 + \lambda \sum_n \sum_m \left\|\mathbf{s}_{n,m}^{(k+1)}\right\|^2 \quad (8)$$

It can be seen that if we set $T_n^{(0)}(\mathbf{p}) = \mathbf{p}$, Eqn. (7) can also be integrated in Eqn. (8). Since the transformation $T_n^{(k)}$ has already been calculated in the last round, the value of $W_n\left(T_n^{(k)}(\mathbf{p})\right)$ can be obtained in the $(k+1)$-th round. In this way, the optimization becomes easier to be solved.

Next, we discuss the solution to the simplified problem. For convenience, we omit all the superscripts, and use $W_n(\mathbf{q})$ to substitute for $W_n\left(T_n^{(k)}(\mathbf{p})\right)$ since it is just a constant. Thus, the general form of the cost function can be written as

$$E = \sum_n \sum_{\mathbf{p}} W_n(\mathbf{q}) \left(F(\mathbf{p}) - F_n\big(T_n(\mathbf{p})\big)\right)^2 + \lambda \sum_n \sum_m \left\|\mathbf{s}_{n,m}\right\|^2 \quad (9)$$

The problem of minimizing E can be treated as joint optimization for the global TSDF representation $F(\mathbf{p})$ and the transformation parameters $\mathbf{s}_{n,m}$. In this paper, we use an iterative approach to solve it.

At first, all the transformation parameters $\mathbf{s}_{n,m}$ are set to 0, thus $T_n(\mathbf{p}) = \mathbf{p}$, and Eqn. (9) can be written as

$$E = \sum_n \sum_{\mathbf{p}} W_n(\mathbf{q}) \big(F(\mathbf{p}) - F_n(\mathbf{p})\big)^2 \quad (10)$$

In this case, the global TSDF value for each point $\mathbf{p}$ can be obtained, and the problem of linear least squares has a solution in closed form

$$F(\mathbf{p}) = \sum_n W_n(\mathbf{q}) F_n(\mathbf{p}) \Big/ \sum_n W_n(\mathbf{q}) \quad (11)$$

With fixed $F(\mathbf{p})$, we need to calculate $\mathbf{s}_{n,m}$. It is obvious that in this case, the cost function can be decomposed into N independent terms E_n $(n = 1, 2, \cdots, N)$

$$E_n = \sum_{\mathbf{p}} W_n(\mathbf{q}) \left(F(\mathbf{p}) - F_n\big(T_n(\mathbf{p})\big)\right)^2 + \lambda \sum_m \left\|\mathbf{s}_{n,m}\right\|^2 \quad (12)$$

Thus the parameters for each transformation T_n can be dealt with separately. For a given n, by concatenating all the parameters $\mathbf{s}_{n,m}$ into one vector $\mathbf{s}_n$, the optimization becomes a problem of non-linear least squares. Let

$$r_{n,\mathbf{p}} = \sqrt{W_n(\mathbf{q})} \left(F(\mathbf{p}) - F_n\big(T_n(\mathbf{p})\big)\right) \quad (13)$$

$$r_{n,m,t} = \sqrt{\lambda}\, s_{n,m,t} \quad (14)$$

where $t = 1, 2, 3$, and $s_{n,m,t}$ is the t-th element in the vector $\mathbf{s}_{n,m}$. Then we have

$$E_n = \sum_{\mathbf{p}} r_{n,\mathbf{p}}^2 + \sum_m \sum_t r_{n,m,t}^2 \quad (15)$$

The residual vector $\mathbf{r}_n$ is defined by combining all $r_{n,\mathbf{p}}$ and $r_{n,m,t}$ together, and its Jacobian matrix $\mathbf{J}_n$ is calculated accordingly. Hence the problem of non-linear least squares can be solved by the

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-398

Gauss-Newton method, in which the vector $\mathbf{s}_n$ to be solved is updated as

$$\mathbf{s}_n^{(l+1)} = \mathbf{s}_n^{(l)} + \Delta\mathbf{s}_n \tag{16}$$

And the incremental vector $\Delta\mathbf{s}_n$ is determined by the following linear equation

$$\mathbf{J}_n^T\mathbf{J}_n\Delta\mathbf{s}_n = -\mathbf{J}_n^T\mathbf{r}_n \tag{17}$$

The case with fixed $\mathbf{s}_{n,m}$ is similar with that in the first step. The cost function can be rewritten as

$$E = \sum_n \sum_{\mathbf{p}} W_n(\mathbf{q}) \left(F(\mathbf{p}) - F_n\big(T_n(\mathbf{p})\big)\right)^2 \tag{18}$$

And its solution is

$$F(\mathbf{p}) = \sum_n W_n(\mathbf{q}) F_n\big(T_n(\mathbf{p})\big) \Big/ \sum_n W_n(\mathbf{q}) \tag{19}$$

When the iterative calculation for $F(\mathbf{p})$ and $\mathbf{s}_{n,m}$ has converged, the next around of iteration for minimizing $E^{(k+1)}$ in Eqn. (8) goes on. At last, after obtaining the final global TSDF representation, 3D construction can be achieved by the existing algorithms such as marching cubes [16].

Implementation Issues

In this part, first we talk about how to calculate $\mathbf{J}_n^T\mathbf{J}_n$ and $\mathbf{J}_n^T\mathbf{r}_n$ in Eqn. (17). Due to the huge number of involved points, both the length of vector $\mathbf{r}_n$ and the height of matrix $\mathbf{J}_n$ are quite large. For more efficient calculation, let

$$\mathbf{r}_n = \left[r_{n,1}, r_{n,2}, \cdots, r_{n,N_\mathbf{p}}, \bar{\mathbf{r}}_n^T\right]^T \tag{20}$$

where $N_\mathbf{p}$ denotes the involved point number, and $\bar{\mathbf{r}}_n$ is a vector combining all $r_{n,m,t}$ together. Then the Jacobian matrix $\mathbf{J}_n$ can also be partitioned into blocks as

$$\mathbf{J}_n = \left[\mathbf{J}_{n,1}^T, \mathbf{J}_{n,2}^T, \cdots, \mathbf{J}_{n,N_\mathbf{p}}^T, \bar{\mathbf{J}}_n^T\right]^T \tag{21}$$

where $\mathbf{J}_{n,i}\ (i = 1,2,\cdots,N_\mathbf{p})$ and $\bar{\mathbf{J}}_n$ are the corresponding Jacobian matrices for $r_{n,i}$ and $\bar{\mathbf{r}}_n$ respectively. Thus

$$\mathbf{J}_n^T\mathbf{J}_n = \sum_i \mathbf{J}_{n,i}^T\mathbf{J}_{n,i} + \bar{\mathbf{J}}_n^T\bar{\mathbf{J}}_n \tag{22}$$

$$\mathbf{J}_n^T\mathbf{r}_n = \sum_i \mathbf{J}_{n,i}^T r_{n,i} + \bar{\mathbf{J}}_n^T\bar{\mathbf{r}}_n \tag{23}$$

The first parts of $\mathbf{J}_n^T\mathbf{J}_n$ and $\mathbf{J}_n^T\mathbf{r}_n$ can be calculated point by point, which only costs a few memory, and is convenient for parallel processing. According to the definition of $r_{n,m,t}$ in Eqn. (14), we can get $\bar{\mathbf{J}}_n^T = \sqrt{\lambda}\,\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. So the second parts of $\mathbf{J}_n^T\mathbf{J}_n$ and $\mathbf{J}_n^T\mathbf{r}_n$ can also be easily obtained. Further, in our implementation for Eqn. (4), for each point $\mathbf{p}$, only 8 nearest neighboring lattice points are assigned with non-zero interpolation coefficients. Therefore, when calculating the Jacobian matrix $\mathbf{J}_{n,i}$, there are at most 24 non-zero values in it, whose positions are determined by the index of the 8 nearest neighboring points lattice for point $\mathbf{p}$. Once the positions are acquired, it is only needed to update the corresponding elements in $\mathbf{J}_n^T\mathbf{J}_n$ and $\mathbf{J}_n^T\mathbf{r}_n$.

Next, we discuss how to efficiently solve Eqn. (17). It can be seen that the variable number is determined by the number of lattice points. Suppose the lattice consists of M points, then the sizes of $\mathbf{J}_n^T\mathbf{J}_n$ and $\mathbf{J}_n^T\mathbf{r}_n$ are $3M \times 3M$ and $3M \times 1$, respectively. Although the scale of the linear equation is much smaller than the total involved point number, we can determine the values of some variables in advance, and further reduce the number of variables to be calculated.

Since only the points near the object surface are assigned with meaningful TSDF values and used in the optimization framework, many lattice points are not included in the neighborhood of any involved points when a uniform lattice is adopted. For these lattice points, we have no information to find their transformations, so their corresponding displacements should be zero. This conclusion can also be inferred by quantitative analysis. Briefly speaking, if the m-th lattice point is not included in the neighborhood of any points near the object surface for the n-th individual TSDF volume, then the $(3m-2)$-th, $(3m-1)$-th, and $3m$-th elements of the vector $\mathbf{J}_{n,i}\ (i = 1,2,\cdots,N_\mathbf{p})$ are zeros. According to Eqn. (22) and Eqn. (23), for the $(3m-2)$-th, $(3m-1)$-th, and $3m$-th rows of the matrix $\mathbf{J}_n^T\mathbf{J}_n$, only the elements in the positions of $(3m-2, 3m-2)$, $(3m-1, 3m-1)$, and $(3m, 3m)$ are with non-zero values, and the $(3m-2)$-th, $(3m-1)$-th, and $3m$-th elements of the vector $\mathbf{J}_n^T\mathbf{r}_n$ are zeros. Thus, the calculated $\mathbf{s}_{n,m}$ must be a zero vector.

As the displacements of some lattice points can be determined without solving Eqn. (17), only the left variables are needed to be calculated. In this way, the scale of the linear equation becomes smaller, which further accelerates the solving process.

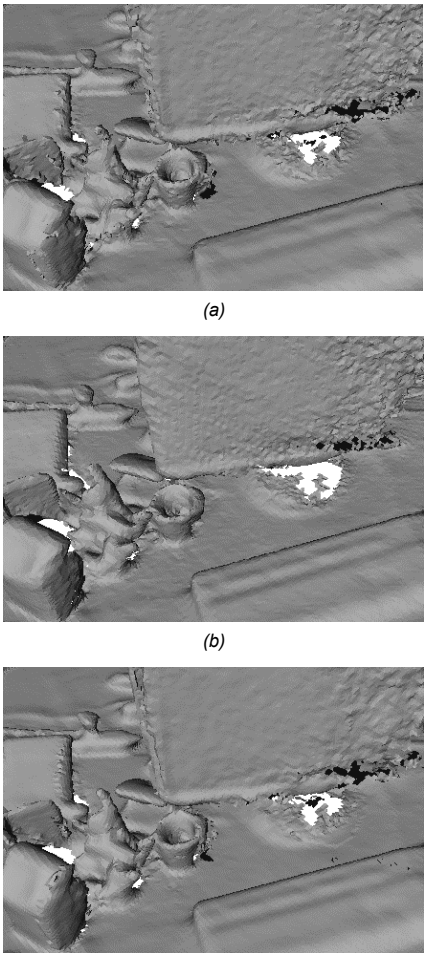
Experimental Results

All our experiments are conducted on the RGB-D SLAM benchmark [17]. Different sequences of depth maps are used in the experiments. Since similar conclusions can be made, only the results for the sequence "fr1/desk" are illustrated in the paper.

Two different experiments are finished. For the first one, each individual TSDF volume is constructed by multiple depth maps. The first 200 depth maps are adopted in the experiment. All the frames are evenly partitioned into 4 segments, and each individual TSDF volume is constructed with 50 frames by KinectFusion [2], [3]. The methods of weighted average integration as well as simultaneous localization and calibration (SLAC) [7] are used for comparison with our proposal. It should be noted that although range camera calibration is utilized in SLAC, the step of integration is still simply implemented by weighted average of all the individual TSDF volumes constructed by the calibrated depth maps. The final reconstruction results are illustrated in Figure 1. It can be seen that the overall performance of weighted average integration is the worst. By introducing non-rigid transformations in either point level or voxel level, both SLAC and our proposal can achieve better results. It is demonstrated that the object edges are well reconstructed by SLAC, while the result of our proposal is smoother, especially for the computer monitor.

Furthermore, by placing two reconstructed models in the same coordinate space, as shown in Figure 2, we can see that the results of weighted average integration and our proposal coincide

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-398



(a)

(b)

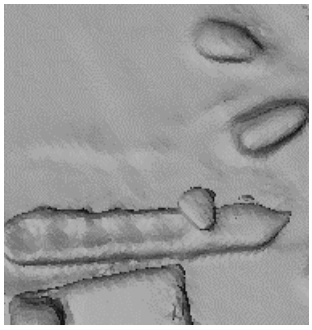
(c)

***Figure 1**. Reconstruction results by (a) weighted average integration, (b) SLAC, and (c) our proposal.*

well, while there are obvious displacements between the results of weighted average integration and SLAC. Since point-level transformations are conducted in SLAC, this kind of unreasonable local displacements are inevitable, which will hinder subsequent step such as color mapping. As we only involve voxel-level transformations for individual TSDF volume correction, significant displacements do not appear.

In the second experiment, the global TSDF volume is growing frame by frame. That is to say, with an incoming depth map, we need to integrate the TSDF volume constructed by all the frames coming before and the one corresponding to the new depth map. SLAC is not applicable for this case. Therefore, we just compare our proposal with the method of weighted average integration. The first 50 depth maps are adopted for the experiments, and some details of the final reconstruction results are illustrated in Figure 3. Since less depth maps are used, the reconstruction results are inferior to those in Figure 1. However, the performance of our proposal is also better than that of weighted average integration, which again demonstrates the effectiveness of introducing non-rigid transformations for individual TSDF volumes. When we place the two reconstructed models in the same coordinate space as in Figure 4, we can see that they also coincide well and there are no unreasonable local displacements.

(a) Results of weighted average integration and SLAC.



(b) Results of weighted average integration and our proposal.

***Figure 2**. Two reconstruction results in the same coordinate space.*

Conclusions and Future Work

In this paper, by taking camera distortions into consideration for TSDF volume integration, a novel method based on voxel-level optimization is proposed. Since perfect alignment usually cannot be obtained, suitable non-rigid transformations are introduced for individual TSDF volumes to make them aligned better. According to the consistency of TSDF values of individual and integrated volumes, a comprehensive cost function involving both the final global TSDF representation and the transformation parameters is defined. An iterative solution is developed for the optimization problem, and some implementation issues for reducing memory cost and computational load are also introduced. It is demonstrated our proposal does not cause unreasonable local displacements, and is effective for more satisfactory 3D reconstruction.

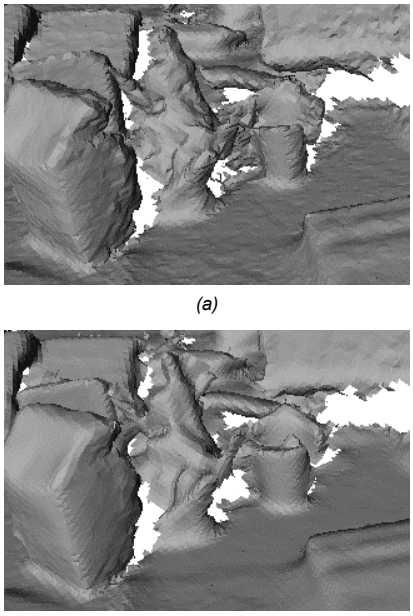
In our finished work, a uniform lattice is adopted for defining the introduced non-rigid transformations, and it remains the same for different TSDF volumes. However, in practice, the points with meaningful TSDF values are not evenly distributed in the whole 3D space, and their distributions are different for each TSDF volume. Therefore, the interpolation results may not be accurate for all the points. If we want to better describe the transformations, more complex lattices should be utilized. In the future, we will focus on how to adaptively determine the most effective lattice for each TSDF volume, and complete its efficient implementation.

References

[1] J. Smisek, M. Jancosek, and T. Pajdla. 3D with Kinect. Consumer Depth Cameras for Computer Vision, 3-25, Springer, 2013.

[2] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera. Proc. ACM Symposium on User Interface Software and Technology, 559-568, 2011.

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-398



(a)

(b)

***Figure 3**. Reconstruction results by (a) weighted average integration and (b) our proposal.*

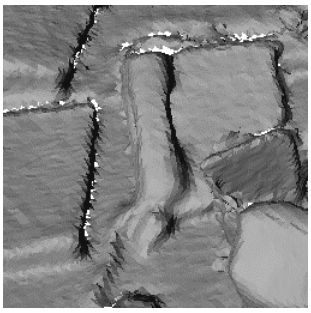


***Figure 4**. Results of weighted average integration and our proposal in the same coordinate space.*

[3] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. Proc. IEEE International Symposium on Mixed and Augmented Reality, 127-136, 2011.

[4] Q.-Y. Zhou and V. Koltun. Dense scene reconstruction with points of interest. ACM Transactions on Graphics, 32(4), 2013.

[5] Q.-Y. Zhou, S. Miller, and V. Koltun. Elastic fragments for dense scene reconstruction. Proc. IEEE International Conference on Computer Vision, 473-480, 2013.

[6] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3D reconstruction at scale using voxel hashing. ACM Transactions on Graphics, 32(6), 2013.

[7] Q.-Y. Zhou and V. Koltun. Simultaneous localization and calibration: Self-calibration of consumer depth cameras. Proc. IEEE International Conference on Computer Vision and Pattern Recognition, 454-460, 2014.

[8] G. Choe, J. Park, Y.-W. Tai, and I. S. Kweon. Exploiting shading cues in Kinect IR images for geometry refinement. Proc. IEEE International Conference on Computer Vision and Pattern Recognition, 3922-3929, 2014.

[9] M. Zollhöfer, M. Nießner, S. Izadi, C. Rhemann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, and M. Stamminger. Real-time non-rigid reconstruction using an RGB-D Camera. ACM Transactions on Graphics, 33(4), 2014.

[10] M. Dou, J. Taylor, H. Fuchs, A. Fitzgibbon, and S. Izadi. 3D scanning deformable objects with a single RGBD sensor. Proc. IEEE International Conference on Computer Vision and Pattern Recognition, 493-501, 2015.

[11] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. Proc. IEEE International Conference on Robotics and Automation, 2724-2729, 1991.

[12] G. Turk and M. Levoy. Zippered polygon meshes from range images. Proc. ACM International Conference and Exhibition on Computer Graphics and Interactive Techniques, 311-318, 1994.

[13] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy. Real-time 3D model acquisition. ACM Transactions on Graphics, 21(3), 438-446, 2002.

[14] D. Herrera C., J. Kannala, and J. Heikkilä. Joint depth and color camera calibration with distortion correction. IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(10), 2058-2064, 2012.

[15] A. Teichman, S. Miller, and S. Thrun. Unsupervised intrinsic calibration of depth sensors via SLAM. Robotics: Science and Systems, 2013.

[16] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. ACM Computer Graphics, 21(4), 163-169, 1987.

[17] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, 573-580, 2012.

Author Biography

Fei Li received his B.S. degree in automation from Beijing University of Aeronautics and Astronautics in 2004, and his Ph.D. degree in control science and engineering from Tsinghua University in 2009. Then he joined Fujitsu Research & Development Center Co. Ltd., Beijing, China. His research interests include pattern recognition, image understanding, robot vision, and computer graphics.

Yunfan Du received his B.S. degree from Northeastern University in 2011, and his M.S. degree from Beijing Institute of Technology in 2014, both in computer science and technology. Since then, he has worked in Fujitsu Research & Development Center Co. Ltd., Beijing, China. His current work focuses on 3D modeling and reconstruction.

Rujie Liu received his B.S., M.S., and Ph.D. degrees in electronic engineering from Beijing Jiaotong University in 1995, 1998, and 2001, respectively. Since then, he has worked as a researcher in Fujitsu Research & Development Center Co. Ltd., Beijing, China. His research interests are in the areas of content-based image retrieval, pattern recognition, and image processing.