3D Res (2017) 8:16
DOI 10.1007/s13319-017-0127-x

3DR EXPRESS

A Hierarchical Optimization Algorithm Based on GPU for Real-Time 3D Reconstruction

Jin-hua Lin · Lu Wang · Yan-jie Wang

Received: 16 February 2017 / Revised: 9 April 2017 / Accepted: 11 April 2017 / Published online: 19 April 2017

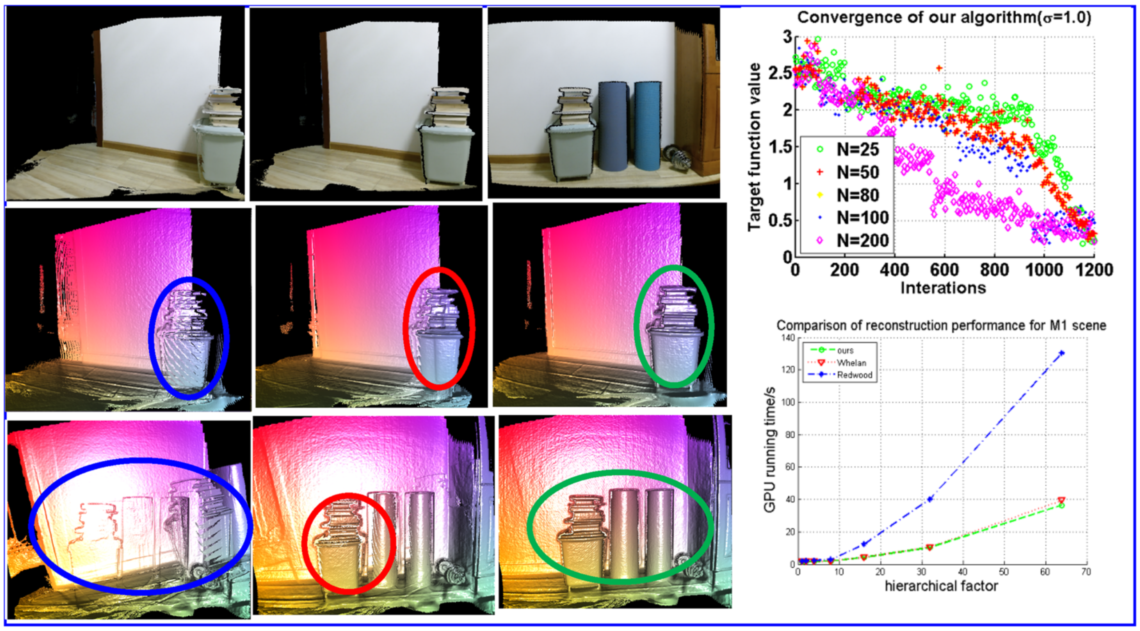
Abstract In machine vision sensing system, it is important to realize high-quality real-time 3D reconstruction in large-scale scene. The recent online approach performed well, but scaling up the reconstruction, it causes pose estimation drift, resulting in the cumulative error, usually requiring a large number of off-line operation to completely correct the error, reducing the reconstruction performance. In order to optimize the traditional volume fusion method and improve the old frame-to-frame pose estimation strategy, this paper presents a real-time CPU to Graphic Processing Unit reconstruction system. Based on a robust camera pose estimation strategy, the algorithm fuses all the RGB-D input values into an effective hierarchical optimization framework, and optimizes each frame according to the global camera attitude, eliminating the serious dependence on the tracking timeliness and continuously tracking globally optimized frames. The system estimates the global optimization of gestures (bundling) in real-time, supports for robust tracking recovery (re-positioning), and re-estimation of large-scale 3D scenes to ensure global consistency. It uses a set of sparse corresponding features, geometric and ray matching functions in one of the parallel optimization systems. The experimental results show that the average reconstruction time is 415 ms per frame, the ICP pose is estimated 20 times in 100.0 ms. For large scale 3D reconstruction scene, the system performs well in online reconstruction area, keeping the reconstruction accuracy at the same time.

J. Lin (✉) · Y. Wang
Machinery and Electronics Engineering, Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China
e-mail: ljh3832@163.com

J. Lin
Machinery and Electronics Engineering, Chinese Academy of Sciences University, Changchun 130033, China

J. Lin · L. Wang
Computer Application Technology, Changchun University of Technology, Changchun 130033, China

Graphical Abstract



Keywords Machine vision · 3D reconstruction · Online volume fusion · GPU · Pose estimation

1 Introduction

At present, the three-dimensional reconstruction technology based on image is developing continuously. It is widely used in robot vision, 3D manufacturing, implicit space tracking and virtual reality. For example, as a user or robot requires real-time scanning of the entire room or multiple spaces, the cumulative 3D model is instantaneously fused to the target application; the robot navigation needs to map the physical world to the virtual world and provide immediate feedback during the scan; These requirements have led researchers to begin a real-time reconstruction of large-scale scenarios.

There has been a lot of work on 3D reconstruction over the past decades. Key to high-quality 3D reconstruction is the choice of representation for multiple sensor measurements. Approaches range from unstructured point-based representations methods to volumetric approaches. While each has tradeoffs, volumetric methods based on implicit truncated signed distance fields have become the general method for highest quality reconstructions. They model continuous surfaces and efficiently perform incremental updates. The most prominent recent example is Kinect Fusion where real-time volumetric fusion of smaller scenes was demonstrated. One inherent issue with these implicit volumetric methods is their lack of scalability due to reliance on a uniform grid. This has become a focus of much recent research, where real-time efficient data structures for volumetric fusion have been proposed. We exploit the sparsity of TSDF to create more efficient spatial subdivision strategies. While this allows for volumetric fusion at scale, pose estimates suffer from drift. Even small pose errors can accumulate to dramatic error in the final 3D model. Most of the research on achieving globally consistent 3D models at scale from RGB-D input requires offline processing and access to all input frames. Globally consistent models are provided by optimizing across the entire pose trajectory, but require minutes or even hours of processing time, meaning real-time revisiting or refinement.

Real-time, drift-free pose estimation is a key focus in the simultaneous localization and mapping (SLAM) literature. Many real-time monocular RGB methods have been proposed, including sparse methods, semi-dense or direct methods. Typically these approaches rely on either pose-graph optimization or bundle adjustment, minimizing re-projection error across frames and distributing the error across the graph. While impressive tracking results have been shown using only monocular RGB sensors, these approaches do not generate detailed dense 3D models, which is the aim of our work. Real-time SLAM approaches typically first estimate poses frame-to-frame and perform correction in a background thread. In contrast, DTAM uses the concept of frame-to-model tracking to estimate the pose directly from the reconstructed dense 3D model. This omits the need for a correction step, but clearly does not scale to larger scenes. Pose estimation from range data typically is based on variants of the iterative closest point (ICP) algorithm. In practice, this makes tracking extremely brittle and has led researchers to explore either the use of RGB data to improve frame-to-frame tracking or the use of global pose estimation correction. These systems are state-of-the-art in terms of online correction of both pose and underlying 3D model. However, they usually need more time to perform online optimization.

In order to solve the above problems, an end-to-end real-time reconstruction system is proposed. The core is a robust attitude estimation strategy, which optimizes the camera trajectory of each frame and fuses all RGB-D input values into an effective local-to-global hierarchical optimization framework. The system is globally associated with each RGB-D frame, which can implicitly and continuously process the cyclic closures, eliminating the need for them. So if the tracking fails, the scan is interrupted or restarted from a completely different corner, the system will be re-positioning in a globally consistent manner, with good reconstruction accuracy and robustness. The main work of this paper includes the following aspects:

First, a global attitude alignment framework is presented. According to all the RGB-D values of the input frame, the inaccurate characteristics of the tracking aging are eliminated, and the scalability of the scene is achieved by the localized global hierarchical decomposition strategy reconstruction.

Second, according to the alignment strategy from sparse to intensive items, the global structure with implicit closure is aligned with the exact attitude of the precise scale to achieve accurate reconstruction of the local surface detail.

Third, an RGB-D fusion method is implemented. The sparse correspondence with RGB feature is used to estimate the global pose. In order to guarantee the reconstructed target being refined according to geometric consistency of dense area, 3D scene is updated continuously.

Fourth, the large-scale reconstruction of the geometric texture scene is achieved, the model of the re-access area is improved, the tracking failure is restored, and the robustness of the drift and continuous closed loop is restored.

2 Related Work

In the past few decades, researchers have done a lot of research on 3D reconstruction methods. The key to high-quality 3D reconstruction is to integrate the underlying representation of multiple sensor measurements. From the point-based unstructured representation [1–4], the 2.5D depth map representation [5, 6] and the height field representation [7] to the volume representation [8, 9], each method is a trade-off between reconstruction quality, rate and scale. The volume fusion method based on truncated signed distance fields (TSDF) is a representative high-volume reconstruction method [10]. The surface of the model is represented continuously and systematically, and the explicit topology is removed.

These implicit volume reconstruction methods relying on uniform grids lack scalability. In response to this deficiency, the researchers have given a variety of effective data structures for real-time volume fusion [11, 12], using TSDF's sparseness to create a more effective spatial segmentation strategy. Although these methods achieve a large-scale volume fusion, they may cause the attitude estimation drift phenomenon, leading to the final 3D model distortion. Even a smaller attitude error, which appears to be negligible on the local scale, accumulates a significant error in the final model.

Global consistent 3D reconstruction based on RGB-D input association usually requires off-line processing. For example, Zhou et al. [13] proposed a globally consistent model based on global attitude trajectory optimization, which takes several minutes or even several hours of off-line processing. Choi et al [14]. proposed globally consistent models by optimizing across the entire pose trajectory. simultaneous localization and mapping (SLAM) is the core of real-time drift-free camera attitude estimation. Researchers have proposed a variety of real-time monocular RGB methods, including sparse methods, semi-density methods and direct reconstruction methods [15, 16]. These methods rely on the optimization or bundling of monocular RGB sensors to minimize the cross-frame re-projection error and cross-regional distribution errors, but do not generate accurate dense 3D models that affect the reconstruction of fine-grained details.

The monocular fusion method increases the binding adjustment of sparse SLAM, achieves dense volume fusion, and the reconstruction effect is better, but only for small scale scenes. SLAM-based reconstruction methods typically use frame-to-frame strategies for gesture estimation and perform corrections in background threads to reduce real-time rates. Newcombe et al. proposed a frame-to-model tracking framework to estimate the gesture directly from the reconstructed dense 3D model, eliminating the need for corrective steps, but not for large-scale scene reconstruction.

The attitude estimation strategy based on the iterative closest point algorithm relies on the measurement of distance data, which makes the robustness of real-time tracking worse. Researchers began to use RGB data and global attitude estimation correction to improve frame-to-frame tracking robustness [17], including posture optimization, closed-loop detection, incremental bundling, and recovery by image or point relocation. These systems perform well on the online posture correction of the standard 3D model, but the online optimization time is longer. When camera tracks in closed-up trajectory, the system limits the free motion and scanning trajectory of camera; when fusion step is done, the camera's attitude is optimized and the previous data is calculated efficiently, but the accuracy of the model correction is limited.

3 System Overview

The system is based on a robust global attitude optimization algorithm that performs continuous posture optimization for each frame of data. It updates corrections based on newly generated attitude estimates supporting free change of camera paths, instantaneous relocation, and frequent re-scanning of the same scene area. Our system owns the robust implementation of sensor block, fast frame to frame motion and no feature area tracking scan. It can achieve large-scale three-dimensional scene real-time reconstruction, as shown in Fig. 1.

First, the RGB-D stream input is obtained from the depth sensor and a set of sparse corresponding term features are used to obtain coarse global alignment. The alignment is corrected by optimizing the density and geometrical consistency to calculate the paired scale of all input frames feature transform (SIFT, Scale Invariant Feature Transform). It detects all SIFT keys that match the previous frame, and filters out the outliers.

Secondly, in order to achieve real-time alignment of global pose, the system uses localized global pose optimization of hierarchical filtering frames. At first level, consecutive n frames constitute a block and are localized to the local level. At second level, all blocks are correlated and globally optimized. This method is similar to the hierarchical sub-graph algorithm, and performs global correlation analysis on all frames. The algorithm is based on the currently visible frustum region optimizing the two-stage attitude alignment based on the sparse corresponding feature of the filter and the dense photo-metric geometric constraints. This hierarchical phase optimization strategy reduces the non-associated nature of each optimization step. Our algorithm is suitable for large-scale scene reconstruction. The system uses a Graphic Processing Unit (GPU) nonlinear iterative solver to handle highly nonlinear optimization problems at two levels.

Finally, the RGB-D frame is quickly re-fused based on a continuously updated representation of the global 3D scene based on the continuously changing global gesture, and the old gesture RGB-D image is removed. The new gestures are reintegrated into RGB-D images. With more RGB-D frames and sophisticated gesture estimation, the volume model is continually improved to ensure the quality of reconstruction of large scenes.

4 Global Alignment

The globally consistent online 3D reconstruction is based on robust global gesture optimization strategies. The system input is captured by the sensor RGB-D stream: $S = \{f_i = (C_i, D_i)\}_i$, C_i is for each frame of time space alignment color, D_i is depth data, the frame rate of the sensor is 30 Hz and the pixel resolution is 640*480. The goal of global alignment is to find a set of 3D counterparts between input frames. It aligns all frames according to the optimal set of rigid camera

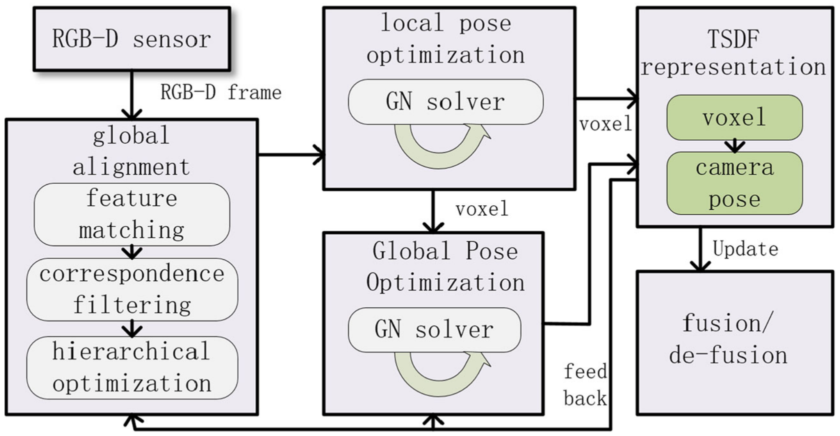


Fig. 1 Block diagram of reconstruction system

transformations. The transformation is mapped from the local camera coordinates of the frame to the spatial coordinate system: $\Gamma_i(p) = R_i p + t_i$, p is a coordinate vector, which represents the mapping of local coordinates to global coordinates. R_i is for the rotation angle, t_i is for the translation size. It is assumed that the first frame is defined as the world coordinate system.

4.1 Matching of Feature Counterparts

The system uses feature detection, feature matching, and corresponding filtering steps to search for sparse correspondence between input frames. Accurate sparse correspondence is critical to obtain a dense optimized convergence area. This section details the search and filtering steps for the corresponding item.

The system detects the SIFT feature for each new frame of input and matches it to all previously visible frames. SIFT represents adjustment of RGB-D scans, it involves several steps including image translation, scaling, rotation, et al. Then the matching between each pair of frames is filtered to produce a valid list of corresponding items as a global gesture optimization input. The search step is performed entirely on the GPU, saving the cost of copying data to the CPU. The GPU and CPU working process is shown in Fig. 2. The system calculates the SIFT key and descriptor for a time of 4–5 ms per frame, and the time for matching a pair of frames in parallel is 0.05 ms. For each newly input RGB-D image, the system can search for up to 20 K frames in real time to match the corresponding items.

4.2 Correspondence Filtering Step

In order to minimize the outliers, the system filters the detected set of identities based on photometric and geometric consistency. Robustness in the detection is built into subordinate optimization .

First, for a pair of input frames f_i and f_j, they are corresponding to the two 3D spatial points detected. The corresponding relationship between the frame-to-point is referred to as the corresponding term. The corresponding key filter detects a set of corresponding items. These corresponding items have a steady state distribution and have a consistent rigid conversion characteristic. The corresponding items are aggregated according to the matching distance. For each newly added correspondence, the rigid transform minimizes the RMSD between the current set of correspondences. If this condition number is high then the system is considered unstable. Therefore, when the re-projection error is high or the conditional analysis is an unstable system, the corresponding term is deleted in the order of the re-projection error until the unstable condition is ended or the corresponding item amount is too small to determine the rigid transformation. When the corresponding result does not produce a valid transformation, the corresponding item corresponding to the frame and associated is discarded.

Secondly, the system detects whether the surface size of the corresponding item is large enough to calculate the cross-surface area between the corresponding key point and the corresponding key point. For each set of three-dimensional space points, it will

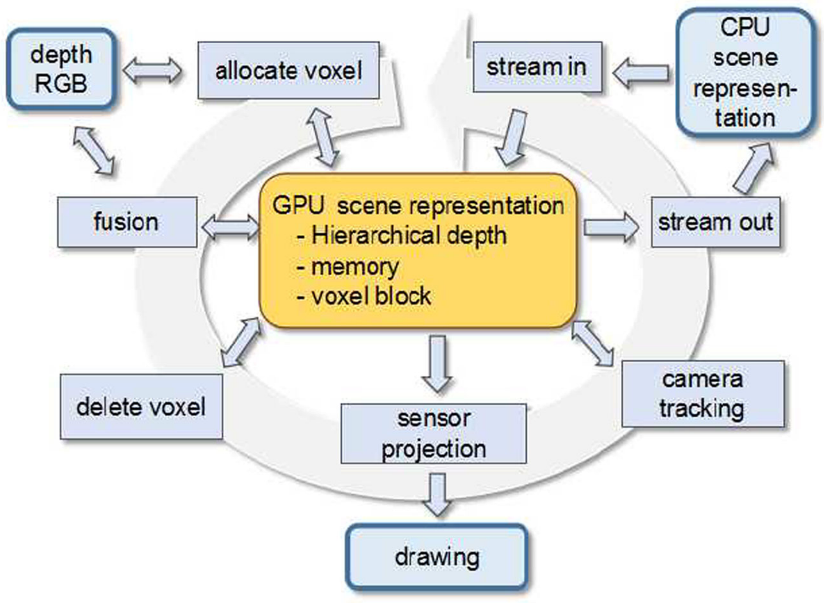


Fig. 2 GPU and CPU streaming process

be projected to the respective 2D plane. The two-dimensional directional bounding box is calculated for the surface area. When the surface area is not large enough, then delete the corresponding item.

Finally, the system performs dense double-sided geometry and photometric verification. It will keep aligning the coordinate system using the transformations calculated from the key-point filter. It measures the average depth of the re-projection in both directions, the method phase offset, and luminosity consistency. In order to improve the efficiency of reconstruction, the system performs the checks on the filtered and down-sampled frames, and when the new RGB-D images are captured, including the filtered and down-sampled color intensities, depths, camera spatial positions, and phase coordinates putting into the GPU cache. From f_i to f_j the total projection error is:

$$E_r(f_i, f_j) = \sum_{x,y} \left\| \Gamma_{ij}(\mathrm{p}_{i,x,y}) - \mathrm{q}_{j,x,y} \right\|_2 \tag{1}$$

in which $\mathrm{p}_{i,x,y} = \mathrm{P}_i^{low}(x, y)$, $\mathrm{q}_{j,x,y} = P_j^{low}(\pi^{-1}(\Gamma_{i,j}\mathrm{p}_{i,x,y}))$. When the position, normal or color value of them does not correspond, it indicates that the occlusion phenomenon occurs, and it is determined that the dense corresponding item is invalid. When the re-projection error is too large (>0.065 m) or the effective counterpart is insufficient (<0.01 wh), the match is invalid. The system implements detection through a single GPU call, allowing each thread to process an image (1 frame). When all tests are passed, the corresponding entry is added to the valid set, which is later used for gesture optimization.

4.3 Hierarchical Optimization

In order to process tens of thousands of RGB-D input frame data in real time, the system adopts the hierarchical optimization strategy. The input continuous frame sequence is divided into blocks, and in the first layer, the intra-block optimization is performed to the local alignment mode. At the second level, the key frames associated with each block are used for global alignment optimization. Table 1 shows the allocation and optimization pseudocode used in this paper for sparse voxel levels.

First, the system performs inter-block gesture optimization operation with 11 consecutive frames in the input RGB-D stream and one block adjacent to each other. The purpose of local attitude optimization is to use the first frame in the block as the reference frame to calculate the best intra-block alignment. That is the best camera attitude transformation. The system searches for valid feature counterparts between all frame pairs and uses the energy minimization method to ensure that all frames are aligned in the best possible way. Since each block contains only a small number of

Table 1 Optimization step of our algorithm

```
Algorithm: Hierarchical optimization
input：struct voxel{
        float SDF;    //Signed distance field
        char RGB;    //input RGB color image
        char weight;    //weight
        float LA;    //albedo
        float RD; //corrected distance
       };
       int level;    //corrected level
output： $\widetilde{\mathbf{D}}$    //corrected image
Begin：//GPU parallel optimization
    allocate(level);    //allocate sparse grades
    initialfuse(level);      //fuse sparse model
    for k=1 to level
    allocate_fine(k,k+1);    //allocate the correction grid
    RGBfuse(k);      //fuse target with color
    amend(k);        //fuse results with shadow correction
    end for
end
```

consecutive frames, the attitude changes within the block are small and the system is initialized to a unit matrix. In order to ensure that the localized posture optimization results are accurate enough, the system uses an optimized local trajectory to apply intensive verification tests for each image within the block. When the re-projection error is too large for the frame image, it will discard the block.

Secondly, the first frame RGB-D data in the block is defined as the key frame of the block. The aggregation key frame is calculated, and the three-dimensional position information of the inter-block feature points in the world space is calculated based on the optimized attitude trajectory of the block. These three-dimensional position information contains the same real world space points, belonging to different frame pairs of geometric information. In order to obtain the key frame feature set, a plurality of feature points matching the feature descriptors and merging in the three-dimensional space are combined into a best three-dimensional representation. The system maps the key frame feature set to the respective key frame space using the corresponding item transformation. Once the global key frame and feature integration are created, the system discards the block data, including the in-block features, descriptors, and counterparts.

Finally, the sparse correspondence of the global keyframe is searched and filtered. If no match is found for the previous keyframe, it is marked as invalid and treated as an alternative key frame for re-verification. Global pose optimization calculates the best global alignment of all global key frames to achieve global alignment of all blocks. Then it will apply the energy minimization method to perform intra-block alignment after each new global key frame finds the corresponding entry. The incremental transformation is calculated using the corresponding inter-block optimization, using the incremental transform to initialize the gesture of the global key frame, which consists of the previous global key frame gestures. After inter-block transformation, the corresponding incremental transform (from local optimization) is applied to the frame in the block, and thus the global uniform transform is obtained from the input frames.

4.4 Global Pose Optimization Strategy

Global attitude alignment is a nonlinear least squares problem for unknown camera parameters. In order to realize the on-line global camera attitude optimization for long scan sequences with more than 20,000 frames, this paper adopts a GPU-based nonlinear iterative solver. Because sparse patterns require different parallel strategies, this paper is based on the Gauss–Newton method, namely:

$$\chi^2 = \arg\min_{\chi} E_{align}(\chi) \tag{2}$$

For ease of representation, we reformulate the objective in the following canonical least-squares form:

$$E_{align}(\chi) = \sum_{i=1}^{R} r_i(\chi)^2 \tag{3}$$

in which $R = 3N_{corr} + |\mathrm{E}| \cdot (|\Gamma_i| + |D_i|)$, N_{corr} is the number of sparse correspondence entries for inter-block alignment.

The system solves the pre-processor conjugate gradient (PCG) solver in parallel with the GPU and uses the Jacobi pre-processor to solve the above equation. Our optimization architecture is shown in Fig. 3. It uses the iterative strategy to exploits the sparseness of the system matrix. The related linear least squares problem are solved by Gauss–Newton iteration, and the optimization process is performed based on the result of the last frame.

The system uses a single kernel to calculate the optimal step size, update the camera down direction and global-local handover to accumulate the final scan results. It also uses the PCG algorithm to calculate the system matrix with the current drop direction. In order to avoid filling in, we multiply the system matrix by applying two separate kernel calls: the first kernel is multiplied by the second kernel call. Because there are different sparse-by-row patterns, using two different kernels to do parallel processing. Specifically, for sparse items, each row is precisely encoded as a corresponding term according to up to two external camera gestures or $2 \times 6 = 12$ non-zero matrix entries. Since the number of operations required is small, the matrix vector can be calculated by assigning a dedicated thread to each 3D block row, i.e., processing the correspondence, and residuals. Because different dimensions use the same operation for evaluation, the number of nonzero terms in each row is equal to the corresponding number of unknown frames for each unknown item. For longer scan sequences, this results in thousands of entries per line, and in order to reduce the amount of memory read and computation for each thread, we chose a simplified method to calculate the matrix vector product. We use the size of a block to compute the dot product of each row direction. Each thread of the block executes the thread based on PCG algorithm. For calculation, the auxiliary list is pre-computed to allow you to find all the correspondence of a variable that populates the individual thread cores for each communication. It adds the entries to the corresponding list of associated variables. The memory of each list is managed using an atomic counter, and the table is recalculated when the set of corresponding items changes.

For precise photo-metric and geometrical alignment items, the number of related residues is quite high. Since the system matrix is fixed during PCG operation, we pre-computed it at the beginning of each non-linear iteration. The required memory is pre-allocated, and we can only update nonzero items

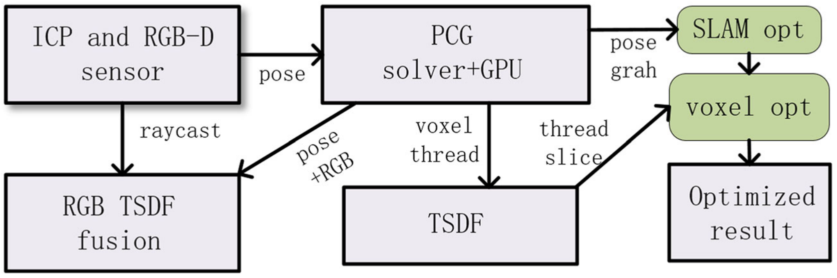


Fig. 3 Pose optimization architecture

through voxel fusion. Note that as the local memory of the shared memory decreases, only a small amount of write operations are required.

As an optimization measure, the system performs the corresponding frame filtering after each optimization, which is robust to the processing of potential communication anomalies (which are mistakenly considered to be valid). That is, we use the parallel reduction operation of the GPU to determine the maximum residual, when all the corresponding items between the two frames i and j are removed. Note that all correspondence between i and j is removed in order to minimize the number of optimization and the number of all bad correspondence. In addition, for frames that do not have a corresponding relationship, they are implicitly removed from the optimization and are marked as invalid.

5 Dynamic 3D Reconstruction

The global consistent reconstruction uses camera hierarchical optimization to update 3D models in real time. The system monitors successive changes in the posture of each frame, updating the volume scene representation by frame fusion and de-fusing. Based on this strategy, once a better attitude estimate is detected, the error in the volume representation due to the cumulative drift or dead reckoning in the non-characteristic region can be fixed.

5.1 TSDF Representation

TSDF is defined in the volume grid of voxels. It is saved in GPU cache. In order to store and process these data, this paper adopts the adaptive feature reconstruction method proposed by Lin et al. [18] to reconstruct large-scale scene, and the blank space does not need to be expressed with the use of spatial voxel index TSDF stored in sparse volume grid, using 8*8*8 voxel block. In addition, in order to achieve dynamic updating of gestures, the system implements the integration of RGB-D frames into TSDF as well as de-fusing operations (ie, adding and removing frames from reconstruction) [19, 20].

5.2 Fusion and De-fusion

For each spatial voxel in a complex scene, $\mathbf{D}(v)$ is the signed distance of the voxel, $\mathbf{W}(v)$ represents the voxel weight, $w_i(v)$ represents the fusion weight of D_i, $d_i(v)$ represented the projection distance (along the z-axis) between voxel and the depth frame D_i. The following fusion is updated for each voxel:

$$\begin{aligned} \mathbf{D}'(v) &= \frac{\mathbf{D}(v)\mathbf{W}(v) + w_i(v)d_i(v)}{\mathbf{W}(v) + w_i(v)}, \\ \mathbf{W}'(v) &= \mathbf{W}(v) + w_i(v) \end{aligned} \tag{4}$$

The integration of each system is updated as follows:

$$\begin{aligned} \mathbf{D}'(v) &= \frac{\mathbf{D}(v)\mathbf{W}(v) - w_i(v)d_i(v)}{\mathbf{W}(v) - w_i(v)}, \\ \mathbf{W}'(v) &= \mathbf{W}(v) - w_i(v) \end{aligned} \tag{5}$$

The system integrates the original posture and reintegrates it with the new gesture to update the reconstructed frame data. The integrated surface measurements are well adapted to the continuously changing attitude estimation flow, which is obtained when the scene is closed.

6 Experimental Results and Analysis

6.1 Experimental Parameters and Evaluation Criteria

In order to test system performance, we use a variety of light changing under the real three-dimensional scene.

Operating system: Windows 8.1;
Development language: C++;
Development tools: Microsoft Visual Studio 2014;
3D graphics programming interface: Direct3D 11;
CPU type/frequency: INTEL Core i5/3.0 GHz;
RAM: 8G;
Graphics processor: NVIDIA GTX 960;
Sensor: Asus Xtion Pro, RGB-D stream frame drawing rate of 30 Hz, and 640 × 480 color and depth resolution capture.

We use the wireless network connection to transfer the captured RGB-D data stream to the system for global attitude optimization and real-time reconstruction of the 3D model. The reconstructed visual feedback is streamed to the system interface to aid in the scanning process. To reduce the bandwidth required, we use data compression based on depth and jpeg color compression. We use the CUDA 7.0 architecture to implement the global attitude alignment framework. Using six large-scale scenes (4 rooms M1–M4, 1 desktop M5, 1 texture wall M6, up to 80 m camera trajectory) to complete the reconstruction effect shows that the attitude alignment without obvious camera drift, with geometry and texture of high local display accuracy. This also shows that the proposed global attitude alignment strategy can be well extended to large spatial and long sequences (more than 10,000 frames). In order to quantitatively evaluate camera parameters and three-dimensional structure estimation accuracy, the following evaluation criteria are defined:

(1) The camera focal length estimation error is defined as:

$$\Delta f = \frac{1}{2}\left(\left|\frac{\hat{f}_1 - f_1}{f_1}\right| + \left|\frac{\hat{f}_2 - f_2}{f_2}\right|\right) \tag{6}$$

in which f_1 and f_2 are the true values of the two camera focal lengths; $\hat{f}_1$ and $\hat{f}_2$ are the estimated values of the two camera focal lengths.

(2) The camera relative rotation estimation error $\Delta\mathbf{R}$ is defined as the rotation axis of the rotation matrix $\hat{\mathbf{R}}\mathbf{R}^{-1}$ and the size of the angle in the angle representation (in degrees). Where $\mathbf{R}$ and $\hat{\mathbf{R}}$ is the true value and the estimated value of the relative rotation matrix of the camera.

(3) The camera relative translation estimation error Δt is defined as the angle between the camera's relative translation vector true value t and the camera's relative translation vector estimate $\hat{t}$.

(4) The RMS error of the 3D point re-projection is RMS2D. Since the re-projection error is not significant, the index is calculated only at the interior point.

(5) The RMS error of the three-dimensional point estimate RMS3D, the three-dimensional point estimation error refers to the Euclidean distance between the three-dimensional point true value and the estimated value. The maximum side length of the bounding box is set to unit 1.

6.2 Quantitative Analysis

6.2.1 Single Scene Quantitative Analysis

In this group of experiments, we select the texture wall model for rapid reconstruction, as shown in Fig. 4. The distance between the two cameras and the center of the model is about 1 to 2 times the height of the model. The true values of the two camera focal lengths are:

$$\begin{aligned} f_1 &= 1400 \\ f_2 &= 1100 \end{aligned} \tag{7}$$

Table 2 gives the average result of the noise intensity, the number of internal points, the external point ratio and the running time, the camera parameter estimation error and the three-dimensional point estimation error of the 15 test results for each test instance. It can be seen from the table that the average three-dimensional reconstruction error of the experiment is between 0.025×10^{-7} and 4.374×10^{-11}, as shown in Table 3. It can be seen from the experimental results that the reconstruction accuracy of the three-dimensional structure decreases smoothly with the increase of noise, and no mutation occurs. It can be seen from Fig. 3 that the reconstruction accuracy of the three-dimensional scene does not change significantly when the external noise ratio increases from 1.5 to 2.0 color levels per pixel. The experimental results show that the effect of model fusion is large when the noise is small, and the model does not change when the noise reaches a certain value. It can be seen that the algorithm correctly identifies the external points and

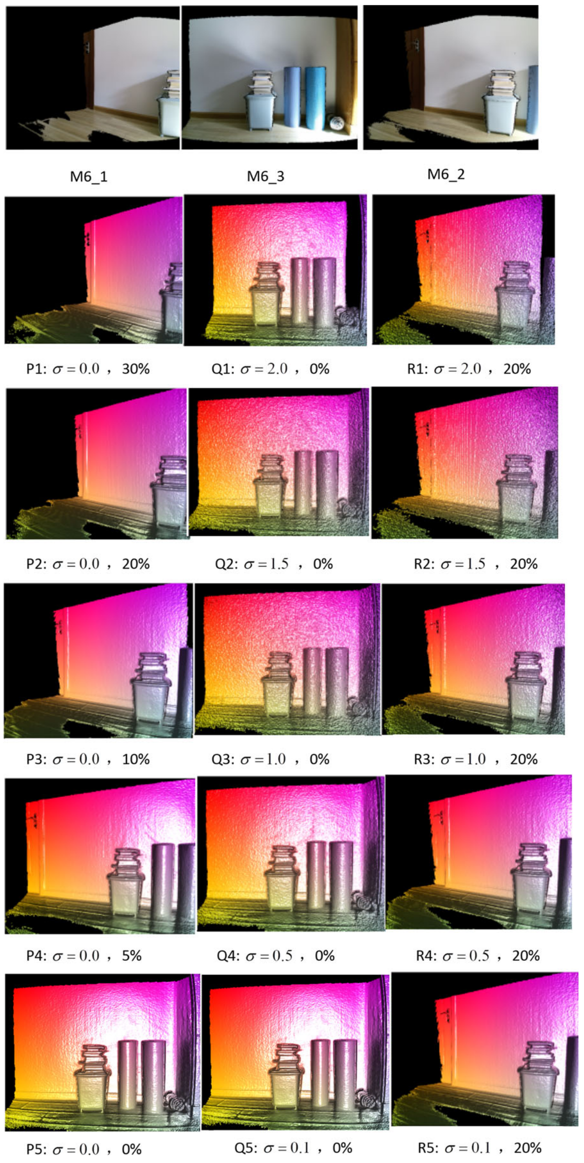


Fig. 4 Texture wall model M6 three-dimensional reconstruction effect map

Table 2 Test results of scene M6 with algorithm performance

Scene	Noise σ	Inner point	Outer point (%)	Time/ms
P1	0.0	4700	1410 (30)	327
P2	0.0	5210	1042 (20)	368
P3	0.0	5730	573 (10)	401
P4	0.0	5980	299 (5)	398
P5	0.0	6230	0 (0)	386
Q1	0.1	6230	0 (0)	438
Q2	0.5	6230	0 (0)	459
Q3	1.0	6230	0 (0)	429
Q4	1.5	6230	0 (0)	444
Q5	2.0	6230	0 (0)	507
R1	0.1	5730	573 (10)	374
R2	0.5	5730	573 (10)	426
R3	1.0	5730	573 (10)	368
R4	1.5	5730	573 (10)	460
R5	2.0	5730	573 (10)	446

successfully reconstructs inter point, external point. The algorithm is robust for noise.

In order to visually observe the effect of noise intensity and external point ratio on the 3D reconstruction error, the 3D reconstruction error of each point is taken as the logarithm of 10, and then it is mapped to the color space and the result is visualized in Fig. 4. The first column of Fig. 3 shows that there is no significant change in the reconstruction accuracy of the 3D structure in the absence of noisy data, while the external point ratio rises from 0 to 30%. Column 2 shows the change in the accuracy of the reconstruction process when the noise intensity is increased from 0.1 to 2.0 in the case where the external point ratio is fixed. It can be seen that the reconstruction accuracy of the 3D structure decreases smoothly with the increase of noise, no mutation occurs. The results of the experiment with 20% of the external points are shown in column 3 of Fig. 3. Our algorithm correctly identifies the external points and successfully reconstructs the interior points. The experimental results show that the proposed algorithm is robust to external points and noise.

6.2.2 *Quantitative Comparison of Large Scale Scenes*

The alignment performance is tested in this group of experiment, it is compared with modern online and offline methods, our method shows a better camera tracking results. First, we mark real large-scale scenes as M1, M2, M3 and M4. We change the camera posture several times for 3D reconstruction. Table 4 shows the trajectory estimation performance of the system's absolute trajectory error measurements in six scenes (including synthetic noise) [21–25]. The real-time reconstruction performance of this system is better. In addition, the system is tested based on the RGB-D benchmarks proposed by Sturm et al. [26–28].

Table 3 Error analysis for scene

Scene	Δf	$\Delta \mathbf{R}$	Δt	RMS2D	RMS3D
P1	8.185×10^{-9}	2.550×10^{-7}	3.337×10^{-7}	4.793×10^{-7}	3.233×10^{-9}
P2	2.398×10^{-8}	3.071×10^{-7}	4.159×10^{-7}	4.842×10^{-7}	3.430×10^{-9}
P3	2.294×10^{-8}	3.539×10^{-7}	6.457×10^{-7}	4.755×10^{-7}	4.007×10^{-9}
P4	2.119×10^{-8}	3.604×10^{-7}	6.497×10^{-7}	4.737×10^{-7}	3.794×10^{-9}
P5	2.876×10^{-10}	2.993×10^{-7}	2.320×10^{-7}	4.828×10^{-7}	3.717×10^{-9}

The benchmark gives the calibration of the motion capture system for the ground scene of the camera attitude estimation. For these scenes, only the small scale scene and simple camera trajectory are covered. The reconstruction effect of this system is comparable to that of the prior art, which is suitable for faster tracking with more cycles closed. The Redwood system, which relies only on geometric registration, has a relative lack of view diversity in the camera trajectory and affects the reconstruction effect. For example, M5 is a desktop with texture features that can not be geometrically treated. Through these scenes, we validate the relevance of the global consensus decision. The system is based on the sparse features of the online alignment to achieve accurate reconstruction, only in each block alignment using dense matching to increase the accuracy. The algorithm achieves the local and global level precision reconstruction. Figure 5 shows the Whelan, Redwood and our algorithm for the reconstruction performance comparison. The data in the graph show the change of the objective function value with the number of iterations in the case of different noise and 3D points. The convergence time of our method scales linearly with the 3D points. The convergence performance is changed under different noise intensity. Our algorithm allows for a large number of 3D points in practical application, it is suitable for real-time 3D reconstruction of large-scale scene.

Table 4 Quantitative comparison of several reconstruction algorithms (error unit: cm)

	M1	M2	M3	M4	M5	M6
DVO SLAM [21]	9.3	1.8	17.0	14.2	1.7	3.3
RGB-D SLAM [22]	11.2	3.1	18.1	16.6	1.8	3.5
MRSMap [23]	21.3	23.2	18.9	89.3	222.1	199.2
Kintinuous	8.2	2.6	5.0	23.2	3.2	5.4
VoxelHashing [24]	2.4	3.1	3.0	2.3	7.9	6.6
Whelan method	1.9	2.5	3.1	2.2	1.8	2.1
LSD-SLAM [25]	–	–	–	–	1.5	1.6
Redwood	26.2	2.1	3.0	6.1	3.0	4.2
Our algorithm	1.7	1.2	2.9	1.1	1.2	1.5

6.3 Qualitative Analysis

This group of experiments uses Titan X for volume reconstruction. For all test scenarios, the system runs at a frame rate of more than 30 Hz, and global intensive optimization runs at less than 500 ms at the end of the sequence. The reconstruction effect is shown in Fig. 6. From the reconstruction results, we can see that the real-time global attitude optimization strategy is superior to the current most advanced online reconstruction system, and the reconstruction quality has reached the standard of off-line reconstruction. Reconstruction has integrity, global alignment without obvious camera drift. Our algorithm has high local precision of geometry and texture reconstruction.

When the new key frame can not be successfully aligned, the reconstruction system needs to be robustly restored to the previous reconstruction area, i.e., the track is lost and the fusion surface measurement is canceled, as shown in Fig. 7. In order to show the tracking failure, with the shade in front of the camera shaking, the color image shows the effect of occlusion. In this case, the fusion map basically shows no transformation. The system can return to the previously scanned area to restore the rebuild area. Since this method globally matches the new frame with all existing data, it does not require time and space to be associated. Thus, the reconstruction can be interrupted and it can continue at a completely different location.

In Fig. 8, the time required for the reconstruction of scene M1 is shown. When the hierarchical factor is less than 8, our algorithm performs equivalent to the Whelan algorithm; when the hierarchical factor reaches 64, our algorithm runs faster than the Whelan algorithm. We can see from this graph, the same scene

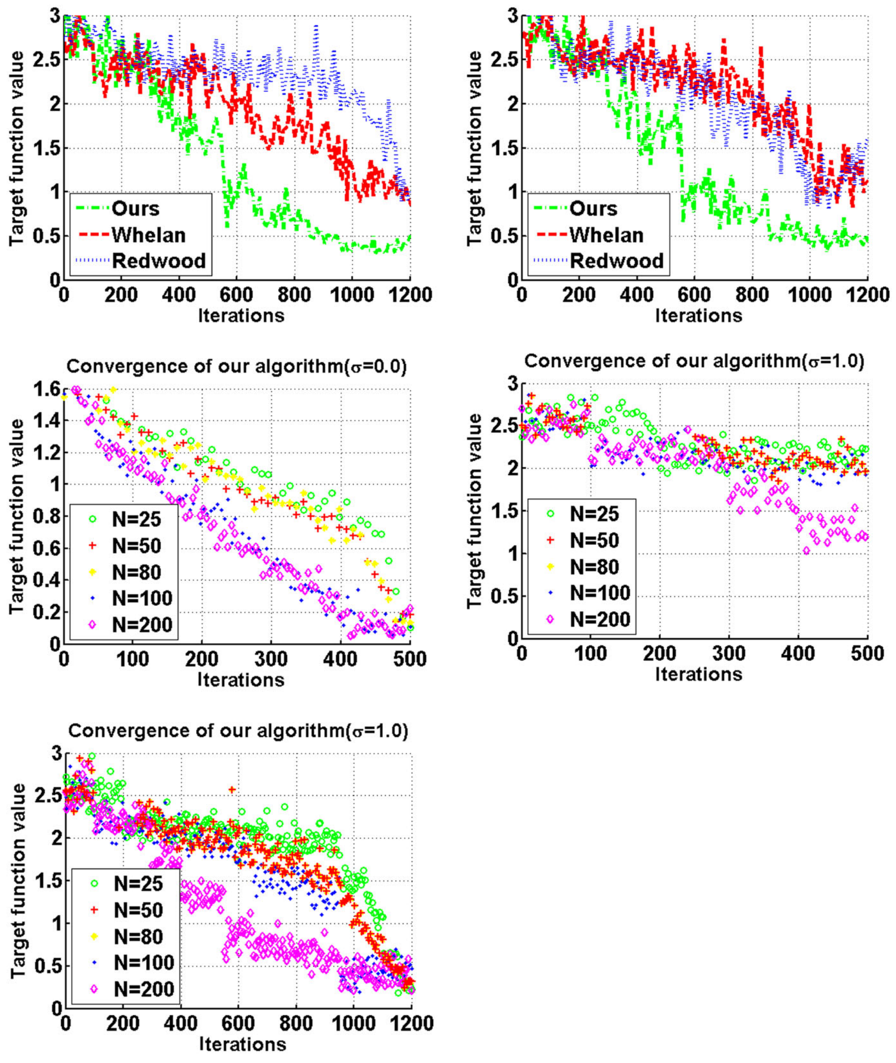
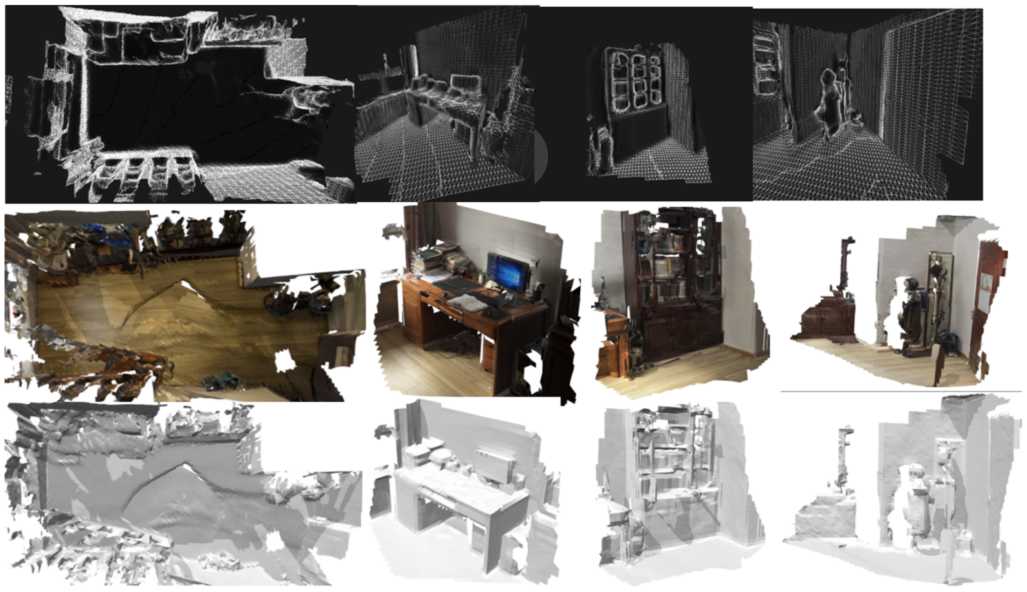


Fig. 5 Comparison of Whelan, Redwood and our algorithm for convergence

is reconstructed at the higher depth of hierarchical. Our algorithm runs faster, that is, when the hierarchical depth increases, our algorithm draws in better speed. The Whelan algorithm uses a uniform hierarchical depth for the irregular blocks of the initial control grid, and the reconstruction effect is good, but the block computation is large.

The Whelan algorithm allocates a certain GPU storage space for each feature area, stores the hierarchical table corresponding to the feature area, it needs a sufficiently large vertex buffer to store the vertex information of each slice. When a large number of feature regions are repeated, the hierarchical table of each layer also increases, and the memory occupancy rate increases. In order to alleviate the occupancy rate of the GPU and improve the reconstruction efficiency, our algorithm uses the hierarchical optimization strategy to reduce GPU memory footprint.

In Fig. 9, the reconstruction result is shown for M3, M4, M7 scene, the number of singular points in the M8 scene is large, and these singular points have the same topological characteristics. Our algorithm only establishes the hierarchical table for the first singular point, and other feature points are reused in turn. The reconstruction results show that our algorithm reconstructs the complex scene in robust manner, while the calculation is small.


(a)

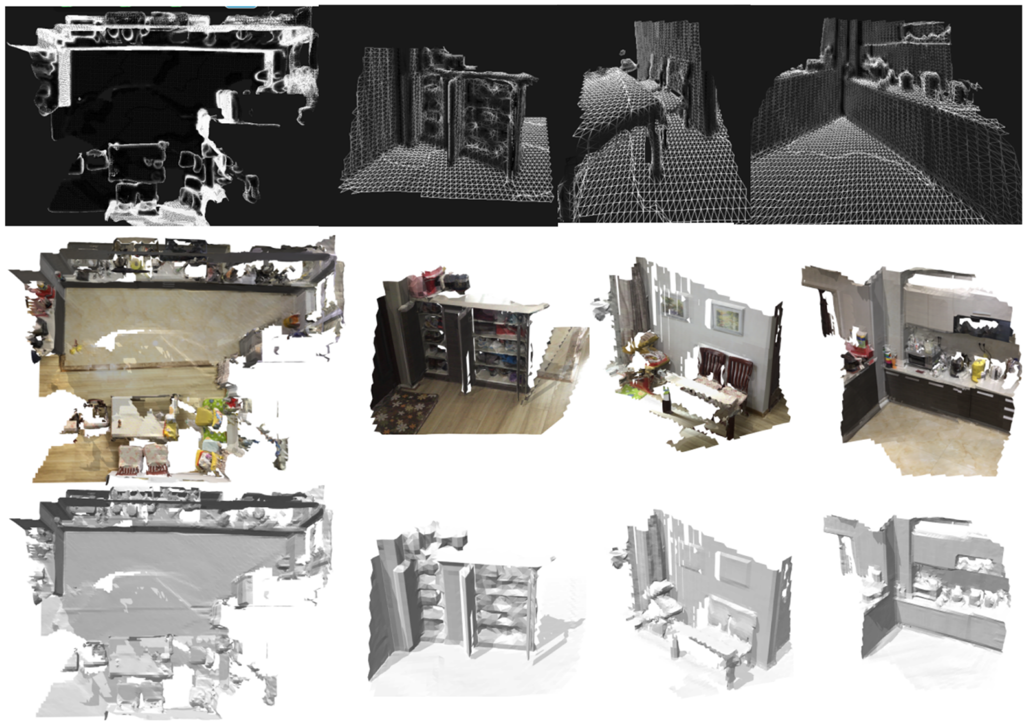

(b)

Fig. 6 System large-scale reconstruction map (depth grid, color map, fusion map). **a** Large scale scene M1 depth grid and reconstruction graph, and **b** large scale scene M2 depth grid and reconstruction graph

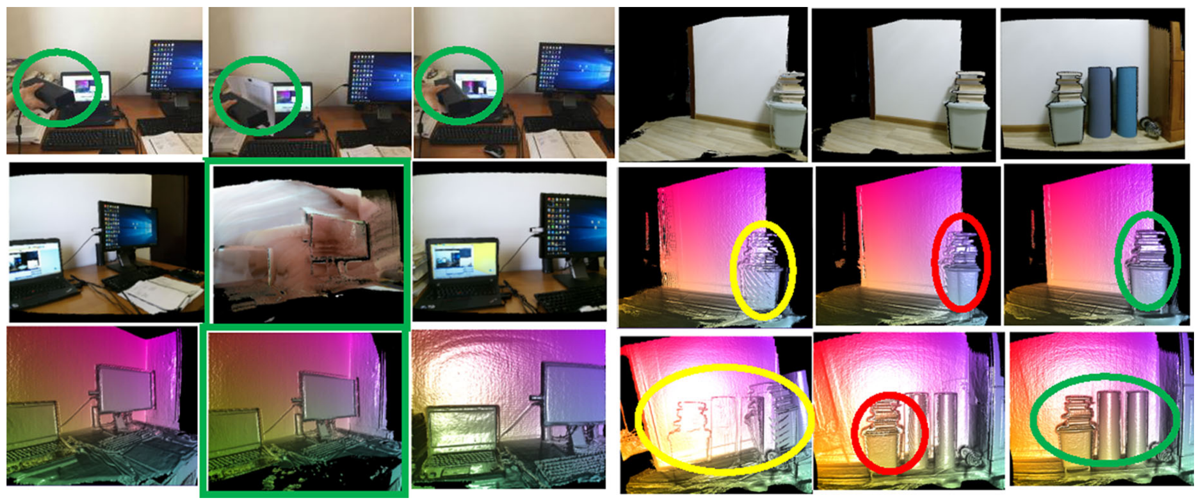


Fig. 7 Reconstruction recovery of our system (M5 and M1)

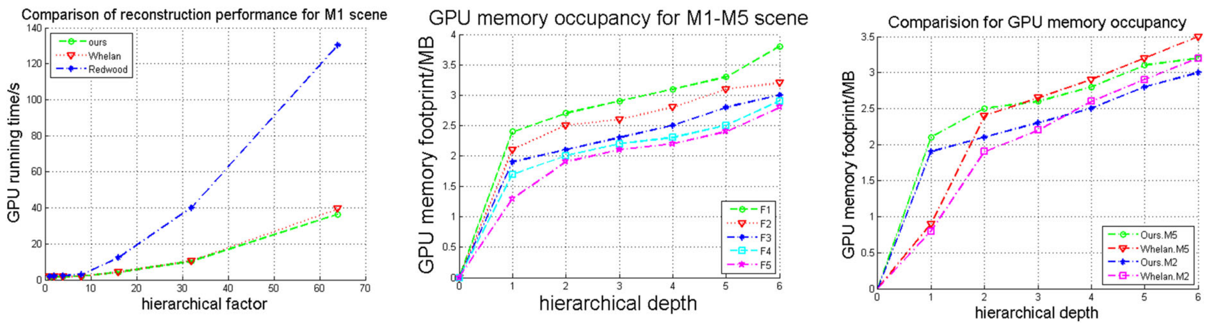


Fig. 8 Time and space complexity comparison for reconstruction of large-scale scene

7 Conclusion

In this paper, an on-line 3D reconstruction method based on global camera attitude updating is proposed. By optimizing the trajectory of each captured frame, it provides a robust tracking and implicit solution to the loop closure problem. In the sparse feature and the dense correspondence, the online SIFT feature extraction combined with the parallel nonlinear attitude optimization framework make the system to solve the global alignment problem in real time. The system monitors the continuous optimization of the posture flow, it also updates the optimization by dynamic fusion and de-fusing. This algorithm is suitable for real-time 3D reconstruction of large-scale scenes. The reconstruction accuracy and integrity are better. For a frame with resolution of 640*480, the average on-line reconstruction time is 415 ms, and the ICP attitude is estimated 20 times, with 100.0 ms, which is suitable for robot vision, AR/VR application and other high-speed tracking area for the reconstruction of visual scene.

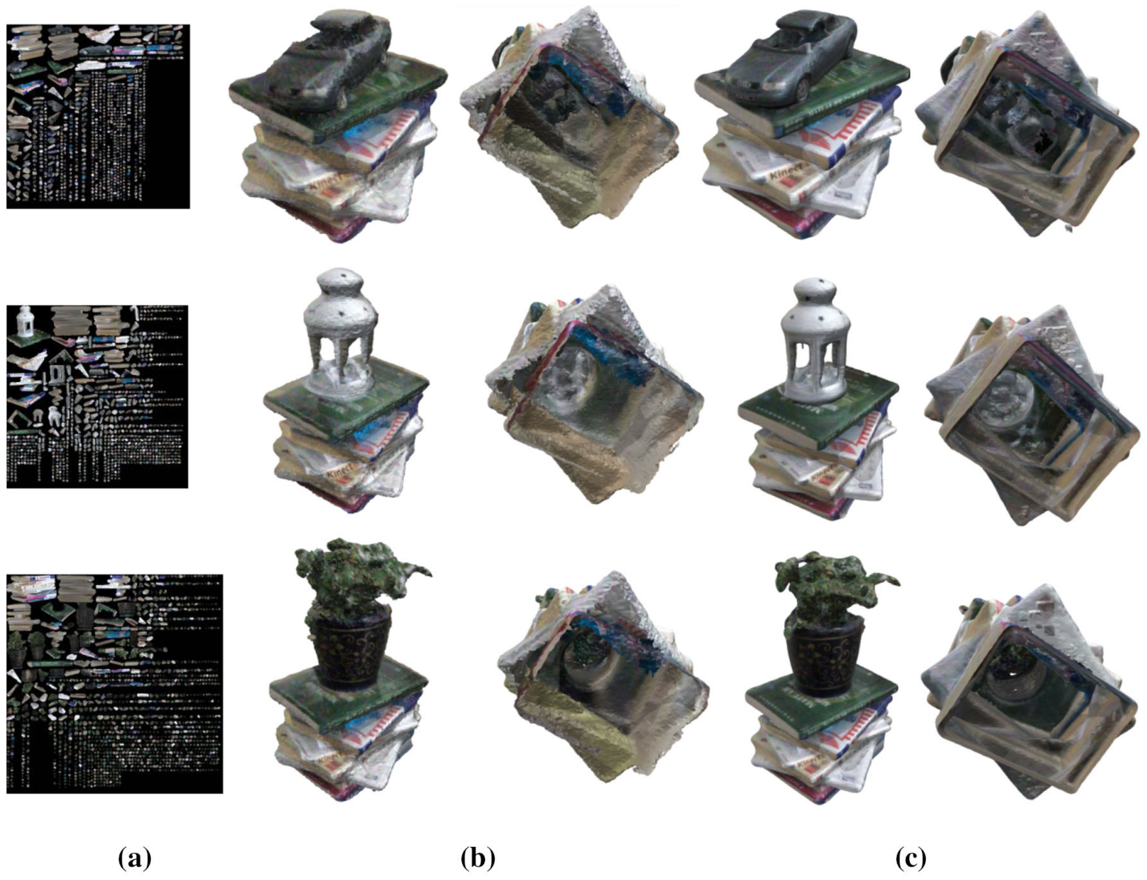


Fig. 9 Comparison between Whelan and our algorithm for 3D reconstruction. **a** initial RGB, **b** Whelan algorithm, and **c** our algorithm

Acknowledgements This work is funded by National High-tech R&D Program (863 Program) (No. 2014AA7031010B), Science and Technology Project of the thirteenth Five-Year Plan (JJZ[2016]345).

References

1. Weise, T., Wismer, T., & Leibe, B., et al. (2009). In-hand scanning with online loop closure. In *IEEE international conference on computer vision workshops* (pp. 1630–1637).
2. Henry, P., Krainin, M., Herbst, E., et al. (2012). RGB-D mapping: Using depth cameras for dense 3d modeling of indoor environments. *International Journal of Robotics Research, 31*(5), 647–663.
3. Keller, M., Lefloch, D., & Lambers, M., et al. (2013). Real-time 3D reconstruction in dynamic scenes using point-based fusion. In *International conference on 3D Vision-3DV* (Vol. 8768, Issue 2, pp. 1–8).
4. Whelan, T., Leutenegger, S., & Salas-Moreno, R. F., et al. (2015). ElasticFusion: Dense SLAM without a pose graph. Robotics: Science and Systems (RSS).
5. Merrell, P., Akbarzadeh, A., & Wang, L., et al. (2007). Real-time visibilitybased fusion of depth maps. In *IEEE international conference on computer vision* (Vol. 8, pp. 1–8).
6. Meilland, M., & Comport, A. (2013). On unifying key-frame and voxel-based dense visual slam at large scales. *IEEE/RSJ International Conference on Intelligent Robots & Systems, 8215*(2), 3677–3683.
7. Gallup, D., Pollefeys, M., & Frahm, J. M. (2010). 3D reconstruction using an n-layer heightmap. *Pattern Recognition, 6376,* 1–10.
8. Wurm, K. M., Hornung, A., & Bennewitz, M., et al. (2010). OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In *IEEE international conference on robotics and automation*.
9. Curless, B., & Levoy, M. (1996). A volumetric method for building complex models from range images. In *In Proceedings of SIGGRAPH*. ACM (pp. 303–312).
10. Newcombe, R. A., Izadi, S., & Hilliges, O., et al. (1996). KinectFusion: Real-time dense surface mapping and tracking. In *Conference on computer graphics and interactive techniques* (Vol. 3, pp. 303–312).
11. Steinbruecker, F., Sturm, J., & Cremers, D. (2014). Volumetric 3D mapping in real-time on a CPU. In *IEEE*

international conference on robotics and automation (pp. 2021–2028).

12. Zollhöfer, M., Thies, J., Colaianni, M., et al. (2014). Interactive model-based reconstruction of the human head using an RGB-D sensor. *Computer Animation and Virtual Worlds, 25*(25), 213–222.
13. Zhou, Q. Y., & Koltun, V. (2014). Color map optimization for 3d reconstruction with consumer depth cameras. *ACM Transactions on Graphics, 33*(4), 1–10.
14. Choi, S., Zhou, Q. Y., & Koltun, V. (2015). Robust reconstruction of indoor scenes. In: *Computer vision and pattern recognition* (pp. 5556–5565).
15. Wikowski, A., Kornuta, T., Stefańczyk, M., et al. (2016). Efficient generation of 3D surfel maps using RGB-D sensors. *International Journal of Applied Mathematics and Computer Science, 1,* 99–122.
16. Kornuta, T., & Laszkowski, M. (2016). Perception subsystem for object recognition and pose estimation in RGB-D images. *Automation, Springer International Publishing, 44*(10), 995–1003.
17. Whelan, T., Johannsson, H., & Kaess, M., et al. (2013). Robust real-time visual odometry for dense RGB-D mapping. In *IEEE international conference on robotics and automation*.
18. Lin, J. H., Wang, Y. J., & Sun, H. H. (2017). A feature-adaptive subdivision method for real-time 3D reconstruction of repeated topology surfaces. *3D Research, 8,* 6. doi:10.1007/s13319-017-0117-z.
19. Qu, Y., Liu, Z., Jiang, Y., et al. (2017). Self-adaptative variable-metric feature point extraction method. *Editorial Office of Optics and Precision Engineering, 25*(1), 188–197. **(In Chinese)**.
20. Liu, Y., Wang, C., Gao, N., et al. (2017). Point cloud adaptive simplification of feature extraction. *Editorial Office of Optics and Precision Engineering, 25*(1), 245–254. **(In Chinese)**.
21. Maier, R., Sturm, J., & Cremers, D. (2014). Submap-based bundle adjustment for 3D reconstruction from RGB-D Data. In *Pattern Recognition* (pp. 54–65).
22. Engel, J., Schöps, T., & Cremers, D. (2014). *LSD-SLAM: Large-scale direct monocular SLAM* (pp. 834–849). Zurich: Springer.
23. Stückler, J., & Behnke, S. (2014). Multi-resolution surfel maps for efficient dense 3D modeling and tracking. *Journal of Visual Communication and Image Representation, 25*(1), 137–147.
24. Nießner, M., Dai, A., & Fisher, M. (2014). Combining inertial navigation and ICP for real-time 3D Surface Reconstruction.
25. Kerl, C., Sturm, J., & Cremers, D. (2013). Dense visual SLAM for RGB-D cameras. In *IEEE international conference on intelligent robots and systems* (pp. 2100–2106).
26. Zhang, L., Wang, Y., Sun, H., et al. (2016). Adaptive scale object tracking with kernelized correlation filters. *Editorial Office of Optics and Precision Engineering, 24*(2), 448–459. **(In Chinese)**.
27. Wang, Y., Zhang, Q., & Zhou, Y. (2015). *Dense 3D mapping for indoor environment based on Kinect-style depth cameras* (Vol. 345, pp. 317–330). Cham: Springer.
28. Sturm, J., Engelhard, N., Endres, F., et al. (2012). A benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 573–580). IEEE.