the Visual Computer manuscript No.
(will be inserted by the editor)

Fast Capture of Textured Full-Body Avatar with RGB-D Cameras

Shuai Lin · Yin Chen · Yu-Kun Lai · Ralph R. Martin · Zhi-Quan Cheng*

Received: 2016-02-21 / Accepted: 2016-04-20

Abstract We present a practical system which can provide a textured full-body avatar within three seconds. It uses sixteen RGB-depth (RGB-D) cameras, ten of which are arranged to capture the body, while six target the important head region. The configuration of the multiple cameras is formulated as a constraint-based minimum set space-covering problem, which is approximately solved by a heuristic algorithm. The camera layout determined can cover the full-body surface of an adult, with geometric errors of less than 5 mm. After arranging the cameras, they are calibrated using a mannequin before scanning real humans. The 16 RGB-D images are all captured within 1 s, which both avoids the need for the subject to attempt to remain still for an uncomfortable period, and helps to keep pose changes between different cameras small. All scans are combined and processed to reconstruct the photo-realistic textured mesh in 2 s. During both system calibration and working capture of a real subject, the high-quality RGB information is exploited to assist geometric reconstruction and texture stitching optimization.

Keywords Full-Body Avatar · High-Quality Texture · RGB-Depth Camera · Global Registration

Shuai Lin
PDL, School of Computer, National University of Defense Technology, China

Yin Chen
Defense Engineering School, PLA University of Science and Technology, China

Yu-Kun Lai · Ralph R. Martin
School of Computer Science & Informatics, Cardiff University, UK

Zhi-Quan Cheng
Avatar Science Company, China
*Corresponding author, E-mail: cheng.zhiquan@avatarscience.com

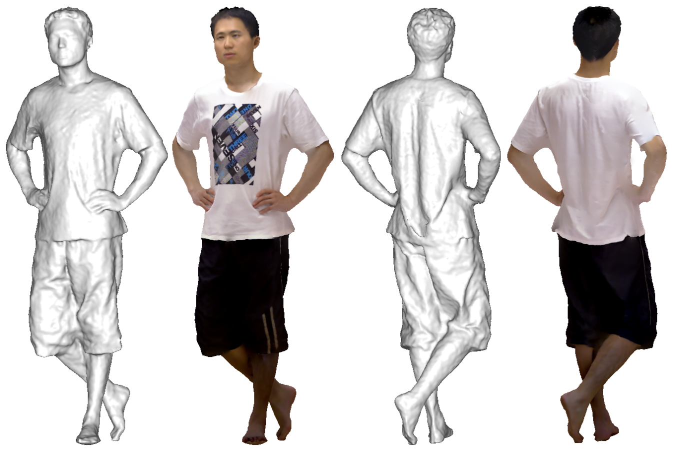


Fig. 1 Two views of a 3D avatar captured using our system, each showing the reconstructed geometry without and with texture.

1 Introduction

Avatars are used in many computer graphics applications as representations of users or other persons. Commodity RGB-D cameras (e.g. Kinect [1], Primesense [2], Xtion Pro Live [3]) provide low-cost scanning hardware which can be used as a basis for capturing 3D personalized avatars in the form of textured full-body models. Typical devices currently can produce both a 640×480 color image and a registered 640×480 depth image, at a rate of 30 frames per second.

The pioneering KinectFusion [4, 5] is a GPU-based capture system using a Kinect camera for both tracking and rigid surface reconstruction, allowing users to incrementally capture geometrically accurate 3D models. It has motivated many follow-up algorithms [6–20]. Generally, such algorithms fuse successive dense RGB-D frames into a signed-distance-function (SDF) volume and perform camera tracking based on this model. This problem may also be described in terms of simultaneous localization and mapping (SLAM), where both the camera pose and the map (i.e. scene geometry) have to be estimated at the same time. Such algorithms have tried many strategies to improve the tracking speed and to expand its spatial mapping capabilities to larger scenes or deformable objects. In particular, the recent DynamicFusion algorithm [20] demonstrates robust performance when reconstructing a non-rigidly deforming human body in real-time.

The goal of avatar capture is to build a visually pleasing result. However, it is well-known that a massive amount of RGB-D data is not essential and that collecting redundant data can significantly increase both scanning and processing time. Until now, a theoretical analysis has not been done to analyze the best multiple camera configuration for full-body avatar capture. Ideally the camera setup should completely cover the entire body, although self-occlusion of a few small areas such as the armpits means that it is almost impossible to achieve full-body coverage with a fixed camera setup. The solution for camera locations should carefully consider various important factors, such as desired proportion of body-surface coverage, geometric error bounds, camera capabilities, and number of cameras, and a trade-off between these is always required.

RGB-D cameras already provide raw high-quality RGB image data which can be used to build an avatar with good visual appearance, by using standard texture mapping techniques. Even if the geometric accuracy is low from a single frame, the textured partial mesh may be visually pleasing.

By considering the configuration of multiple cameras and by taking advantage of RGB texture, we have devised a practical system to capture personalized avatars with 16 RGB-D cameras. The system can produce pleasing 3D textured results (see Figure 1), as well as having very high capture speed (it takes about 3 s, including all processing, to deliver the final avatar). Our main contributions includes:

– A configuration solution for multiple RGB-D cameras for full-body avatar capture. Finding such an optimal configuration is an NP-hard problem. We utilize a heuristic algorithm to approximately solve it, taking into account constraints determined by the features of the RGB-D cameras. Our solution uses 16 appropriately configured cameras, enabling us to provide almost full-body surface coverage with 5 mm geometric tolerance.
– A color-aware reconstruction method. The high-quality RGB information is fully exploited, both to provide effective correspondences during the geometric reconstruction of partial textured scans, and to provide accurate color constraints during post-optimization to provide seamless texture stitching.
– A fast avatar capture system. The hardware can scan a subject within 1 s to obtain the initial raw data, then perform the computations needed to produce an avatar in about 2 s. Overall, a textured full-body avatar is delivered in only 3 s, the fastest speed so far as reported.

The remainder of this paper is organized as follows. After surveying related work in Section 2, we address four main aspects of our system: the camera configuration (Section 3), data capture (Section 4), geometry processing (Section 5) and texture mapping (Section 6). Various results, a performance analysis, and comparisons are given in Section 7, and finally conclusions are drawn in Section 8.

2 Related work

We first briefly review related avatar capture research, concentrating on geometry capture and visual texture mapping. We consider representative papers concerning major developments leading to the state of the art in these fields.

After the advent of the KinectFusion system [4, 5] using a single camera, various methods [6, 8–20] have been proposed for full-body capture, using RGB-D data for camera pose estimation, dense mapping and full SLAM pipelines.

The latest template-free DynamicFusion algorithm [20] achieves real-time non-rigid reconstruction of a 3D geometric avatar; it generalizes the volumetric truncated SDF fusion technique [4, 5] to the non-rigid case, as well as providing a real-time approach to estimating a volumetric warp to give the surface of the scanned avatar.

However, DynamicFusion [20] takes over 60 s to capture sufficient data to produce a full-body avatar, as the scanning camera moves in a closed loop around the subject. It is obvious that much redundant input data is collected. Some partial scans are inessential and provide little benefit to the final result. However, no prior work appears to have been done to evaluate the amount of input data required to achieve a desired geometric error.

Besides the limitations of slow capture time resulting from redundant data input, RGB texture information is often not utilized in SLAM systems [4–6, 8–13, 15–20]. An exception is [14], which uses a global optimization approach for mapping the color images produced by an RGB-D camera onto the geometric model reconstructed from the corresponding depth data. It improves the quality of reconstructed color maps by use of multiple RGB color datasets: visual quality is rapidly optimized from a few RGB images using a texture mapping technique.

While [14] only targeted rigid static objects, non-rigid subjects were considered in [21–24]. These methods do not utilize a SLAM scheme, and their non-rigid reconstruction steps are very time-consuming. In contrast by utilizing a SLAM scheme, and integrating color information, we achieve a short processing time.

As shown by [25, 26], pre-configured multiple cameras can greatly shorten the scanning time. Sparse mapping of data from multiple carefully configured cameras can rapidly produce a suitable geometric avatar. However, their configurations do not take into account the SLAM scheme used.

We optimize a texture consistency term that takes full sparse image information into account and is naturally coupled with multiple calibrated cameras (with respect to localization and orientation). Seamlessly texturing a 3D model from a set of RGB images has also been investigated by [27, 28]. They try to select a single input RGB image per triangular mesh face and minimize seams, but these approaches are computationally expensive because of the need for an expensive combinatorial optimization. The computational burden can be alleviated by using the configuration information to discard many useless potential combinations.

3 Camera configuration

Determining a suitable multi-camera configuration can be posed as a constraint-based minimum set space-covering problem. More formally, given a set of N RGB-D cameras, the union of each camera's field-of-view (Fov) should cover the surface of the human subject (Surface(s)). The objective is to find the minimal N that meets the coverage requirements, while also satisfying the constraint that the maximal depth error $\mathbf{e}_d$ should be less than δ:

$$\underset{N}{\operatorname{argmin}} \left(\bigcup_{n=1}^{N} \mathrm{Fov}(n) \supseteq \mathrm{Surface}(s) \right) \quad \text{such that} \quad \mathbf{e}_d < \delta. \tag{1}$$

Typical applications demand that errors in the captured avatar model should be less than a few millimetres. Specifically, we set Surface(s) to be the full surface of an adult subject s. To take a concrete example, for the body, we set a target for geometric errors to be less than $\delta_{body} = 5$ mm, while for the head, we set $\delta_{head} = 1.6$ mm. In practice, self-occlusion means that some areas (e.g. the armpits) are very tricky to capture with a reasonable number of cameras, so we replace the actual body by a proxy constructed from cylinders, and aim for full coverage of that. For a general subject, the body surface is simplified to a cylinder $\mathrm{C}(r_b, h_b)$, with radius $r_b = 0.45$ m, and height $h_b = 1.7$ m, while the head surface is simplified to a cylinder $\mathrm{C}(r_h, h_h)$, with radius $r_h = 0.2$ m, and height $h_h = 0.4$ m.

The constraint-based minimum set space-covering problem (shorten as c-space-covering) as given in Equation (1) is a typical minimum set space-covering problem, and is known to be NP-hard [29].

The maximal depth error e_d depends on the properties of the RGB-D cameras. We utilize Xtion Pro Live cameras [3], which have an angular field-of-view of 57° horizontally and 43° vertically, and which can capture points whose depths lie in the range 0.4–3 m when used in near mode. The accuracy of the Xtion Pro Live has been thoroughly analyzed in [30]. The maximal depth error e_d increases quadratically with distance d according to $e_d = 2.85 \times 10^{-6} d^2$. For the body, to ensure that e_d does not exceed 5 mm, the distance between the subject surface and the camera center should not exceed 1.32 m, while for the head, the distance must be less than 0.75 m.

Algorithm 1 Heuristic algorithm for constraint-based minimum set space-covering problem

```
Input: $\mathrm{Fov}(n), \mathrm{Surface}(s), \delta_{head}, \delta_{body}$
Output: N cameras configuration
 1: Decompose Surface$(s)$ into $S_{\text{body}}$ and $S_{\text{head}}$;
 2: // Handle the head surface
 3: $S_{\text{head}}$ is simplified to a cylinder $\mathrm{C}(r_h, h_h)$;
 4: $N_{\text{head}}$=1;
 5: Place 1st camera with $\delta_{head}$ constraint, covering one region of $\mathrm{C}(r_h, h_h)$;
 6: repeat
 7:     $N_{\text{head}}$++;
 8:     Adding $N_{\text{head}}$-th camera with $\delta_{head}$ constraint: intersecting $\bigcup_{n=1}^{N_{\text{head}}-1} \mathrm{Fov}(n)$ and maximizing $\bigcup_{n=1}^{N_{\text{head}}} \mathrm{Fov}(n)$;
 9: until ( $\bigcup_{n=1}^{N_{\text{head}}} \mathrm{Fov}(n) \supseteq \mathrm{C}(r_h, h_h)$).
10: Re-arrange the head cameras to equal distance spacing.
11: // Handle the body surface
12: $S_{\text{body}}$ is simplified to a cylinder $\mathrm{C}(r_b, h_b)$;
13: $N_{\text{body}}$=1;
14: Place 1st camera with $\delta_{body}$ constraint, covering one corner region of $\mathrm{C}(r_b, h_b)$;
15: repeat
16:     $N_{\text{body}}$++;
17:     Adding $N_{\text{body}}$-th camera with $\delta_{body}$ constraint: intersecting $\bigcup_{n=1}^{N_{\text{body}}-1} \mathrm{Fov}(n)$ and maximizing $\bigcup_{n=1}^{N_{\text{body}}} \mathrm{Fov}(n)$;
18: until ( $\bigcup_{n=1}^{N_{\text{body}}} \mathrm{Fov}(n) \supseteq \mathrm{C}(r_b, h_b)$).
19: Re-arrange the body cameras to equal distance spacing.
20: $\mathbf{N} = N_{\text{head}} + N_{\text{body}}$;
```

As the human is represented by a combined cylindrical proxy, we utilize a heuristic algorithm to find an approximate camera arrangement. We successively place cameras to cover as much as possible of the subject, each camera orienting to the proxy with a proper distance constrained by the accuracy requirement. Consequently, the proxy could be covered by several horizontal rings of cameras, and the task is simplified as one of deciding the number and heights of the rings, and how many cameras to place in each ring. The framework of the heuristic solution is addressed by Algorithm 1, which determines that 16 cameras provide the desired solution.

A configuration of 16 RGB-D cameras, computed by this algorithm, is used by our capture system. It is illustrated in Figure 2: the system uses ten for the body, and six for the head.

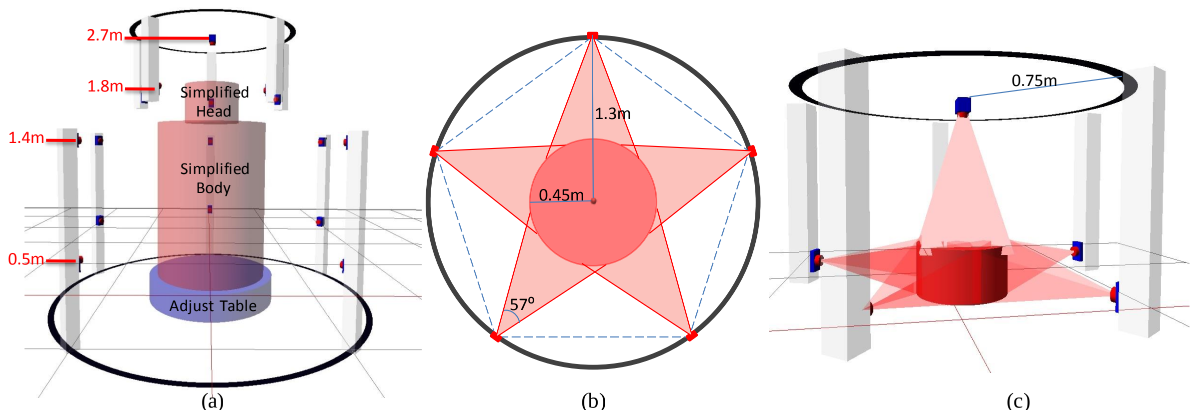


Fig. 2 System configuration with 16 RGB-D cameras. (a) Ten cameras capture the body, simplified as a cylinder (0.45 m radius, 1.7 m height) standing on an adjustable platform. Ten cameras are organized into five vertical pairs, uniformly located around a circle with radius 1.3 m. The two cameras in each pair are placed at 0.5 m and 1.4 m above the floor. Six cameras capture the head, simplified as a cylinder (0.2 m radius, 0.4 m height). Five of these are placed uniformly around a circle with radius 0.75 m, 1.8 m above the floor, while the final camera is placed vertically above the head 2.7 m from the floor. (b) Top-view of the body capture space, formed by the union of the fields-of-view of the cameras. (c) The head capture space formed by the other six cameras.

– Head capture. Six cameras are computed from Algorithm 1 Lines 3-9, then re-arranged at equal angles around the head (Line 10) as shown in Figure 2. Five cameras are placed on a horizontal ring 1.8 m above the floor to cover the sides the cylinder representing the head, uniformly located around a circle of radius 0.75 m. The subject stands on an adjustable platform to raise his or her height so that the subject's nose is at a height of 1.8 m. One camera is placed vertically above the head, looking down, at a height of 2.7m.
– Body capture. Ten cameras are computed from Algorithm 1 Lines 12-18, then re-arranged into five vertical pairs at equal angles around the body (Line 19) as shown in Figure 2.b, lying around a circle of radius 1.3 m. In each pair, one is placed 0.5 m above the floor, and the other is 1.4 m above it. The fields-of-view of these ten cameras form a polyhedral volume (see Figure 2.b), covering the simplified body $\mathrm{C}(r_b, h_b)$.

4 Scanning and offline multi-camera calibration

After placing the cameras close to the above layout, they are used to scan a full-sized human mannequin, placed on the adjustable platform. The scanned data is cleaned in a preprocessing step, then these cameras are calibrated to determine their precise layout before they are used for real avatar capture. The calibration is performed just once in an offline process.

4.1 Scanning

Figure 3 shows the mannequin data captured by the 16 cameras, in RGB and depth images. The cameras must essentially sequentially capture data to avoid interference between the active cameras: if multiple depth maps are simultaneously scanned, major quality degradation is found to occur in overlapping areas. Each camera captures only a single RGB-D frame, then turns off, allowing the next camera to capture a frame and so on.

To shorten the scanning time, in practice certain cameras may be paired such that in each pair, scanning regions do not overlap, allowing the cameras in a pair to work simultaneously. Specially, when a head-related camera is scanning, one body-related camera without scanning overlap works at the same time. After the 6 head cameras and related body cameras have scanned, the final 4 body-related cameras capture data sequentially. A camera takes under 0.1 s to turn on, capture data, and turn off. Therefore, the whole scanning time using 16 cameras is about 1 s.

4.2 Data pre-processing

After scanning, we perform pre-processing on the initial raw RGB-D data. The steps include data cleaning, RGB-D to surface conversion, and platform removal.

We adapt the fast-speed filtering technique in [31] to perform data cleaning. We first employ the Sobel operator to detect boundary pixels at depth edges, and remove unreliable boundary points (pixels near depth edges) by thresholding depth gradient magnitudes. The main difference between our method and the one in [31] is that we only consider spatial filtering rather than spatitemporal filtering. Such spatial filtering can effectively denoise the depth data obtained by the RGB-D cameras.

Each frame is a partial scan of the captured subject, comprising a color image registered with a depth image. This

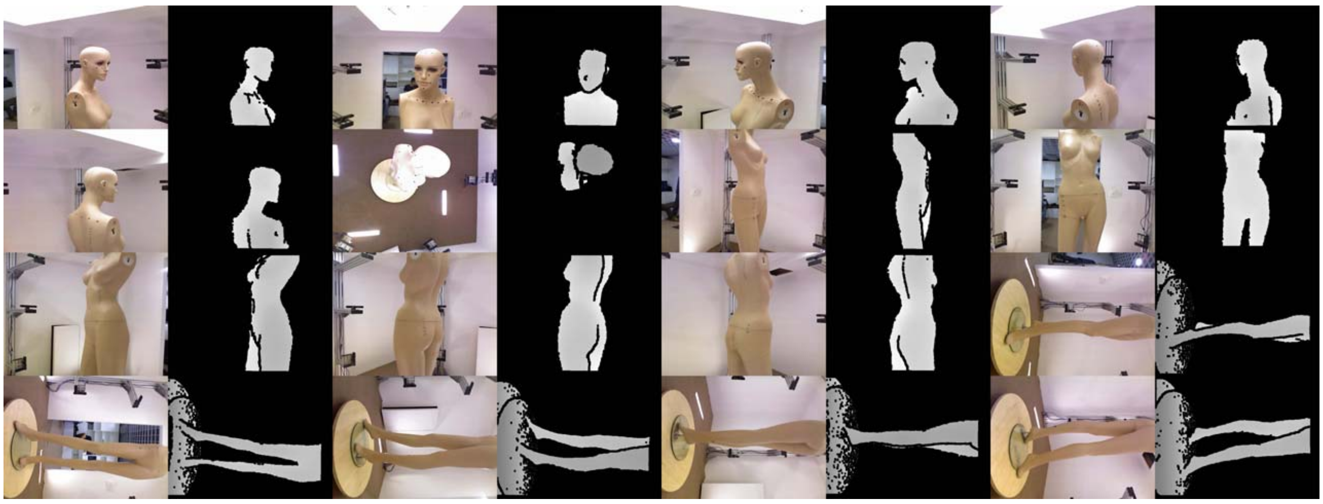


Fig. 3 Mannequin raw data captured by all 16 cameras, as RGB and depth images.

pair of images is converted to a 3D mesh surface with texture information using the OpenNI package [32]. The conversion is fast, taking about 1 ms per frame.

The partial surfaces resulting from the lower body cameras still contain data captured from the platform upon which the subject stands. Following the floor removal step in [26], we estimate a lowest horizontal plane P for all points. Any points below P and up to 10 mm above it are treated as outliers belonging to the floor and removed.

All preprocessing steps take in total about 10 ms on a single CPU.

4.3 Offline multi-camera calibration

The goal of multi-camera calibration is to allow the 3D coordinates of every point captured by every camera to be put into a common reference frame.

A mannequin is serving as the calibration template, since it is mostly similar to a real adult in the shape space. The mannequin is accurately pre-scanned offline by an accurate laser scanner. Obviously, this template model can be provided to users of our capture system, who therefore do not need to also own a laser scanner, just a mannequin. The mannequin is placed in the center of our system, and each camera captures its own view. We then employ a matching algorithm [33] to determine rigid alignment between each partial scan and the whole template. This provides an initial alignment for each partial scan (see Figure 4.b), as well as the initial parameters of our revised SLAM-based multi-camera calibration.

Based on the initial alignment, we optimize the multi-camera calibration by a revised rigid SLAM scheme. We utilize the SLAM scheme, rather than the bundle adjustment (e.g. [34]), to achieve the global error optimization, since it is fast due to the linearized numerical solution. Our approach is based on [13], which performs real-time volumetric 3D mapping on a CPU, based on depth information. We extend it by integrating texture color information to optimize the calibration.

In an RGB-D camera frame, a 3D point is defined in homogeneous coordinates as $\mathbf{p} = (X,Y,Z,1)^{\mathrm{T}}$. We reconstruct a point from its pixel coordinates $\mathbf{x} = (x,y)^{\mathrm{T}}$ and a corresponding depth measurement $Z = Z(\mathbf{x})$ using the inverse projection function π^{-1}:

$$\mathbf{p} = \pi^{-1}(\mathbf{x},Z) = \left(\frac{x-o_x}{f_x}Z, \frac{y-o_y}{f_y}Z, Z, 1\right)^{\mathrm{T}}, \tag{2}$$

where f_x, f_y are the focal lengths and o_x, o_y are the coordinates of the camera center in the standard pinhole camera model. f_x, f_y, o_x, and o_y are already known intrinsic parameters of the RGB-D camera. The pixel coordinates for a point can be computed using the projection function π:

$$\mathbf{x} = \pi(\mathbf{p}) = \left(\frac{Xf_x}{Z} + o_x, \frac{Yf_y}{Z} + o_y\right)^{\mathrm{T}}. \tag{3}$$

With the previously defined projection function π and rigid camera motion $\mathbf{T}$ between two cameras, we can derive a warping function τ, which computes the location of a pixel from the first image I_1 in the second image I_2, given the rigid motion $\mathbf{T}$:

$$\mathbf{x}' = \tau(\mathbf{x},\mathbf{T}) = \pi_2\Big(\mathbf{T}\pi_1^{-1}\big(\mathbf{x}, Z_1(\mathbf{x})\big)\Big). \tag{4}$$

Based on the warping function τ we define the intensity error r_Z for a pixel $\mathbf{x}$ as

$$r_Z = Z_2\big(\tau(\mathbf{x},\mathbf{T})\big) - Z_1(\mathbf{x}), \tag{5}$$

Similarly, the depth error is given as

$$r_Z = Z_2\big(\tau(\mathbf{x},\mathbf{T})\big) - \Big[\mathbf{T}\pi_1^{-1}\big(\mathbf{x}, Z_1(\mathbf{x})\big)\Big]_Z, \tag{6}$$

where $[\cdot]_Z$ returns the Z component of a point, i.e., $[\mathbf{p}]_Z = Z$. Therefore, the second term is the depth of the transformed point, which was reconstructed from the first depth image.

The previous work [13] gave a probabilistic formulation for the estimation of the camera motion $\mathbf{T}$ given the depth error r_Z. They determined the motion $\mathbf{T}^*$ by maximizing an iteratively re-weighted (w) least squares formulation:

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} \sum_{i=1}^{N} w_i r_{Z,i}^2, \tag{7}$$

where N is the total number of RGB-D images; in our system, $N = 16$. We extend the previous formulation by incorporating the photometric color error r_C. Therefore, we model the photometric and the depth error as a bivariate random variable $\mathbf{r} = (r_C, r_Z)^{\mathrm{T}}$, In the bivariate case, Equation (7) becomes:

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} \sum_{i=1}^{N} w_i \mathbf{r}_i^{\mathrm{T}} \mathbf{r}_i. \tag{8}$$

The error function (8) we seek to minimize is nonlinear in the motion parameters $\mathbf{T}$. Therefore, we linearize it around the current motion estimate $\mathbf{T}_k$ using a first order Taylor expansion. Furthermore, we employ a coarse-to-fine scheme (in a similar way to the multi-resolution octree implementation in [13]) to account for a larger range of camera motions.

Consequently, the revised rigid SLAM scheme finds the camera motions, giving the multi-camera calibration. Figure 4 shows intermediate results in the steps of this process, including coarse alignment, and the revised rigid-SLAM optimization integrating color information. The textured template of the mannequin, reconstructed by our approach, is also shown in Figure 4.

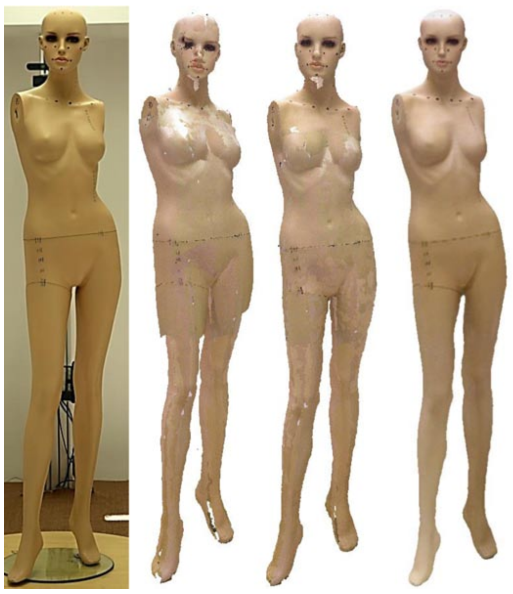


Fig. 4 Calibration with mannequin. Left to right: 2D snapshot from a high-resolution digital camera, 3D coarse alignment from partial scans, alignment after the revised rigid SLAM optimization integrating the color information, and the final textured result.

5 Geometric reconstruction of a real subject

After calibrating the camera locations, the system is ready to capture real subjects.

Firstly, the subject stands on the platform, which automatically adjusts according to the subject's height, placing the nose at a height of 1.8 m above the ground. The subject then holds as stationary a pose as possible for about a second. During this time, the system scans the raw RGB-D images, as described in Section 4.1.

Next, the raw RGB-D images are pre-processed as in Section 4.2.

Thirdly, by using the multi-camera calibration information as the initialization, revised rigid SLAM optimization (as in Section 4.3) is performed to find the global rigid alignment of the multiple overlapping partial datasets.

Fourthly, a non-rigid optimization algorithm is used to compensate for any misalignments due to deformation. A human subject cannot hold perfectly still during scanning, leading to small non-rigid deformation between scans. In each overlapping region, we first build an embedded graph on each overlapped surface, following [35], and use it to determine deformation. Local transformations are defined at each node of the embedded graph, with the transformation at each vertex being obtained by a weighted average of transformations at neighboring nodes. Each weight is proportional to the inverse of the distance between the vertex and the node. Corresponding points on overlapping surfaces are established using ORB features [36], which allow real-time performance without use of GPUs, and provide good invariance of RGB images to changes in viewpoint and illumination. We prune the ORB feature matching pairs using local region constraints, and only use a few high-confidence pairs as the correspondences. Driven by these correspondences, the embedded deformation algorithm [35] can quickly perform non-rigid registration.

Finally, we make use of the volumetric SDF method in [13] to fuse the warped triangle meshes. The resulting triangle mesh is closed, since holes are filled when reconstructing from the octree volume.

6 Texture stitching

After reconstructing the geometric mesh, the texture information is attached to it using a texture stitching approach, which minimizes texture seams between adjacent triangles.

We assign a label l_j to each face $f_j (j = 1, \ldots, F)$ that identifies its associated single input image I_i from the input RGB images $\{I_1, \ldots, I_N\}$ $(N = 16)$. This projection between faces and images uses the previous camera parameters and the geometric mesh M. This is essentially a discrete label assignment problem, which can be formulated as:

$$\arg\min_{l_1,\ldots,l_F} \sum_{j=1}^{F} E_{data}(l_j) + \lambda \sum_{\{f_j,f_k\}\in \mathrm{M}, f_j\in \mathscr{N}(f_k)} E_{smooth}(l_j,l_k), \tag{9}$$

where λ is the weight of the smoothness term, empirically set to $\lambda = 10$, and $\mathscr{N}(\cdot)$ is the set of neighboring faces.

We experimentally set $E_{data}(l_j) = \sin^2\theta$, where θ is the angle between the local viewing direction of the corresponding camera and the face normal. It is smaller for RGB images with better quality (smaller angle θ), seeking to texture each face from the best image that contains it.

$E_{smooth}(l_j,l_k) = \int_{e_{(j,k)}} \left\| I_{l_j}\left(\phi_{l_j}(p)\right) - I_{l_k}\left(\phi_{l_k}(p)\right)\right\|^2 dp$, to measure the integrated difference of the colors along each shared edge ($e_{(j,k)}$) between adjacent triangles f_j and f_k. ϕ_{l_j} is the perspective map from mesh face f_j into image I_{l_j}. This term is zero for pairs of faces with the same label.

Equation (9) is a typical discrete energy minimization problem. Different to [28] using the slow tree-reweighted message passing solution, we solve Equation (9) using the efficient α-expansion solution [37]. As explained in a recent survey [38], the global minimum can be computed rapidly by α-expansion graph cut, especially in multi-camera configurations.

More important, before solving Equation (9), the angle θ is first used to remove obviously unreasonable labeling candidates. If the image I_i does not map on the triangular face f_j, θ_i is set as 180° and directly removed from the candidates. Then, all labeling candidates are reversely sorted by the corresponding angle θ. Assuming the first candidate with smallest angle θ_1, we directly remove the candidates whose corresponding angles are larger than $\theta_1 + 30°$. This discards some useless combinations, and helping to reduce the computation costs.

Benefiting from the efficient α-expansion solution [37] coupled with angle thresholding scheme, Equation (9) could always be solved about 0.7 second. This means our texture stitching is fast, avoiding the slow computation time as [28].

After texture stitching, local luminosity variations along optimized texture boundaries are further handled by using Poisson blending [39]. Furthermore, small holes without mapping images are also filled by methods in [39] to produce the mapped textures. Note that, we combine and layout all 16 RGB images to form a large texture image. Given the reconstructed geometric model and the 16 RGB views around it, the establishment of texture coordinate is a simple process: each point on the model can be back-projected to views to retrieve color at that point, by utilizing the known camera parameters and calibration information.

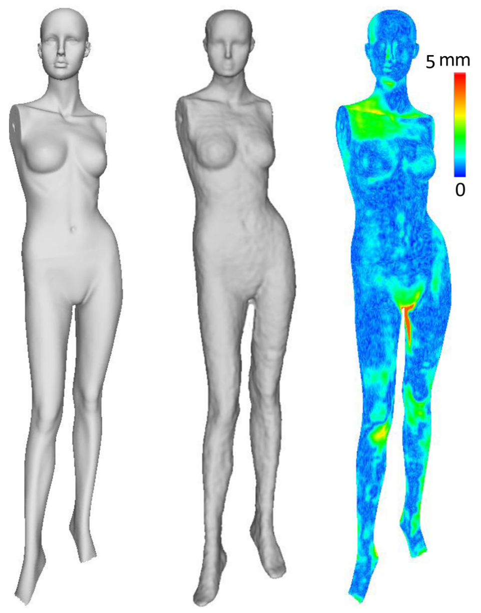


Fig. 5 Accuracy of reconstruction, compared to the laser-scanned mannequin template. Left to right: accurate template, reconstructed mesh, L_2 distance error on the template surface.

7 Results

7.1 Evaluation

Our system meets its goals of rapidly constructing an avatar with photorealistic visual appearance, and whose geometric errors do not exceed 5 mm.

The visual quality of the result is determined by two considerations: accuracy of captured geometry, and visual appearance. The visual appearance is provided by texture stitching of the RGB images mapped onto the geometric surface. Geometric accuracy is determined by the RGB-D camera, whose accuracy falls offf quadratically with distance as noted in [30]. Our configuration ensures geometric errors are less than 5 mm on the body regions where data is captured. The remaining holes due to self-occlusion are small (see the calibrated data in Figure 4), and cover less than 5% of the whole surface area. Figure 5 validates our reconstructed avatar against an accurate template of the mannequin produced by a higher quality 3D scanner, which has an accuracy of about 0.1 m. The L_2 distance error between the reconstructed mesh and the template is shown on the right. The maximal distance error is about 5 mm.

7.2 Experimental tests

The system has been validated by capturing and reconstructing personalized avatars for many human subjects; typical results are shown in Figures 1 and 6. The reconstructed surfaces contain rich geometric details specific to the subject,

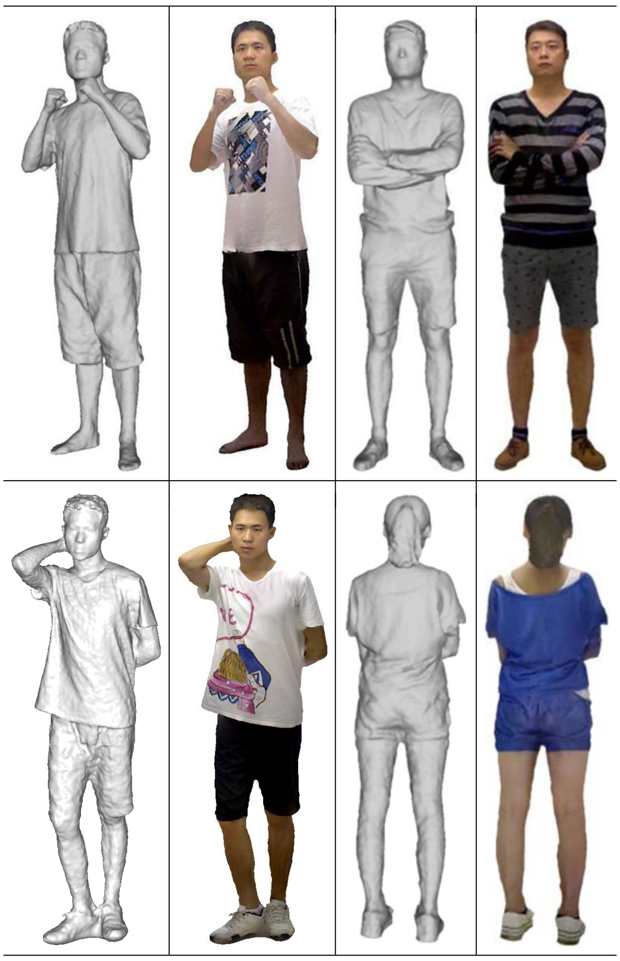


Fig. 6 Avatars of several different people, showing for each subject the reconstructed surface and textured model. Personal geometric details, such as the face, salient cloth wrinkles, and hairstyle, are well captured.

Table 1 Typical running times.

scanning	pre-process	rigid SLAM	non-rigid	texturing
1 s	0.1 s	0.1 s	0.8 s	1 s

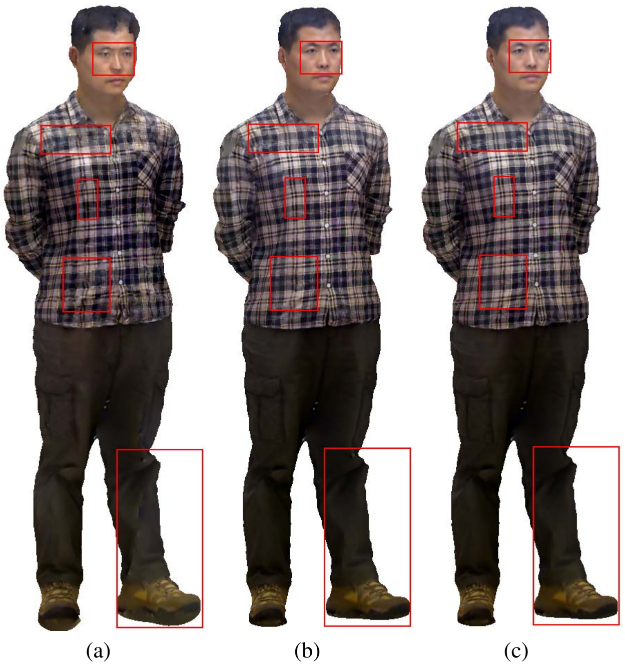


Fig. 7 Effectiveness of use of color information during rigid and non-rigid geometric optimization. (a) Original SLAM method [13] without using color. (b) Our rigid SLAM scheme integrating color information. (c) Our non-rigid scheme using RGB correspondences. Compare results in the regions marked by red rectangles.

Table 2 Comparison of typical RGB-D methods. N_c: number of RGB-D cameras, T_s: scanning time, T_p: processing time, E: maximal L_2 geometric distance error, C: color results supplied.

	From [25]	From [20]	Our method
N_c	3	1	16
T_s	30 s	>60 s	1 s
T_p	360 s	real-time	2 s
E	10 mm	a few mm	5 mm
C	color	no	texture

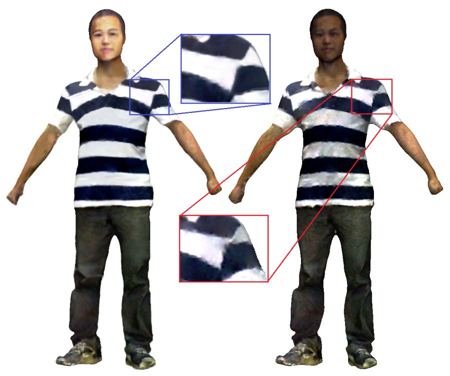
such as their individual faces, hairstyles and folds of their clothing. Moreover, since an RGB-D camera captures both color and depth images simultaneously, the final avatar is textured, as also shown in Figure 6.

Running times for a PC laptop with a 1.6 GHz Intel Core i7 processor are reported in Table 1. The whole capture process takes about 3 s, including 1 s for scanning, and 2 s for data processing. (Multi-camera calibration is performed offline and is not included in these timings). As pre-computed calibration matrices are used, alignment using rigid SLAM scheme is very fast. Reconstruction time is mainly spent on non-rigid registration and texture stitching.

Thanks to the effective use of RGB color information in the rigid SLAM framework, our revised color-aware method produces better results than the original depth version [13]. As shown by Figure 7, in the regions marked by red rectangles, visual alignment is more accurate than produced by the method in [13]. While the combined RGB and depth method is slightly slower than the depth-only version, the speed is still acceptable.

7.3 Comparison

The results of a comparison between our system and other typical RGB-D camera systems described in [20, 25], are shown in Table 2. Here T_s and T_p are the scanning time and data processing times respectively, while E is the maximal L_2 distance error.


(a) Front

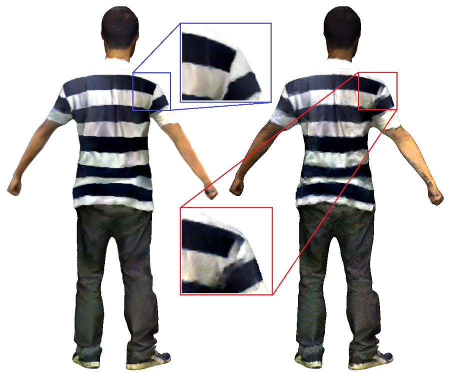

(b) Back

Fig. 8 Visual comparison of our result (left), and those using using the method in [25] (right).

A key advantage of our system is the scanning speed, about 1 s, which is short enough to allow the person being captured to hold a fairly static pose stably and comfortably.

The differences between our method and the one in [20] should be further examined. Initially it seems that the latter approach is better in terms of geometric accuracy. In fact, their accuracy of a few mm can only be achieved if a long enough time (> 60 s) is spent to acquire sufficient dense scans; in practice subjects will find it very difficult (as well as inconvenient) to stand still for such a length of time. More importantly, as is also the case for [13], RGB information is not utilized—yet it can definitely improve the visual appearance of the reconstructed avatar.

A visual comparison of the results of our approach and those provided by [25] is shown in Figure 8. The input RGB-D raw data was provided by the author of [25]. We manually selected 16 RGB-D images of one subject, and the processing steps were automatically performed to build the results. Our stitched texture appearance is noticeably better than the one produced by [25], which simply utilizes the color assignment on the reconstructed geometry without texture optimization.

We finally discuss the impact of varying number of RGB-D images. From representative KinectFusion [4] and DynamicFusion [20] work, it is easy to imagine the effect of using a large number RGB-D images—the reconstructed mesh would be more detailed, at the expense of processing much redundant data. In fact, we met the requirement for geometric error to be under 5 mm in earlier work [26], using six RGB-D images produced by two Kinects to capture a personalized avatar. As noted in that paper, however, the results do not cover a full adult full body, resulting in noticeable artifacts. A comparison between our current results and those of [26] are shown in Figure 9 for a typical example.

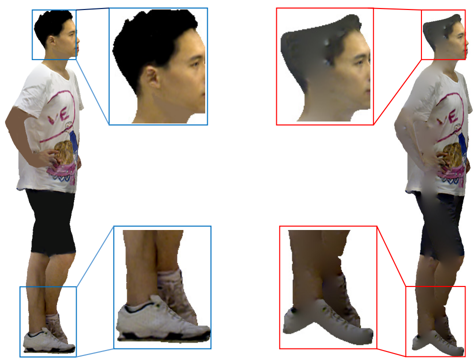


Fig. 9 Varying the number of RGB-D images. Left: our result using sixteen RGB-D images. Right: result from [26] using six RGB-D images.

8 Conclusions

We have described a fast photorealistic avatar capture system based on a configuration of multiple cameras, allowing a user to quickly and easily produce a textured avatar. Data scanning takes about 1 s, avoiding the need for the user to hold steady in an awkward pose for a long time. After scanning, the textured avatar is automatically reconstructed on a commodity PC in about 2 s.

As a next step, we plan to generalize the color-aware SLAM framework from the rigid to non-rigid case, following [20]. Furthermore, we hope to improve the geometric quality by using high-quality structured light scanners. Other techniques will be investigated to accelerate the scanning and computation to provide real-time results.

Acknowledgments. This work was supported by the Natural Science Foundation of China (No. 61272334).

References

1. Microsoft, “Kinect,” 2015. http://www.xbox.com/kinect.
2. Primesense, “Carmine sensor,” 2015. http://www.primesense.com/solutions/3d-sensor/.
3. ASUS, “Xtion pro live,” 2015. http://www.asus.com/Multimedia.
4. 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," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp. 559–568, 2011.

5. R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *10th IEEE International Symposium on Mixed and Augmented Reality*, pp. 127–136, 2011.
6. T. Whelan, J. B. McDonald, M. Kaess, M. F. Fallon, H. Johannsson, and J. J. Leonard, "Kintinuous: Spatially extended Kinect-Fusion," in *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, (Sydney, Australia), pp. 127–136, Jul 2012.
7. D. S. Alexiadis, D. Zarpalas, and P. Daras, "Real-time, full 3-d reconstruction of moving foreground objects from multiple consumer depth cameras," *IEEE Transactions on Multimedia*, vol. 15, pp. 339–358, Feb. 2013.
8. M. Niessner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3d reconstruction at scale using voxel hashing," *ACM Transactions on Graphics*, vol. 32, pp. 169:1–169:11, Nov. 2013.
9. J. Chen, D. Bautembach, and S. Izadi, "Scalable real-time volumetric surface reconstruction," *ACM Transactions on Graphics*, vol. 32, pp. 113:1–113:16, July 2013.
10. J. Sturm, E. Bylow, F. Kahl, and D. Cremers, "Copyme3d: Scanning and printing persons in 3d," in *German Conference on Pattern Recognition*, pp. 405–414, 2013.
11. F. Steinbrücker, C. Kerl, and D. Cremers, "Large-scale multi-resolution surface reconstruction from RGB-D sequences," in *IEEE International Conference on Computer Vision*, pp. 3264–3271, 2013.
12. Q.-Y. Zhou and V. Koltun, "Dense scene reconstruction with points of interest," *ACM Transactions on Graphics*, vol. 32, pp. 112:1–112:8, July 2013.
13. F. Steinbrücker, J. Sturm, and D. Cremers, "Volumetric 3D mapping in real-time on a CPU," in *Int. Conf. on Robotics and Automation*, pp. 2021–2028, IEEE, 2014.
14. Q.-Y. Zhou and V. Koltun, "Color map optimization for 3d reconstruction with consumer depth cameras," *ACM Transactions on Graphics*, vol. 33, pp. 155:1–155:10, July 2014.
15. M. Zollhöfer, M. Niessner, S. Izadi, C. Rehmann, 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*, vol. 33, pp. 156:1–156:12, July 2014.
16. H. Afzal, K. A. Ismaeil, D. Aouada, F. Destelle, B. Mirbach, and B. E. Ottersten, "Kinect Deform: Enhanced 3d reconstruction of non-rigidly deforming objects," in *2nd International Conference on 3D Vision*, pp. 7–13, 2014.
17. T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. Mcdonald, "Real-time large-scale dense RGB-D slam with volumetric fusion," *International Journal of Robotics Research*, vol. 34, pp. 598–626, Apr. 2015.
18. R. Or-El, G. Rosman, A. Wetzler, R. Kimmel, and A. M. Bruckstein, "RGBD-fusion: Real-time high precision depth recovery," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5407–5416, 2015.
19. C. Kerl, J. Stueckler, and D. Cremers, "Dense continuous-time tracking and mapping with rolling shutter RGB-D cameras," in *IEEE International Conference on Computer Vision*, 2015.
20. R. A. Newcombe, D. Fox, and S. M. Seitz, "DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 343–352, 2015.
21. Y. Cui, W. Chang, T. Nöll, and D. Stricker, "KinectAvatar: Fully automatic body capture using a single kinect," in *ACCV 2012 International Workshops, Part II*, pp. 133–147, 2012.
22. H. Li, E. Vouga, A. Gudym, L. Luo, J. T. Barron, and G. Gusev, "3d self-portraits," *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 187:1–187:9, 2013.
23. M. Dou, H. Fuchs, and J. Frahm, "Scanning and tracking dynamic objects with commodity depth cameras," in *IEEE International Symposium on Mixed and Augmented Reality*, pp. 99–106, 2013.
24. M. Dou, J. Taylor, H. Fuchs, A. W. Fitzgibbon, and S. Izadi, "3d scanning deformable objects with a single RGBD sensor," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 493–501, 2015.
25. J. Tong, J. Zhou, L. Liu, Z. Pan, and H. Yan, "Scanning 3d full human bodies using kinects," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 4, pp. 643–650, 2012.
26. Y. Chen, G. Dang, Z.-Q. Cheng, and K. Xu, "Fast capture of personalized avatar using two kinects," *Journal of Manufacturing Systems*, vol. 33, no. 1, pp. 233 – 240, 2014.
27. V. S. Lempitsky and D. V. Ivanov, "Seamless mosaicing of image-based texture maps," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–6, 2007.
28. R. Gal, Y. Wexler, E. Ofek, H. Hoppe, and D. Cohen-Or, "Seamless montage for texturing models," *Computer Graphics Forum*, vol. 29, no. 2, pp. 479–486, 2010.
29. R. M. Karp, "Reducibility among combinatorial problems," in *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York.*, pp. 85–103, 1972.
30. K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012.
31. C. Richardt, C. Stoll, N. A. Dodgson, H.-P. Seidel, and C. Theobalt, "Coherent spatiotemporal filtering, upsampling and rendering of RGBZ videos," *Computer Graphics Forum*, vol. 31, no. 2, pp. 247–256, 2012.
32. OpenNI, "OpenNI organization," 2015. http://openni.org/.
33. Z.-Q. Cheng, Y. Chen, R. R. Martin, Y. Lai, and A. Wang, "Supermatching: Feature matching using supersymmetric geometric constraints," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 11, pp. 1885–1894, 2013.
34. K. Amplianitis, M. Adduci, and R. Reulke, "Calibration of a multiple stereo and rgb-d camera system for 3d human tracking," in *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* (I. Colomina and M. Prat, eds.), vol. XL-3/W1, pp. 7–14, 2014.
35. R. W. Sumner, J. Schmid, and M. Pauly, "Embedded deformation for shape manipulation," *ACM Transactions on Graphics*, vol. 26, no. 3, 2007.
36. E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *IEEE International Conference on Computer Vision*, pp. 2564–2571, 2011.
37. Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.
38. J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. X. Kausler, T. Kröger, J. Lellmann, N. Komodakis, B. Savchynskyy, and C. Rother, "A comparative study of modern inference techniques for structured discrete energy minimization problems," *International Journal of Computer Vision*, vol. 115, no. 2, pp. 155–184, 2015.
39. P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 313–318, 2003.