Revisiting Depth Image Fusion with Variational Message Passing

Diego Thomas
Kyushu University, Fukuoka, Japan
thomas@ait.kyushu-u.ac.jp

Ekaterina Sirazitdinova
Konica Minolta / National Institute of Informatics
ekaterina.sirazitdinova@konicaminolta.eu

Akihiro Sugimoto
National Institute of Informatics, Tokyo, Japan
sugimoto@nii.ac.jp

Rin-ichiro Taniguchi
Kyushu University, Fukuoka, Japan
rin@kyudai.jp

Abstract

The running average approach has long been perceived as the best choice for fusing depth measurements captured by a consumer-grade RGB-D camera into a global 3D model. This strategy, however, assumes exact correspondences between points in a 3D model and points in the captured RGB-D images. Such assumption does not hold true in many cases because of errors in motion tracking, noise, occlusions, or inconsistent surface sampling during measurements. Accordingly, reconstructed 3D models suffer unpleasant visual artifacts. In this paper, we visit the depth fusion problem from a probabilistic viewpoint and formulate it as a probabilistic optimization using variational message passing in a Bayesian network. Our formulation enables us to fuse depth images robustly, accurately, and fast for high quality RGB-D keyframe creation, even if exact point correspondences are not always available. Our formulation also allows us to smoothly combine depth and color information for further improvements without increasing computational speed. The quantitative and qualitative comparative evaluation on built keyframes of indoor scenes show that our proposed framework achieves promising results for reconstructing accurate 3D models while using low computational power and being robust against misalignment errors without post-processing.

1. Introduction

For many applications in robotics, computer vision and augmented reality, reconstructing high quality dense 3D models from hand-held RGB-D cameras is a critical step. For example, detailed 3D models are used for path planning in autonomous driving, or to insert virtual contents in a realistic way for mixed reality applications. The process relying on a single RGB-D camera to build a 3D model of a real scene consists of tracking the camera motion and incrementally building the 3D model (this is also called RGB-D SLAM). Over the last decade many efforts have been done to improve the performance of RGB-D SLAM systems.

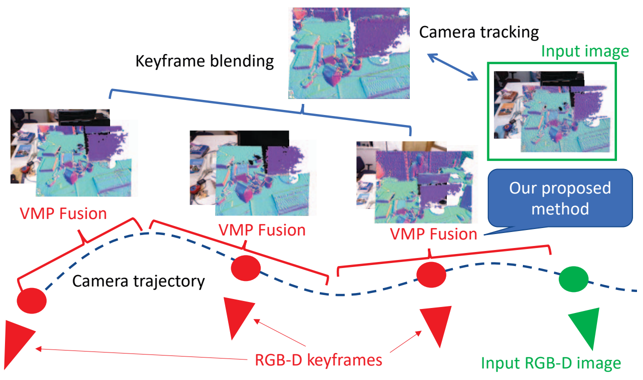

Figure 1. We employ the keyframe-based 3D mapping strategy and fuse input depth measurements into the keyframes in real-time using our proposed VMP Fusion.

On one hand, for accurate camera tracking, visual feature points [15], joint usage of geometric and photometric properties [14], and loop-closure algorithms [26, 23] have been proposed. On the other hand, many efforts have been done to design efficient global model representations of fused depth images. For example, surfels [17], Gaussian Mixture Models [3], implicit functions [8], and parametric surfaces [22] were proposed. Several techniques were also introduced to improve visual appearance of the built 3D model by more accurate color mapping [4, 25, 30]. However, the depth fusion itself (i.e., integration of multiple depth measurements into a single high resolution 3D map) has been less studied and the basic technique called the running average is still used as state-of-the-art.

The largely adopted strategy for depth image fusion relies on integration of depth measurements into a volumetric Truncated Signed Distance Function (TSDF) [2] using the running average. However, some limitations arise when using this technique: (1) the TSDF requires a trade-off be-

2475-7888/19/$31.00 ©2019 IEEE
DOI 10.1109/3DV.2019.00044

tween the resolution and the memory consumption; (2) visual blur artifacts appear in reconstructed 3D models because of inevitable errors and drifts in estimated camera poses and depth image distortion. In addition, the TSDF representation accounts only for geometry.

The keyframe-based 3D mapping [11, 13, 14], however, allows to render a synthetic RGB-D image at any viewpoint by blending nearby keyframes and, moreover, to freely adjust the resolution of the synthesized RGB-D image by changing the virtual camera parameters. Keyframe resolution is in general much higher than that of a volumetric TSDF, and, thus, the keyframe-based 3D mapping produces 3D models of higher quality. In addition, it is easy to increase a keyframe resolution if needed (with low memory usage counterpart), and to adapt to camera path corrections (after loop closure, for example). However, to create high quality RGB-D keyframes, the running average in the image plane is used, which results in visual artifacts such as blurs or double edges in the built keyframe.

In order to keep the advantage of using keyframes for 3D mapping, we take a new theory-supported probabilistic approach to depth fusion. Namely, we formulate the depth fusion problem using a Bayesian network, and propose to use the Variational Message Passing (VMP) scheme [24] for fusion. The main idea is to represent each depth value in a depth image as a probabilistic function centered at the current estimate and with variance decreasing over time. At each integration of a newly coming RGB-D image, the probabilistic function is updated by looking at consistency in both depth and color at possible projection locations (depth and color information are naturally combined in the fusion process). By using the VMP, the variational distribution of the depth values at pixels is optimally updated using neighborhood information, reducing the number of outliers and noise while keeping sharp details. As a result, robust and accurate 3D reconstruction is achieved even if exact point correspondences are not available. Fig. 1 shows an overview of our proposed keyframe-based 3D mapping.

2. Related Work

The development of RGB-D cameras has motivated a lot of research on real-time 3D mapping. Below, we review different 3D representations used in RGB-D SLAM systems, and the corresponding RGB-D image fusion strategies.

Early work on RGB-D SLAM employed surfels [17] to build a dense 3D model from a sequence of captured RGB-D images [7, 10]. Surfels are small surface elements oriented with the surface normal and with size varying depending on the sampling resolution. 3D models are built by incrementally updating center and orientation of each surfel using the running average. Additional visibility constraint is often used to reduce outliers. The 3D representation using surfels allows to obtain visually appealing results and is still often used in dense 3D mapping systems [28]. The main drawback of using surfels is in the fact that in this technique the data is not organized, so it is difficult to modify the 3D model at run time, or to maintain the model at large scale.

Izadi *et al*. [8] proposed KinectFusion, a method that successfully employs the volumetric TSDF for dense 3D modeling. Their idea is to represent a 3D surface with an implicit function embedded in a 3D space. Namely, the TSDF takes positive values outside the surface, and negative values inside. Once the TSDF is computed, the surface can be extracted at the zero crossing. By discretizing the 3D space into voxels and computing the TSDF at each voxel, the 3D surface is recovered using marching cubes or ray tracing. Since KinectFusion development, the volumetric TSDF has been largely used in RGB-D SLAM systems [6, 9, 19, 27, 31], and follow-up research has been reported using efficient discretizations such as octrees [1, 29] or hash-tables [12, 16]. However, in all these extensions, the TSDF is always built using the running average. It can be explained by the fact that running average can be implemented on GPU and, therefore, has real-time performance. When the camera pose is wrongly estimated, however, the values from different points on the surface are averaged, leading to blurs and "ghost effects".

Rajput *et al*. [18] proposed to improve the depth fusion strategy by using local information. They proposed a recursive Total Variation (TV) algorithm that works directly in the 3D volume of the TSDF field. However, only support from voxels in the ray direction is used for the regularization. Therefore, information from neighbors on the actual surface is limited and the regularization works more as a smoothing filter than a way to enhance details. In contrast, we propose to use the connectivity of pixels in a key-frame to obtain richer information about the local geometry of the scene.

In this work, we introduce a robust probabilistic framework for depth fusion into keyframes that reduces noise in depth measurements while preserving sharp features. By passing messages in a Bayesian network, our proposed method allows to use neighbouring information as well as depth and color consistency when fusing depth measurements.

3. Proposed method

We employ the keyframe-based strategy for dense 3D mapping using an RGB-D camera. At run-time, keyframes are blended to create a reference RGB-D image that is used to compute the current camera pose. New keyframes are then added depending on the camera motion. With every new input RGB-D image, the closest keyframe is updated using the VMP [24] framework where the depth at each pixel is represented as a probability distribution. The probability distribution of each pixel in the closest keyframe's depth image is incrementally updated with each input RGB-

D image. This is done by passing messages through a network of nodes connecting the pixels in the keyframe with those in the input RGB-D image. These messages represent the consistency of the pixel's RGB-D values with respect to the current RGB-D observations and the estimated camera pose (the color of the pixels in each keyframe is fixed).

Compared to the running average, our proposed method has the advantage that it does not require exact point correspondences. Our proposed method is distinguished in its usage of neighbourhood information and in its ability to smoothly combine geometric and color information to improve robustness while keeping computations efficient.

3.1. Probabilistic depth mapping

We use a probabilistic representation of a keyframe's depth image, which allows multi-modal fusion of RGB-D data (*e.g.*, combining color, geometry, and local neighbourhood), without relying on perfect point correspondences.

We denote by D the keyframe's depth image, which records in each pixel the depth value from the camera optical center to the measured surface. We associate pixel i in D with a random variable H_i that follows the Gaussian distribution with mean μ_i and precision (*i.e.*, inverse of variance) σ_i (we represent 2D pixels with a 1D index to ease notations). Then, the log conditional probability of H_i is

$$\ln P_i(H_i|\mu_i,\sigma_i) = \phi_i(\mu_i,\sigma_i)^\top \mathbf{u}(H_i) + \ln \sigma_i - \frac{\sigma_i \mu_i^2}{2} - \ln\sqrt{2\pi}, \quad (1)$$

where $\phi_i(\mu_i,\sigma_i) = \begin{bmatrix}\sigma_i\mu_i & -\frac{\sigma_i}{2}\end{bmatrix}^\top$ is the natural parameter vector, and $\mathbf{u}(H_i) = \begin{bmatrix}H_i & H_i^2\end{bmatrix}^\top$ is the natural statistic vector. At any time, the current depth value estimate $\widehat{\mu_i}$ is obtained as the expectation of H_i at that time.

Given the 4×4 pose and intrinsic matrices T and K, we compute the corresponding 3D point as:

$$\mathbf{vtx}_i = TK^{-1}(\widehat{\mu_i}u_i, \widehat{\mu_i}v_i, \widehat{\mu_i}, 1)^\top, \quad (2)$$

where $\mathbf{vtx}_i$ are the homogeneous coordinates of the 3D point that corresponds to pixel $i = (u_i, v_i)$ (with dept value $\widehat{\mu_i}$) in the world coordinate system. Then, our goal is to optimize the mean estimate $\widehat{\mu_i}$ and the precision estimate $\widehat{\sigma_i}$ for all pixels given a live stream of RGB-D measurements. In section 4.2, we detail our optimization procedure using VMP. Note that equation (2) can also be written as:

$$\mathbf{vtx}_i = T(\mathbf{v_{ref}} + \widehat{\mu_i}\mathbf{Z}_i), \quad (3)$$

where $\mathbf{v_{ref}} = (0,0,0,1)^\top$ and $\mathbf{Z}_i = K^{-1}(u_i, v_i, 1, 0)^\top$.

3.2. Motion tracking

To create accurate dense 3D models, reliable camera motion tracking is required. We use the built keyframes and the point-plane ICP [20] formulated in the Lie algebra to align successive input RGB-D images and track the camera's motion. Aligning the input RGB-D image to the closest keyframe is not effective because the keyframes are few and there may exist significant differences between the closest keyframe and the input RGB-D image (such as occlusions and different surface sampling resolution). Therefore, it is better to blend several keyframes to create a current synthetic RGB-D image from the last estimated camera pose, which is used to align the input RGB-D image.

We create the synthetic RGB-D image by rasterizing (*i.e.*, filling the triangles of) the projections of the k-nearest keyframes (we used $k = 3$ in our experiments)[1] into the synthetic image plane. We use the Z-buffer algorithm to handle occlusions. Fig. 1 shows an example of a synthetic RGB-D image created from three keyframes.

4. Fusion using variational message passing

We formulate the fusion of depth measurements using the VMP [24] that takes into account local information, color, and geometry without relying on exact point correspondences. Our contributions here are three-fold:

- We formulate the data fusion process in a probabilistic framework.
- We derive the message passing algorithm for our specific problem.
- We propose a graphical model for the optimization by using the confidence range, which allows real-time optimization.

4.1. Probabilistic formulation on a graphical model

VMP [24] is an algorithm that optimizes a factorized variational distribution using a message passing procedure on a graphical model. Similar to belief propagation, VMP proceeds by sending messages between nodes in the network and updating posterior beliefs using local operations at each node. We refer the reader to [24] for more details.

Variational inference. Below we briefly review variational inference with a particular focus on our case. Our variational model is composed of random variables denoted by $X = (V, H)$ where V are the visible variables (each variable corresponds to a depth measurement) and H are the hidden variables (each variable corresponds to a depth value in the keyframe). Then by using the Bayesian network shown in Fig. 2 (where each node corresponds to a variable), the joint distribution $P(X)$ can be expressed in terms of the conditional distribution of each node:

$$P(X) = \prod_i P(X_i \mid pa_i), \quad (4)$$

[1]A larger value of k may increase completeness and accuracy of the synthetic image, but at the cost of additional computational time. We experimentally found that the best trade-off is obtained with $k = 3$.

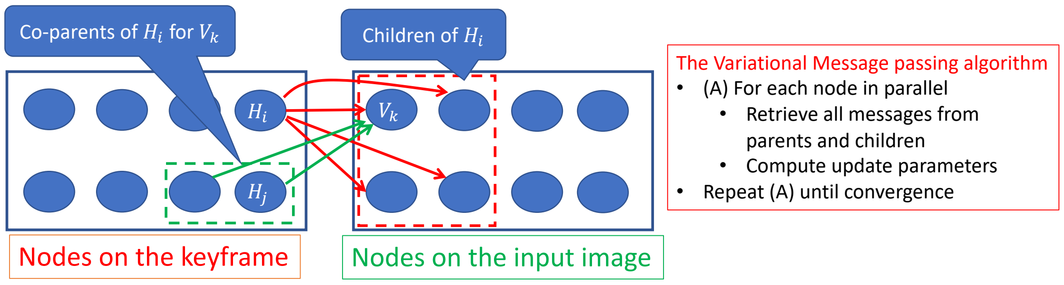
Co-parents of H_i for V_k
Children of H_i
H_i
H_j
V_k
The Variational Message passing algorithm
• (A) For each node in parallel
• Retrieve all messages from parents and children
• Compute update parameters
• Repeat (A) until convergence
Nodes on the keyframe
Nodes on the input image

Figure 2. Variational message passing algorithm. Nodes H_i and H_j correspond to the keyframe RGB-D image. Node V_k corresponds to the input RGB-D image. The set of children and co-parents of a node H_i form the Markov blanket of the node H_i.

where pa_i denotes the set of variables corresponding to the parents of node i, and X_i denotes the variable (or the group of variables) associated with node i.

Our goal is to estimate H that maximizes $P(H|V)$. Directly inferring $P(H|V)$ is, however, hard and in general not tractable. Therefore, variational inference finds a simpler variational distribution $Q(H)$ that closely approximates the true variational distribution $P(H|V)$ by minimizing the Kulback-Leiber divergence between $P(H|V)$ and $Q(H)$ under the condition that Q has a factorized form: $Q(H) = \prod_i Q_i(H_i)$. Then, the optimization becomes tractable. As demonstrated in [24], the optimized form Q_i^* of the i^{th} factor is given by:

$$\ln Q_i^*(H_i) = < \ln P(H \mid V) >_{\sim Q_i(H_i)} + const, \qquad (5)$$

where $< \cdot >_{\sim Q_i(d_i)}$ denotes the expectation with respect to all factors except for $Q_i(H_i)$.

Since the variational distribution $Q_i(H_i)$ depends on the expectations over variables in the Markov blanket of the node H_i only[2], we can write:

$$\ln Q_i^*(H_i) = < \ln P(H_i|\nu_i, \sigma_i) > \\ + \sum_{k \in ch_i} < \ln P(V_k|cp_k^i) > + \, const,$$

where ch_i is the set of children of node H_i and cp_k^i is the set of co-parents of node H_i for V_k over the graphical model. The optimization of Q_i can therefore be expressed as a local computation at the node H_i involving neighbouring (*i.e.* parent and child) nodes in the graph. Note that the equations for all the factors are coupled since the solution for each $Q_i(H_i)$ depends on expectations with respect to the other factors $Q_{j \neq i}$. The variational optimization proceeds by initializing each of $Q_i(H_i)$ and then cycling through each factor in turn by replacing the current distribution with a revised one.

Graphical model construction. We create one node H_i for each pixel i in the keyframe's depth image, and one node V_k for each pixel k in the input depth image[3]. To build the graph we connect each node in H with all their children nodes in V that could be a measurement of that (surface) point (thus creating directed edges). At each new input frame, these edges are recomputed depending on the current estimates of the mean and the precision of the random variable for each node (pixel) in the keyframe's depth image. As the 3D model becomes better, we restrict the connections only to confident pixels, which reduces the number of edges and speeds up the message passing procedure. Note that the edges are newly created for each input RGB-D image. Fig. 3 illustrates the process of creating the graph.

In every new input RGB-D image, we select the confident pixels as those in a confidence bounding box that are close enough to the corresponding point in the keyframe and that are roughly oriented in the same direction (a threshold of 3 cm and 40 degrees in our experiments). The confidence bounding box for the i^{th} node H_i is defined by a projected confidence range segment that is the 2D projection of the 3D segment r_i in the input depth image given the keyframe and current camera poses (as shown in Fig. 3). We compute r_i as $r_i = [\mathbf{vtx}_i - \frac{0.3}{\sqrt{-2\sigma_i}}\mathbf{Z}_i, \mathbf{vtx}_i + \frac{0.3}{\sqrt{-2\sigma_i}}\mathbf{Z}_i]$, where $\mathbf{vtx}_i$ is the 3D point on the keyframe computed using equation (3) for the i^{th} pixel.

4.2. Message passing procedure

In the original VMP algorithm, messages are passed until convergence. We, however, execute only a single message passing procedure for each input image, because (1) the convergence after reading multiple image measurements is preferable to avoid overfitting to a single noisy input image, and (2) the computational time is reduced, converging faster to a global solution. Note that at each iteration, only the current RGB-D image is used to update the messages. This is an approximation of the global solution.

We now write the functional form of the conditional probability functions, which defines the exact form of the messages to be sent through the network. For the VMP algorithm to work efficiently, all conditional probability distributions have to be in the exponential family and conju-

[2]The Markov blanket of a node is defined as the set of parents, children and co-parents of that node.

[3]We use H and V to denote both variables and nodes.

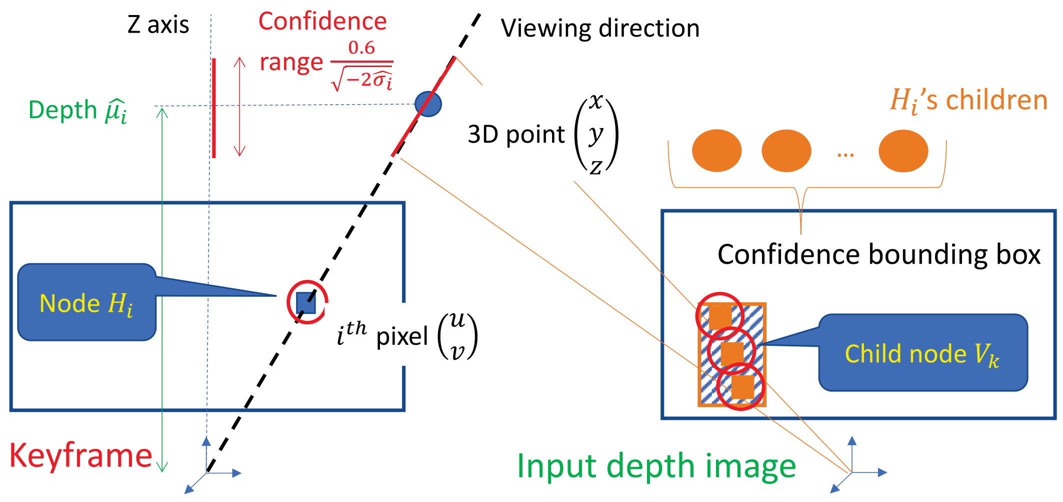
Z axis
Confidence range $\frac{0.6}{\sqrt{-2\hat{\sigma}_i}}$
Viewing direction
Depth $\hat{\mu}_i$
3D point $\begin{pmatrix}x\\y\\z\end{pmatrix}$
H_i's children
...
Confidence bounding box
Node H_i
i^{th} pixel $\begin{pmatrix}u\\v\end{pmatrix}$
Child node V_k
Keyframe
Input depth image

Figure 3. We build the graph that connects nodes on the keyframe's depth image with nodes on the input depth image by projecting confidence segments onto the input depth image. Given the current estimate of the mean and the precision values of each pixel in the keyframe's depth image, a 3D segment is identified for each node in the depth image. For each node, the confidence segment is projected onto the input depth image, which defines the confidence bounding box, from which the children are selected.

gate[4] with respect to the distributions over the parent variables (called a conjugate-exponential model).

For each node V_k that is a child of H_i, the log-conditional probability of V_k has to be written in the same form as in equation (1):

$$\ln P(V_k|H_i, cp_k^i) = \phi_{V_k,H_i}(V_k, cp_k^i)^\top \mathbf{u}(H_i) + \lambda_i(V_k, cp_k^i),$$

where ϕ_{V_k,H_i} is the natural parameter vector of V_k with respect to H_i, cp_k^i is the set of co-parents of H_i with respect to V_k (*i.e.*, all parents of V_k except for H_i) and λ_i is a normalization function. ϕ_{V_k,H_i} and λ_i are defined below.

We aim to relax the assumption of perfect point correspondences availability. To do so, we have to explicitly account for the fact that a parent H_i (in the keyframe's depth image) of a node V_k (in the input depth image) may not correspond to the measurement of the same point, and that the 3D points corresponding to V_k and H_i may be completely different. Nevertheless, if one of the parents of V_k corresponds to the measurement of the same point, then we make $\ln P(V_k|H_i, cp_k^i)$ close to 0. One possible solution for that is to use products and minimum of distances.

We write the log-conditional probability of V_k as:

$$\begin{aligned}\ln P(V_k|H_i, cp_i) = \frac{-\sigma_D}{2}(||\mathbf{vtx}_i-\mathbf{p}_k||^2+\alpha||\mathbf{C}_i-\mathbf{I}_k||^2)\\ \times \min_{j\in cp_k^i}(||\mathbf{vtx}_j-\mathbf{p}_k||^2+\alpha||\mathbf{C}_j-\mathbf{I}_k||^2)\\ + \ln\sigma_D - \ln\sqrt{2\pi},\end{aligned} \quad (6)$$

where σ_D is the precision of the depth sensor and $\mathbf{p}_k$ is the 3D point corresponding to the pixel k in the depth image (and $\alpha = 10$ in our experiments). $\mathbf{C}$ and $\mathbf{I}$ are the keyframe's and input color images (respectively). Note that we combine geometric and color distances in the conditional probability of V_k. As a consequence, when the messages are passed, consistency in both color and geometry is considered, which is smoother compared to using a hard threshold to reject correspondences.

Defining

$$\begin{aligned}\phi_{V_k,H_i}(V_k, cp_k^i)[0] = \sigma_D(T\mathbf{Z}_i(T\mathbf{v}_{\mathbf{ref}\,i}-\mathbf{p}_k))\\ \times \min_{j\in cp_k^i}(||\mathbf{vtx}_j-\mathbf{p}_k||^2+\alpha||\mathbf{C}_j-\mathbf{I}_k||^2)\end{aligned}$$

and

$$\begin{aligned}\phi_{V_k,H_i}(V_k, cp_k^i)[1] = -\frac{\sigma_D}{2}||T\mathbf{Z}_i||^2\\ \times \min_{j\in cp_k^i}(||\mathbf{vtx}_j-\mathbf{p}_k||^2+\alpha||\mathbf{C}_j-\mathbf{I}_k||^2)\end{aligned}$$

allows us to write equation (6) with $\phi_{V_k,H_i}(V_k, cp_k^i) = \left[\phi_{V_k,H_i}(V_k, cp_k^i)[0] \quad \phi_{V_k,H_i}(V_k, cp_k^i)[1]\right]^\top$ for each node H_i that is a parent of node V_k, where $\mathbf{v}_{\mathbf{ref}\,i}$ and $\mathbf{Z}_i$ are defined in equation (3). We also define

$$\begin{aligned}\lambda_i(V_k, cp_k^i) = -\frac{\sigma_D}{2}(||T\mathbf{v}_{\mathbf{ref}\,i}-\mathbf{p}_k||^2+\alpha||\mathbf{C}_i-\mathbf{I}_k||^2)\\ \times \min_{j\in cp_k^i}(||\mathbf{vtx}_j-\mathbf{p}_k||^2+\alpha||\mathbf{C}_j-\mathbf{I}_k||^2)+\ln\sigma_D-\ln\sqrt{2\pi}.\end{aligned}$$

Note that $||\mathbf{vtx}_i - \mathbf{p}_k||^2 = ||T(\mathbf{v}_{\mathbf{ref}\,i} + d_i\mathbf{Z}_i) - \mathbf{p}_k||^2$.

Now we can write the exact form of the messages to be sent. The message from a parent node H_i to a child node V_k is the expectation of the natural statistic vector:

$$m_{H_i->V_k} = <\mathbf{u}(H_i)>.$$

[4] A parent distribution $P(X|Y)$ is said to be conjugate to a child distribution $P(W|X)$ if $P(X|Y)$ has the same functional form, with respect to X, as $P(W|X)$.

The message from a child node V_k to a parent node H_i is

$$m_{V_k->H_i} \quad = \quad \phi_{V_k,H_i}(<\mathbf{u}(V_k)>, \{m_{H_j->V_k}\}_{j\in cp_k^i}).$$

We can simply use the measured data as the expectations for the nodes from the input depth image. The expectation of the natural statistic vector for the nodes in the keyframe's depth image can be computed by re-parameterizing (1) with respect to $\phi_i(\mu_i, \sigma_i)$, which gives:

$$\ln P(H_i|\phi_i(\mu_i,\sigma_i)) = \phi_i(\mu_i,\sigma_i)^\top \mathbf{u}(H_i) + \tilde{g}(\phi_i(\mu_i,\sigma_i)),$$

where

$$\begin{aligned}\tilde{g}(\phi_i(\mu_i,\sigma_i)) = \ln(-2\phi_i(\mu_i,\sigma_i))[1] \\ +\frac{\phi_i(\mu_i,\sigma_i))[0]^2}{4\phi_i(\mu_i,\sigma_i))[1]} - \ln\sqrt{2\pi}.\end{aligned}$$

Then,

$$< \mathbf{u}(H_i) >= -\frac{\mathrm{d}\tilde{g}(\phi_\mathrm{i}(\mu_\mathrm{i},\sigma_\mathrm{i})))}{\mathrm{d}\phi_\mathrm{i}(\mu_\mathrm{i},\sigma_\mathrm{i}))} = \begin{bmatrix}\mu_i \\ \mu_i^2 + \frac{1}{\sigma_i}\end{bmatrix}.$$

At every iteration, we update the expectations of the statistic vectors of all the nodes and re-compute all messages between the connected nodes in the graph.

Once all messages from all children are computed, the parameters of the variational distribution of the nodes H_i in the keyframe's depth image are updated as follows:

$$\phi_i^t(\mu_i,\sigma_i) = \phi_i^{t-1}(\mu_i,\sigma_i) + \sum_{k\in ch_i} m_{V_k->H_i},$$

where t is the current iteration index. ϕ_i^0 is the prior variational distribution of H_i initialized as $\phi_i^0 = \begin{bmatrix}0 & -200.0\end{bmatrix}$. With the updated natural parameter vector ϕ_i^t, the current mean and precision values for the keyframe's node are updated. Then, a bilateral smoothing filter is applied to the obtained depth image[5].

In VMP, messages can be passed in any order, which is well-suited for efficient GPU implementation. In our case, we allocated one thread for each node in the keyframe's depth image. In each thread, for each child we compute the messages from the co-parents and then send all messages from the children to the node corresponding to the thread.

5. Experimental results

We show the performance of our proposed method with synthetic and real data captured with a Kinect V1, which provides RGB-D images at 30 fps at 640×480 pixels. We used keyframes with the resolution of either 640×480 or 320×240 pixels. We compared our proposed depth fusion method (denoted by *VMPFusion*(resolution)) with the method using the running average on keyframes [13] (denoted by *Average*), the method using the running median on keyframes (denoted by *Median*) and the method using the running average on a TSDF volume built in [8] with voxel size of 0.006 meters for synthetic data, and of 0.01 or 0.005 meters for real data (called *KinFu*(voxel size)). *Median* is our own modification of [13] where the current depth value for each pixel is computed as the median of the 100 best depth measurements. If there are already more than 100 measurements attached to a pixel, the farthest from the median is removed and the new measurement is inserted. Note that most extensions of KinectFusion [8] improve robustness of tracking and scalability, but the fusion technique of [8] is still state-of-the-art for small static scenes.

We implemented our proposed method in Swift with some parts running on the GPU (with metal shaders). We run the code on an Apple iMac Pro with a 2.5 GHz Intel Xeon W CPU and a Radeon Pro Vega 64 16GB GPU. Our method runs on average with about 25 fps for the resolution of 640×480 and 30 fps for 320×240. We observed that those frame rates drop to 4 (640×480) and 15 (320×240) when we do not use the confidence range[6].

5.1. Quantitative evaluation

We quantitatively evaluated the quality of produced depth images of some selected keyframes using the ICL-NUIM synthetic benchmark dataset [5]. The dataset consists of two synthetic scenes: a living room (called *lr*) and an office room (called *of*), and four RGB-D image sequences for each of these scenes. We used the sequences with synthetic sensor noise added. We selected one RGB-D image sequence from each scene (those with smaller number of RGB-D frames), and evaluated the accuracy of some selected keyframes for the methods mentioned above. For each sequence, we used the ground truth camera trajectory provided in the dataset to compute errors.

Figure 4 shows the error maps for the first keyframe generated by all methods, while Table 1 shows the average and the standard deviation (in mm) of the errors in generated depth images. For *KinFu*, we built the whole TSDF volume using the whole RGB-D image sequence, with a voxel size of 0.006 m (the size of the voxel grid was $1000 \times 1000 \times 1000$). We optimized the voxel size so that all points in the first keyframes (used for our evaluation) are inside the volume, while the size of the volumetric data is minimized on the GPU. We then rendered the TSDF volume using ray tracing at the camera pose for the keyframe to generate the estimated keyframe.

Because we used the ground truth camera poses, this is actually the ideal case for *Average*, *Median*, and *KinFu*, which rely on perfect point correspondences. Nevertheless, Table 1 shows that our proposed method is able to generate

[5]In our experiments we used a filter of the size of 3 pixels and with standard deviation of 1cm.

[6]The code is available at http://limu.ait.kyushu-u.ac.jp/e/member/member0042.html

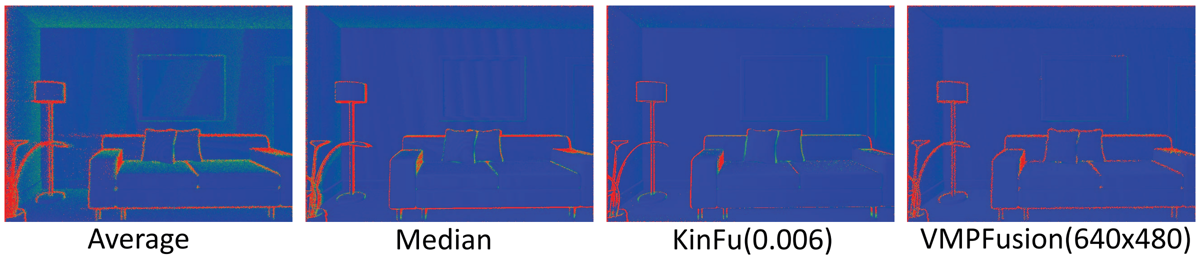

Figure 4. Error maps obtained by comparing depth images of the first frame of the ICL-NUIM synthetic dataset "Lounge" with the first keyframe obtained with *Average*, *Median*, *KinFu*, and *VMPFusion* (better seen in color).

Table 1. Quantitative evaluation of our proposed method (*VMPFusion* against *Average*, *Median*, and *KinFu* on the ICL-NUIM dataset. The scores are the means and standard deviations of the errors are mm (the lower the better).

	lrkt2 (KF$_1$)	lrkt2 (KF$_4$)	lrkt2 (KF$_6$)	ofkt2 (KF$_1$)	ofkt2 (KF$_7$)	ofkt2 (KF$_{15}$)
Average	40 ± 0.08	42 ± 0.07	41 ± 0.07	67 ± 0.08	22 ± 0.07	16 ± 0.06
Median	24 ± 0.07	28 ± 0.07	33 ± 0.08	36 ± 0.08	15 ± 0.06	16 ± 0.06
KinFu(0.006)	$\mathbf{20 \pm 0.06}$	$\mathbf{20 \pm 0.05}$	$\mathbf{19 \pm 0.05}$	32 ± 0.08	20 ± 0.05	22 ± 0.06
VMPFusion(640×480)	23 ± 0.07	24 ± 0.06	20 ± 0.06	$\mathbf{31 \pm 0.06}$	$\mathbf{13 \pm 0.06}$	$\mathbf{15 \pm 0.06}$

accurate results, sometimes even better than *KinFu*. Fig. 4 shows that *VMPFusion* reconstructs smooth surfaces while keeping sharp edges, thanks to combining depth, color, and neighbouring information in the fusion process.

Although the code of the TV method proposed in [18] is not publicly available, the authors report in the paper accuracy of about 103 to 200 mm, while our proposed method obtained results of about 20 to 30 mm. Moreover, the method in [18] runs at about 5 fps while our proposed method runs at more than 25 fps.

5.2. Qualitative evaluation

We evaluated the proposed method qualitatively on the public TUM dataset [21] and our own captured data obtained with a Kinect V1. For each sequence, we selected some keyframes and evaluated the visual quality of the produced normal image. Note that we do not show the results obtained with *Average* because *Average* is clearly inferior to *Median* as seen in Table 1.

Figure 5 shows some selected results. *VMPFusion* proves to be able to produce accurate and sharp keyframes. In particular, the keyboard in the first row illustrates that *VMPFusion* shows more details than *KinFu*. Moreover, as we can see in the last row, *VMPFusion* is more robust to errors in camera pose estimation. When the camera pose is wrongly estimated, *Median* and *KinFu* fuse depth measurements from wrong correspondences, which leads to "ghost" effects (second column in the last row) and blurs (third column in last row). In this case, we superimposed the color and normal images and we observe significant deformations between the color and depth images of the keyframes generated by *Median* and *KinFu*, while there was almost no deformation between the color and depth images generated with *VMPFusion*.

We evaluated the effect of changing the resolution of the volumetric grid and the keyframe's depth images. For RGB-D SLAM applications in mobile devices, or at large scale, finding a good compromise between reconstruction quality, memory consumption, and computational time is of utmost importance. The most straightforward and the most important parameter to change when searching for a good compromise is the resolution of the 3D model (*i.e.*, voxels or pixels size). As seen on Fig. 5, using a larger voxel size leads to a blurred 3D reconstruction, because 3D points on the surface are obtained by linear interpolation of the voxels summit. On the contrary, we observe only few losses in the details of the keyframes obtained with *VMPFusion*, due to the fact that the depth value of each pixel is directly optimized with the raw input measurements.

We also conducted an ablation study of *VMPFusion* by modifying α in equation (6) that controls the importance of color compared to the depth measurement during the message passing. This study evaluates the advantage of combining both geometric and color information to improve the quality of depth fusion. Fig. 6 shows that increasing the value of α allows to preserve sharper edges and more details (see the red rectangles). However, as we can see in the green circle, giving to much importance to the color consistency also creates some unpleasant artifacts because of blurs in the color image and misalignment between depth and color images (the depth and color cameras are not perfectly synchronized). Note that the results obtained with *VMPFusion* were not as complete as those obtained with *KinFu* because we do not add new points in the key-frame. This limitation will be addressed in future work.

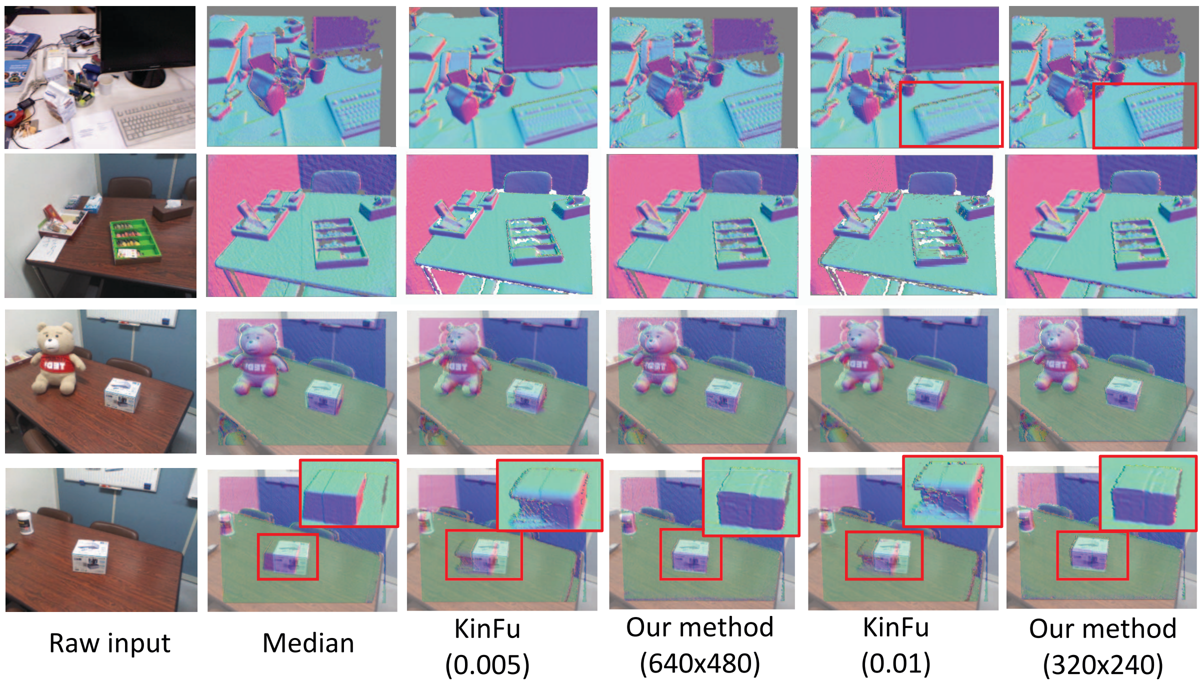

Figure 5. Results obtained on the TUM dataset (xyz) when using *Median*, *KinFu* and *VMPFusion* (better seen in color). The last row shows the case where the camera tracking seemed to fail.

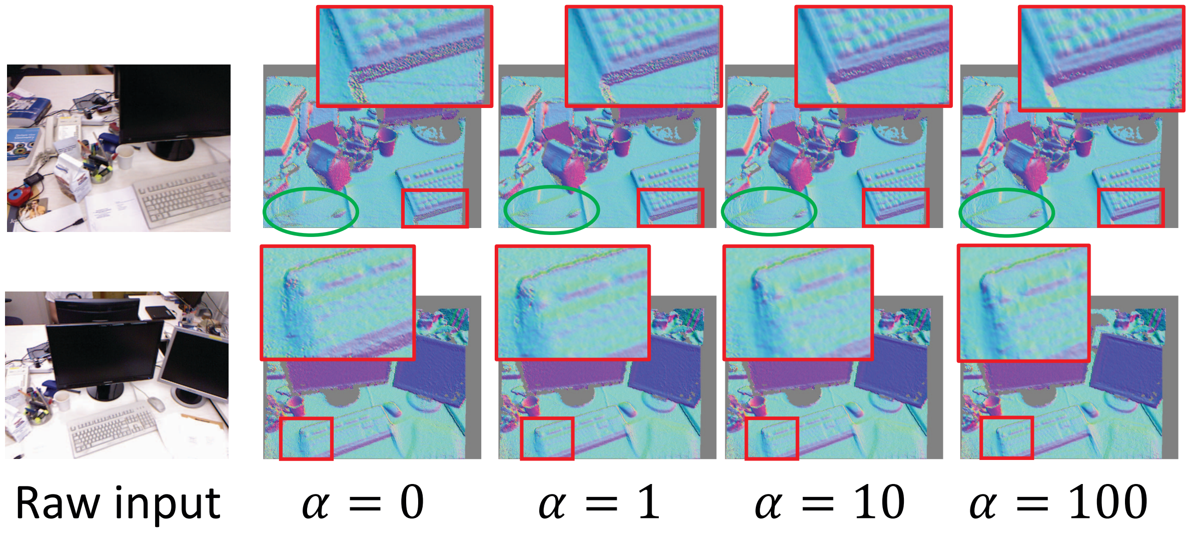

Figure 6. Results obtained on our own dataset using different α's for *VMPFusion*(640×480) (better seen in color).

6. Conclusion

We formulated depth fusion for keyframe-based 3D mapping in the probabilistic framework. Using the variational message passing in the graphical representation of the depth image, we integrate the depth values at each pixel by taking into account its neighborhood information to maintain robustness and accuracy. Our proposed probabilistic fusion framework does not rely on exact point correspondences, but allows us to combine both depth and color. This cannot be done using the TSDF representation with the running average fusion. Moreover, GPU timings of our proposed depth fusion allow real-time applications. Our experimental results show the superior quality of the reconstructed 3D models compared to the running average strategy and the state-of-the-art volumetric TSDF fusion.

Acknowledgement

A part of this work was supported by JSPS KAKENHI Grant Number JP19K20297. E. Sirazitdinova was supported by the NII international internship program during her stay at NII.

References

[1] J. Chen, D. Bautembach, and S. Izadi. Scalable real-time volumetric surface reconstruction. *ACM Trans. Graph.*, 32(4):113:1–113:16, July 2013. 2

[2] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM. 1

[3] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz. Accelerated generative models for 3d point cloud data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5497–5505, 2016. 1

[4] Y. Fu, Q. Yan, L. Yang, J. Liao, and C. Xiao. Texture mapping for 3d reconstruction with rgb-d sensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4645–4653, 2018. 1

[5] A. Handa, T. Whelan, J. McDonald, and A. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China, May 2014. 6

[6] P. Henry, D. Fox, A. Bhowmik, and R. Mongia. Patch volumes: Segmentation-based consistent mapping with rgb-d cameras. In *3D Vision - 3DV 2013, 2013 International Conference on*, pages 398–405, June 2013. 2

[7] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663, 2012. 2

[8] 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*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM. 1, 2, 6

[9] O. Kähler, V. A. Prisacariu, and D. W. Murray. Real-time large-scale dense 3d reconstruction with loop closure. In *ECCV 2016*, pages 500–516, 2016. 2

[10] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision - 3DV 2013*, pages 1–8, June 2013. 2

[11] C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2100–2106. IEEE, 2013. 2

[12] J. Li, W. Gao, H. Li, F. Tang, and Y. Wu. Robust and efficient cpu-based rgb-d scene reconstruction. *Sensors*, 18(11):3652, 2018. 2

[13] M. Meilland and A. I. Comport. On unifying key-frame and voxel-based dense visual slam at large scales. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3677–3683. IEEE, 2013. 2, 6

[14] M. Meilland and A. I. Comport. Super-resolution 3d tracking and mapping. In *2013 IEEE International Conference on Robotics and Automation*, pages 5717–5723. IEEE, 2013. 1, 2

[15] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 1

[16] M. NieBner, M. Zollhofer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Trans. Graph.*, 32(6):169:1–169:11, Nov. 2013. 2

[17] H. Pfister, M. Zwicker, J. Van Baar, and M. Gross. Surfels: Surface elements as rendering primitives. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 335–342. ACM Press/Addison-Wesley Publishing Co., 2000. 1, 2

[18] M. Rajput, E. Funk, A. Börner, and O. Hellwich. Recursive total variation filtering based 3d fusion. In *SIGMAP*, pages 72–80, 2016. 2, 7

[19] H. Roth and M. Vona. Moving volume kinectfusion. *British Machine Vision Conference (BMVC)*, 2012. 2

[20] A. Segal, D. Haehnel, and S. Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435, 2009. 3

[21] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 7

[22] D. Thomas and A. Sugimoto. A flexible scene representation for 3d reconstruction using an rgb-d camera. In *2013 IEEE International Conference on Computer Vision*, pages 2800–2807, Dec 2013. 1

[23] D. Thomas and A. Sugimoto. A two-stage strategy for real-time dense 3d reconstruction of large-scale scenes. In L. Agapito, M. M. Bronstein, and C. Rother, editors, *ECCV 2014 Workshops*, pages 428–442. Springer International Publishing, 2015. 1

[24] C. B. W. John. Variational message passing. *Journal of Machine Learning Research*, pages 661–694, 2005. 2, 3, 4

[25] M. Waechter, N. Moehrle, and M. Goesele. Let there be color! large-scale texturing of 3d reconstructions. In *European Conference on Computer Vision*, pages 836–850. Springer, 2014. 1

[26] T. Weise, T. Wismer, B. Leibe, and L. V. Gool. In-hand scanning with online loop closure. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1630–1637, Sept 2009. 1

[27] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald. Kintinuous: Spatially extended kinectfusion. In *3rd RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2012. 2

[28] M. X, G. W, and H. Z. Dense rgb-d slam with multiple cameras. *Sensors*, 18(7), 2018. 2

[29] M. Zeng, F. Zhao, J. Zheng, and X. Liu. Octree-based fusion for realtime 3d reconstruction. *Graph. Models*, 75(3):126–136, May 2013. 2

[30] Q.-Y. Zhou and V. Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. *ACM Transactions on Graphics (TOG)*, 33(4):155, 2014. 1

[31] Q.-Y. Zhou, S. Miller, and V. Koltun. Elastic fragments for dense scene reconstruction. In *Computer Vision (ICCV),*

2013 IEEE International Conference on, pages 473–480, Dec 2013. 2