Confidence-based Refinement of Corrupted Depth Maps

Satoshi Ikehata* and Kiyoharu Aizawa*
* University of Tokyo, Tokyo, Japan
E-mail: {ikehata, aizawa} @hal.t.u-tokyo.ac.jp

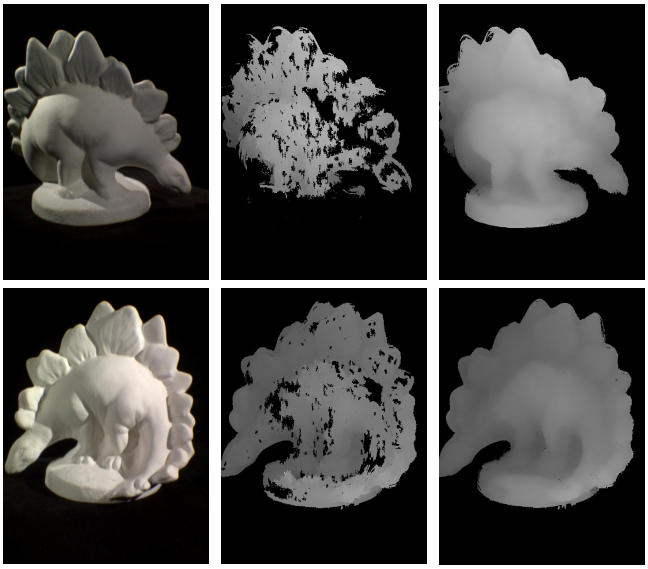

Fig. 1. The depth map refinement results by our method, where (a) Input images, (b) Initially acquired depth maps (by match propagation [11]) and (c) Depth maps refined by our method.

***Abstract*—This paper present a practical depth-map refinement system designed for highly corrupted multiple depth maps. We define a pixel-wise confidence measurement of depth value and apply the three-steps depth-map refinement scheme (*i.e.*confidence-based depth-map fusion, confidence-weighted bundle optimization and super-pixel-based planar propagation) to maximize the whole reliability of depth maps.**

Our experimental result shows that our refinement algorithm can dramatically improve highly corrupted depth maps acquired by previous approaches.

I. Introduction

Depth map is an image where distances between the camera and the scene are assigned to each pixel. Recently, the depth-map processing has become one of most attractive areas in image processing and computer vision since many researchers have reported that depth information helps various visual tasks such as image-based rendering [10], background subtraction [23], object tracking [5] and so on.

As is well known that there are roughly two ways to acquire depth maps. One is the image-based approach so called stereo or multi-view stereo (MVS) and using active sensors such as Time of Flight (TOF) sensors or structured light sensors such as MicrosoftKinect is the other way. Most existing image-based depth-map estimation algorithms are further classified into two categories, one is the feature-based approach, and the other is the optimization-based approach. In the feature-based approach, each pixel's depth value is recovered from the local matching with the highest stereo confidence score [8]. While effective in a narrow baseline case, this winner-takes-all strategy is easily disrupted under the large projective distortion and repetitive textures where correct correspondences are difficult to be found. To avoid incorrect matching, some works find sparse salient matches around textured areas initially, and then propagate them into surroundings [11], [9]. Furthermore, various optimization-based algorithms have been proposed, which introduce spatial constraints of depth maps in their cost function [19], [20], [13].

Unfortunately, while dramatic developments in the image-based and the active-sensor-based depth-map acquisition, the accuracy of depth maps is yet problematic. The active sensors are still poor around the object boundaries due to the occlusion problem and sensor satiations. And most existing image-based algorithms including both feature-based and optimization-based approach do not consider any consistency of depth maps among views (especially when there are more than two views).

The primary contributions of this work are twofold. First, we present a practical depth-map refinement system designed for highly corrupted multiple depth maps which enforces the consistency of depth maps. We define a pixel-wise confidence measurement of depth value and apply the three-steps depth-map refinement scheme to maximize the whole reliability of depth maps. Secondly, we propose a confidence-based dense track extraction algorithm, which is a core part of the bundle optimization for multiple depth maps. While Li et al. [12] were the first to introduce the bundle adjustment to merge depth maps, they extracted dense tracks by simply connecting the successive two-view correspondences and then optimized each track with re-projection errors, therefore their method cannot be applied when no explicit correspondence among views is provided (*e.g.*, only depth maps and camera parameters are provided). In contrast that, we try to use *all* of the views simultaneously to find the optimal tracks by taking advantages of confidence scores. In our experimental results, we apply our method to corrupted depth maps acquired by multi-view stereo algorithms and show that our method improve the accuracy of depth maps dramatically.

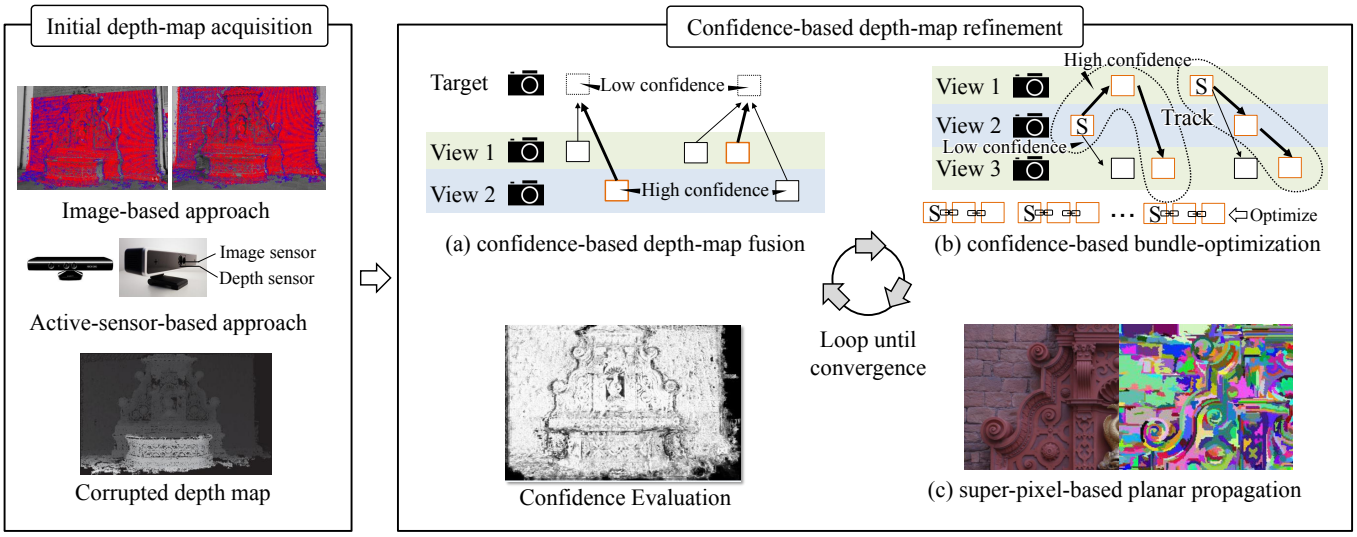

Fig. 2. System overview.

II. Confidence-based Simultaneous Depth-maps Refinement

In this section, we present the three-steps depth refinement algorithm which is composed by, namely, (a) *confidence-based depth-map fusion*, (b) *confidence-based bundle optimization* and (c) *super-pixel-based planar propagation* (see Figure 2). We define a confidence metric which evaluates the consistency of depth maps among views by borrowing the investigation of Hu *et al*. [8] which quantitatively evaluated thirteen kinds of stereo confidence metrics using both indoor and outdoor datasets.

Henceforth we rely the following assumptions:
(1) There are at least two initial depth maps of a static scene.
(2) Each view has aligned depth and color image.
(3) The position and intrinsic parameters of sensors are known.

When initial depth maps are recovered by the image-based approach (*e.g*., multi-view stereo), those assumptions are mostly satisfied. And fortunately, we can satisfy those assumptions when we use active sensors since recent active depth sensors can also capture color images and camera parameters could be recovered by the applying structure-from-motion method [17] with them. To align depth and color images, we can usually use the calibration software provided by the manufacturer, otherwise other calibration methods such as [4] are available.

A. Corresponding points among depth maps

Corresponding points of two depth maps are uniquely determined if the camera parameters of each viewpoint and *one of each* depth value are known. It means that even if either of depth value is incorrect, we can get the correct match (and corresponding scene 3D point), which is the important characteristics of a depth map.

Now, we assume that $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in image i and j are corresponding each other, their relationship is represented as follows,

$$\begin{aligned}\tilde{\boldsymbol{x}}_j &= f^{ij}(\boldsymbol{x}_i) \sim K_j(R_j^T R_i)K_i^{-1}\tilde{\boldsymbol{x}}_i - \frac{1}{d_i}K_j R_j^T(t_j - t_i),\\ \tilde{\boldsymbol{x}}_i &= f^{ji}(\boldsymbol{x}_j) \sim K_i(R_i^T R_j)K_j^{-1}\tilde{\boldsymbol{x}}_j - \frac{1}{d_j}K_i R_i^T(t_i - t_j),\end{aligned} \tag{1}$$

where $d_{i,j}$ are depth values of each pixel, $K_{i,j}$ are intrinsic matrices, $R_{i,j}$ are rotation matrices and $t_{i,j}$ are translations of each camera. Note that in Eq. (1), each point is represented by a homogeneous coordinate system.

When a correspondence ($\boldsymbol{x}_i = [u_i, v_i]^T$, $\boldsymbol{x}_j = [u_j, v_j]^T$) and each camera matrix $P^i = K_i(R_i^T| - R_i^T\boldsymbol{t}_i)$, $P^j = K_j(R_j^T| - R_j^T\boldsymbol{t}_j)$ are provided, we can compute the 3D point X by the triangulation as follows,

$$\begin{aligned}&A\tilde{X} = 0\\ &A = \begin{bmatrix} P^i_{11} - P^i_{31}u_i & P^i_{12} - P^i_{32}u_i & P^i_{13} - P^i_{33}u_i & P^i_{14} - P^i_{34}u_i \\ P^i_{21} - P^i_{31}v_i & P^i_{22} - P^i_{32}v_i & P^i_{23} - P^i_{33}v_i & P^i_{24} - P^i_{34}v_i \\ P^j_{11} - P^j_{31}u_j & P^j_{12} - P^j_{32}u_j & P^j_{13} - P^j_{33}u_j & P^j_{14} - P^j_{34}u_j \\ P^j_{21} - P^j_{31}v_j & P^j_{22} - P^j_{32}v_j & P^j_{23} - P^j_{33}v_j & P^j_{24} - P^j_{34}v_j, \end{bmatrix}\end{aligned} \tag{2}$$

where $X = [\tilde{X}_{(1)}/\tilde{X}_{(4)}, \tilde{X}_{(2)}/\tilde{X}_{(4)}, \tilde{X}_{(3)}/\tilde{X}_{(4)}]$. Inversely, a 3D point X and camera matrices give the projections into each viewpoint are as follows,

$$\begin{aligned}d_i(\boldsymbol{x}_i)[u_i, v_i, 1]^T &= P^i[X_{(1)}, X_{(2)}, X_{(3)}, 1]^T\\ d_j(\boldsymbol{x}_j)[u_j, v_j, 1]^T &= P^j[X_{(1)}, X_{(2)}, X_{(3)}, 1]^T.\end{aligned} \tag{3}$$

Here we emphasize that one of each depth value is enough to find the correspondence between two views and we can re-project the 3D point recovered from the correspondence into each view with depth values. It means that one depth value gives estimation of the depth value of corresponding pixels, which is the core of our confidence-based refinement scheme described in following sections.

B. Confidence metric for evaluating consistency of depth maps

Within the field of the stereo matching, there have been much researches about the confidence metric to accurately estimate how trustworthy the correspondences are (*e.g.*, [6], [8]). However, unfortunately, there is little research about the confidence of depth [19], [16] or limited in the domain of active sensors [15]. Therefore, we design a new heuristic confidence metric for our algorithm by importing ideas from stereo confidence metrics [8].

Inspired by [19], we define the confidence of the depth as the maximum of the confidence of two-view correspondences given by Eq. (1),

$$C_i(\boldsymbol{x}_i) \triangleq \max_j c_{ij}(\boldsymbol{x}_i), \tag{4}$$

where $\boldsymbol{x}_i$ is a pixel in the i-th view and c_{ij} is a confidence metric for each two-view correspondence from $\boldsymbol{x}_i$ to the j-th view. Note that maximizing c_{ij} can implicitly compensate for the case where an accurate depth value wrongly gives small c_{ij} because of the occlusion.

Requirements for our confidence metric in our algorithms are threefold:
(1) Our confidence metric can estimate how trustworthy the provided depth values are.
(2) It leads as small as possible false positive errors.
(3) Our confidence metric should give a high score for high consistency among views.

Under those requirements, we defined c_{ij} as follows,

$$c_{ij}(\boldsymbol{x}_i) \triangleq Conf_{\text{geo}} * Conf_{\text{photo}} \leq 1, \tag{5}$$

$$Conf_{\text{geo}} \triangleq \frac{1}{1 + \frac{|\boldsymbol{x}_i - f^{ji}(\boldsymbol{x}_j)|}{\eta_a}}, \tag{6}$$

$$Conf_{\text{photo}} \triangleq \frac{1}{1 + \frac{|F(\boldsymbol{x}_i, \boldsymbol{x}_j) - \min_{\boldsymbol{p}_i \in N_{\boldsymbol{x}_i}} F(\boldsymbol{p}_i, \boldsymbol{x}_j)|}{\eta_b}}, \tag{7}$$

where $\boldsymbol{x}_j = f^{ij}(\boldsymbol{x}_i)$ and η_a and η_b are weights to regularize each confidence to 1, and the function F evaluates the photo-consistency of two views locally (we will discuss about this function later). $\boldsymbol{p}_i$ are pixels in the i-th view collected from the epipolar line of $\boldsymbol{x}_j$. c_{ij} is composed of two terms, one evaluates the geometry-consistency and the other evaluates the photometry-consistency. Note that we observed that introducing both of them decreased the false positive errors rather than introducing either of them.

We simply use the well known metric called *forward-backward error* [19] as the geometry-consistency term which gives high score when the correspondence is bidirectional. As for the photometry-consistency term, we are inspired by the Hu *et al*. [8]'s LRD (Left-Right Difference) which gives a high score if the photoconsistency of the correspondence is locally minimum since otherwise, it means that there exists a better correspondence.

There are many options for the photoconsistency function F. The most commonly used function in the stereo case may be a color distance or block-based similarity measures such as SSD or NCC. However in case viewpoints are widely separated, we should choose the metric which have a high reliability under large perspective distortions. Here we use the DAISY descriptor [21], which are robust to distortions yet less computational cost comparing to well known local descriptors such as SURF [2].

C. Confidence-based depth-map fusion

Our confidence-based depth-map fusion scheme relies the fact that each of depth value in corresponding points gives another depth value (see section II-A). It means that reliable pixels which correspond to unreliable pixels can give better estimation of depth of those unreliable pixels, which is the core concept of our fusion scheme.

The algorithm is described in Algorithm. 1. Assume we have calculated confidence values for each pixel as described in II-B, and we try to refine the unreliable depth $d(\boldsymbol{x}_i)$ of $\boldsymbol{x}_i$ in i-th view. Firstly, we search for all pixels $\boldsymbol{x}_j$ from other views which can project onto $\boldsymbol{x}_i$ by using Eq. (1). While we may find many correspondences, we try to find the *best* correspondence between $\boldsymbol{x}_i$ and the pixel with most reliable depth value since an inaccurate correspondence can rather disrupt $d(\boldsymbol{x}_i)$. Therefore we choose the pixel with the highest confidence score as follows,

$$\boldsymbol{x}_{max} \triangleq \text{argmax}_{\boldsymbol{x}_j} C_j(\boldsymbol{x}_j). \tag{8}$$

Note that we use $C_j(\boldsymbol{x}_j)$ instead of $c_{ji}(\boldsymbol{x}_j)$ since the priority of the consistency in $c_{ji}(\boldsymbol{x}_j)$ gives a low confidence score if $d_i(\boldsymbol{x}_i)$ is inaccurate. In addition, we remove $\boldsymbol{x}_j$ whose corresponding point is occluded in i-th view by a simple photo-consistency check (*i.e.*, we only collect pixels where the ℓ_2 color distance from $\boldsymbol{x}_i$ is less than 10).

Although the correspondence of $\boldsymbol{x}_i$ and $\boldsymbol{x}_{max}$ gives another $d_i(\boldsymbol{x}_i)$ by Eq. (3), we should prevent the accuracy of $d(\boldsymbol{x}_i)$ from decreasing by this fusion. Therefore, we only update $d(\boldsymbol{x}_i)$ when these two confidence checks between $C(\boldsymbol{x}_i)$ and $C(\boldsymbol{x}_{max})$ are satisfied,

$$C(\boldsymbol{x}_{max}) > \varepsilon_a,\ C(\boldsymbol{x}_{max}) - C_i(\boldsymbol{x}_i) > \varepsilon_b \tag{9}$$

where ε_a and ε_b are thresholds set to 0.5 and 0.2, respectively, in our implementation (note that confidence values are from 0 to 1). The first condition in Eq. (9) is not necessary but important to avoid any inefficient propagation.

While our depth-map fusion scheme refines low-confidence pixels, especially where depth values are not assigned initially, there are two limitations. First, only low-confidence pixels are refined since the direction of our propagation is from high-confidence pixels to low-confidence pixels. Secondly, refinement is based on per-pixel, two-view correspondences without considering about sub-pixel correspondences and multi-view geometry. Thus, after this step we apply bundle optimization described in the next section for further refinement.

D. Confidence-based bundle optimization

Our confidence-based bundle optimization algorithm mainly builds on the standard bundle adjustment scheme which is

Algorithm 1 Confidence-based depth-map fusion

$C_{max} \leftarrow 0$, C and c are already calculated
for $j = 1$ to $n - 1$ **do**
  for $i = j + 1$ to n **do**
    for all $\boldsymbol{x}_j$ in j-th view **do**
      $\boldsymbol{x}_i \leftarrow f^{ji}(\boldsymbol{x}_j)$
      if $C_j(\boldsymbol{x}_j) > C_{max}(\boldsymbol{x}_i)$, $C_j(\boldsymbol{x}_j) - C_i(\boldsymbol{x}_i) > \varepsilon_b$ and $C_j(\boldsymbol{x}_j) > \varepsilon_a$ **then**
        $C_{max}(\boldsymbol{x}_i) \leftarrow C_j(\boldsymbol{x}_j)$, $\boldsymbol{x}_{max} \leftarrow d_{new}$
      end if
    end for
    for all $\boldsymbol{x}_i$ in i-th view **do**
      $\boldsymbol{x}_j \leftarrow f^{ij}(\boldsymbol{x}_i)$
      if $C_i(\boldsymbol{x}_i) > C_{max}(\boldsymbol{x}_j)$, $C_i(\boldsymbol{x}_i) - C_j(\boldsymbol{x}_j) > \varepsilon_b$ and $C_i(\boldsymbol{x}_i) > \varepsilon_a$ **then**
        $C_{max}(\boldsymbol{x}_j) \leftarrow C_i(\boldsymbol{x}_i)$, $d_j(\boldsymbol{x}_j) \leftarrow d_{new}$
      end if
    end for
  end for
end for

mainly applied in the structure-from-motion framework [17] to simultaneously refine 3D points and camera parameters by minimizing the sum of re-projection errors (*i.e.*, ℓ_2 distances between projections of original and refined 3D points). Since bundle adjustment builds on an assumption that re-projection errors obey the zero-mean Gaussian distribution [1], outliers in the input sequence of points easily disrupt the optimization, which is a critical issue when we apply the bundle adjustment to correspondences extracted from highly corrupted depth maps. Unlike Li et al. [12] which simply connected matching pixels of stereo pairs to build a track and then filtered out the pixels as outliers whose re-projection error is less than a certain threshold, we use the confidence score to intelligently extract the tracks from highly corrupted depth maps and apply confidence-weighed bundle adjustment to refine whole track.

1) Dense Track Extraction: We define a track as a sequence of pixels over views which assumed to be pointing at the same 3D point. $t_k = \{x_1^k, x_2^k, ...x_{N(t)}^k\}$.

We illustrate the algorithm in Table I. Track extraction process is described by two steps, *i.e.*, the selection of the starting point and the order of next viewpoints. The starting points of each track are critically important since inaccurate depth values can lead unreliable tracks. Therefore we firstly set a pixel with the highest depth-confidence score (C_i), as a start point of each track. Then we collect correspondences in the track considering two factors for selecting the next view; first is the reliability of the correspondence between current view and the next view (c_{ij}), and second is the possibility of a correspondence between the next view and the view after the next (C_j) because we would like to build the longer tracks. We finish the track extraction if the confidence of current pixel (C_i) is under a threshold (0.5 in our implementation).

The important property of our track extraction algorithm is twofold. First, our method intelligently chooses the first viewpoint from all of viewpoints which is practical when using depth maps where there is no explicit *first view* (*e.g.*, captured by active sensors). Secondly, when tracks are extracted, we can also acquire the confidence of each pixel, which could be used as the weight for the bundle optimization described in the next section.

2) Confidence-weighted Bundle Adjustment: When tracks are built, we firstly reconstruct a 3D point for each track by applying triangulation in a similar manner with Eq. (2). Then we apply the weighted bundle adjustment for each track as follows,

$$\min_X E = \sum_{i=1}^{N_t} w_i^2 f(\boldsymbol{x}_i, P_i X), \tag{10}$$

$$\begin{aligned} w_i &= C_1 && (i = 1), \\ w_i &= C_1 \textstyle\prod_{k=2}^{i} c_{(k-1,k)} && (else). \end{aligned} \tag{11}$$

Here, N_t is the number of points included in a track and P is a projective matrix for each viewpoint. $\boldsymbol{x}_i$ is the i-th points in the track and X is the 3D point corresponding to the track initialized by the triangulation result. f is the ℓ_2 re-projection error and w_i is a penalty weight reflecting confidence scores (we discuss about the weight later). The objective is minimized with the Levenberg-Marquardt algorithm. When the optimization is finished, we can get a set of refined 3D point cloud.

Unlike Li et al. [12] which used current re-projection errors as the penalty weight, we relies confidence of each correspondence which has higher score the former in the track. Therefore, our method not only filters out outliers, but also refine unreliable correspondences by propagating reliable information of former part of the tracks. It means that even if the depth value of the last pixel is not accurate, the optimization is not disrupted by the pixel since the weight is very small, in spite that, the inaccurate pixel could receive the benefit from the optimization. One problem is that we acquire the refined quasi-dense 3D points as the output of our bundle optimization. To get refined dense depth maps, we finally apply super-pixel based planar fitting described in the next section.

E. Super-pixel based planar propagation

While refined 3D points are not dense, we can use them as the seeds of segmentation based approach [9], [22] to get dense depth maps. This approach assumes that the region with the similar observation has a loose change of structure and approximate the underlying local structure as a simple plane. Although this approach is usually used for the last step of stereo algorithms to remove outliers, it could be also applied to estimate depth maps.

Firstly we divide images into super-pixels by mean-shift color segmentation [3]. We assume that the depth map of each super pixel is approximated by a plane and depth values inside the super pixel is represented as follows,

$$d(\boldsymbol{x}_i) = au_i + bv_i + c. \tag{12}$$

where d is the depth value provided by the projection of a refined 3D point, $\boldsymbol{x}_i = [u_i, v_i, 1]^t$ is the i-th point in the

TABLE I
ALGORITHM OF CONFIDENCE-BASED TRACK EXTRACTION

1.	Create $List$, $Match$ and $Track$. $List$ and $Match$ are stacks of pixels and $Track$ is the stack of $Match$.
2.	Put all pixels into $List$ with confidence scores C_i and c_{ij}
2.	Sort $List$ in a descending order of C_i.
3.	Pull the top of $List$ and set it on x_i and put x_i into $Match$ if it has not been included in $Track$.
4.	Put x_j in j-th iamge into $Match$, which satisfies following conditions. (1) x_j is a correspondence of x_i. (2) $j = \mathrm{argmax}_k c_{ik} C_k$. (3) j-th view has not been included in $Match$. If all views have been included in $Match$ or $c_{ij}C_j$ is less than a certain threshold, go to 4. Otherwise set x_j on x_i then iterate 3.
5.	If the size of $Match$ is less than three, then go to 7. Otherwise, go to 4.
6.	Put $Match$ into $Track$
7.	If $List$ is not empty, clear $Match$ and go back to 2.
8.	Get a set of reliable tracks in $Track$.

super-pixel and a, b, c are the plane parameters. While three projections is enough to estimate the parameters, we use all projections to calculate the plane parameters using the voting approach [22] for a robust estimation. After plane parameters for each super pixel are estimated, we can recover depth values of regions with no projection from refined 3D points by Eq. (12), and finally get refined dense depth maps.

We note that our three-step refinement scheme works well when it is applied for several times (at least twice). The computational cost is less problematic since there is no large scale optimization in our scheme and each bundle optimization could be processed in parallel because it is performed on track-level.

III. EXPERIMENTAL RESULTS

A. Implementation and Materials

We implemented the proposed approach in C/C++ platform. We used DAISY library provided by Tola et al. [21] and *levmar* [14]; the library for Levenberg-Marquartdt algorithm. All experiments are performed on a 3.06 GHz Intel Core2 Duo CPU and 4GB RAM. The main computational costs are the calculation of confidence metrics (each needs about 1 minute per image). For evaluating our algorithm, we test our method on two kinds of standard datasets; Strecha et al.'s [18] (outdoor datasets) and Middlebury [7] (indoor datasets). First, by using outdoor datasets, we quantitatively evaluate how each of our step refine the depth maps. Then we show the 3D reconstruction result from indoor datasets.

B. Evaluation of contribution for each refinement step

We applied our method to the output of Strecha et al. [19]'s algorithm, which indicates middle-level performance in the comparison in Middlebury online evaluation (many regions of recovered depth maps are corrupted). In this experiment, we use the *fountain-P11* datasets since the ground truth data is provided by [21] which are captured by a laser scanner. Those examples contained eleven images, however we only used five of eleven images. Because of the limited memory space, we down sampled the images to 768 x 512 from the original resolution of 3,072 x 2,048. For evaluating depth map quantitatively, we introduced *relative depth error histograms* as the measurement of accuracy which proposed by Strecha et al [18] which is defined as follows.

$$\mathbf{h}_k \propto \sum_{ij} \delta_k(|D_l^{ij} - D_s^{ij}|0.01 D_\sigma^j) \tag{13}$$

D_l^{ij} and D_s^{ij} are the ground truth depth and the estimated depth value of camera j and position i. D_σ^j is the variance of the correct depth values in image j given by ground truth data. $\delta_k()$ returns 1 if $|D_l^{ij} - D_s^{ij}|$ is in $[k \times 0.01 D_\sigma^j, (k+1) \times 0.01 D_\sigma^j]$, otherwise returns 0. Then we construct histograms of the relative errors, which regularized by the total number of the pixels which are seen at least from two views. If the error is larger than the range of $k > 20$, we include it in the histogram corresponding to $k = 20$.

We illustrate the experimental result in Figure 3, Figure 4 and Figure 5. In Figure 4, we construct error maps whose colors are decided by the error level. When k is increased, the color changes from blue ($k = 0$) to red ($k = 20$). And green areas are not used in the evaluation because they don't have ground truth values or they could not be seen from more than two views. The experimental results show that depth-fusion step reduced high-level errors ($k = 20$) of input depth maps, in contrast our confidence-weighted bundle optimization and super-pixel-based planar propagation dramatically reduced errors of depth maps especially with low-level errors ($k < 3$). These results show our three-step scheme worked as we had intended. One example of refined depth map is presented in Figure 5.

C. Evaluation with Middlebury datasets

In this experiment, we demonstrate the effectiveness of our method by Middlebury *dinoSparseRing* dataset [7] which includes 16 images captured from largely different viewpoints and have little textures. In this experiment, the input depth maps are recovered from one of local feature-based approaches called match propagation [11] using the SURF descriptors [2] as seeds. Experimental results are illustrated in Figure 1. Since the dataset has less texture, the initially recovered depth maps have large amount of holes. However, by applying our three-step refinement scheme (in this case five times), we can recover refined depth maps where most of holes are filled in.

Finally, we illustrate the qualitative comparison with Hu et al. [8] in Figure 6. To demonstrate the performance of our track extraction algorithm and confidence-weighted bundle optimization, we applied the bundle optimization algorithm which is similar manner with Hu et al. [8](*i.e.*, extract tracks from two successive image pairs and set a penalty weight by re-projection errors). Note that the algorithm is not completely same with them since original work extracts tracks based on dense two-view matches not from depth maps. The experimental result show that our intelligent system of bundle

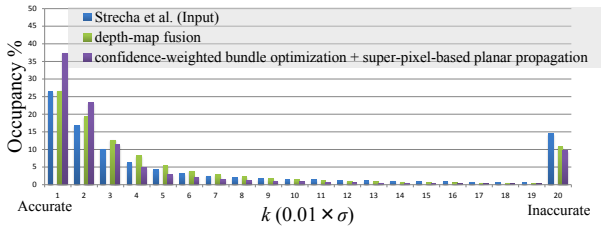

Fig. 3. Error histograms of *fountain-P11* datasets. We averaged results of { 3,4,5,6,7 }-th images.

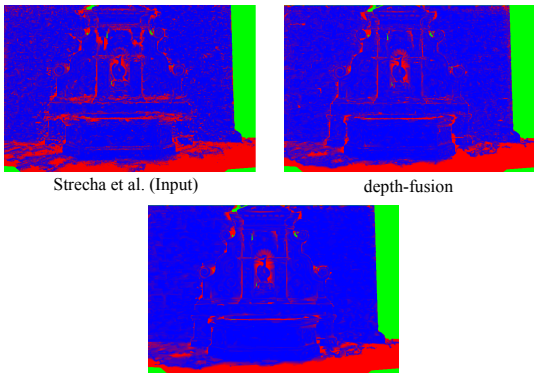

Fig. 4. Examples of error maps. Here, we show only the error maps corresponding to 5-th image of *fountain-P11* datasets.

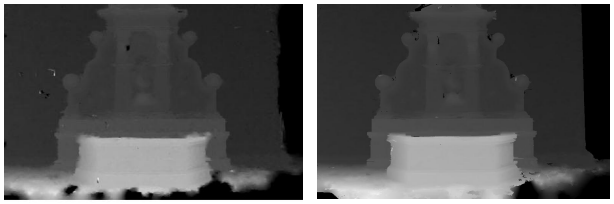

Fig. 5. Examples of the depth map refinement. The left one is the input depth map and the right one is refined by our three-step scheme (our refinement scheme is applied once).

optimization works better than simple scheme of Hu et al. [8].

IV. Conclusion

In this paper, we proposed a novel framework of the confidence-based depth-map refinement scheme. Our experimental results showed that our algorithm can estimate depth maps in detail. Future work will focus on tuning our parameters empirically by training on some data, and applying our algorithms to images captured by active sensors.

References

[1] R. h. B. Triggs, P. Mclauchlan and A. Fitzgibbon. Bundle adjustment - a modern synthesis. *Vision Algorithms: Theory and Practice, LNCS*, 2000.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. *CVIU*, 110(3):346–359, 2008.

[3] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE PAMI*, 24(5):603–619, 2001.

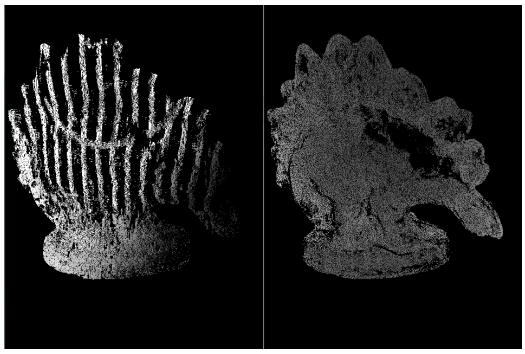

Fig. 6. Comparison of refined depth maps by bundle optimization algorithms, (a) the similar manner with Hu et al. [8], (b) our method.

[4] J. K. D. Herrera and J. Heikkila. Accurate and practical calibration of a depth and color camera pair. *CAIP*, 2011.

[5] A. Ess, B. Leibe, K. Schindler, and L. V. Gool. A mobile vision system for robust multi-person tracking. *In ICCV*, 2007.

[6] M. M. G. Egnal and R. P. Wildes. A stereo confidence metric using single view imagery with comparison to five alternative approaches. *In Proc. of International Conferenc of Vision Interface*, 2004.

[7] http://vision.middlebury.edu/mview/.

[8] X. Hu and P. Mordohai. Evaluation of stereo confidence indoors and outdoors. *In CVPR*, 2010.

[9] M. Jancosek and T. Pajdla. Segmentation based multi-view stereo. *Computer Vision Winter Workshop*, 2009.

[10] P. Kauffa, N. Atzpadina, C. Fehna, O. S. M. Mullera, A. Smolica, and R. Tangera. Depth map creation and image-based rendering for advanced 3dtv services providing interoperability and scalability. *Signal Processing: Image Communication*, 22(2):217–234, 2007.

[11] M. Lhuillier and L. Quan. Match propagation for image-based modeling and rendering. *IEEE PAMI*, 24(8):1140–1146, 2002.

[12] J. Li, E. Li, Y. Chen, L. Xu, and Y. Zhang;. Bundled depth-map merging for multi-view stereo. *In CVPR*, 2010.

[13] Y. Liu, X. Cao, Q. Dai, and W. Xu. Continuous depth estimation for multi-view stereo. *In CVPR*, 2009.

[14] M. I. A. Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++, Jul. 2004. Accessed on 31 Jan. 2005.

[15] L. P. T. W. M. Reynolds, J. Dobos and G. J. Brostow. Capturing time-of-flight data with confidence. *In Proc. of Computer Vision and Pattern Recognition*, 2011.

[16] L. W. P. M. P. Merrell, A. Akbarzadeh and J. M. Frahm. Real-time visibility-based fusion of depth maps. *In Proc. of IEEE 11th International Conference on Comput Vision*, 2007.

[17] M. Pollefeys, F. Verbiest, and L. V. Gool. Surviving dominant planes in uncalibrated structure and motion recovery. *In ECCV2002, Lecture Notes in Computer Science*, 2351:837–851, 2002.

[18] C. Strecha, W. V. Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. *In CVPR*, 2008.

[19] C. Strecha, T. Tuytelaars, and L. V. Gool. Dense matching of multiple wide-baseline views. *In ICCV*, 2003.

[20] M. F. Tappen and W. T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. *In ICCV*, 2003.

[21] E. Tola, V. Lepetit, and P. Fua. A fast local descriptor for dense matching. *In CVPR*, 2008.

[22] Z. Wang and Z. Zheng. A region based stereo matching algorithm using cooperative optimization. *In CVPR*, 2008.

[23] G. Zhang, J. Jia, W. Xiong, T. T. Wong, P. A. Heng, and H. Bao. Moving object extraction with a hand-held camera. *In ICCV*, 2007.