High Quality Depth Map Upsampling and Completion for RGB-D cameras

Jaesik Park, *Student Member, IEEE*, Hyeongwoo Kim, Yu-Wing Tai, *Senior Member, IEEE*, Michael S. Brown, *Member, IEEE*, and In So Kweon, *Member, IEEE*

***Abstract*—This paper describes an application framework to perform high quality upsampling and completion on noisy depth maps. Our framework targets a complementary system setup which consists of a depth camera coupled with an RGB camera. Inspired by a recent work that uses a nonlocal structure regularization, we regularize depth maps in order to maintain fine details and structures. We extend this regularization by combining the additional high-resolution RGB input when upsampling a low-resolution depth map together with a weighting scheme that favors structure details. Our technique is also able to repair large holes in a depth map with consideration of structures and discontinuities by utilizing edge information from the RGB input. Quantitative and qualitative results show that our method outperforms existing approaches for depth map upsampling and completion. We describe the complete process for this system, including device calibration, scene warping for input alignment, and even how our framework can be extended for video depth-map completion with consideration of temporal coherence.**

***Index Terms*—Depth Map Upsampling, Depth Map Completion, RGB-D cameras**

I. Introduction

Active depth cameras are becoming a popular alternative to stereo-based range sensors. In particular, 3D time-of-flight (3D-ToF) cameras and active pattern cameras (e.g. Microsoft Kinect) are widely used in many applications. 3D-ToF cameras use active sensing to capture 3D range data at frame-rate as per-pixel depth. A light source from the camera emits a near-infrared wave that is then reflected by a scene and captured by a dedicated sensor. Depending on the distance of objects in a scene, the captured light wave is delayed in phase compared to the original emitted light wave. By measuring the phase delay the distance at each pixel can be estimated. Active pattern cameras emit a pre-defined pattern into a scene and use a camera to observe the deformation and translation of the pattern to determine depth map. This can be done using an infrared projector and camera to avoid human detection and interference with other visible spectrum imaging devices.

Jaesik Park, Hyeongwoo Kim, Yu-Wing Tai and In So Kweon are with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Republic of Korea.
E-mail: jspark@rcv.kaist.ac.kr, hyeongwoo.kim@rcv.kaist.ac.kr, yuwing@kaist.ac.kr, iskweon77@kaist.ac.kr

Michael S. Brown is with School of Computing, National University of Singapore (NUS), Singapore.
E-mail: brown@comp.nus.edu.sg

Manuscript received XXXX XX, XXXX; revised XXXX XX, XXXX.

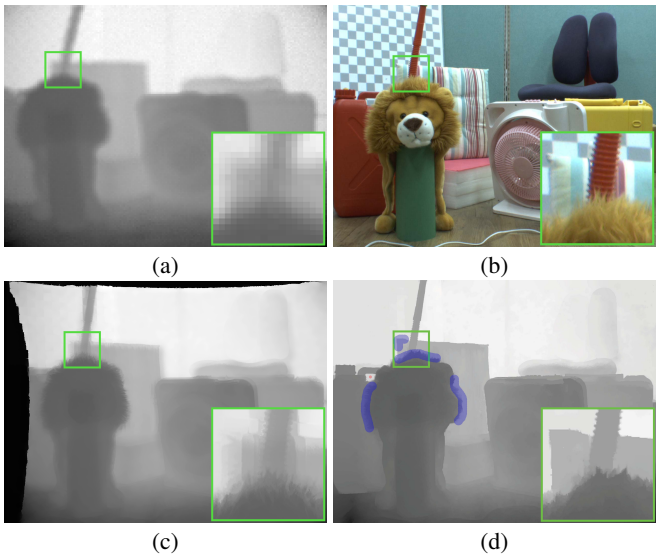


Fig. 1. (a) Low-resolution depth map (enlarged using nearest neighbor upsampling), (b) high-resolution RGB image, (c) result from [29], (d) our result. User scribble areas (blue) and the additional depth sample (red) are highlighted. The dark areas in (c) are the areas without depth samples after registration. Full resolution comparisons are provided in the supplemental materials.

The quality of the depth maps captured by these 3D cameras, however, are relatively low. In the 3D-ToF cameras, the resolution of the captured depth maps are typically less than 1/4th the resolution of a standard definition video camera. In addition, the captured depth maps are often corrupted by significant amounts of noise. In active pattern cameras, although the captured depth map has a comparable resolution to the resolution of RGB cameras, the depth map usually contains many holes due to the disparity between the IR projector and the IR camera. In addition, the captured depth map contains quantization errors and holes due to the estimation errors in pattern matching and the presence of bright light sources and non-reflective objects in a scene.

The goal of this paper is to propose a method to estimate a high quality depth map from the depth sensors through upsampling and completion. To aid this procedure, an auxiliary high-resolution conventional camera is coupled with the depth camera to synchronously capture the scene. Related work [29], [3], [6] also using coupled device setups for depth map upsampling have focused primarily on image filtering techniques such as joint bilateral filtering [7], [17] or variations. Such filtering techniques can often over-smooth results, especially in areas of structure details.

We formulate the depth map upsampling and completion problem using constrained optimization. Our approach is inspired by the recent success of nonlocal structure regularization for depth map construction from depth-from-defocus [8]. In particular, we describe how to formulate the problem into a least-squares optimization that combines nonlocal structure regularization together with an edge weighting scheme that further reenforces fine details. We also employ scene warping to better align the imagery to the auxiliary camera input. While our methodology is heuristic in nature, the result is a system that is able to produce high-quality depth maps superior in quality to prior work. In addition, our approach can be easily extended to incorporate simple user markup to correct errors along discontinuity boundaries without explicit image segmentation (e.g. Fig. 1).

A shorter version of this work appeared in [22], which focuses on the depth map upsampling problem for 3D-ToF cameras. This paper shows that the framework is also applicable in depth map completion for active pattern cameras (e.g. Microsoft Kinect) with additional implementation details, discussions and experiments for depth map completion. A new technical section is included that extends our framework to depth video completion. Our new experimental results demonstrate high quality temporarily coherent depth video, which out-performs our single frame approach in [22] for the depth video completion problem.

The remainder of our paper is organized as follow: In Sec. II, we review related works in depth map upsampling and completion. Our system setup and preprocessing steps are presented in Sec. III, followed by the optimization framework in Sec. IV. In Sec. V, we present our experimental results. Finally, we conclude our work in Sec. VI.

II. Related Work

Previous work on depth map upsampling and completion can be classified as either *image fusion techniques* that combine the depth map with the high quality RGB image or *super-resolution techniques* that merge multiple misaligned low quality depth maps. Our approach falls into the first category of image fusion which is the focus of the related work presented here.

Image fusion approaches assume there exists a joint occurrence between depth discontinuities and image edges and those regions of homogenous color have similar 3D geometry [31], [26]. Representative image fusion approaches include [5], [29], [3], [6]. In [5], Diebel and Thrun performed upsampling using an MRF formulation with the data term computed from the depth map and weights of the smoothness terms between estimated high-resolution depth samples derived from the high-resolution image. Yang et al. [29] used joint bilateral filtering [7], [17] to fill the hole and interpolate the high-resolution depth values. Since filtering can often over-smooth the interpolated depth values, especially along the depth discontinuity boundaries, they quantized the depth values into several discrete layers. Joint bilateral filtering was applied at each layer with a final post processing step to smoothly fuse the discrete layers. This work was later extended by [30] to use a stereo camera for better discontinuity detection in order to avoid over-smoothing of depth boundaries. Chan et al. [3] introduced a noise-aware bilateral filter that decides how to blend between the results of standard upsampling or joint bilateral filtering depending on the depth map's regional statistics. Dolson et al. [6] also used a joint bilateral filter scheme, however, their approach includes additional time stamp information to maintain temporal coherence for the depth map upsampling in video sequences.

The advantage of these bilateral filtering techniques is that they can be performed quickly; e.g. Chan et al. [3] reported near real-time speeds using a GPU implementation. However, the downside is that they can still over-smooth fine details. Our approach is more related to [5] because we formulate the problem using a constrained optimization scheme. However, our approach incorporates a nonlocal structure (NLS) term to help preserve local structures. This additional NLS term was inspired by Favaro [8], which demonstrated that NLS filtering is useful in maintaining fine details even with noisy input data. We also include an additional weighting scheme based on derived image features to further reinforce the preservation of fine detail. In addition, we perform a warping step to better align the low-resolution and high-resolution input. Huhle et al. [11] proposes NLS based filtering algorithm targeting a sparse 3D point cloud. However, data alignment and outlier rejection are not needed in this approach. It also only considers the NLS term, while our approach uses several additional terms. Our experimental results on ground truth data shows that our application framework can outperform existing techniques for the majority of scenes with various upsampling factors. Since our goal is high-quality depth maps, the need for manual cleanup for machine vision related input is often unavoidable. Another advantage of our approach is that it can easily incorporate user markup to improve the results.

Recently, the Kinect sensor has been widely used in the research community. The Kinect's depth accuracy and mechanism are extensively analyzed in [16]. In [12], Izadi et al. proposed KinectFusion for static scene modeling. This approach aligns multi-view depth maps into pre-allocated volumetric representation by using a signed-distance function. It is adequate for modeling objects within a few cubic meters [20]. Also, this method can be applied to depth map refinement since it averages noisy depth maps from multiple view points. Compared to Izadi et al. [12], we focus our work as a depth map upsampling and completion for a single RGB-D image configuration. While our configuration is more restrictive, several research groups have utilized Kinect to build RGB-D datasets [13], [24] under various scenes and objects for scene categorization, object recognition and segmentation. Since the Kinect depth map tends to have holes, it is necessary to fill in these missing depth values before they can be used for recognition and segmentation tasks. In this paper, using the same optimization framework for depth map upsampling, we demonstrate our approach can achieve high quality depth map completion for these Kinect RGB-D data.

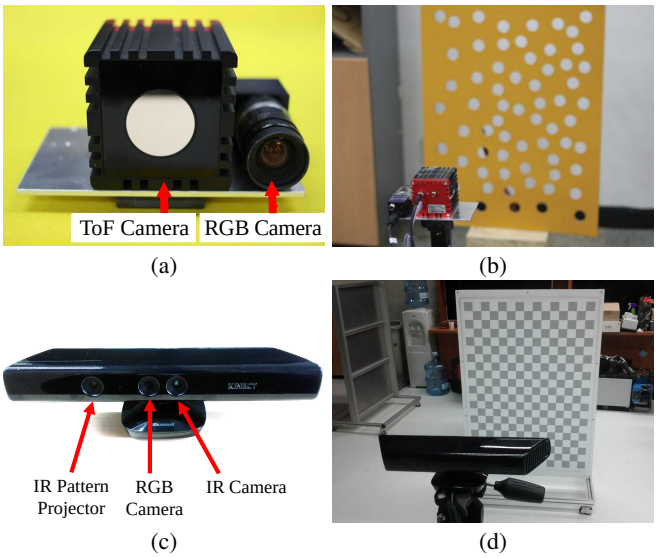


Fig. 2. (a) Our 3D-ToF imaging setup. It uses a 3D-ToF camera which captures images at 176×144 resolution that is synchronized with a 1280×960 resolution RGB camera. (b) The calibration configuration for 3D-ToF imaging setup that uses a planar calibration pattern with holes to allow the 3D-ToF camera to be calibrated. (c) The Kinect imaging setup. The Kinect depth camera which captures depth map at 640×480 resolution that is synchronized with a 1280×1024 resolution RGB camera. (d) The calibration configuration for Kinect imaging setup, which utilizes a planar checker board.

III. System Setup and Preprocessing

In this section, we describe our RGB-D imaging step and the preprocessing step to register the depth camera and conventional RGB camera and to perform initial outlier rejection on the depth input.

A. System Configuration

Figure 2 (a) shows our hardware configuration consisting of a 3D-ToF camera and a high-resolution RGB camera. For the ToF camera, we use a SwissRanger^{TM} SR4000 [1] which captures a 176×144 depth map. For the RGB camera, we use the Point Grey Research Flea RGB camera with a resolution of 1280×960 pixels. Figure 2 (c) shows the Microsoft^{TM} Kinect depth range sensor. The Microsoft Kinect is equipped with an IR pattern projector, an IR camera for depth measurement, and an RGB camera. The resolution of the captured depth map depends on the resolution of the IR camera. The default resolution of the Kinect is 640×480 and it is capable to capture depth map and RGB image up to 1280×1024 (at a low frame rate) resolution.

B. Camera Calibration

Since the RGB-D data captured from the system in Sec. III-A have slightly different viewpoints (both for 3D-ToF setting and Kinect), we need to register the camera according to the depth values from the low-resolution depth map. This process requires intrinsic and extrinsic camera parameters.

In this work, we utilize prior art for the heterogeneous camera rig calibration. For 3D-ToF camera setting, we use the method proposed by Jung et al. [14]. This method uses a planar calibration pattern consisting of holes (Figure 2 (b)). This unique calibration pattern allows us to detect the positions on the planar surface that are observed by the 3D-ToF camera. In the case of the Kinect, we utilize the Kinect calibration Toolbox [10] which simultaneously estimates the camera parameters and fixed pattern noise of the depth map. In this case, a conventional checkerboard is utilized.

The captured depth map does not response linearly to the actual depth in the real world. To calibrate the response of the depth camera, we capture a scene with a planar object and move the planar object towards the camera. The actual movement of the planar object is recorded and is used to fit a depth response curve to linearize the depth map before warping the depth map to align with the RGB image.

Besides our proposed calibration method, Pandey et al. [21] introduces an automatic extrinsic calibration method for RGB-D cameras. They optimize 6D extrinsic parameters by maximizing mutual information. The mutual information is evaluated by measuring the correlation between surface reflectivity from a range scanner (or known as intensity image in ToF camera case) and intensity values from an RGB camera in their joint histogram. Talyor and Nieto [25] introduced a method that also takes the intrinsic calibration parameters into account. They use *normalized* mutual information to avoid drift and apply a particle swam optimization [15] for robust estimation. Compared to the automatic methods, our procedure requires a calibration pattern as shown in Fig. 2 (b) and (d). The usage of the calibration pattern allows us to achieve higher accuracy in calibration. However, in the case when pre-calibration is impossible, the automatic approaches can be a good substitution for our framework.

C. Depth Map Registration

With the estimated camera parameters, we first undistort the depth map and RGB image. After that, the linearized depth map is back-projected as a point cloud and points are projected into the RGB camera coordinate. Numerically, for any point, $\mathbf{x}_t = (u, v)^\top$, on the low-resolution depth map with depth value d_t, we can compute its corresponding position in the high-resolution RGB image by the following equation:

$$s\mathbf{x}_c = \mathsf{K_c}\left[\begin{array}{c|c} \mathsf{R} & \mathsf{t}\end{array}\right]\mathsf{K_d}^{-1}[\mathbf{x}_t \;\; d_t]^\top \tag{1}$$

where $\mathsf{K_c}$ and $\mathsf{K_d}$ are the intrinsic parameters of the RGB and depth camera respectively, and R and t are the rotation and translation matrices which describe the rotation and translation of the the RGB camera and the depth camera with respect to the 3D world coordinate. We obtain the scaling term s by calculating the relative resolution between the depth camera and the RGB camera. Since the depth map from the depth camera is noisy, we impose a neighborhood smoothness regularization using the thin-plate spline to map the low-resolution depth map to the high-resolution image. The thin-plate spline models the mapping of 3D points by minimizing the following equation:

$$\arg\min_{\alpha} \sum_j ||\mathrm{x}'_j - \sum_i \alpha_i R(\mathrm{x}_j - \mathrm{x}_i)||^2 \tag{2}$$

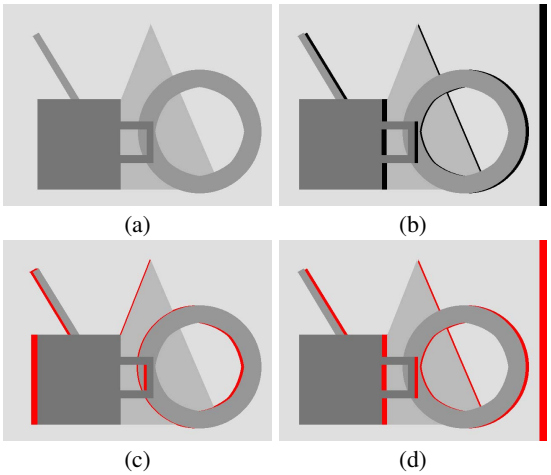


Fig. 3. A synthetic example for validating our outlier detection algorithm. (a) A depth map of a synthetic scene. (b) A depth map of the same scene with a translated view point. (c) Detected occlusion map for the view point change. (d) Detected disocclusion map of (b).

where $\alpha = \{\alpha_i\}$ is a set of mapping coefficients that defines the warping of 3D space, $R(r) = r^2 \log r$ is the radial basis kernel of the thin-plate spline, and $\{i, j\}$ are index of correspondents $\{\mathrm{x}'_i = [s\mathbf{x}_c \;\; d]^\top, \mathrm{x}_i = [\mathbf{x}_t \;\; d]^\top\}$ given by Eq. (1). Thus, for any 3D point, x, the new location after warping is given by:

$$\sum_i \alpha_i R(\mathrm{x} - \mathrm{x}_i) \tag{3}$$

With the thin-plate spline regularization, the entire point cloud is warped as a 3D volume. This can effectively suppress individual mapping errors caused by noisy estimation of depth.

D. Outliers Detection

Since there is disparity between the depth camera and the RGB camera, occlusion/disocclusion occurs along depth discontinuities. In addition, we found that the depth map from the 3D-ToF camera contains depth edges that are blurred by mixing the depth values of two different depth layers along depth boundaries[1]. These blurred depth boundaries are unreliable and should be removed before our optimization framework.

For each depth sample in the depth image, we reproject its location to the RGB image using the method described in Sec. III-C. The reprojected depth map allows us to determine occluded and disoccluded regions. We expand the detected occluded and disoccluded regions such that the unreliable depth samples along depth discontinuities are rejected. To demonstrate the idea, we show a synthetic example in Fig. 3. Fig. 3 (a) shows the depth map of the synthetic scene and Fig. 3 (b) show the reprojected depth map. The occluded regions and disoccluded regions are detected and shown in Fig. 3 (c) and Fig. 3 (d) respectively. Since our optimization method handles both depth upsampling and depth completion simultaneously in the same framework, the rejected depth samples along depth boundaries will not degrade the performance of upsampling. Instead, we found that our framework performs worse if the outliers were not rejected before optimization framework.

[1]In the case of the Kinect, unreliable depth samples were already rejected leaving holes in the captured depth map.

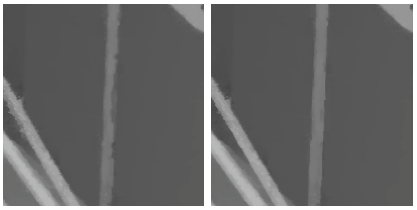


Fig. 4. Comparison of our result without (a) and with (b) the NLS term. The same weighting scheme proposed in Sec. IV-B is used for both (a) and (b). Although the usage of NLS does not significantly affect the RMS error, it is important in generating high quality depth maps especially along thin structure elements.

IV. Optimization Framework

This section describes our optimization framework for upsampling the low-resolution depth map given the aligned sparse depth samples and the high-resolution RGB image. Note that this framework is also applied to depth map completion. Similar to the previous image fusion approaches [5], [29], [6], we assume the co-occurrences of depth boundaries and image boundaries.

A. Objective Function

We define the objective function for depth map upsampling and completion as follows:

$$E(\mathbf{D}) = E_{\mathbf{d}}(\mathbf{D}) + \lambda_{\mathbf{s}} E_{\mathbf{s}}(\mathbf{D}) + \lambda_{\mathbf{N}} E_{\mathrm{NLS}}(\mathbf{D}) \tag{4}$$

where $E_{\mathbf{d}}(\mathbf{D})$ is the data term, $E_{\mathbf{s}}(\mathbf{D})$ is the neighborhood smoothness term, and $E_{\mathrm{NLS}}(\mathbf{D})$ is a NLS regularization. The term $\lambda_{\mathbf{s}}$ and $\lambda_{\mathbf{N}}$ are the relative weights to balance the three terms. Note that the smoothness term and NLS terms could be combined into a single term, however, we keep them separate here for sake of clarity.

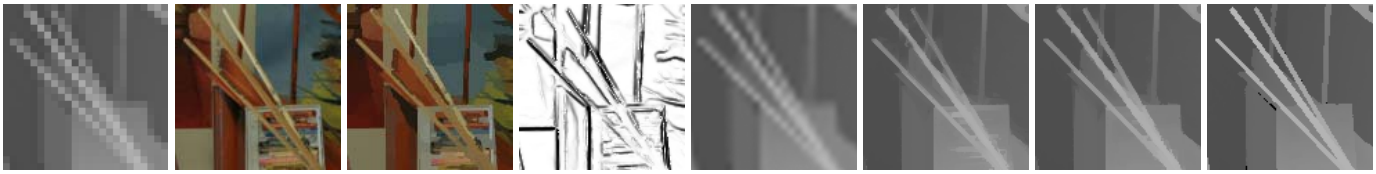
Our data term is defined according to the initial sparse depth map:

$$E_{\mathbf{d}}(\mathbf{D}) = \sum_{p \in \mathcal{G}} \left(\mathbf{D}(p) - \mathbf{G}(p)\right)^2, \tag{5}$$

where $\mathcal{G}$ is a set of pixels, which has the initial depth value, $\mathbf{G}(p)$. Our smoothness term is defined as:

$$E_{\mathbf{s}}(\mathbf{D}) = \sum_p \sum_{q \in \mathcal{N}(p)} \frac{w_{pq}}{\mathrm{W}_p} \left(\mathbf{D}(p) - \mathbf{D}(q)\right)^2, \tag{6}$$

where $\mathcal{N}(p)$ is the first order neighborhood of p, w_{pq} is the confidence weighting that will be detailed in the following section, and $\mathrm{W}_p = \sum_q w_{pq}$ is a normalization factor. Combining Eq. (5) and Eq. (6) forms a quadratic objective function which is similar to the objective function in [18]. The work in [18] was designed to propagate sparse color values to a gray high-resolution image that is similar in nature to our problem of propagating sparse depth values to the high-resolution RGB image.



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 5. (a) Low-resolution depth map (enlarged using nearest neighbor upsampling). (b) High-resolution RGB image. (c) Color segmentation by [27]. (d) Edge saliency map. (e) Guided depth map by using bicubic interpolation of (a). (f) Our upsampling result without the guided depth map weighting, depth bleeding occurred in highly textured regions. (g) Our upsampling result with guided depth map weighting. (h) Ground truth. We subsampled the depth value of a dataset from Middlebury to create the synthetic low-resolution depth map. The magnification factor in this example is $5\times$. The sum of squared difference(SSD) between (f) and (g) comparing to the ground truth are 31.66 and 24.62 respectively. Note that the depth bleeding problem in highly textured regions has been improved.

The difference between our method and that of [18] is the definition of w_{pq}. Work in [18] defined w_{pq} using intensity difference between the first order neighborhood pixels to preserve discontinuities. We further combine segmentation, color information, and edge saliency as well as the bicubic upsampled depth map to define w_{pq}. The reason for this is that we find the first order neighborhood does not properly consider the image structure. As the result, propagated color information in [18] was often prone to bleeding errors near fine detail. In addition, we include a NLS regularization term that helps to preserve thin structures by allowing the pixels on the same nonlocal structure to reinforce with each other within a larger neighborhood. We define the NLS regularization term using an anisotropic structural-aware filter [4]:

$$E_{\text{NLS}}(\mathbf{D}) = \sum_{p} \sum_{r \in \mathcal{A}(p)} \frac{\kappa_{pr}}{\text{K}_p} \left(\mathbf{D}(p) - \mathbf{D}(r)\right)^2, \qquad (7)$$

where $\mathcal{A}(p)$ is a local window (e.g. 11×11) in the high-resolution image, κ_{pr} is the weight of the anisotropic structural-aware filter, and $\text{K}_p = \sum_r \kappa_{pr}$ is normalization constant. κ_{pr} is defined as:

$$\begin{aligned} \kappa_{pr} &= \frac{1}{2}\left(\exp(-(p-r)^\top \Sigma_p^{-1}(p-r)) + \right. \\ &\quad \left. \exp(-(p-r)^\top \Sigma_r^{-1}(p-r))\right), \\ \Sigma_p &= \frac{1}{|\mathcal{A}|} \sum_{p' \in \mathcal{A}(p)} \nabla I(p') \nabla I(p')^\top. \end{aligned} \qquad (8)$$

Here, $\nabla I(p) = \{\nabla_x I(p), \nabla_y I(p)\}^\top$ is the x- and y- image gradient vector at p, and I is the high-resolution color image. The term Σ_q is defined similarly to Σ_p. This anisotropic structural-aware filter defines how likely p and q are on the same structure in the high-resolution RGB image, i.e. if p and r are on the same structure, κ_{pr} will be large. This NLS filter essential allows similar pixel to reinforce each other even if they are not first-order neighbors. To maintain the sparsity of the linear system, we remove neighborhood entries with $\kappa_{pr} < t$. A comparison of our approach to illustrate the effectiveness of the NLS regularization is shown in Fig. 4.

B. Confidence Weighting

In the section, we describe our confidence weighting scheme for defining the weights w_{pq} in Eq. (6). The value of w_{pq} defines the spatial coherence of neighborhood pixels. The larger w_{pq} is, the more likely that the two neighborhood pixels having the same depth value. Our confidence weighting is decomposed into four terms based on color similarities (w_c), segmentation (w_s), edge saliency (w_e), and guided bicubic interpolated depth map (w_d).

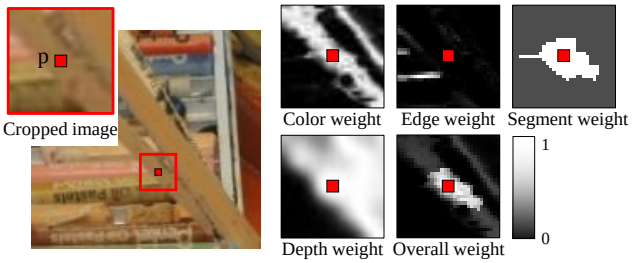


Fig. 6. Visualization of different weighting terms and the overall weighting term.

The color similarity term is defined in the YUV color space as follows:

$$w_c = \exp -\left(\sum_{I \in YUV} \frac{(\mathbf{I}(p) - \mathbf{I}(q))^2}{2\sigma_{\mathbf{I}}^2}\right), \qquad (9)$$

where $\sigma_{\mathbf{I}}$ controls the relative sensitivity of the different color channels.

Our second term is defined based on color segmentation using the library provided in [27] to segment an image into super pixels as shown in Fig. 5(c). For the neighborhood pixels that are not within the same super pixel, we give a penalty term defined as:

$$w_s = \begin{cases} 1 & \text{if } \mathbf{S}_{\text{CO}}(p) = \mathbf{S}_{\text{CO}}(q) \\ t_{\text{SE}} & \text{otherwise,} \end{cases} \qquad (10)$$

where $\mathbf{S}_{\text{CO}}(\cdot)$ is the segmentation label, t_{SE} is the penalty factor with its value between 0 and 1. In our implementation, we empirically set it equals to 0.7.

Inspired by [2], we have also included a weight that depends on the edge saliency response. Different from the color similarity term, the edge saliency responses are detected by a set of Gabor filters with different sizes and orientations.[2] The edge saliency map contains image structures rather than just color differences between neighborhood pixels. We combine the responses of different Gabor filters to form the edge saliency

[2]We use two scale 7×7, and 15×15 filter kernels with 8 orientations.

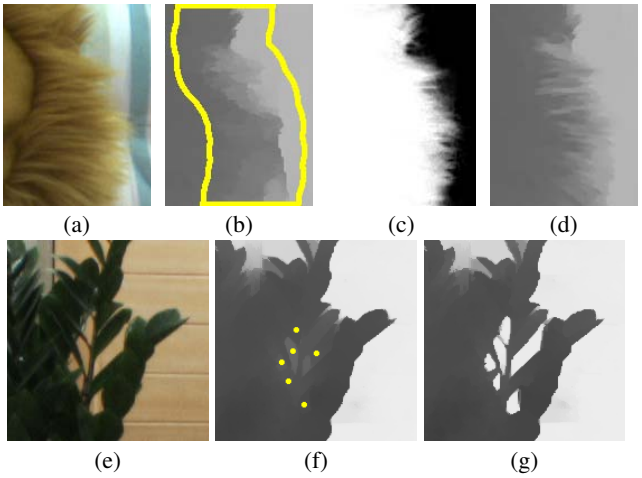


Fig. 7. Depth map refinement via user markup. (a)(e) Color image of small scale structure. (b)(f) Upsampled depth map before user correction. The user scribble areas in (b), and the user added depth samples in (e) are indicated by the yellow lines and dots respectively. (c) The extracted alpha mattes for depth refinement. (d)(g) Refined depth maps.

map as shown in Fig. 5(d). Our weighting is computed as:

$$w_e = \frac{1}{\sqrt{s_x(p)^2 + s_x(q)^2 + 1}}, \tag{11}$$

where $s_x(\cdot)$ is the value of $x-$axis edge saliency map if p and q are $x-$axis neighborhoods. The edge saliency enhances smoothness of depth boundary along edge.

Allowing the depth values to propagate freely with only very sparse data constraint can lead to notable depth bleeding. Here, we introduce the guided depth map to resolve this problem. The guided depth map weighting is similar to the intensity weighting in bilateral filter. Since we do not have a depth sample at each high-resolution pixel location, we use bicubic interpolation to obtain the guided depth map, $\mathbf{D}_g$, as shown in Fig. 5(e). Similar to the bilateral filter, we define the guided depth map weighting as follow:

$$w_d = \exp -(\frac{(\mathbf{D}_g(p) - \mathbf{D}_g(q))^2}{2\sigma_g^2}), \tag{12}$$

Combining the weight defined from Eq. (9) to Eq. (11) by multiplication, we obtain the weight $w_{pq} = w_s w_c w_e w_d$. Note that except for the edge saliency term, all the weighting defined in this subsection can be applied to the weighting κ_{pq} via multiplication to the NLS regularization term. Figure 6 illustrates the effects of each weighting term and the combined weighting term.

C. User Adjustments

Since the goal is high-quality depth refinement, there may be cases that some depth frames are going to require user touch up, especially if the data is intended for media related applications. Our approach allows easy user corrections by direct manipulation of the weighting term w_{pq} or by adding additional sparse depth sampling for error corrections.

For the manipulation of the weighting term, we allow the user to draw scribbles along fuzzy image boundaries, or along the boundaries where the image contrast is low. These fuzzy boundaries or low contrast boundaries represent difficult regions for segmentation and edge saliency detection. As a result, they cause depth bleeding in the reconstructed high-resolution depth map as illustrated in Fig. 7(b). Within the scribble areas, we compute an alpha matte based on the work by Wang et al. [28] for the two different depth layers. An additional weighting term will be added according to the estimated alpha values within the scribble areas. For the two pixels p and q within the scribble areas, if they belong to the same depth layer, they should have the same or similar alpha value. Hence, our additional weighting term for counting the additional depth discontinuity information is defined as:

$$\exp -(\frac{(\alpha(p) - \alpha(q))^2}{2\sigma_\alpha^2}), \tag{13}$$

where $\alpha(\cdot)$ is the estimated alpha values within the scribble areas. Figure 7(d) shows the effect after adding this alpha weighting term. The scribble areas are indicated by the yellow lines in Fig. 7(b) and the corresponding alpha matte is shown in Fig. 7(c).

Our second type of user correction allows the user to draw or remove depth samples on the high-resolution depth map directly. When adding a depth sample, the user can simply pick a depth value from the computed depth map and then assign this depth value to locations where depth samples are "missing". After adding the additional depth samples, our algorithm generates the new depth map using the new depth samples as a hard constraint in Equation (4). The second row of Fig. 7 shows an example of this user correction. Note that for image filtering techniques, such depth sample correction can be more complicated to incorporate since the effect of new depth samples can be filtered by the original depth sample within a large local neighborhood. Removal of depth samples can also cause a hole in the result of image filtering techniques.

D. Evaluation on the Weighting Terms

Our weighting term, w_{pq}, is a combination of several heuristic weighting terms. Here we provide some insight to the relative effectiveness of each individual weighting term and their combined effect as shown in Fig. 8 (f). Our experiments found that using only the color similar term can still cause propagation errors. The edge cue is more effective in preserving structure along edge boundary, but cannot entirely remove propagation errors. The effect of the segmentation cue is similar to the color cue as the segmentation is also based on color information, but generally produces sharper boundary with piecewise smoothed depth inside each segment than simply using the color cue. The depth cue[3] is good in avoiding propagation bleeding, but it is not effective along the depth boundaries, which do not utilize the co-occurrence of image edges and depth edges. After combining the four different cues together, the combined weighting scheme shows the best results. The results produced with the combined weighting term can effectively utilize the structures in the

[3]The depth cue produces better results than other cues in this synthetic examples because the depth cues have no noise and no missing values. It is blurry because it is from a low resolution depth map.

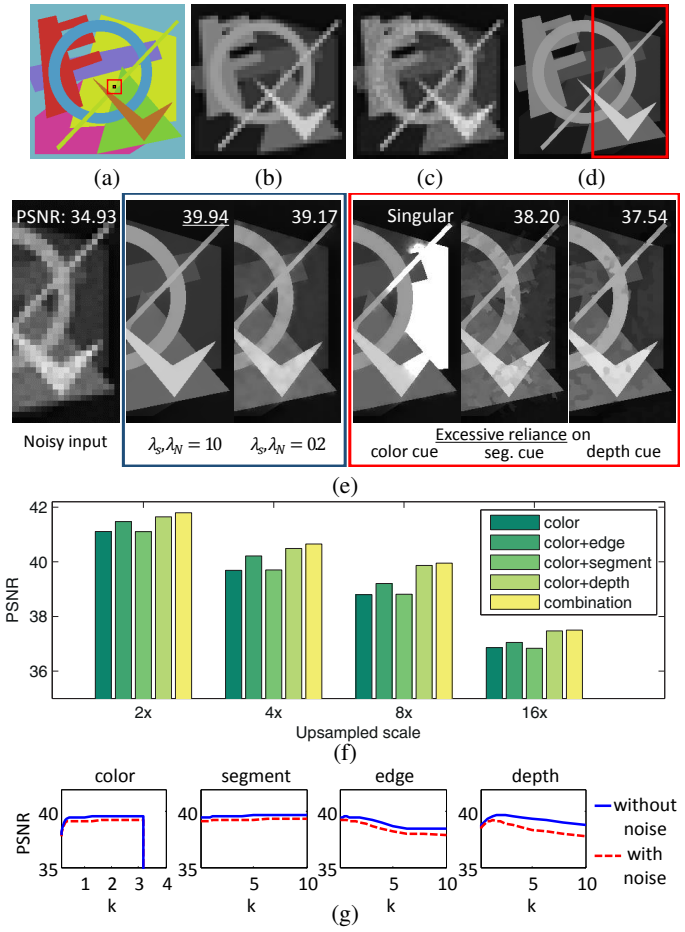


Fig. 8. A synthetic example for self-evaluation of our weighting term. (a)(b) A synthetic image pair consisting of a high-resolution color image and low-resolution depth image. (c) A low-resolution depth map with noise. (d) Our 8× upsampled depth map with the combined weighting term. (e) A zoomed-up images of upsampling results for various parameter configurations. (f) The plot of PSNR accuracy against the results with the combined weighting term and the results with additional weighting term. The combined weighting term consistently produce the best results under different upsampling scales. (g) In 8× upsampling, the four weights are manipulated individually and its corresponding accuracies are displayed in PSNR. Larger k implies giving more reliance to the corresponding weighting term.

high-resolution RGB image while it can avoid bleeding by including the depth cue which consists with the low-resolution depth map.

To further justify the relative effectiveness of our weight terms, we manipulate the weight values w_c, w_s, w_e and w_d individually. Since the weight terms are combined by a multiplication, simply multiplying an individual weight term by a constant factor does not affect the relative effectiveness. Therefore, we apply an exponential function $f(w,k) = w^k$. If one of the weights is mapped as $w' = f(w,k)$ where $0 < k < 1$, the relative importance of this term decreases since w' is likely to be 1. In contrast, If $k \gg 1$, the weight term dominates the overall weight. $k = 0$ means that the corresponding weight term is not used and $k = 1$ means no manipulation (equivalent to our empirical choice). Figure 8 (g) shows the evaluation results on the four weight terms. The graphs reach the maximum accuracy around $k = 1$, which implies our empirical choice is reasonable. Figure 8 (g) also

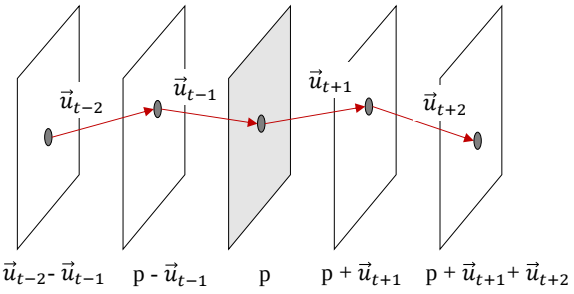


Fig. 9. Long range tracking of depth samples. From the dense flow fields $\vec{u}$ which is acquired from sequential color image pairs, we accumulate $\vec{u}$ to get correspondences from distant frames.

shows that the choice of $t_{\mathbf{se}}$ do not significantly alters the accuracy and the PSNR drops in most cases when $k \gg 1$. This implies that excessive reliance on a specific weight term is undesirable. For the color cue, when $k > 3$, the linear system[4] become singular and the solution is invalid.

Regarding $\lambda_{\mathbf{s}}$ and $\lambda_{\mathbf{N}}$ defined in Eq. (4), we find out that varying ratio between the two values shows subtle changes in PSNR. Instead, we observe that the magnitude of $\lambda_{\mathbf{s}}$ and $\lambda_{\mathbf{N}}$ is significant to the processed depth map accuracy. Figure 8 (e) shows an example of choosing $\lambda_{\mathbf{s}}$ and $\lambda_{\mathbf{N}}$ for the noisy depth map. When there is high-level of noise, larger lambda is preferable since the optimization is less stick to the input depth. However, if it is too large, the output can be over-smoothed. Throughout this paper, we choose $\lambda_{\mathbf{s}} = \lambda_{\mathbf{N}} = 10$ for noisy depth input and $\lambda_{\mathbf{s}} = \lambda_{\mathbf{N}} = 0.2$ for high-fidelity depth input.

E. Extension to RGB-D Video

In this section, we extend our framework to handle RGB-D video depth completion. A major difference between single image depth completion and video depth completion is that video depth completion requires temporal coherence. To achieve temporal coherent depth completion, we substitute our Eq. (5) as

$$E_{\mathbf{d}}(\mathbf{D}) = \sum_{p} \left(\mathbf{D}(p) - \mathbf{G}'(p)\right)^2, \tag{14}$$

where $\mathbf{G}'(p)$ is median value of depth from temporal correspondences as illustrated in Fig. 9.

We find the temporal correspondence of depth value by finding dense temporal correspondence across the RGB video. In our implementation, we use the optical flow algorithm by Liu [19]. We assume that the depth sensor does not move drastically as well as the motion of moving object inside a scene. From the computed dense optical flows, we trace the depth samples in $[t-3,t+3]$ frames. Since modern depth sensors can record approximately 30 frames per second, the period within $[t-3,t+3]$ corresponds to 0.2 seconds. After collecting the multiple correspondences of $\mathbf{D}(p)$ in $[t-3,t+3]$ frames, we compute the median depth values and use it as $\mathbf{G}'(p)$ in Eq. (14). This gives us a reliable and temporally consistent data term. For the special case such as Kinect,

[4]We optimize overall energy by solving a linear equation. Details are discussed in Sec. IV-F.

whose depth maps show consistent holes over the frames, we apply our algorithm on each frames independently for hole filling and take median depth value of tracked points over $[t-3,t+3]$ frames.

Section V-C demonstrates the effectiveness of our video depth completion extension versus the results of using the original frame-by-frame depth completion.

F. Efficient Energy Optimization

We now describe our approach to optimize Eq. (4). To optimize our quadratic energy function, similar to the work in Diebel et al. [5], we can first register the low-resolution depth map with the color image and interpolate the depth values as an initialization (described in Sec. III-C). After that, we apply an iterative convex optimization technique. Specifically, if we take the first derivative w.r.t. $\mathbf{D}$ on Eq. (4) and set the derivative equal to zero, i.e. $\frac{\Delta E(\mathbf{D})}{\Delta \mathbf{D}} = \mathbf{0}$, we get:

$$\mathrm{A}\mathbf{d} = \mathbf{g}, \tag{15}$$

where A is a $n \times n$ Laplacian matrix with weight terms, n is number of pixels in RGB domain, $\mathbf{d}$ is the desired depth values, and $\mathbf{g}$ is observed depth which is conditionally filled with measured depth $\mathbf{G}$. For $\mathbf{D}(p)$, the elements of A and $\mathbf{g}$ are filled as:

$$\mathrm{A}_{pp} = \begin{cases} 1+\lambda_{\mathbf{s}}+\lambda_{\mathbf{N}} & \text{if } p \in \mathcal{G} \\ \lambda_{\mathbf{s}}+\lambda_{\mathbf{N}} & \text{otherwise,} \end{cases} \tag{16}$$

$$\mathrm{A}_{pq} = -\frac{\lambda_{\mathbf{s}} w_{pq}}{\mathrm{W}_p}, \tag{17}$$

$$\mathrm{A}_{pr} = -\frac{\lambda_{\mathbf{N}} \kappa_{pr}}{\mathrm{K}_p}, \tag{18}$$

$$\mathbf{g}_p = \begin{cases} \mathbf{G}(p) & \text{if } p \in \mathcal{G} \\ 0 & \text{otherwise,} \end{cases} \tag{19}$$

where $q \in \mathcal{N}(p)$ and $r \in \mathcal{A}(p)$ indicate neighborhoods for smoothness term and NLS regularization term respectively, A_{pq} indicates A's element at p-th row and q-th column, and $\mathbf{g}_p$ indicates p-th element of vector $\mathbf{g}$. In the RGB-D video case, $\mathbf{G}'(p)$ is used instead of $\mathbf{G}(p)$. To solve Eq. (15), we applied the built-in linear solver in Matlab^{TM} 2009b, or known as a backslash operator '\'.

V. Results and Comparisons

We demonstrate our approach applied to both depth map upsampling and depth map completion. For depth map upsampling, we present quantitative evaluation and real world examples. For depth map completion, we refine the raw Kinect depth which exhibits missing regions (i.e. holes). The system configuration for experiments is 3Ghz CPU, 8GB RAM. The computation time is summarized in Table III.

A. Depth Map Upsampling

1) Evaluations using the Middlebury stereo dataset: We use synthetic examples for quantitative comparisons with the results from previous approaches [5], [29], [9]. The depth map from the Middlebury stereo datasets [23] are used as the ground truth. We downsampled the ground truth depth map by different factors to create the low-resolution depth map. The original color image is used as the high-resolution RGB image. We compare our results with bilinear interpolation, MRF [5], bilateral filter [29], and a recent work on guided image filter [9]. Since the previous approaches do not contain a user correction step, the results generated by our method for these synthetic examples are all based on our *automatic* method in Sec. IV-A and Sec. IV-B for fair comparisons. Table I summaries the RMSE (root-mean-square error) against the ground truth under different magnification factors for different testing examples. To demonstrate the advantages of our combined weight, $w_{pq} = w_s w_c w_e w_d$, defined in Sec. IV-B, we have also applied our algorithm using each weight independently. As shown in Table I, our combined weight consistently achieves the lowest RMSE among all the test cases especially for large scale upsampling. The qualitative comparison with the results from [5] and [29] under $8\times$ magnification factor can be found in Fig. 10.

Synthetic	Time(Sec.)	Real-world	Time(Sec.)
Art	21.60	Lion	18.60
Books	26.47	Office with person	16.65
Moebius	24.07	Lounge	18.28
		Classroom	19.00

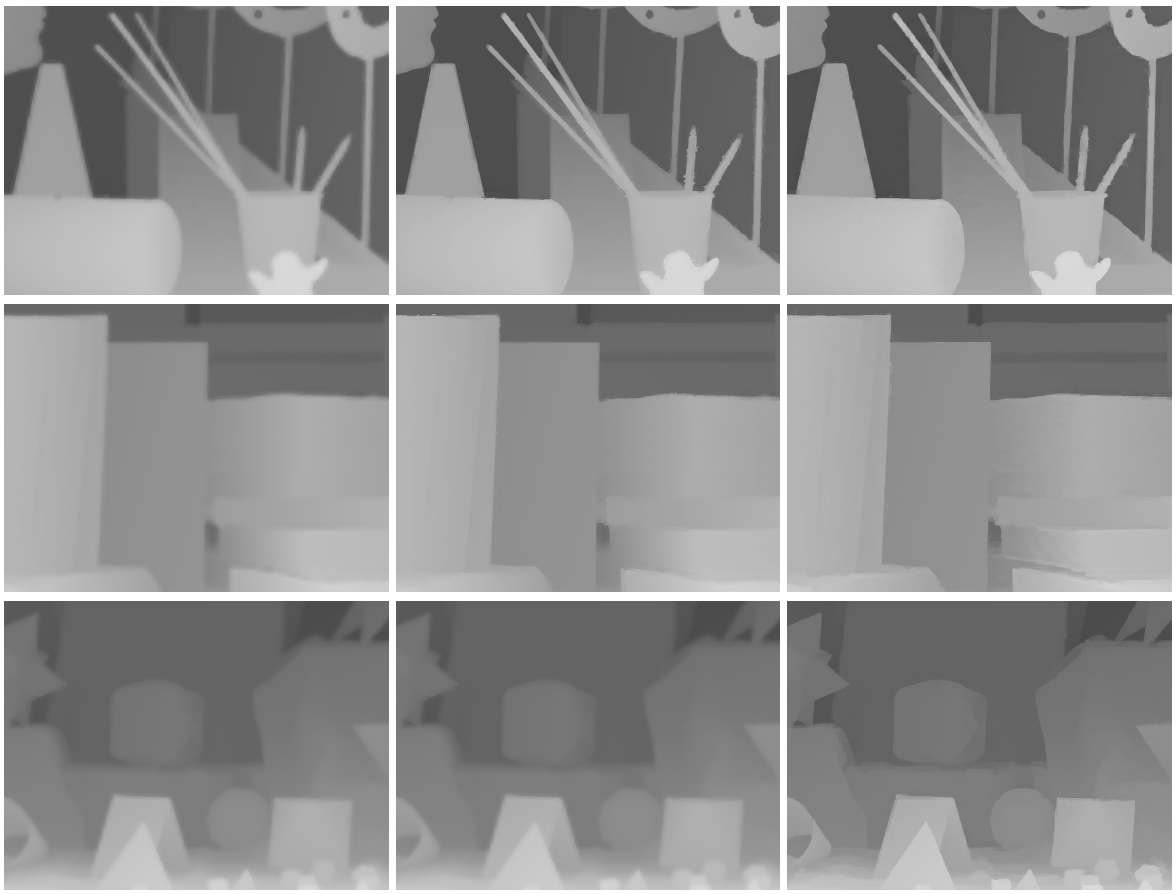
TABLE II
RUNNING TIME OF OUR ALGORITHM FOR 8X UPSAMPLING. THE UPSAMPLED DEPTH MAP RESOLUTION IS 1376×1088 FOR SYNTHETIC AND 1280×960 FOR REAL-WORLD EXAMPLES. THE ALGORITHM WAS IMPLEMENTED USING UNOPTIMIZED MATLAB CODE.

In terms of depth map quality, we found that the MRF method in [5] produces the most blurred result. This is due to its simple use of neighborhood term, which considers only the image intensity difference as the neighborhood similarity for depth propagation. The results from bilateral filtering in [29] are comparable to ours with sharp depth discontinuities in some of the test examples. However, since segmentation and edge saliency are not considered, their results can still suffer from depth bleeding highly textured regions. Also, we found that in the real world example in Fig. 1, the results from [29] tend to be blurry.

2) Robustness to Depth Noise: The depth map captured by depth cameras exhibit notable levels of noise. We compare the robustness of our algorithm and the previous algorithms by adding noise. We also compare against the Noise-Aware bilateral filter approach in [3]. We observe that the noise characteristics in depth camera depends on the distance between the camera and the scene. To simulate this effect, we add a conditional Gaussian noise:

$$p(\mathbf{x}, k, \sigma_d) = k \exp -(\frac{\mathbf{x}}{2(1+\sigma_d)^2}), \tag{20}$$

where σ_d is a value proportional to the depth value, and k is the magnitude of the Gaussian noise. Although the actual noise distribution of depth camera is more complicated than the Gaussian noise model, many previous depth map upsampling algorithms do not consider the problem of noise in the low-resolution depth map. This experiment therefore attempts an objective comparison on the robustness of different



(a) (b) (c)

Fig. 10. Qualitative comparison on Middlebury dataset. (a) MRFs optimzation [5]. (b) Bilateral filtering with subpixel refinement [29]. (c) Our results. The image resolution are enhanced by $8\times$. Note that we do not include any user correction in these synthetic testing cases. The results are cropped for the visualization, full resolution comparisons are provided in the supplemental materials.

	Art				Books				Moebius			
	$2\times$	$4\times$	$8\times$	$16\times$	$2\times$	$4\times$	$8\times$	$16\times$	$2\times$	$4\times$	$8\times$	$16\times$
Bilinear	0.56	1.09	2.10	4.03	0.19	0.35	0.65	1.24	0.20	0.37	0.70	1.32
MRFs [5]	0.62	1.01	1.97	3.94	0.22	0.33	0.62	1.21	0.25	0.37	0.67	1.29
Bilateral [29]	0.57	0.70	1.50	3.69	0.30	0.45	0.64	1.45	0.39	0.48	0.69	1.14
Guided [9]	0.66	1.06	1.77	3.63	0.22	0.36	0.60	1.16	0.24	0.38	0.61	1.20
Ours segment weight	0.60	1.12	2.54	4.01	0.21	0.38	0.79	1.28	0.23	0.42	0.92	1.47
Ours color weight	0.52	0.81	1.45	3.09	0.19	0.32	0.57	1.06	0.21	0.32	0.56	1.08
Ours edge weight	0.67	1.73	3.62	6.68	0.25	0.51	1.05	2.03	0.26	0.57	1.19	2.12
Ours depth weight	0.46	0.85	1.66	3.49	0.18	0.34	0.68	1.36	0.18	0.33	0.67	1.31
Ours combined weight	0.43	0.67	1.08	2.21	0.17	0.31	0.57	1.05	0.18	0.30	0.52	0.90

TABLE I

QUANTITATIVE COMPARISON ON MIDDLEBURY DATASET. THE ERROR IS MEASURED IN RMSE FOR 4 DIFFERENT MAGNIFICATION FACTORS. THE PERFORMANCE OF OUR ALGORITHM IS THE BEST AMONG ALL COMPARED ALGORITHM. NOTE THAT NO USER CORRECTION IS INCLUDED IN THESE SYNTHETIC TESTING EXAMPLES.

algorithms towards the effect of noisy depth map. The results in term of RMSE are summarized in Table II. We note that some of our results are worse than results from the previous methods, this is because the previous methods tends to over-smooth upsampled depth map and therefore have higher noise tolerance. Although we can adjust parameters to increase noise tolerance by increasing smoothness weight, we keep the same parameter setting as in Sec. V-A1 to provide a fair comparison.

3) ToF Depth Upsampling: Figure 13 shows upsampled ToF depth maps which is taken from real scenes. Since the goal of our paper is to obtain high quality depth maps, we include user corrections for the examples in the top and middle row. We show our upsampled depth as well as a novel view rendered by using our depth map. The magnification factors for all these examples are $8\times$. These real world examples are challenging with complicated boundaries and thin structures. Some of the objects contain almost identical colors but with different depth values. Our approach is successful in distinguishing the various depth layers with sharp boundaries.

	Art				Books				Moebius			
	2×	4×	8×	16×	2×	4×	8×	16×	2×	4×	8×	16×
Bilinear	3.09	3.59	4.39	5.91	2.91	3.12	3.34	3.71	3.21	3.45	3.62	4.00
MRFs [5]	1.62	2.54	3.85	5.70	1.34	2.08	2.85	3.54	1.47	2.29	3.09	3.81
Bilateral [29]	1.36	1.93	2.45	4.52	1.12	1.47	1.81	2.92	1.25	1.63	2.06	3.21
Guided [9]	1.92	2.40	3.32	5.08	1.60	1.82	2.31	3.06	1.77	2.03	2.60	3.34
NAFDU [3]	1.83	2.90	4.75	7.70	1.04	1.36	1.94	3.07	1.17	1.55	2.28	3.55
Ours	1.24	1.82	2.78	4.17	0.99	1.43	1.98	3.04	1.03	1.49	2.13	3.09

TABLE III
QUANTITATIVE COMPARISON ON MIDDLEBURY DATASET WITH ADDITIVE NOISE. OUR ALGORITHM ACHIEVES THE LOWEST RMSE IN MOST CASES. NOTE THAT ALL THESE RESULTS ARE GENERATED WITHOUT ANY USER CORRECTION. BETTER PERFORMANCE IS POSSIBLE AFTER INCLUDING USER CORRECTION.

B. Depth Map Completion

This section provides our depth map completion results on Kinect's raw depth data where the depth map and the RGB image are of the same resolution. For all depth map completion results in this section, we do not include any user markup or post-processing. Figure 11 compares our depth map completion results with results obtained using joint bilateral filtering [17] and colorization [18]. The scene in the first row is captured by our calibrated Kinect and the second scene is from the NYU RGB-D dataset [24]. The NYU RGB-D dataset provides RGB images, depth maps and also pre-calculated camera parameters for alignment. In both scenes, filter based result [17] in Fig. 11 (c) shows an over-smoothed depth map. Results from colorization [18] in Fig. 11(d) are very similar to our results in Fig. 11(e) as we both use optimization based method for depth map completion. However, upon careful comparisons in the highlighted regions, our method produces sharper depth boundaries and does not over-smooth fine structures.

Figure 14 shows more depth completion results using the NYU RGB-D dataset. The input depth map from the Kinect has many holes which are indicated as black. Our refined and completed depth maps are shown in Fig. 14(c), which well align with object boundaries in RGB image. Note that some holes are very big and the available depth samples within these holes are limited.

Compared to filter based methods [29], [9], [3], it is worth noting that our optimization based method does not need a manual filter size adjustment for completing large depth holes.

We have also evaluated the quality of our depth completion using "ground truth" depth maps captured using KinectFusion [12]. As previously discussed in Sec. II, KinectFusion integrates raw Kinect depth maps into a regular voxel grid. From this voxel grid we can construct a textured 3D mesh to serve as the ground truth depth maps. Since KinectFusion integrates noisy depth maps which are captured at various view points, compared to the single raw Kinect depth, the fused depth has less holes and less noise. In our evaluation, we applied our algorithm on the single raw Kinect depth with the RGB image guidance. The result in Fig. 12 shows that our depth completion result is visually similar to the KinectFusion depth. Especially in Fig. 12, we are able to complete thin and metal structure of the desk lamp whose depth measurement is not captured reliably from the raw Kinect depth map. In addition, the boundary regions and holes are also reasonably completed.

C. Video Depth Completion

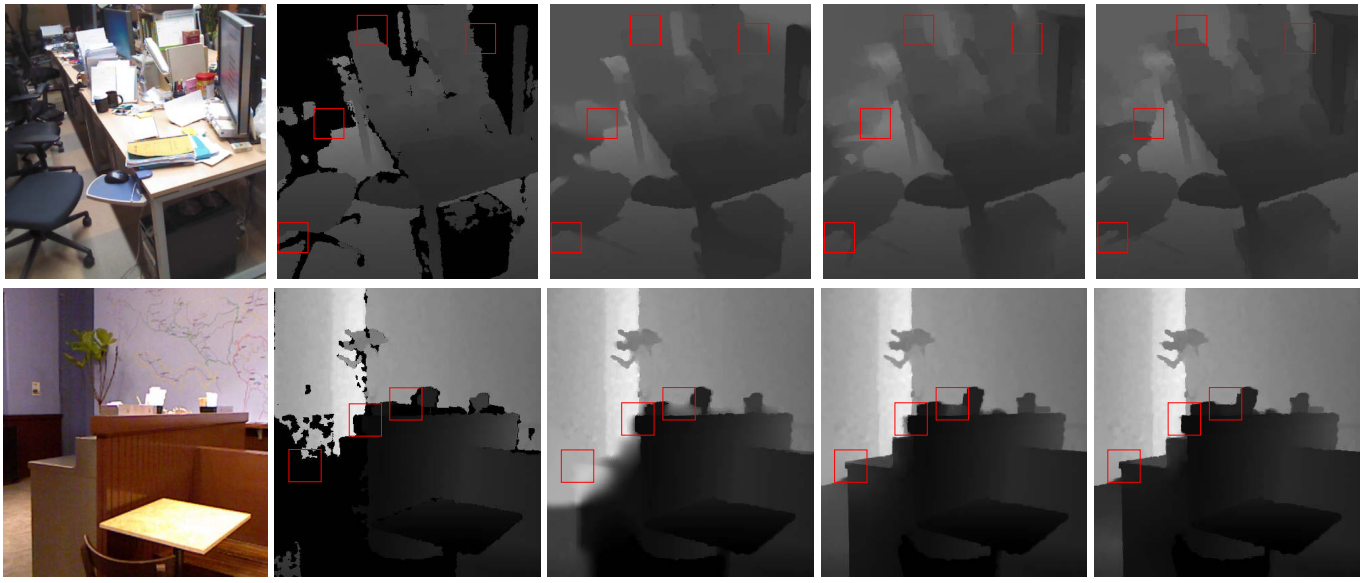
Finally, we show our video depth completion results in Fig. 15. In our experiments, we also use the NYU RGB-D dataset [24] since it is an public data set and it provides synchronized depth-color video pairs in 30 frame rates. As shown in Fig. 15, the depth map without temporal coherency flickers and has noticeable depth inconsistency. After using our approach described in Sec. IV-E, however, the artifacts are significantly reduced. In addition, the quality of the depth completion results in each individual frame improved.

VI. DISCUSSION AND SUMMARY

We have presented a framework to upsample a low-resolution depth from 3D-ToF camera and to repair raw depth maps from a Kinect using an auxiliary RGB image. Our framework is based on a least-square optimization that combines several weighting factors together with nonlocal structure filtering to maintain sharp depth boundaries and to prevent depth bleeding during propagation. Although the definitions of our weighting factors are heuristic, each of them serves different purposes to protect edge discontinuities and fine structures in the repaired depth maps. By combining the different weighting factors together, our approach achieved the best quality results comparing to individual usage of the weighting factors. Our experimental results show that this framework out-performs previous work in terms of both RMSE and visual quality. In addition to the automatic method, we have also discussed how to extend our approach to incorporate user markup. Our user correction method is simple and intuitive and does not require any addition modifications for solving the objective function defined in Sec. IV-A. Lastly, we described how to extend this framework to work on video input by introducing an additional data term to exploit temporal coherency.

ACKNOWLEDGEMENT

We are grateful to anonymous reviewers for their constructive comments. This research was supported by the National Strategic R&D Program for Industrial Technology, Korea (No. 10031903) and the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP)



(a) (b) (c) (d) (e)

Fig. 11. Depth map completion results. (a) RGB Images, (b) Registered raw depth maps from Kinect, Depth map completion using (c) Joint bilateral filter [17], (d) Colorization [18], and (e) Our method. Note the high lighted regions.

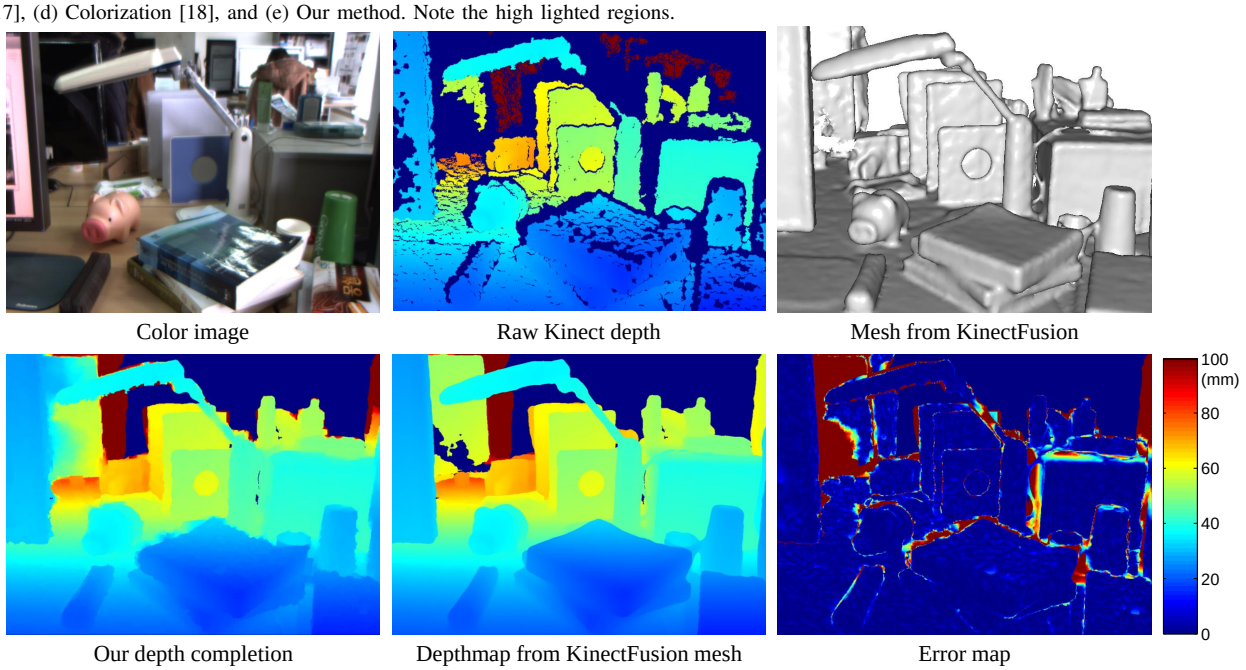


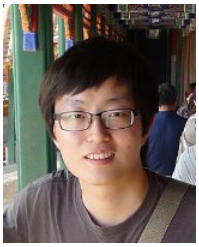
Fig. 12. Qualitative comparison of our depth completion result with KinectFusion [12]. From the 3D mesh of KinectFusion, we generated a depth map at the color image coordinate. Next, we apply our depth completion approach to the raw Kinect depth map. The two depth maps shows quite similar structure and similar depth values in various regions (especially a thin structure of the desk lamp and boundary regions of cups and holes in the books.

(No. 2010-0028680). Michael S. Brown was supported by A*STAR Science and Engineering Research Council, Public Sector Research Funding Grant (No. 1121202020).

REFERENCES

[1] SwissRanger^{TM} SR4000 data sheet, http://www.mesa-imaging.ch/prodview4k.php.
[2] P. Bhat, C. L. Zitnick, M. F. Cohen, and B. Curless. Gradientshop: A gradient-domain optimization framework for image and video filtering. *ACM Trans. Graph.*, 29(2), 2010.
[3] D. Chan, H. Buisman, C. Theobalt, and S. Thrun. A noise-aware filter for real-time depth upsampling. In *ECCV Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications*, 2008.
[4] J. Chen, C. Tang, and J. Wang. Noise brush: interactive high quality image-noise separation. *ACM Trans. Graphics*, 28(5), 2009.
[5] J. Diebel and S. Thrun. An application of markov random fields to range sensing. In *Proc. of Neural Information Processing System (NIPS)*, 2005.
[6] J. Dolson, J. Baek, C. Plagemann, and S. Thrun. Upsampling range data in dynamic environments. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2010.
[7] E. Eisemann and F. Durand. Flash photography enhancement via intrinsic relighting. *ACM Trans. Graphics*, 23(3):673–678, 2004.
[8] P. Favaro. Recovering thin structures via nonlocal-means regularization with application to depth from defocus. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2010.
[9] K. He, J. Sun, and X. Tang. Guided image filtering. In *Proc. of European Conf. on Computer Vision (ECCV)*, 2010.
[10] D. Herrera, J. Kannala, and J. Heikkilä. Joint depth and color camera calibration with distortion correction. *IEEE Trans. PAMI*, 2012.
[11] B. Huhle, T. Schairer, P. Jenke, and W. Straßer. Fusion of range and color images for denoising and resolution enhancement with a non-local filter. *Computer Vision and Image Understanding (CVIU)*, 114:1336–

1345, 2010.
[12] 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 *24th annual ACM symposium on User interface software and technology, ser. UIST '11*, pages 559–568, 2011.
[13] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3-d object dataset: Putting the kinect to work. In *In Proc. of Int'l Conf. on Computer Vision (ICCV)*, 2011.
[14] J. Jung, Y. Jeong, J. Park, H. Ha, J. D. Kim, and I. S. Kweon. A novel 2.5d pattern for extrinsic calibration of tof and camera fusion system. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011.
[15] J. Kennedy and R. Eberhart. Particle swam optimization. In *In Proc. of Int'l Conf. on Neural Networks*, pages 1942–1948, 1995.
[16] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12:1437–1454, 2012.
[17] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele. Joint bilateral upsampling. *ACM Trans. Graphics*, 26(3):96, 2007.
[18] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. *ACM Trans. Graphics*, 23(3):689–694, 2004.
[19] C. Liu. Beyond pixels: Exploring new representations and applications for motion analysis. *Doctoral Thesis. Massachusetts Institute of Technology*, May 2009.
[20] S. Meister, S. Izadi, P. Kohli, M. Hämmerle, C. Rother, and D. Kondermann. When can we use kinectfusion for ground truth acquisition? In *Workshop on Color-Depth Camera Fusion in Robotics, IROS*, 2012.
[21] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice. Automatic extrinsic calibration of vision and lidar by maximizing mutual information. *Journal of Field Robotics, Special Issue on Calibration for Field Robotics*, 2014. In Print.
[22] J. Park, H. Kim, Y.-W. Tai, M. Brown, and I. Kweon. High quality depth map upsampling for 3d-tof cameras. In *In Proc. of Int'l Conf. on Computer Vision (ICCV)*, 2011.
[23] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1/2/3):7–42, 2002.
[24] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *Proc. of European Conf. on Computer Vision (ECCV)*, 2012.
[25] Z. Taylor and J. Nieto. Automatic calibration of lidar and camera images using normalized mutual information. In *In Proc. of IEEE Int'l Conf. on Robotics and Automation*, 2013.
[26] A. Torralba and W. T. Freeman. Properties and applications of shape recipes. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 383–390, 2003.
[27] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.
[28] J. Wang, M. Agrawala, and M. Cohen. Soft scissors: An interactive tool for realtime high quality matting. *ACM Trans. Graphics*, 26(3), 2007.
[29] Q. Yang, R. Yang, J. Davis, and D. Nistér. Spatial-depth super resolution for range images. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2007.
[30] J. Zhu, L. Wang, R. Yang, and J. Davis. Fusion of time-of-flight depth and stereo for high accuracy depth maps. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2008.
[31] A. Zomet and S. Peleg. Multi-sensor super-resolution. In *Proceedings. Sixth IEEE Workshop on Applications of Computer Vision (WACV)*, pages 27–31, 2002.



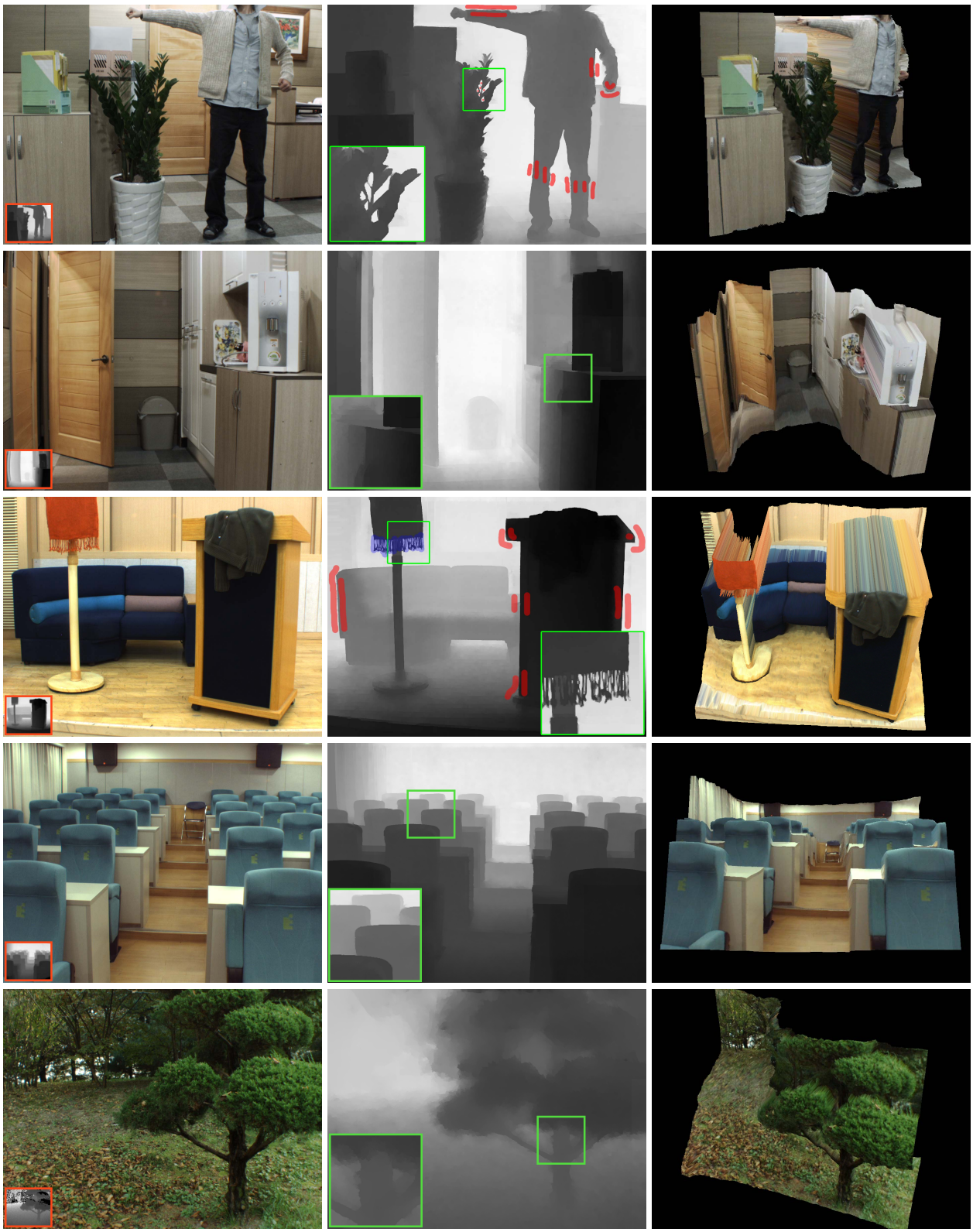
Jaesik Park received his Bachelor degree (Summa cum laude) in media communication engineering from Hanyang University in 2009 and his Master degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST) in 2011. He is currently pursuing his PhD degree in KAIST. He received the Microsoft Research Asia Fellowship in 2011. From April 2012 to Oct. 2012, he worked as a full time internship in Microsoft Research Asia (MSRA). His research interests include depth map refinement, rigid/non-rigid 3D reconstruction. He is a member of the IEEE.

Hyeongwoo Kim received his Bachelor degree in Electrical Engineering from Yonsei University in 2005 and his Master degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST) in 2007. He is currently pursuing his PhD degree in KAIST. From Sept. 2008 to Sept. 2009, he worked as a full time internship in Microsoft Research Asia (MSRA). From Jan. 2001 to June 2011, he worked as a visiting researcher at NASA. His research interests include photometric stereo and depth refinement.

Yu-Wing Tai received the BEng (first class honors) and MPhil degrees in compute science from the Hong Kong University of Science and Technology (HKUST) in 2003 and 2005 respectively, and the PhD degree from the National University of Singapore (NUS) in June 2009. He joined the Korea Advanced Institute of Science and Technology (KAIST) as an assistant professor in Fall 2009. He regularly serves on the program committees for the major Computer Vision conferences (ICCV, CVPR, and ECCV). His research interests include computer vision and image/video processing. He is a member of the IEEE and ACM.

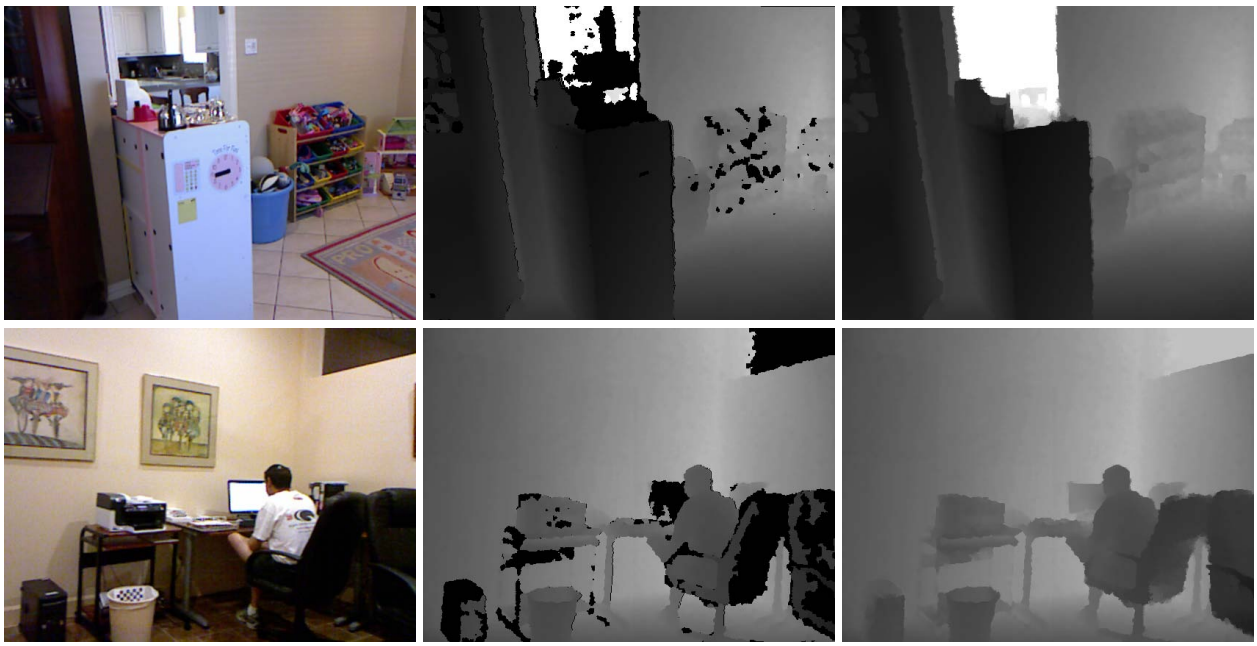
Michael S. Brown obtained his BS and PhD in Computer Science from the University of Kentucky in 1995 and 2001 respectively. He is currently an Associate Professor and Vice Dean (External Relations) in the School of Computing at the National University of Singapore. Dr. Brown's research interests include computer vision, image processing and computer graphics. He has served as an area chair for CVPR, ICCV, ECCV, and ACCV and is currently an associate editor for the IEEE Transactions on Pattern Analysis and Machine Intelligence. He is a member of the IEEE.

In So Kweon received the BS and MS degrees in mechanical design and production engineering from Seoul National University, Seoul, Korea, in 1981 and 1983, respectively, and the PhD degree in robotics from the Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, in 1990. He worked for the Toshiba R&D Center, Japan, and joined the Department of Automation and Design Engineering, KAIST, Seoul, Korea, in 1992, where he is now a professor with the Department of Electrical Engineering. He is a recipient of the best student paper runner-up award at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '09). His research interests are in camera and 3D sensor fusion, color modeling and analysis, visual tracking, and visual SLAM. He was the program co-chair for the Asian Conference on Computer Vision (ACCV '07) and was the general chair for the ACCV '12. He is also on the editorial board of the International Journal of Computer Vision. He is a member of the IEEE and the KROS.



(a) (b) (c)

Fig. 13. Depth map upsampling on ToF-RGB system. (a) Our input, the low-resolution depth maps are shown on the lower left corner (Ratio between the two images are preserved). (b) Our results. User scribble areas (blue) and the additional depth sample (red) were high-lighted. (c) Novel view rendering of our result. Note that no user markup is required in our results in the third row.



(a) (b) (c)

Fig. 14. Depth map completion examples using NYU RGB-D dataset [24]. (a) RGB Images. (b) Registered raw depth maps. (c) Our refinement. For these results no user markup is applied.

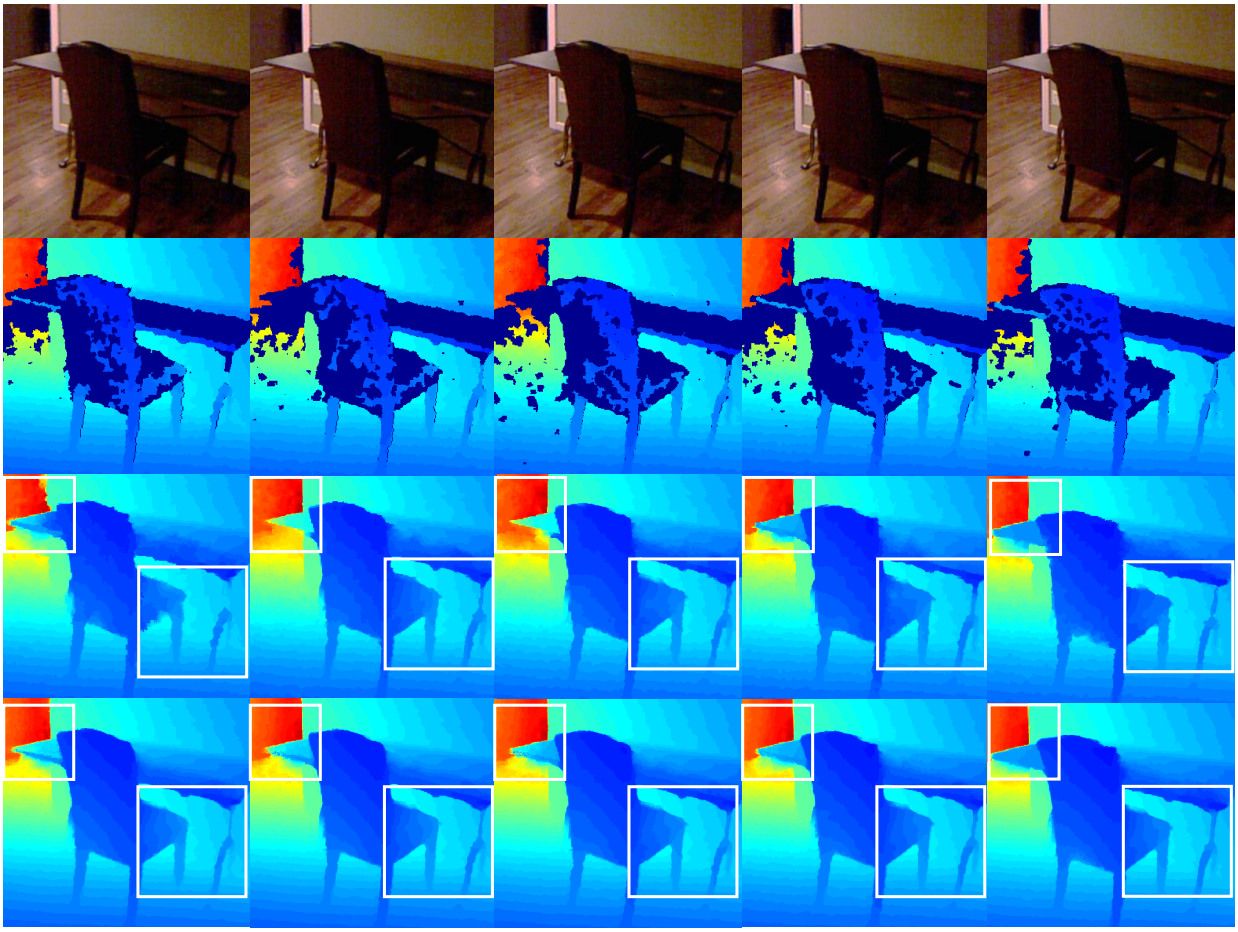


Fig. 15. Depth map video completion results on NYU RGB-D dataset [24]. Each rows show sequence of color images, raw Kinect depth, our depth completion without temporal coherency and with temporal coherency, which is described in Sec. IV-E. The white boxes highlight improvements when the temporal coherency is considered; the large holes in raw depth is reliably filled and the depth boundaries become stable.