JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 37, 1379-1403 (2021)
DOI: 10.6688/JISE.202111_37(6).0011

Fast Stereo Matching with Recursive Refinement and Depth Upsizing for Estimation of High Resolution Depth

WEI-JONG YANG[1], HSING-JU CHOU[1] AND PAU-CHOO CHUNG[2]
[1]*Institute of Computer and Communication Engineering*
[2]*Department of Electrical Engineering*
National Cheng Kung University
Tainan, 701 Taiwan
E-mail: weijongx@hotmail.com; ghost0473@gmail.com; pcchung@ee.ncku.edu.tw

In this paper, a fast and precise stereo matching system, which is mainly composed of adaptive stereo matching, recursive refinement and depth upsizing subsystems, is proposed to estimate high resolution depth maps. The adaptive stereo matching subsystem, which adaptively uses color gradient, color intensity and texture census costs, can reduce the most of computation and achieve good primary depth maps. The recursive refinement subsystem consists of median filtering, left-right check, hole filling, and bilateral filtering functions. The refinement subsystem is performed recursively to reduce the most errors occurred in occlusion regions such that we can obtain high-precision depth maps. Finally, the depth upsizing subsystem by referring original high-resolution images performs depth value upscaling, texture-selected interpolation, and weighted depth voting processes to finally obtain the high-precision and high-resolution depth maps. To evaluate the proposed algorithms in three subsystems, the performances of the proposed stereo matching system are exhibited in step-by-step approaches. In comparison of the existing methods, the accuracy performance and computation time of the proposed stereo matching system are finally demonstrated. For future 3D ultra-high-resolution videos, the proposed stereo matching system with low computation will be a good solution to extract high-precision and high-resolution depth information.

Keywords: computer vision, disparity estimation, high-resolution depth map, stereo image processing, stereo vision, recursive refinement

1. INTRODUCTION

Depth estimation, which is also known as disparity estimation, can be used to retrieve the distances of the objects from paired stereo images for many applications such as smart robotics [1, 2] self-driving vehicles [3, 4] and 3D data representation [4-7]. Thus, the disparity estimation is an important topic in computer vision. Especially for 3D video broadcasting systems, the multiple views including ultra-high definition (UHD) colour texture frames and their associated UHD depth maps are needed for 3D-HEVC coders [8, 9]. Generally, the stereo matching algorithms can be classified into three major categories: global, local and semi-global approaches. The global approach utilizes data term and smooth team to construct their energy functions to compute the optimal depth map globally. Graph-cut [10-12], belief propagation [13-16] and dynamic programming [17] are the typical global stereo matching algorithms. To reduce the computation, the local approach only examines a small region around the reference pixel to estimate the depth value through cost aggregation of neighbouring pixels. The local ste-

Received October 1, 2019; revised May 7 & August 19 & September 28, 2020; accepted November 6, 2020.
Communicated by Jen-Hui Chuang.

reo matching algorithms [18-20] need to balance cost functions and aggregation strategies to achieve the final depth estimation. To fasten the computation, the arrangement of parallel structures [21, 22] and the computation with GPU cores could be considered. Semi-global methods based on the local approach further jointly consider the estimated depths in the previous pixels to compute the accumulated costs [23-26] to achieve better results. Recently, the convolution neural networks (CNNs) [27-29] with deep learning methods won the best systems in several vision-based classification competitions. To compute the stereo matching cost, the first CNN model was effectively proposed for depth estimation [30]. Recently, several effective end-to-end CNN models [31-35] for stereo matching have been proposed. Based on the correlation between paired stereo images, the CNN-based depth estimation models are designed successfully. Generally, the CNN-based solutions limited by training dataset performs good only for certain data-dependent applications. Especially for general super resolution images, a fast stereo matching system, which can estimate high-precision and high-resolution depth information, is needed for future 3D video applications in cooperation with 3D-HEVC coding standard.

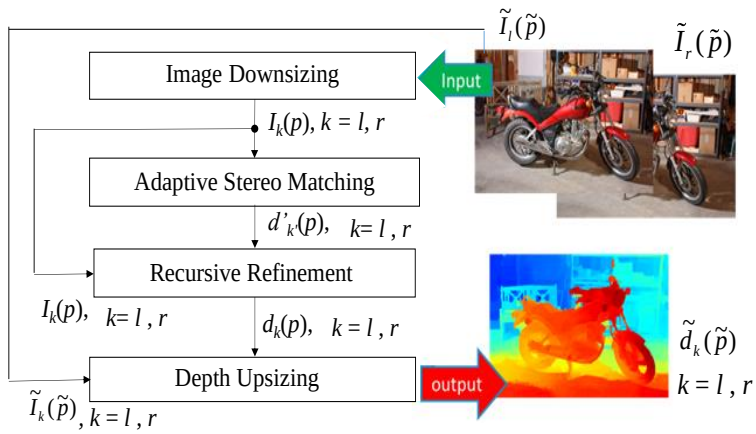


Fig. 1. Flow diagram of the proposed stereo matching system.

In order to estimate high-precision and high-resolution depth maps with low computational complexity, as shown in Fig. 1, we propose a fast and precise stereo matching system, which is composed of adaptive stereo matching, recursive refinement and depth upsizing subsystems. After downsizing images, the proposed adaptive stereo matching subsystem uses the adaptive costs to estimate the primary low-resolution depth maps. Then, the recursive refinement subsystem combines left right checking, median filtering and bilateral filtering functions to iteratively enhance the primary low-resolution depth maps to become high-precision depth maps. Finally, the depth upsizing subsystem utilizes depth rescaling, selected interpolations and depth voting procedures to further upsample high-precision depth maps to become high-precision and high-resolution depth maps. The details of these three proposed subsystems are described in Sections 2 and 3 while the simulations and discussions are exhibited in Section 4. Finally, the conclusions of this paper are addressed in Section 5.

2. ADAPTIVE STEREO MATCHING SUBSYSTEM

2.1 The Proposed System and its Notations

As shown in Fig. 1, the original UHD left and right images, $\tilde{I}_l(\tilde{p})$ and $\tilde{I}_r(\tilde{p})$ after image downsizing are denoted as the left and right downsized low-resolution (LR) images, $I_l(p)$ and $I_r(p)$, respectively. The positions in the original UHD and the downsized LR images are respectively set as $\tilde{p} = (\tilde{x}, \tilde{y})$ and $p = (x, y)$ while their estimated depth maps are respectively denoted as $\tilde{d}(\tilde{p})$ and $d(p)$ at $\tilde{p}$ and p positions. The subscripts r and l respectively represent the right and left indices of the images and depth maps. If the stereo matching method is directly conducted in the original UHD images, it will result in unbearably large computation for real-time applications. Thus, the proposed stereo matching system is first conducted in the LR images to reduce the computation while the precision and the resolution will be recovered by the recursive refinement and depth upsizing subsystems, respectively.

2.2 Adaptive Stereo Matching Cost

With the left and right downsized LR images, we design an adaptive cost function to characterize the matching level between the reference and target pixels. For simplicity, in this paper, we only set the left image as the reference image to estimate the left depth map. To estimate the right depth map, we can simply exchange right to left indices l and r. If the reference pixel at p in the left image and the target pixel at p_d with the disparity d in the right image are closer, the cost function $C(p, d)$ should be smaller, where $p_d = (x + d, y)$.

The proposed adaptive cost function, which combines color gradient, color intensity and texture costs, is expressed as

$$\begin{aligned} C(p, d) = {} & w_\alpha(J_s(p)\phi(C_a(p, d), \alpha_a)) + (1 - J_s(p))\phi(C_a(p, d), \alpha_a)) \\ & + (1 - w_\alpha)\phi(C_c(p, d), \alpha_c), \end{aligned} \tag{1}$$

where $C_a(p, d)$ represents the color intensity cost defined by:

$$C_a(p,d) = \left(\frac{1}{3} \sum_{i \in \{B,G,R\}} (I_l^{(i)}(p) - I_r^{(i)}(p_d))^2 \right)^{1/2}, \tag{2}$$

where the superscript, i denotes the color channel index, *i.e.*, *R*, *G*, or *B*. In Eq. (1), $C_g(p, d)$ denotes the color gradient cost defined as

$$C_g(p,d) = \left(\frac{1}{3} \sum_{i \in \{B,G,R\}} (g_l^{(i)}(p) - g_r^{(i)}(p_d))^2 \right)^{1/2} \tag{3}$$

with $g_l^{(i)}(p)$ and $g_r^{(i)}(p_d)$ respectively denoting the gradients of $I_l^{(i)}(p)$ and $I_r^{(i)}(p_d)$ computed by horizontal Sobel's operator. Finally, $C_c(p, d)$ indicates the texture cost expressed by the census transform as

$$C_c(p.d) = \sum_i^N B_l(p) \oplus B_r(p_d), \tag{4}$$

where $B_l(p)$ and $B_r(p_d)$ are bit streams, which are formed by N binarized surrounding pixels with the threshold of the pixel at p in the left LR image and that of the pixel at p_d in the right LR image. The operator, "⊕" denotes "eXclusive-OR" to compute Hamming distance of two bit streams. In Eq. (1), $\varphi(C, \alpha)$ is a normalization function, which maps the cost C to the range of 0 and 1 by a factor α, expressed as,

$$\varphi(C, \alpha) = 1 - ((C/\alpha) + 1)^{-1}. \tag{5}$$

In Eq. (1), the gradient flatness indication J_s is defined as

$$J_s(p) = \begin{cases} 1, & \sum_{p_i \in S(p)} |g_{\max}(p_i)| > \tau_2 \\ 0, & \text{otherwise} \end{cases}, \tag{6}$$

where the maximum of color gradients is given as

$$g_{\max}(p_l) = \max_{i \in \{R,G,B\}} g_l^{(i)}(p_t). \tag{7}$$

In Eq. (6), the flatness is determined by the sum of $g_{\max}(p_l)$ in the region defined by

$$S(p) = \{p_l \mid |g_{\max}(p_l)| < \tau_1 \text{ and } |p - p_l| \le L_1\}, \tag{8}$$

which is horizontally-extended of the target pixel between $+L_1$ and $-L_1$. In Eq. (6), if the sum of absolute gradients is larger than τ_2, $J_s(p)$ =1 such that the adaptive cost in Eq. (1) with the color gradient cost will be replaced by the color intensity cost to achieve the better depth estimation. The constants, w_a, α_g, α_a, α_c, τ_1, τ_2 and L_1 will be determined by experiments. The proposed cost function normally uses the color gradient cost instead of the color intensity one.

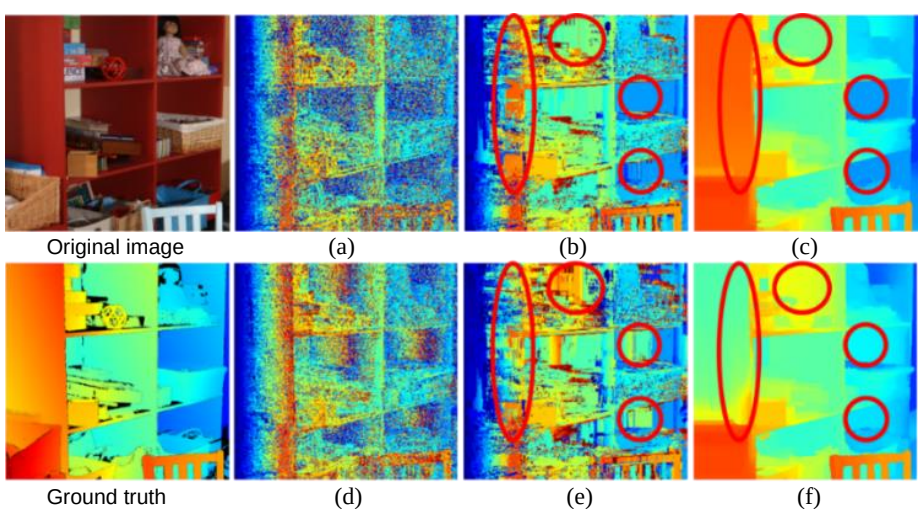


Fig. 2. Estimated depth maps with the costs by: (a) initial color gradient; (b) aggregated color gradient; (c) refined color gradient; (d) initial color intensity; (e) aggregated color intensity; (f) refined color intensity.

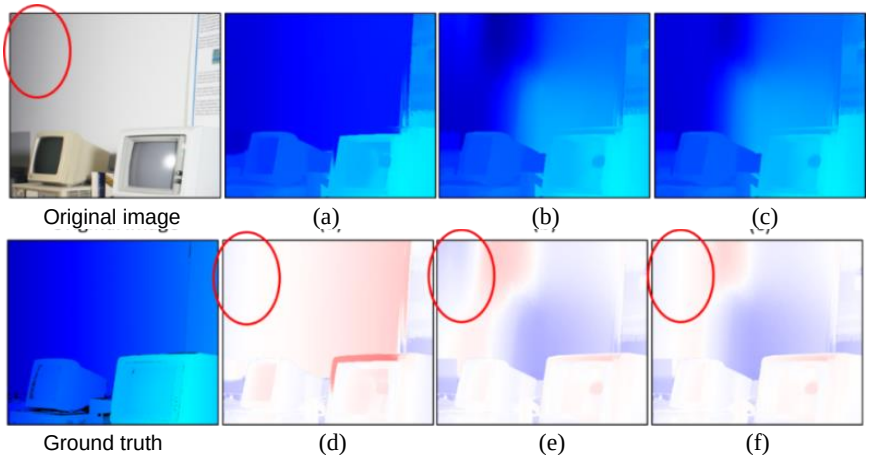


Fig. 3. Estimated depth maps with the costs by: (a) initial color gradient; (b) aggregated color gradient; (c) refined color gradient; (d) initial color intensity; (e) aggregated color intensity; (f) refined color intensity.

Figs. 2 and 3 show two examples of the estimated depth maps achieved by initial, aggregated, and refined cost functions by using either color gradient or color intensity. For smooth regions in Fig. 2 marked by circles, the color gradient cost shows better results than the color intensity cost. On the contrary, Fig. 3 shows that the color intensity cost performs better than the color gradient cost in the gradually-changing color regions, which are marked by circles. Thus, the color cost function stated in Eq. (1) adaptively uses either color gradient or color intensity cost controlled by the gradient flatness indication function defined in Eq. (6) could achieve better depth estimation.

2.3 Cross Aggregation

With the cost function defined in Eq. (1), the cost aggregation is computed by a cross support region [38], which iteratively extends the similar pixels from p in the arm basis as

$$A_{(x,y)}(p) = \max\{L|\Delta(p, p + r(x, y)) < \tau(l) \text{ for } 0 < l \le L_{\max}\}, \quad (9)$$

where $\Delta(p, p_t)$ denotes the intensity difference of the pixels at p and p_t and $A_{(x,y)}(p)$ is the arm length of support region extended from the pixel at p in search of similar pixels. The adaptive similarity threshold is designed as $\tau(l) = 0$ for $l > L_3$, $\tau(l) = \tau_3$ for $l \le L_2$ and $\tau(l) = \tau_4$ for $L_2 < l \le L_3$ with $\tau_4 < \tau_3$. The adaptive similarity threshold gives the larger threshold for larger distanced pixels and the smaller threshold for smaller distanced pixels. Fig. 4 shows the growth of cross support region, which is obtained by the vertical arm growth from the pixel at p and the horizontal arm growth from each pixel in vertical arm to find the whole cross support region.

It is noted that we should find the support regions, $S_l(p)$ and $S_r(p_d)$ for the matching points in the reference and target images, respectively. The aggregation cost of the pixel p in the left image with disparity d in the right image is computed as

$$C_{agg}(p,d) = \sum_{p_i \in R_I} C(p_i,d) / \|S_I\|, \quad S_I = S_l(p) \cap S_r(p_d), \tag{10}$$

where S_I is the intersection of $S_l(p)$ and $S_r(p_d)$ and $\|S_I\|$ denotes the number of pixels in S_I. After the computation of aggregation cost, the estimated depth value for the left image at p is obtained by the winner-takes-all (WTA) criterion by minimizing the aggregation cost as,

$$d_l'(p) = \arg\min_d C_{agg}(p,d). \tag{11}$$

Similarly, we could switch the left and right images by using the same procedures to estimate the depth map $d_r'(p)$ for the right image.

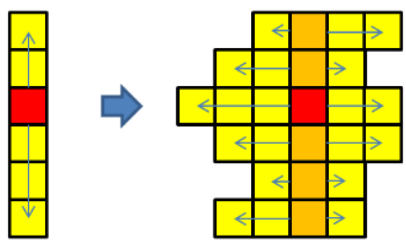


Fig. 4. Growth of the cross support region $S(p)$: vertical growth first followed by horizontal growth.

After the adaptive stereo matching subsystem, the estimated depth maps, $d_l'(p)$ and $d_r'(p)$ could not be so precise in occluded, smooth texture and noisy areas. Besides, the left and right depth maps are still in low resolution. We further propose the recursive refinement and depth upsizing subsystems to further improve the estimated left and right primary depth maps to become high-precision and high-resolution depth maps.

3. RECURSIVE REFINEMENT AND DEPTH UPSIZING SUBSYSTEMS

As shown in Fig. 1, with $d_l'(p)$ and $d_r'(p)$ at hands, we should utilize the recursive refinement to obtain high-precision depth map, $d_l(p)$ and $d_r(p)$ and depth upsizing processes to enhance depth map to become high-precision and high-resolution ones.

3.1 Recursive Refinement Subsystem

As shown in Fig. 5, the depth maps are first processed by median filtering (MF) process to remove the noisy depth values, followed by left right check (LRC) process to detect all matched depth values. Finally, the bilateral filtering (BF) process is used to enhance the depth values. The detailed formulations of all processes and their recursive computation procedures are explained as follows.

To remove the dot-like noises, the median filtering (MF) process is used to remove the separated noises first. For simplicity, after the MF process, as shown in Fig. 5, the filtered left and right depth maps are denoted as $d_l''(p)$ and $d_r''(p)$, respectively.

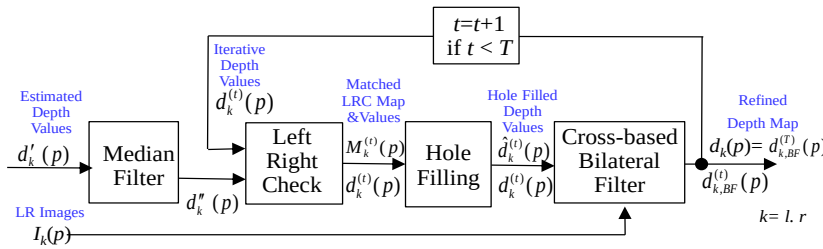


Fig. 5. Flow diagram of the proposed recursive refinement subsystem.

(A) Left Right Checking

By setting $d_l^{(1)}(p) = d_l''(p)$ and $d_r^{(1)}(p) = d_r''(p)$ in the first iteration, the left right checking (LRC) process examines not only the matching level of left and right depth values but also the color similarity in the associated intensities. From left image point of view, the LRC matching index map, $M_l^{(t)}(p)$ at p is given as

$$M_l^{(t)}(p) = \begin{cases} 1, & d_l^{(t)}(p) = d_r^{(t)}(p_{d_l^{(t)}}) \text{ and } \left\| I_l(p) - I_r(p_{d_l^{(t)}}) \right\| \le \tau_{AD} \\ 0, & \text{otherwise} \end{cases}, \tag{12}$$

where the superscript, t is the iteration index and τ_{AD} is a fixed threshold for detecting color similarity. Similarly, we can compute the LRC matching index map, $M_r^{(t)}(p)$ for the right image. Ignoring all unmatched depth values, the LRC outputs the matching map and their depth values.

(B) Iterative Hole Filling

For bilateral filtering (BF) process, the depth pixels in a selected BF window, $S_{\text{BF}}(p)$ are used to compute the refined BF results by excluding the un-matched depth values. In some cases, the pixels in the BF window could be all un-matched depth pixels to make the BF process be impossible. These cases mostly occur in occlusion regions. If the left (right) side image is treated as the target image, the occlusion regions will be in the right (left) side of foreground objects. Generally, we can refer the depth values from the depth values of the background in the other side. In order to solve this problem, the hole filling (HF) value for the left view is first performed as

$$\hat{d}_l^{(t)}(p) = \begin{cases} d_l^{(t)}(\hat{p}_l(p)), & \text{if } \hat{p}_l \in S_{BF}^{(t)}(p) \text{ with } M_l^{(t)}(\hat{p}_l)=1 \\ d_r^{(t)}(p), & \text{otherwise,} \end{cases} \tag{13}$$

where the nearest matching point of p pixel in the left view is expressed as

$$\hat{p}_l(p) = \arg\min_{p_i} \left\{ \left\| p - p_i \right\| \middle| M_l^{(t)}(p_i)=1,\ d_l^{(t)}(p) - d_l^{(t)}(p_i) \le 0 \right\}. \tag{14}$$

Similarly, the hole filling value located for the right view will be

$$\hat{d}_r^{(t)}(p) = \begin{cases} d_r^{(t)}(\hat{p}_r(p)), & \text{if } \hat{p}_r \in S_{BF}^{(t)}(p) \text{ with } M_l^{(t)}(\hat{p}_r)=1 \\ d_l^{(t)}(p), & \text{otherwise} \end{cases} \tag{15}$$

where the nearest matching point at p pixel in the right view is given as

$$\hat{p}_r(p) = \arg\min_{p_i}\{|p - p_i| \mid M_l^{(t)}(p_i) = 1, \quad d_l^{(t)}(p) - d_l^{(t)}(p_i) \geq 0\}. \tag{16}$$

(C) Cross-based bilateral filtering

After hole filling process, the bilateral filtering (BF) process by removing the unmatched ones is computed as

$$d_{k,BF}^{(t)}(p) = \begin{cases} \sum_{p_i \in R(p)} d_k^{(t)}(p_i) w(p, p_i) / \left\| S_{BF}^{(t)}(p) \right\|, & S_{BF}^{(t)}(p) \neq \varphi \\ \hat{d}_k^{(t)}(p), & \text{otherwise} \end{cases} \tag{17}$$

$$S_{BF}^{t)}(p) = \left\{ p_t \left| M^{(t)}(p_t) = 1 \text{ and } p_t \in S_{cross}(p) \right. \right\} \tag{18}$$

$$w(p, p_t) = w(p, p_c) \times w(p_c, p_t) \tag{19}$$

$$w(p, p_c) = ((\|I(p) - I(p_c)\|/\alpha_a)^2 + 1)^{-1} \times ((|p - p_c|/\alpha_d)^2 + 1)^{-1} \tag{20}$$

where $d_{k,BF}^{(t)}(p)$ and $d_k^{(t)}(p)$ denote the depth values after BF and MF processes in the tth iteration, respectively and $S_{BF}^{(t)}(p)$ defines the set of the matched pixels in the BF window at pixel p. By using cross computation algorithm [37], we can fasten the BF filtering process. For recursions, as shown in Fig. 5, we can iteratively perform LRC, HF and BF processes several times. The final refined depth maps can be finally obtained after T iterations as $d_l(p) = d_{l,BF}^{(T)}(p)$ and $d_r(p) = d_{r,BF}^{(T)}(p)$, respectively.

3.2 Depth Upsizing Subsystem

After the recursive refinement, the refined depth maps are still in low resolution. As shown in Fig. 6, we only have spare depth values, which need to be horizontally and vertically upsampled by 4×4 times to become high resolution depth maps. Fig. 7 shows the flow diagram of the proposed depth upsizing subsystem, which is composed of depth value updating, texture-selected interpolations and weighted depth voting algorithms. The depth upsizing subsystem could help to achieve high-resolution (HR) depth maps and further enhance the precision of HR depth maps [38].

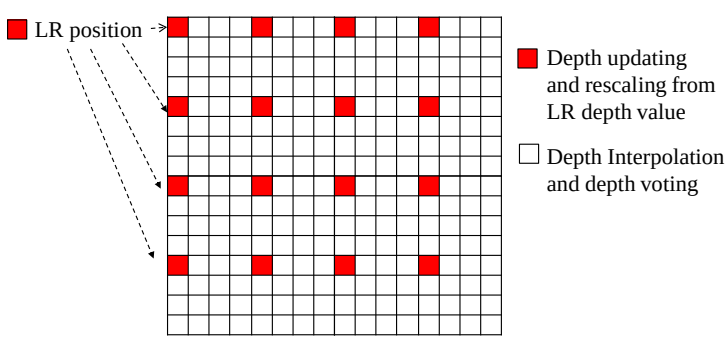


Fig. 6. HR depth pixels related to LR depth pixels for 4×4 upsizing.

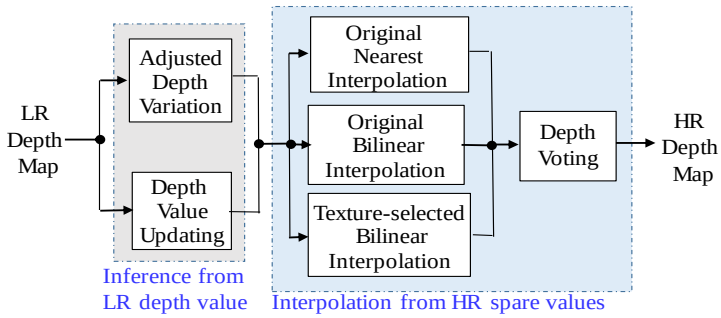


Fig. 7. Flow diagram of the proposed depth upsizing subsystem.

(A) Depth Value Updating

Conceptually, one-pixel shift in low-resolution (LR) images is equal to K-pixels shift in HR images if the LR images are downsized by a factor of $K \times K$. As shown in Fig. 7, the direct depth value performed in spare pixels of the HR image is given by

$$\tilde{d}(\tilde{p})|_{\tilde{p}\to K.p} = K \cdot d(p), \tag{21}$$

where $\tilde{p}$ is the HR position, which is marked red color in Fig. 6, corresponding to the position of p in LR image. The other positions, which are marked in white color in Fig. 6, are computed by interpolations. Fig. 8 (a) shows the matched LR images; however, Fig. 8 (b) exhibits the true HR images, which could not be well matched. It is obvious that the true HR depth values in LR positions in Fig. 6 must have some deviations. To update the depth values, due to 4×4 downsizing, we can magnify the depth values {0, 1, 2, …} to {0, 4, 8, …}, however, the true depth in HR image could be deviated in {+2, –2}.

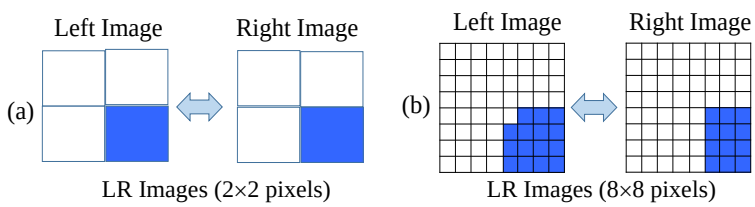
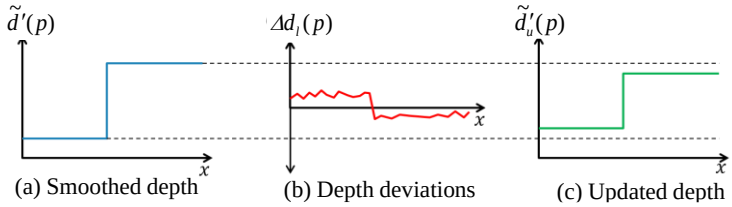


Fig. 8. (a) Matched low-resolution left and right images and (b) their high-resolution left and right images with 4×4 upsampling.

Thus, we need to refine the HR depth value by checking it around $Kd_l(p)$ for the left HR image in range of $-K/2$ and $K/2$. Thus, we suggest that the deviation of upscaling left depth value is performed in the search of the texture similarity in a small patch as:

$$\Delta d_l(p)\big|_{\tilde{p}\to p} = \underset{-K/2<d_i<K/2}{\arg\min} \sum_{p_i \in S_p(p)} \left\| \tilde{I}_l(\tilde{p}_i) - \tilde{I}_r(\tilde{p}_{Kd(p)+d_i}) \right\| \tag{22}$$

where $\Delta d_l(p)$ denotes the possible deviation at p in the LR image with the consideration of HR image and $S_p(p)$ denotes the patch region in HR image with respect to p at the LR image.



(a) Smoothed depth (b) Depth deviations (c) Updated depth

Fig. 9. Concept of depth value updating procedure: (a) direct scaled depth; (b) searched depth deviations; (c) final update depth.

Fig. 9 (a) shows the magnified depth (blue curve) obtained by Eq. (21) with respect to the HR image while Fig. 9 (b) exhibits the possible deviations (red curve) estimated by Eq. (22), where $\Delta d_l(p)$ is depth variations along the horizontal line. To achieve the best depth rescaling with smoothing concept, the depth values as shown in Fig. 9 (c) will be better. Thus, the updating depth values stated in Eq. (22) should be smoothed by the averaged deviation to achieve the correct depth value as Fig. 9 (c) (green curve). Thus, the final depth value updating with the averaged deviation is given by:

$$\tilde{d}_u'(p) = \tilde{d}(p) + \frac{\sum_{p_i \in S_F(p)} \Delta d_l(p_i)}{|S_F(p_i)|}, \tag{23}$$

$$S_F(p) = \{p_i |\, M_i(p) = 1 \text{ and } p_i \in F(p)\}, \tag{24}$$

where $\tilde{d}_u'(p)$ denotes the updating depth value at p point in the LR image and $S_F(p)$ represents the set of depth pixels, which satisfy the LRC and flat conditions horizontally. The LRC condition of depth value is defined as

$$M_l(p) = \begin{cases} 1, & d_l(p) = d_r(p_{d(p)}) \\ 0, & \text{otherwise} \end{cases}, \tag{25}$$

and the flat depth region is defined as

$$F(p)=\{p_i |\, |d(p_i)-d(p_i-(1,0))| < 1 \text{ and } [p: p_i] \in F(p)\}. \tag{26}$$

In summary, the above procedure, as shown in Fig. 9, the depth deviations will be adjusted in the average sense to the upscaled depth values.

(B) Texture-Selected Interpolation

With the above depth updating procedure, we only have sparse HR depth values just in the relative LR depth positions, which are marked red color in Fig. 6. The last problem for the depth resizing subsystem is the interpolation process. For depth interpolation, we only use 4 LR depth values, d_1, d_2, d_3 and d_4 respectively at p_1, p_2, p_3 and p_4 to compute the interpolated result at p as shown in Fig. 10, where α_1, α_2, β_1, and β_2 denote the horizontal and vertical distances from p to $\overline{p_1 p_3}$, $\overline{p_2 p_4}$, $\overline{p_1 p_2}$ and $\overline{p_3 p_4}$ lines, respectively.

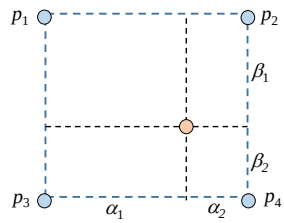


Fig. 10. Interpolation of high resolution at p from four known depth values at p_1, p_2, p_3 and p_4.

For the normal bilinear interpolation, the interpolated depth at p could be given by,

$$\tilde{d} = \frac{d_1\alpha_2\beta_2 + d_2\alpha_1\beta_2 + d_3\alpha_2\beta_1 + d_4\alpha_1\beta_1}{(\alpha_1 + \alpha_2)(\beta_1 + \beta_2)}. \tag{27}$$

As to the nearest neighbour interpolation, the interpolated depth at p could be obtained by

$$\tilde{d} = \left\{ d_i \left| \min_i \| p - p_i \| \right. \right\}. \tag{28}$$

Without considering texture information, the bilinear interpolation will produce burr effect along object boundaries while the nearest neighbor method will obtain step-like depth map in HR depth maps.

Thus, we propose a texture-selected interpolation method to eliminate the inappropriate depth pixels by referring both HR and LR images. With respect to the interpolated pixel in the HR image, we first pick the one of four associated LR pixels, which has closest intensity to the interpolated pixel as the anchor pixel (green color) as shown in Fig. 11. At the same time, if the reminding three pixels have large depth differences from the depth value of anchor pixel, we will eliminate the pixels (red) from the interpolation process. Fig. 11 shows several examples of the selected pixels, where the heights represent depth values. The green depth pixel represents the anchor pixel, which has the closest texture to the interpolated pixel in the HR image. The selected (blue) pixels, whose depth values are closer to the depth value of the green (anchor) pixel will be selected for the interpolation, where the red pixels will be removed from interpolation since their depth values are far away from the depth value of the anchor pixel. Including anchor (green) pixel, the texture-selected interpolation will involve four pixels as shown in Fig. 11 (a), three pixels as shown in Fig. 11 (b), two pixels as shown in Figs. 11 (c) and (d), and one pixel as shown in Fig. 11 (e) for interpolation. The interpolation for these 5 cases will be performed differently as the follows.

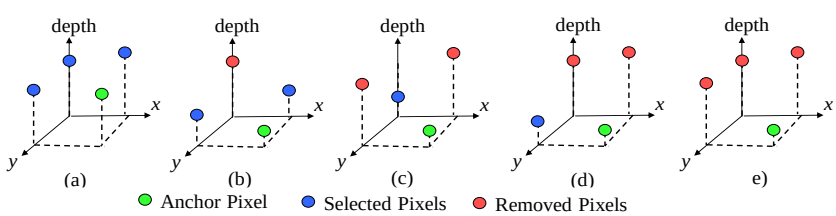


Fig. 11. Texture-selected depth pixels with: (a) three selected; (b) two selected; (c) one selected (diagonal); (d) one selected (edge); (d) no selected (blue) pixels and one anchor (green) pixel.

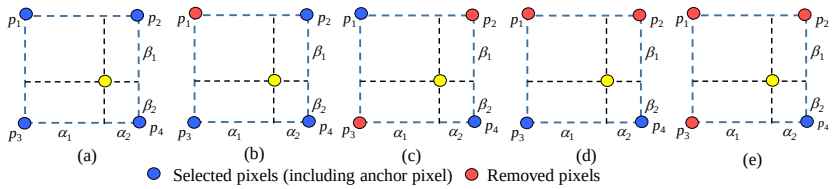


Fig. 12. Interpolation cases with: (a) three; (b) two (diag.); (c) two (edge) and (d) one selected (blue) depth pixels by texture-based selections related to Fig. 11.

Related to Fig. 11, Fig. 12 shows five interpolation cases. If there are four selected pixels, as shown in Fig. 12 (a), the interpolation will be performed as the normal bilinear interpolation stated in Eq. (28).

If there are three selected depth pixels, as shown in Fig. 12 (b), the interpolation will be performed the average of two linear interpolations as

$$\tilde{d} = \frac{d_1\alpha_2 + d_3\alpha_2}{2(\alpha_1+\alpha_2)} + \frac{d_2\beta_2 + d_4\beta_1}{2(\beta_1+\beta_2)}. \tag{29}$$

If there are two selected pixels in diagonal case, as shown in Fig. 12 (c), the interpolation will be approximated by a linear interpolation as

$$\tilde{d} = \frac{d_1(\alpha_2+\beta_2) + d_4(\alpha_1+\beta_1)}{\alpha_1+\alpha_2+\beta_1+\beta_2}. \tag{30}$$

If there are two selected pixels in edge case, as shown in Fig. 12 (d), the interpolation becomes a linear interpolation as

$$\tilde{d} = \frac{d_3\alpha_2 + d_4\alpha_1}{(\alpha_1+\alpha_2)}. \tag{31}$$

If the selected pixel is just one, the interpolation just directly copies the depth value, $\tilde{d} = d_4$ as shown in Fig. 12 (e). Fig. 13 shows the depth maps achieved by various interpolation methods. The proposed texture-selected interpolation outperforms the traditional nearest neighbor and bilinear interpolations. The proposed texture-selected interpolation does not have any blur effects along edges and well match up with the ground true results.

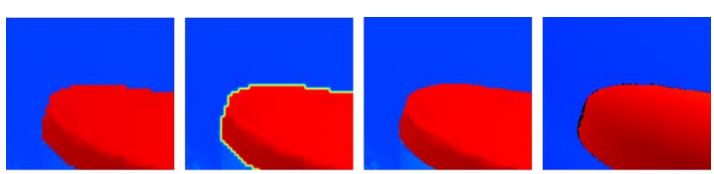


Fig. 13. Interpolation results achieved by: (a) nearest neighbor; (b) original bilinear; (c) texture-selected interpolation methods and (d) ground truth.

(C) Weighted Depth Voting

Actually we have had HR depth maps after texture-based interpolation. To further

refine the HR depth map, we further propose a simple weighed depth voting method to polish the HR depth values in a limited depth region based on the weights computed by similarity and pixel distances. As shown in Fig. 7, we collect all possible candidate values from all interpolations to compute the final voting result. To reduce the computation, the horizontal weight is computed from accumulation of all candidate weights as

$$w_h^{(t)}(d,p) = \sum_{p_i \in S_{\text{cand}}(p) \forall \left|\tilde{d}^{(t-1)}(p_i) - \tilde{d}\right| < 0.5} w(p,p_t) \quad , \tag{32}$$

where $w(p, p_t)$ is defined in Eq. (20). Based on the horizontal weights, the vertical weight can be obtained as

$$w_v^{(t)}(d,p) = \sum_{p_i \in \text{Vset}(\tilde{p})} w_h^{(t)}(d,p) w(p,p_i) \, . \tag{33}$$

Then, from the vertical weights, we search the final top 2 depth values, which are expressed by

$$\hat{d}_1^{(t)}(\tilde{p}) = \arg\max_d w_v^{(t)}(d,p) \, , \tag{34}$$

$$\hat{d}_2^{(t)}(\tilde{p}) = \arg\max_d \{ w_v^{(t)}(d,p) \left| d \neq \hat{d}_1^{(t)}(p_c) \right. \} . \tag{35}$$

If the difference of these two depth values is less than 1, the final depth will be the weighted depth value as

$$\tilde{d}_w^{(t)}(\tilde{p}) = \frac{\hat{d}_1^{(t)}(\tilde{p}) w_v^{(t)}(\hat{d}_1^{(t)}(\tilde{p}),p) + \hat{d}_2^{(t)}(p_c) w_v(\hat{d}_2^{(t)}(\tilde{p}),p)}{w_v^{(t)}(\hat{d}_1^{(t)}(\tilde{p}),p) + w_v(\hat{d}_2^{(t)}(\tilde{p}),p)} \, . \tag{36}$$

Otherwise we will select the one, which has the largest weight, as

$$\tilde{d}^{(t)}(\tilde{p}) = \begin{cases} \tilde{d}_w^{(t)}(\tilde{p}), & \left| \hat{d}_1^{(t)}(\tilde{p}) - \hat{d}_2^{(t)}(\tilde{p}) \right| \leq 1 \\ \hat{d}_1^{(t)}(\tilde{p}), & \text{otherwise.} \end{cases} \tag{37}$$

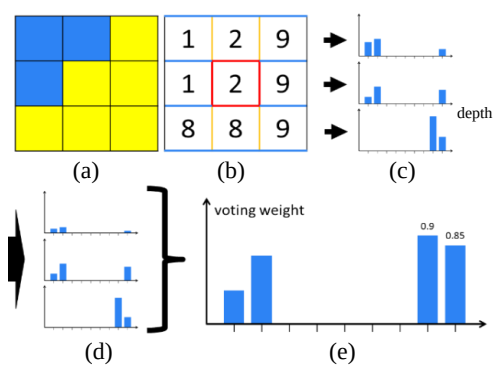


Fig. 14. Depth voting process: (a) color image region; (b) their depth values; (c) computed horizontal voting weights; (d) recursive horizontal voting weights; (e) final vertical voting weights.

Fig. 14 shows the concept of the depth weight voting method, where the bar charts represent the voting weights of different depth values. Figs. 14 (a) and (b) exhibit 3×3 image pixel values and their corresponding depth values, respectively. Fig. 14(c) shows three bar charts of voting weights versus depth values for three horizontal lines. With respect to the central pixel, the higher weight means higher similarity. Fig. 14 (d) shows the voting weights obtained in the previous step while Fig. 14 (e) exhibits the vertical weights, which are the sums of all horizontal results for final voting. With weights 0.9 and 0.85, the top 2 depth values are 8 and 9. Since the difference is 1, the voting depth for the central pixel, marked in Fig. 14 (b), becomes the weighted average of these two values, which is 8.49.

4. EXPERIMENTAL RESULTS

In the proposed stereo matching system, we have suggested several innovated algorithms. To evaluate the effectiveness of the algorithms, we first setup several step-by-step experiments and determine their best control parameters at the same time. By using the selected control parameters, the proposed stereo matching system is finally compared to the well-known stereo matching methods. By using Middlebury HR database [39], the environment of experiments is conducted by C++ programming language with OpenCV 2.4.10 and parallel computation codes with NVIDIA CUDA 5.0, where the hardware system is with Intel(R) Core (TM) i7-4770k @ 3.50GHz platform, RAM 16GB and GPU with GeForce GTX TITAN. For calculation of cost function, we select w_a = 0.2, α_g = 10, α_a = 1, α_c = 50, τ_1 = 1, τ_2 = 10, and L_1 = 10. For cost aggregation, we choose τ_3 = 25, τ_4 = 15, and L_3 = 30 and L_4 = 60. For disparity refinement, we set τ_{AD} = 1, α_a = 1 and α_d = 20. For depth upsizing, we pick τ_b = 5.

4.1 Evaluation of the Proposed Algorithms

In the first set of experiments, we test different cost functions with different refinement methods.

(A) Evaluation of Cost Functions

With texture census cost, the cost functions combined with the costs of color gradient or/and color intensity as well as the proposed refinement algorithm are evaluated, where the traditional refinement is composed of traditional MF, LRC, and BF processes. In depth upsizing unit, we only use depth updating and nearest interpolation. Fig. 15 shows the subjective performances of five depth maps. The results exhibit that the proposed adaptive cost is better than the color intensity cost only. Figs. 16-17 show the error performances achieved by various algorithms for all pixels and non-occluded pixels. The pixels are detected as the error ones if they satisfy

$$\frac{\left|d_{ground\ truth}(p)-d_{result}(p)\right|}{disparity\_range} > 0.5\%. \tag{39}$$

From Figs. 16 and 17, the proposed cost function with integrated cost plus the proposed refinement algorithm performs the best error performances.

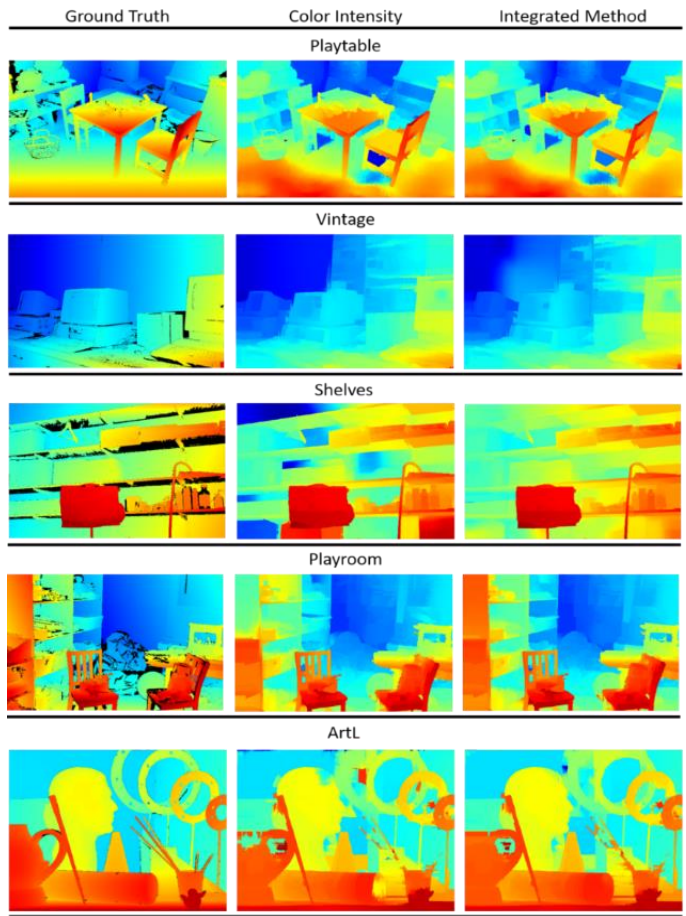


Fig. 15. Depth maps achieved by color intensity and integrated (gradient and intensity) cost functions.

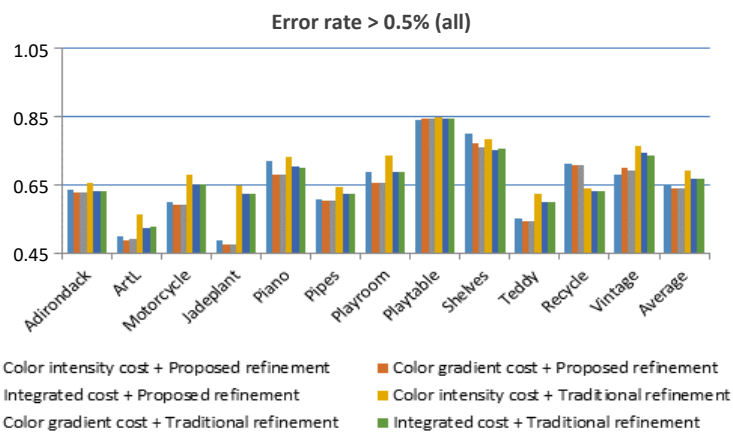


Fig. 16. Error rates with error > 5% for all pixels achieved by various cost functions and refinements.

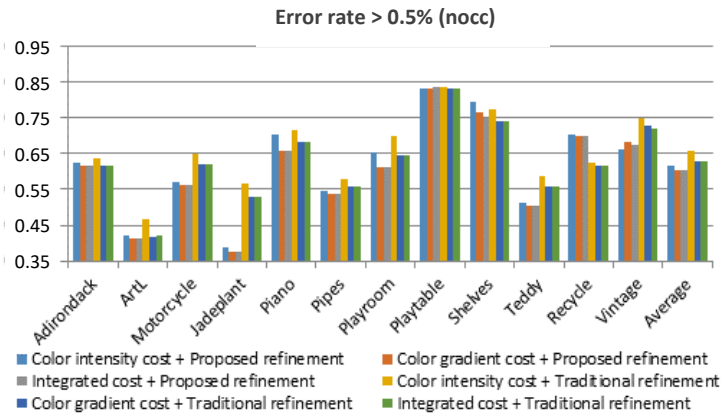


Fig. 17. Error rates with error > 5% for non-occluded pixels achieved by various cost functions and refinements.

(B) Evaluation of Recursive Refinement Algorithm

To understand the effectiveness of recursive refinements, we conduct 1 (no recursion), 5, and 10 iterations to test the error and PSNR performances, where the depth upsizing only includes depth updating and nearest interpolation. From Figs. 18 and 19, we found the recursive refinements can help to improve the performances. The 10 iterations perform the best from the objective performances. Fig. 20 shows the subjective performance of the proposed recursive refinement method. The results also exhibit that the iterations can help to enhance the depth maps. However, we prefer 5-iteration refinement to reduce the computation and over smoothing.

Fig. 18. Error rates with error > 5% for all pixels achieved by the proposed recursive refinements with 1, 5 and 10 iterations.

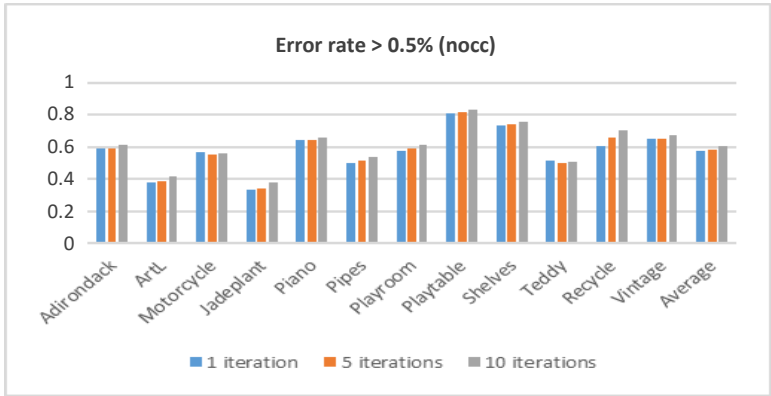


Fig. 19. Error rates with error > 5% for non-occluded pixels achieved by the proposed recursive refinements with 1, 5 and 10 iterations.

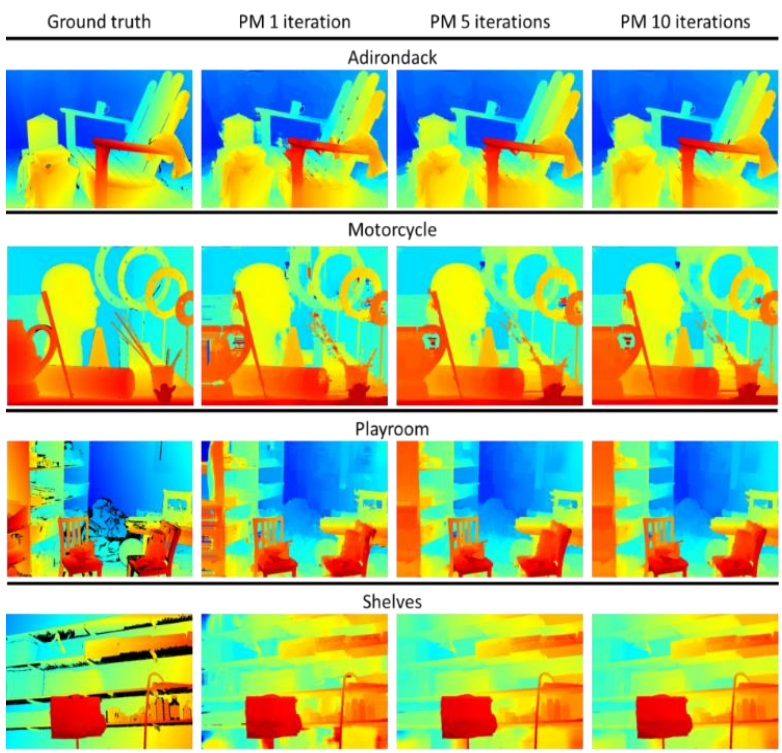


Fig. 20. Depth maps achieved by the proposed recursive refinements with 1, 5, and 10 iterations.

(C) Evaluation of Depth Upsizing and Voting Algorithms

To evaluate the proposed depth upsizing algorithm, in the experiments, there are 6 different combinations of upsizing methods needed to be evaluated with error and PSNR performances. To simplify the notations in Figs. 21-23, there are 6 combinations, which

are set as: A = "update+nearest"; B = "update+bilinear" and C = "update+texture-selected interpolation". In addition, to evaluate the results achieved by the depth voting algorithm, we further add 3 interpolation methods with depth voting by setting D = "A + depth voting"; E = "B+depth voting"; F = "C+ depth voting". To evaluate these six combinations, we exhibit the differential error performances of all methods with respect to the traditional method (rescale+nearest) are exhibited for better observation of their differences. The results show that the proposed depth value updating can improve the accuracy in most of cases. The proposed selected-texture interpolation (C method) performs the best interpolation methods while the depth voting method (F method) can further improve their precisions.

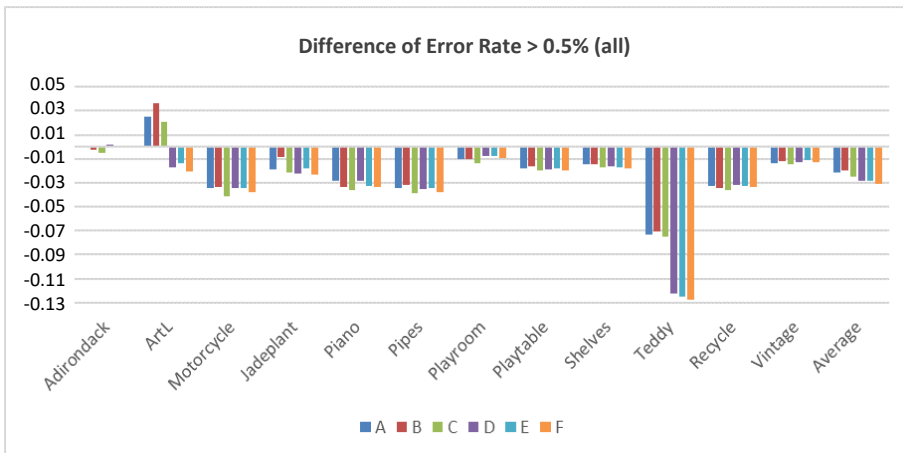


Fig. 21. Differential error rates with error > 5% for all pixels achieved by various depth resizing methods.

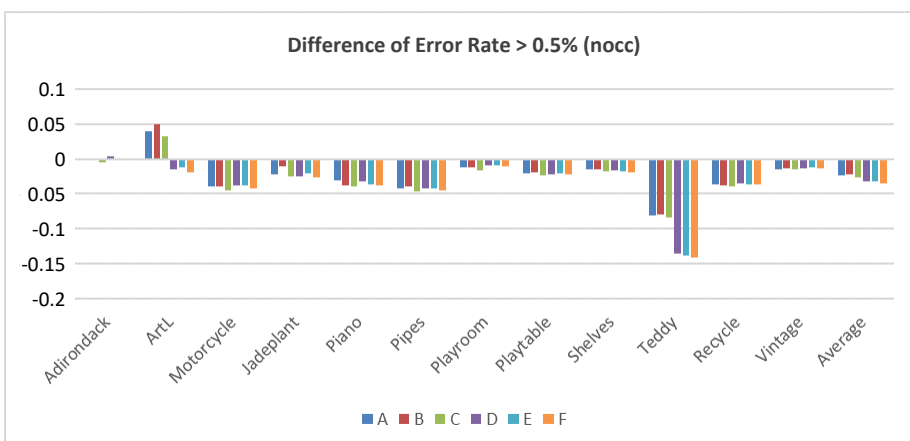


Fig. 22. Differential error rates with error > 5% for non-occluded pixels achieved by various depth resizing methods.

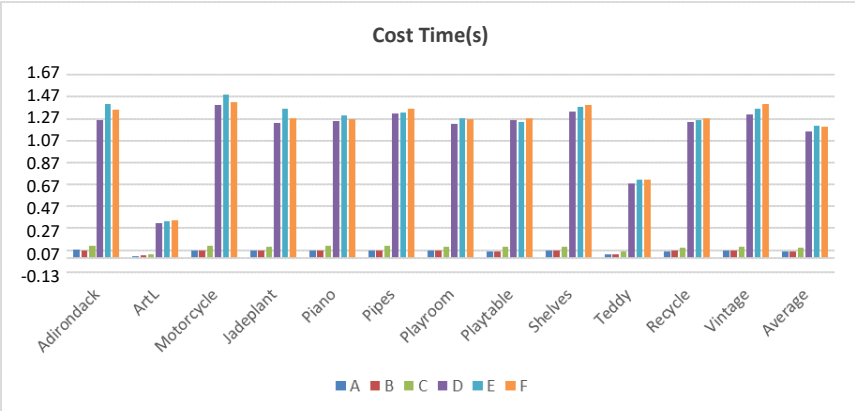


Fig. 23. Differential computation time (sec) required by various depth resizing methods.

4.2 Comparison to the Existed Methods

By using Middlebury dataset, we select three best HR methods, including top down segmented regression (MC-CNN+TDSR) [30], semi-global matching (SGM) [24] and permeability filtering (PFS) [25] to compare the proposed stereo matching system (PM). The PM is composed of the proposed adaptive stereo matching, recursive refinement, and depth resizing subsystems. For quality assessments, the error performances of all methods are root mean square (RMS) errors for all pixels and non-occluded pixels exhibited in Figs. 24 and 25, respectively. The results show that the proposed method (PM) is worse than the CNN-based method but better than the SGM and PFS methods. TABLE I shows the computation times for each test images acquired by all the stereo matching methods. The results exhibit that the proposed method is the fastest method. Comparing to the CNN-based method, the proposed method is about 460 times faster. It is noted that the MC-CNN+TDSR method [30], which utilizes three deep sub-networks to compute the similarity score of the input left and right patches, spends much more computations than the simple adaptive stereo matching cost used in the proposed stereo matching system.

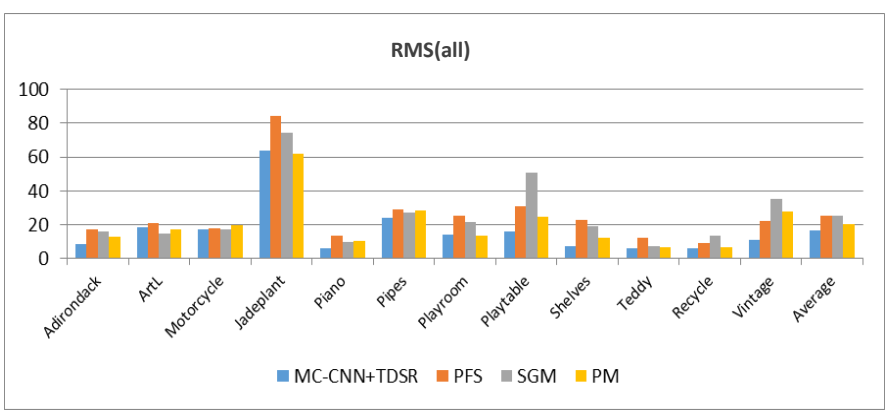


Fig. 24. RMS error performances of different stereo matching methods for all pixels.

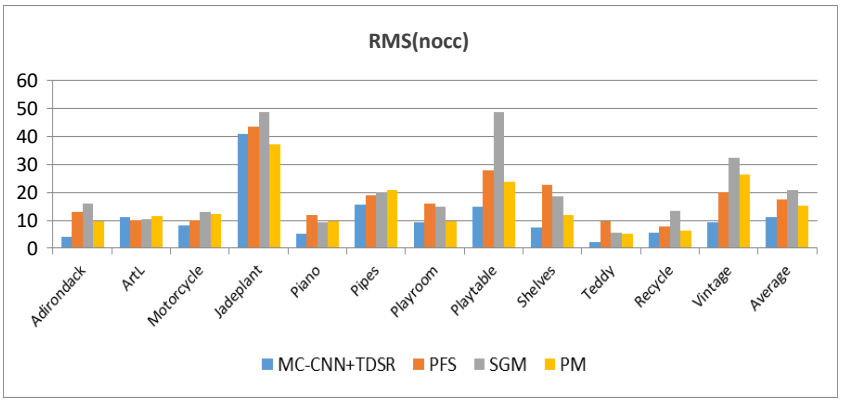


Fig. 25. RMS error performances of different stereo matching methods for non-occluded regions.

Table 1. Computation time needed by various stereo matching methods.

time(s)	Adirondack	ArtL	Motorcycle	Jadeplant	Piano	Pipes	Playroom	Playtable	Shelves	Teddy	Recycle	Vintage	Average
MC-CNN+TDSR	534	170	687	752	563	703	644	619	584	311	546	723	569.6667
PFS	28.4	5.55	29.6	65	22	31.1	32.7	27	24.8	8.92	24.1	82.2	31.78083
SGM	55.4	13.3	53.3	97.1	46.9	55.5	56.5	49.2	45.5	21.2	49.4	123	55.525
PM	1.54	1.54	1.07	1.51	1.07	1.09	1.06	1.11	1.05	0.59	1.33	1.8	1.23

4.3 Comparison to the CNN-based Methods

To compare the latest end-to-end CNN model for stereo matching, we adopted DispNet [35]. The comparisons are conducted on KITTI [40] and Middlebury datasets. We trained DispNet with KITTI 2012 and 2014 as the training data, and separated 5% images as the validation data. However, since the training data is too small in Middlebury, the results achieved by the DispNet in Middlebury dataset is based on pre-trained weights obtained from KITTI dataset. For fair comparisons, we tested the proposed method for KITTI and Middlebury datasets are with same parameters settings.

The simulation results achieved by the DispNet and the proposed method are tested on KITTI dataset. In the DispNet, the error rate in validation data is nearly 3%, which is relatively better than the traditional stereo matching methods generally while the proposed method achieves 12.5% error rates. As shown in Fig. 26, most errors achieved by the proposed method are small and in smooth road areas. The proposed method performs well on objects such as cars, trees, and advertising board. For autonomous driving applications, the proposed method is still a qualified solution. When the performance test changes to HD images on Middlebury, the first problem for the DispNet is that the size of the input images are too large. We have to down scale their resolutions to prevent from out of memories on VGA card. It is noted that the performance of DispNet is highly dependent on the training data. As shown in Fig. 27, although the relative disparities estimated by the DispNet are correct, the overall depth values are with large errors since the depth values are usually small in KITTI training data. Besides, the depths around the object boundary are also blurred. In summary, the CNN-based models can obtain very good results if they are trained by same specific training data. Once the scenes are changed from time to time in videos, we may have very poor results since most of CNN

models are data dependent. The proposed method based on the stereo matching concept will have more stable performance with fixed parameter settings. The proposed method has better flexibility to deal with different categories of scenes, which are mostly existed in high- definition and high-quality 3D movies.

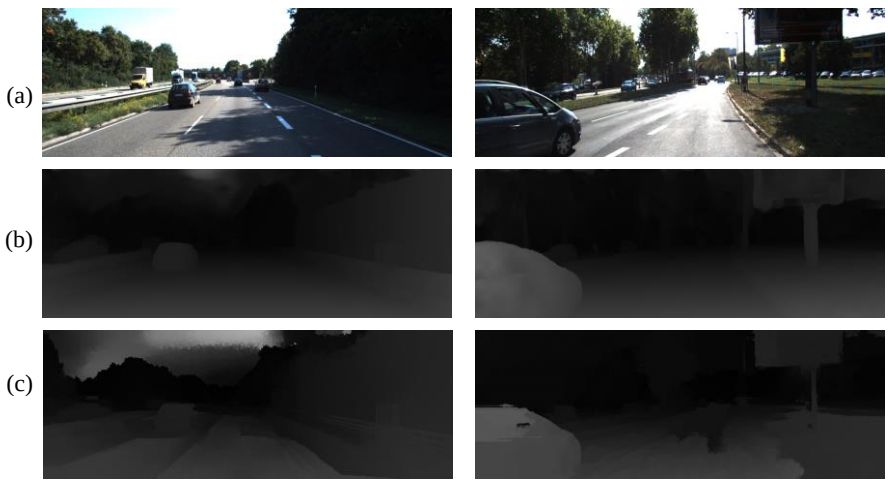


Fig. 26. Compared depth maps estimated by: (a) input image; (b) DispNet [35]; (c) the proposed method in KITTI dataset.

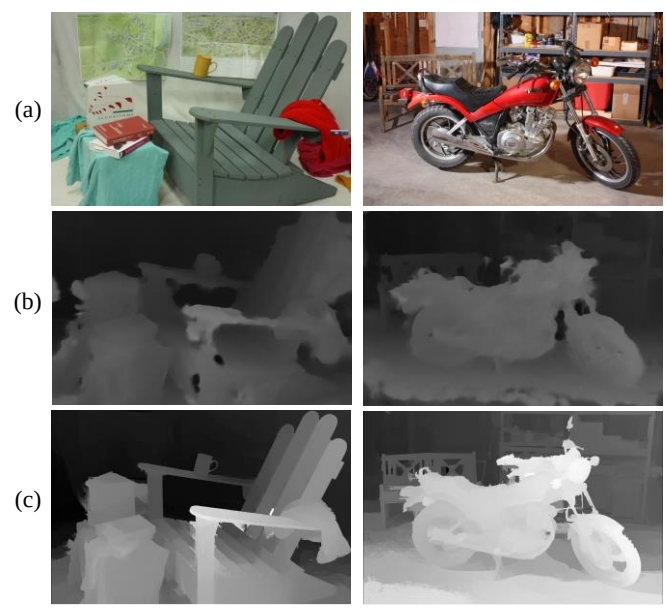


Fig. 27. Compared depth maps estimated by: (a) input image; (b) DispNet [35]; (c) the proposed method in Middlebury data set.

5. CONCLUSIONS

In this paper, we proposed a fast and precision stereo matching system, which is composed of adaptive stereo matching, recursive refinement and depth upsizing subsystems. For fast computation, in general, the concepts of simple image downsizing and depth upsizing approaches can be used for any stereo matching algorithms in use of traditional or deep learning approach. The proposed stereo matching system suggested a simple adaptive matching subsystem, which adaptively combines census cost as well as the color gradient and color intensity costs to estimate the low resolution depth maps. To improve the accuracy of depth maps, the recursive refinement subsystem, which contains median filtering, left right checking, hole filling, and bilateral filtering algorithms can effectively remove the most of errors. The depth upsizing subsystem, which is composed of depth value updating, texture-selected interpolation and depth voting methods, can be used to achieve the high-resolution (HR) depth maps. Simulation results verified that the proposed stereo matching system could retrieve the high-quality and high-resolution depth maps with very low computation. The proposed stereo matching system with low computation will be a good solution for future ultra-high 3D video applications.

REFERENCES

1. F. Ma, L. Carlone, and U. Ayaz, "Sparse depth sensing for resource-constrained robots," *International Journal of Robotics Research*, Vol. 38, 2019, pp. 935-980.
2. K. Zhang, J. Chen, Y. Li, and X. Zhang, "Visual tracking and depth estimation of mobile robots without desired velocity information," *IEEE Transactions on Cybernetics*, Vol. 50, 2020, pp. 361-373.
3. Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-LiDAR from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8445-8453.
4. W.-J. Yang, J.-F. Yang, G.-C. Chen, P.-C. Chung, and M.-F. Chung, "An assigned color depth packing method with centralized texture depth packing formats for 3D VR broadcasting services," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, Vol. 9, 2019, pp. 122-132.
5. A. I. Purica, E. G. Mora, and B. Ionescu, "Multiview plus depth video coding with temporal prediction view synthesis," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 26, 2016, pp. 360-374.
6. M. Sharma, S. Chaudhury, and B. Lall, "A novel hybrid Kinect-variety-based high-quality multiview rendering scheme for glass-free 3D displays," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 27, 2017, pp. 2098-2117.
7. H. Cho, S. Jung, and H. Jee, "Real-time interactive AR system for broadcasting," in *Proceedings of IEEE Virtual Reality*, 2017, pp. 353-354.
8. Y.-W. Liao, M.-J. Chen, C.-H. Yeh, J.-R. Lin, and C.-W. Chen, "Efficient inter-prediction depth coding algorithm based on depth map segmentation for 3D-HEVC," *Multimed Tools and Applications*, Vol. 78, 2019, pp. 10181-10205.

9. M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Fast 3D-HEVC depth map encoding using machine learning," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 30, 2020, pp. 850-861.
10. V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," in *Proceedings of IEEE International Conference on Computer Vision*, 2001, pp. 508-515.
11. Y.-C. Wang, C.-P. Tung, and P.-C. Chung, "Efficient disparity estimation using hierarchical bilateral disparity structure based graph cut algorithm with a foreground boundary refinement mechanism," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 23, 2013, pp. 784-801.
12. Y. Boykov, O. Veksier, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, 2001, pp. 1222-1239.
13. J. Sun, N.-N. Zheng, and H.-Y. Shum, "Stereo matching using belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, 2003, pp. 787-800.
14. A. Klaus, M. Sormann, and K. Karner, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," in *Proceedings of International Conference on Pattern Recognition*, 2006, pp. 15-18.
15. X. Sun, X. Mei, S. Jiao, M. Zhou, and H. Wang, "Stereo matching with reliable disparity propagation," in *Proceedings of International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, 2011, pp. 132-139.
16. Q. Yang, L. Wang, R. Yang, H. Stewénius, and D. Nistér, "Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, 2009, pp. 492-504.
17. O. Veksler, "Stereo correspondence by dynamic programming on a tree," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 384-390.
18. G.-S. Hong and B.-G. Kim, "A local stereo matching algorithm based on weighted guided image filtering for improving the generation of depth range images," *Displays*, Vol. 49, 2017, pp. 80-87.
19. K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 19, 2009, pp. 1073-1079.
20. A. Hosni, M. Bleyer, M. Gelautz, and C. Rhemann, "Local stereo matching using geodesic support weights," in *Proceedings of IEEE International Conference on Image Processing*, 2009, pp. 2093-2096.
21. K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 19, 2009, pp. 1073-1079.
22. X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang, "On building an accurate stereo matching system on graphics hardware," in *Proceedings of Computer Vision Workshops*, 2011, pp. 467-474.
23. Q. Yang, "Stereo matching using tree filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37, 2014, pp. 834-846.

24. H. Hirschmuller, "Stereo processing by semi-global matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, 2008, pp. 328-341.
25. C. Ciglaa and A. A. Alatan, "Information permeability for stereo matching," *Signal Processing: Image Communication*, Vol. 28, 2013, pp. 1072-1088.
26. D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, Vol. 47, 2002, pp. 7-42.
27. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, 2012, pp. 1097-1105.
28. J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431-3440.
29. S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, 2017, pp. 1137-1149.
30. J. Zbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *Journal of Machine Learning Research*, Vol. 17, 2016, pp. 1-32.
31. J. Pang, W. Sun, J. S. Ren, C. Yang and Q. Yan, "Cascade residual learning: A two-stage convolutional neural network for Stereo Matching," in *Proceedings of IEEE International Conference on Computer Vision Workshops*, 2017, pp. 878-886.
32. J. Chang and Y. Chen, "Pyramid stereo matching network," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5410-5418.
33. P. Knöbelreiter, C. Reinbacher, A. Shekhovtsov, and T. Pock, "End-to-end training of hybrid CNN-CRF models for stereo," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1456-1465.
34. B. Ummenhofer *et al.*, "DeMoN: Depth and motion network for learning monocular stereo," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5622-5631.
35. N. Mayer, E. Ilg, P. Hausser, and P. Fischer, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4040-4048.
36. K.-J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, 2006, pp. 650-656.
37. K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 19, 2009, pp. 1073-1079.
38. Q. Yang, R. Yang, J. Davis, and D. Nister, "Spatial-depth super resolution for range images," in *Proceedings of Computer Vision and Pattern Recognition*, 2007, pp. 1-8.
39. D. Scharstein, R. Szeliski, and H. Hirschmüller, "Middlebury," http://vision.middlebury.edu/stereo/.

40. A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354-3361.

Wei-Jong Yang (楊惟中) was born in Hsin-Chu, Taiwan in 1990. He received his B.S. degree in Computer Science from Tunghai University, Taiwan in 2012 and the M.S. degree in Computer Science and Information Engineering from the National University of Tainan, Taiwan in 2015. Currently, he is a Ph.D. student with the Institute of Computer and Communication Engineering, Department of Electrical Engineering in the National Cheng Kung University, Taiwan. His current research interests include video coding and processing, pattern recognition, machine learning and deep learning systems.

Hsing-Ju Chou (周興儒) was born in Tainan, Taiwan in 1993. He received his B.S. degree in Electrical Engineering from National Cheng Kung University, Taiwan in 2017 and the M.S. degree in the Institute of Computer and Communication Engineering, Department of Electrical Engineering in the National Cheng Kung University, Taiwan. His current research interests include CUDA computing, pattern recognition, and deep learning systems.

Pau-Choo Chung (詹寶珠) received the Ph.D. degree in Electrical Engineering from Texas Tech University, Lubbock, TX, USA, in 1991. She was with the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 1991 and became a Full Professor in 1996. She applies most of her research results to healthcare and medical applications. Dr. Chung is a member of the Phi Tau Phi Honor Society, was a member of the Board of Governors of CAS Society from 2007 to 2009 and from 2010 to 2012, and is currently an ADCOM Member of the IEEE CIS and the Chair of CIS Distinguished Lecturer Program. She also is an Associate Editor of IEEE Transaction on Neural Networks and the Editor of Journal of Information Science and Engineering, the Guest Editor of Journal of High Speed Network, the Guest Editor of IEEE Transaction on Circuits and Systems-I, and the Secretary General of Biomedical Engineering Society of China. She is one of the Co-Founders of Medical Image Standard Association (MISA) in Taiwan and is currently on the Board of Directors of MISA. Her research interests include image/video analysis and pattern recognition, biosignal analysis.