REAL-TIME FREE VIEWPOINT IMAGE RENDERING BY USING FAST MULTI-PASS DYANAMIC PROGRAMMING

Norishige Fukushima [†], *Toshiaki Fujii* [‡], *Yutaka Ishibashi*[†], *Tomohiro Yendo*[*], *Masayuki Tanimoto*[*]

[†] Graduate School of Engineering, Nagoya Institute of Technology
[‡] The Graduate School of Science and Engineering, Tokyo Institute of Technology
[*] Graduate School of Engineering, Nagoya University

ABSTRACT

In this paper, we introduce a free viewpoint image generation method with an optimization method for a view dependent depth map. Image based rendering (IBR) can render photo-realistic images from natural images, and ray space or light field is IBR method for 3D representation. To generate free viewpoint images naturally from light filed data which are captured by camera array, disparity maps on the virtual views are required. For high quality image generation, an accurate disparity map is required; however, computational cost of depth map optimization is usually huge. For real-time rendering, we use a semi-global disparity estimation method called multi-pass dynamic programming (MPDP), which applies the dynamic programming method to the depth map multi-directionally. The proposing algorism speeding up MPDP method and this optimization effect and the proposing occlusion detection improve synthesized image quality. The experimental results show that fast MPDP can interpolate virtual view and PSNR of this synthesized image to the actual image is 29.2 dB. On the contrary, synthesized images from belief propagation which is one of the best optimization algorism have 29.4 dB. In addition, the MPDP computational time is almost real-time, and that time is 51.5 ms, meanwhile belief propagation takes 397.7 ms.

Index Terms **—** free viewpoint image, ray space, light field, free viewpoint television, stereo matching, multi-pass dynamic programming, occlusion detection

1. INTRODUCTION

In the last two decades, IBR (Image Based Rendering) grows popular as photo-realistic rendering method. Light field rendering [1] and Ray space [2] have been known as 3D representation method in the context of IBR and both methods can render free viewpoint image. Free viewpoint television is an application of these free viewpoint imaging [3]. According to the plenoptic sampling theory [4], however, the rendering range where synthesized image has enough quality is limited. If the generated view is out of range, it is aliased, i.e. ghosted or blurred. We have proposed the method of [5] which combines depth estimation and ray space interpolation can render a free viewpoint image in full range without aliasing. In the depth information process, we compute a disparity map on a free viewpoint image not on the reference image in the camera array. In the common stereo matching, the disparity maps are generated up to the number of the reference views and these disparity maps correspond to the each reference views. However, in our approach, a disparity map on the required viewpoint, i.e. there is no reference camera, is computed, and this disparity map is reshaped by the viewpoint. Thus we call the disparity map view dependent disparity map. The later work of us [13], improve the view dependent disparity map quality by the proposed method of Multi Pass Dynamic Programming (MPDP), which is Markov Random Field (MRF) optimization method. This disparity map optimization makes a virtual view quality high. The optimization method computes three cycle of Scan-line Optimization (SO) [6], which is one of a Dynamic Programming (DP) for MRF optimization. In the state-of-the-arts stereo algorisms reported in [6], top performance algorism well optimizes depth map, however computational cost becomes high. For the real-time application, such as our view synthesis processing, DP approach is reasonable because of the light computational cost. Nonetheless the SO is fast, the computing cost is $O(d^2)$, where d is the number of disparity candidates. Thus the more the candidates, the higher computing cost exponentially. In addition, occlusion area is ignored and then, these regions tend to be aliased.

For improvement of the free viewpoint image generation and the view dependent disparity map computation, we speed up multi pass dynamic programming approach with min convolution inspired by the Semi-global optimization approach [9] and process an outlier filtering for the disparity map. Moreover, we propose an occlusion detection method for virtual view interpolation.

Organization of this paper is as follows. Section 2 shows how to generate the free viewpoint images and the disparity maps on the view. Section 3.1 explains disparity map optimization method by proposing multi-pass dynamic programming method and section 3.2 speeds it up. A noise filtering method for the disparity map is shown in section 3.3. In section 4, we introduce the occlusion detection method and section 5 reveals the efficacy of the proposing method by an experiment. Finally, we conclude this paper in section 6.

2. FREE VIEWPOINT IMAGE GENERATION

In this section, we explain how to generate a free viewpoint image from an aligned camera array. Now, we generate a free viewpoint image on the virtual viewpoint $\mathbf{v}=(x, y, z)$. The generating ray $I_{\mathbf{v}}(\mathbf{p})$, which $\mathbf{p}=(u,v)$ is a pixel, on the free viewpoint image is by weighted additions of reference images I_i (i is camera index number) on the camera array and expressed by

$$I_{\mathbf{v}}(\mathbf{p}) = \sum_{i \in V_{\mathbf{v}}(\mathbf{p})} w_i(\mathbf{p}) I_i(\mathbf{p} + \mathbf{d}_{\mathbf{v}}(\mathbf{p}, i))$$

, where $w_i(\mathbf{p})$ is weighting function and $\mathbf{d}_{\mathbf{v}}(\mathbf{p},i)$ is view dependent disparity information which indicates corresponded point on reference images I_i. $V_{\mathbf{v}}(\mathbf{p})$ is a subset which includes the nearest rays from the generating ray. Figure 1 shows an example of ray interpolation from W x H camera array. In this condition, the coordinate of the point across the camera plane is defined by ray space, and the point (s,t) becomes $s = x + uz, \quad t = x + vz$.

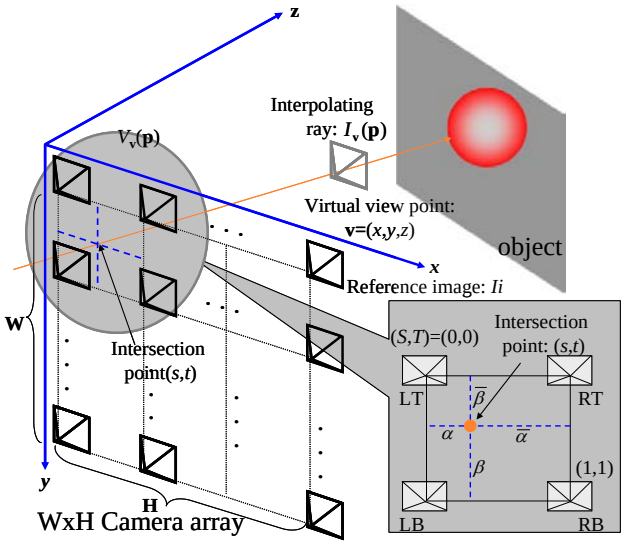

Figure 1. Ray interpolation from Camera array

In the camera array coordinate of (s,t) , the cameras have integer index number, in our setup. So that S and T, which is the nearest cameras of the ray, become $S=\lfloor s \rfloor, T=\lfloor t \rfloor$. In the case of such a two dimensional camera array, the subset of $V_v(\mathbf{p})$ contain four views and defined as:

$$V_v(\mathbf{p}) = \left\{ \begin{pmatrix} S \\ T \end{pmatrix} \begin{pmatrix} S+1 \\ T \end{pmatrix} \begin{pmatrix} S \\ T+1 \end{pmatrix} \begin{pmatrix} S+1 \\ T+1 \end{pmatrix} \right\}. = \{\mathrm{LT}, \mathrm{RT}, \mathrm{LB}, \mathrm{RB}\}$$

The interpolating ray divides the integer view grid internally, and these vertical and horizontal dividing weights become:

$$\alpha = s - S,\ \bar{\alpha} = 1.0 - \alpha$$

$$\beta = t - T, \bar{\beta} = 1.0 - \beta$$

At the next step, we interpolate the ray from the reference images along view dependent disparity. The interpolating ray divides the camera array plane. The disparity information on the virtual ray is given; the distances from the intersection point to reference cameras define the divided disparity and weight of reference rays. Figure 2 shows the *x*-*z* plane of camera array when the ray divides camera array plane. After that setup, the ray on free viewpoint image is computed from weighted linear interpolation along divided disparities:

$$I_v(\mathbf{p}) = \bar{\alpha}\bar{\beta}\mathrm{LT}(u+\alpha \mathrm{d}, \mathrm{v}+\beta \mathrm{d}) + \alpha\bar{\beta}\mathrm{RT}(u-\bar{\alpha}\mathrm{d}, \mathrm{v}+\beta \mathrm{d}) + \bar{\alpha}\beta\mathrm{LB}(u+\alpha \mathrm{d}, \mathrm{v}-\bar{\beta}\mathrm{d}) + \alpha\beta\mathrm{RB}(u-\bar{\alpha}\mathrm{d}, \mathrm{v}-\bar{\beta}\mathrm{d})$$

Now, we explain how to estimate view dependent disparity map. In the stereo matching, disparity map is generally computed by photometric consistency of reference left and right images. According to the stereo matching case, The view dependent disparity estimation method uses a photometric pixel dissimilarity E_p . It is computed from the subset of required ray and its combination of sum of truncated absolute difference:

$$E_p(\mathbf{p}, d) = \frac{1}{M} \sum_{i,j \in v} \min(|I_i(\mathbf{p}+\mathbf{d}_v(\mathbf{p}, i)) - I_j(\mathbf{p}+\mathbf{d}_v(\mathbf{p}, j))|, T_p)$$

, where M is number of combination and T_p is a truncated value of photometric cost. Then these costs are aggregated with support window, where W is a set of support region and $\mathbf{q}$ is a pixel in the set. The number of elements in the set is N:

$$E_{\mathrm{data}}(\mathbf{p}) = \frac{1}{N} \sum_{q \in W} E_p(\mathbf{p}+\mathbf{q})$$

Finally, a view dependent disparity on pixel $\mathbf{p}$ is estimated by minimizing this cost function.

$$\mathrm{disp}(\mathbf{p}) = \underset{d}{\operatorname{argmin}} E_{data}(\mathbf{p}, d)$$

This simple procedure, however, generates uncertainty depth, and is lack of robustness. In the next section, we introduce how optimize and improve this view dependent depth by an advanced dynamic programming algorism.

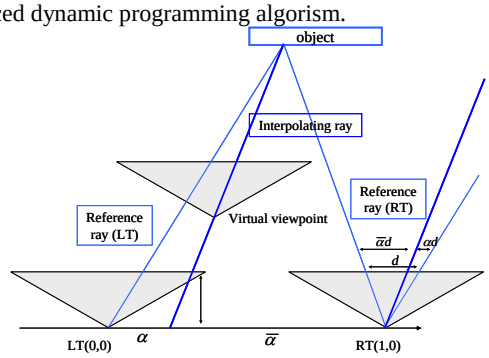

Figure 2. Interpolating ray and divided disparity

3. MULTI PASS DYNAMIC PROGRAMMING

3.1 Markov random field optimization by multi-pass dynamic programming

The top performance methods of stereo matching algorisms adopt MRF optimization model. MRF is solved by the following energy minimization process:

$$\mathrm{disp}(\mathbf{p}) = \underset{d}{\operatorname{argmin}} \sum_{\mathrm{p}} E_{\mathrm{data}}(\mathbf{p}, d) + \lambda E_{\mathrm{smooth}}(\mathbf{p}, d)$$

,where E_{smooth} is smoothness penalty if the disparity on the pixel $\mathbf{p}$ is different from neighboring pixels and balanced by a constant parameter λ. Total minimization of this function is NP-hard problem, however, some approximation approach is proposed, such as *Graph Cut* (GC) [7] , *Belief Propagation* (BP) [8] DP and so on. GC and BP methods can optimize disparity map highly precise, but these methods take high computational cost. These are not suitable for real-time application. Most of real-time stereo algorisms belong to the DP optimization.

In the DP, E_{smooth} is simplified as one directional penalty. On the contrast, GC and BP consider 4-connected direction. To solve one directional MRF, we use DP optimization, especially SO- like approach.

At first, we build up $F_{\mathbf{r}}$ which is sum of total minimum cost from start point to current pixel,

$$F_{\mathbf{r}}(\mathbf{p}, d_{\mathbf{p}}) = \min_{d_{\mathbf{r}}} \{F_{\mathbf{r}}(\mathbf{p}-\mathbf{r}, d_{\mathbf{r}}) + \lambda\varphi(d_{\mathbf{p}}, d_{\mathbf{r}})\} + E_{\mathrm{data}}(\mathbf{p}, d_{\mathbf{p}})$$

, where $\mathbf{r}$ indicates direction of smoothness, such as left (-1,0), right (1,0), up (0,1) and down (0,-1). $d_{\mathbf{p}}$ and $d_{\mathbf{r}}$ is the disparity of the current pixel and of the previous one respectively. The function φ is a penalty function which defined as various models, which is Potts, linear and truncated linear model;

Potts model:

$$\varphi(a,b) = \delta^{\mathrm{inv}}(a,b) = \begin{cases} 0 & (a=b) \\ 1 & (a \neq b) \end{cases}$$

Linear model:

$$\varphi(a,b) = |a-b|$$

Truncated linear model:

$$\varphi(a,b) = \min(|a-b|, T_g)$$

, where T_g expresses geometrical truncated cost value. After building up $F_{\mathbf{r}}$ cost, a disparity map is obtained by track back this cost function.

$$\mathrm{disp}(\mathbf{p}-\mathbf{r}) = \underset{d_r}{\operatorname{argmin}}(F_{\mathbf{r}}(\mathbf{p}-\mathbf{r}, d_{\mathbf{r}}) + \lambda\varphi(\mathrm{disp}(\mathbf{p}), d_{\mathbf{r}}))$$

DP is a fast optimization method. However, streaking noise effect is inevitable. To overcome this problem, applying dynamic programming with multi-direction [9, 14] is one of the solutions.

In the method of Kim et al. [14], firstly the DP of forward and backward direction along horizontal scan-line is processed, and then the buildup cost of F_{forward} and F_{backward} are summed. Finally the summed cost is involved by vertical direction DP. In our conventional approach, MPDP [13], similar optimization scheme is used over view dependent disparity estimation. The computational order of the approach is $O(3d^2)$. In detail, we compute SO 3 times, and SO approach computes the kth costs $d_{\mathbf{p}}$ per $d_{\mathbf{r}}$ which has the kth candidates, where k is the number of disparity labels, such as:

for $d_{\mathbf{p}}$ **from** 1 **to** k :

  for $d_{\mathbf{r}}$ **from** 1 **to** k :

$$F_{\mathbf{r}}(\mathbf{p}, d_{\mathbf{p}}) = \min_{d_{\mathbf{r}}}\{F_{\mathbf{r}}(\mathbf{p}-\mathbf{r}, d_{\mathbf{r}}) + \lambda\varphi(d_{\mathbf{p}}, d_{\mathbf{r}})\} + E_{\text{data}}(\mathbf{p}, d_{\mathbf{p}}).$$

However, if there is not tracking back process, we can reduce the order in O(d) by using min convolution and distance transform [10] introduced by Refs. [8,11].

We will show 3 smoothness models step by step. At first we replace $F_{\mathbf{r}}(\mathbf{p}\text{-}\mathbf{r},d)$ as $h(d) = F_{\mathbf{r}}(\mathbf{p}-\mathbf{r}, d)$.

In the case of Potts model, we can define Potts model cost $F_{\mathbf{r}}^{\text{Potts}}$ as:

$$F_{\mathbf{r}}^{\text{Potts}}(\mathbf{p}, d_{\mathbf{p}}) = \min(h(d_{\mathbf{p}}), \min_{d_{\mathbf{r}}} h(d_{\mathbf{r}}) + \lambda) + E_{data}(\mathbf{p}, d_{\mathbf{p}})$$

which means that which is smaller minimal cost of h + smoothness penalty or h cost expresses Potts model cost.

In the case of linear model, we use distance transform and 2 step min convolutions;

for $d_{\mathbf{p}}$ **from** 1 **to** k :

$$f(d_{\mathbf{p}}) = \min(h(d_{\mathbf{p}}), h(d_{\mathbf{p}}-1) + \lambda)$$

for $d_{\mathbf{p}}$ **from** $k-2$ **to** 0 :

$$F_{\mathbf{r}}^{\text{Linear}}(\mathbf{p}, d_{\mathbf{p}}) = \min(f(d_{\mathbf{p}}), f(d_{\mathbf{p}}+1) + \lambda) + E_{data}(\mathbf{p}, d_{\mathbf{p}})$$

In the case of linear truncation model, we apply linear model min convolution firstly, and then we truncate overshoot error with Potts model convolution.

$$F_{\mathbf{r}}^{\text{T-Linear}}(\mathbf{p}, d_{\mathbf{p}}) = \min(F_{\mathbf{r}}^{\text{Linear}}(\mathbf{p}, d_{\mathbf{p}}), \min_{d_{\mathbf{r}}} h(d_{\mathbf{r}}) + T_{\text{g}}) + E_{data}(\mathbf{p}, d_{\mathbf{p}})$$

The computational order of each model is $O(d)$, however, DP track back requires full search, so that we cannot avoid square order computation if nothing is done. Therefore we restrict MPDP without track back. This idea is simple; summing any directional cost $F_{\mathbf{r}}$, D = {left, right, up, down}

$$S(\mathbf{p}, d) = \sum_{\mathbf{r}\in D} F_{\mathbf{r}}(\mathbf{p}, d).$$

After computing the cost function, we can obtain the disparity map by winner-takes-all strategy:

$$\text{disp}(\mathbf{p}) = \operatorname*{argmin}_{d} S(\mathbf{p}, d).$$

This fast MPDP (FMPDP) method can solve disparity map by O(d) order. This approach is similar to Semi Global Matching (SGM) approach proposed by Hirschmuller [9]. If we use linear truncation model and set truncate threshold T_g=1, our method becomes same representation. In the paper [9], SGM use mutual information as the data term and compute 8 or more direction for noise suppression.

3.2 Outlier reduction

After computation of disparity map by MPDP or FMPDP, there are still streaking noise, so that we perform an outliner filtering [12].

$$\text{disp}^{\text{new}}(u,v) = \begin{cases} d_h & \text{if } \text{disp}(u-1,v) = \text{disp}(u+1,v) = d_h \\ d_v & \text{if } \text{disp}(u,v-1) = \text{disp}(u,v+1) = d_v \\ \text{disp}(u,v) & \text{otherwise} \end{cases}$$

This filter reduces weak streaking noise and its computational cost is quit low.

4. OCCLUTION DITECTION

Occlusion handling is important for view generation. Usually object boundary is occluded and then directs weighed interpolation makes synthesized image blur or ghosted.

We treat 8 occlusion cases; in the case of two views occlusion, left side image (LT and LB) is occluded from synthesized image , right side, top side and bottom side cases, in the case of three view occlusion only each view (LT,RT,LB,RB) is visible.

The occlusion mask occ(u,v) is computed by filtering estimated disparity map and threshold th. This threshold determines how pixel jump is occlusion. In this paper, this parameter is set to th = 2. The occlusion mask is computed from the following equations;

for d **from** $\text{disp}(u,v)$ **to** k :

  if $\text{disp}(u+\overline{\alpha}d, v) - \text{disp}(u,v) \geq th \quad occ(u,v) = O_L$

for d **from** $\text{disp}(u,v)$ **to** k :

  if $\text{disp}(u-\alpha d, v) - \text{disp}(u,v) \geq th \quad occ(u,v) = O_R$

for d **from** $\text{disp}(u,v)$ **to** k :

  if $\text{disp}(u, v+\overline{\beta}d) - \text{disp}(u,v) \geq th \quad occ(u,v) = O_T$

for d **from** $\text{disp}(u,v)$ **to** k :

  if $\text{disp}(u, v-\beta d) - \text{disp}(u,v) \geq th \quad occ(u,v) = O_B$

where k is the max label of disparity. After this filtering, there still exists full occlusion region. In this case, we replace the region where labeled as occlusion with non-occlusion region for avoiding complexity.

This occlusion mask controls the weighted parameter of α, β when synthetic ray is interpolated. If a pixel position in the mask has the left side occlusion flag O_L, the weighting parameter becomes $\alpha = 1$ to eliminate left side image and the case of other two view occlusion, the weight of ray is obeyed to the same rule. In the case of three view occlusion, such as O_L and O_T are flagged, the weight of only visible views becomes 1 and the others are set to 0. The computational order of this filtering becomes $O(d)$.

5. EXPERIMENTAL RESULTS

In the experiment, we use Tsukuba sequence of 5x5 camera array setup. The image resolution is 384 x 288 pixels. We have generated center view (S,T)=(2,2) from views {(1,1), (3,1), (1,3), (3,3)} and evaluated by PSNR on Y channel. We compare the 4 methods, which are Block Matching whose windows size is 5x5 (BM55), MPDP, FMPDP and BP. We employ BP as top performance optimization method. In addition, we use ground true (GT) disparity map for view generation.

Table.1 shows that PSNR of each method and occlusion detection effect, and also indicates computational time. These results have been computed by Intel Core i7 processor (2.67Hz Quad Core with *Hyper Threading Technology*), and the source codes are compiled by *Visual C++* and are parallelized by *Open MP*. The BP method has generated the highest quality image, and MPDP and FMPDP are the second best quality. From the aspect of computational cost, block matching has been the fastest and FMPDP is the second fastest. See Figure 3, which shows the free viewpoint images and dependent disparity map, and then BM55 has been highly noisy. The following results show that, fast multi-pass dynamic programming is suitable for real-time

applications and satisfies rendering image quality. Each method with occlusion detection improved PSNR except for BM55. It is because that occlusion detection highly depends on accuracy of disparity map, so that BM55 method is not enough quality to detect occlusions.

Table 1. Comparison of each method with / without occlusion detection and its computational time

	GT	BM55	BP	MPDP	FMPDP
no occ.[dB]	28.7	27.2	28.9	28.4	28.6
occ.[dB]	32.8	27.1	29.4	28.9	29.2
time[ms]	NA	38.1	397.7	76.2	51.5

Figure 3. Free viewpoint images (left) and view dependent disparity map with occlusion map (right); From top to Bottom methods are GT, BM55, BP, MPDP, FMPDP. Blue regions are horizontal (left and right) occlusions and red regions are vertical (top and bottom) occlusions. Green regions are full occlusion regions.

6. CONCLUSION

In this paper, we proposed the method which generates free viewpoint image with an optimization procedure, called multi-pass dynamic programming and its occlusion detection for real-time rendering. Experimental results show that the synthesized image from the camera array rendered by fast multi pass dynamic programming method has been 29.2 dB. On the contrary, images from belief propagation which is one of the best optimization algorism has been 29.4 dB and Block matching which is simplest and fastest method has been 27.2 dB. In addition, the MPDP computational time is almost real-time, and that time is 51.5 ms, meanwhile belief propagation takes 397.7 ms.

As our future work, we apply this algorism to larger images, such as high definition image, and implement multi-pass dynamic programming method on GPU.

7. REFERENCES

[1] M. Levoy and P. Hanrahan, “Light Field Rendering,” Proc. ACM Conference on Computer Graphics (SIGGRAPH'96), pp. 31-42, 1996.

[2] T. Fujii, T. Kimoto, M. Tanimoto, “Ray Space Coding for 3D Visual Communication,” Picture Coding Symposium '96, pp. 447-451, 1996.

[3] M. Tanimoto, “Overview of free viewpoint television,” Signal Processing: Image Communication, Vol. 21, Issue 6, pp. 454-461, July 2006.

[4] J. X. Chai, S. C. Chan, H. Y. Shum, X. Tong, “Plenoptic sampling,” Proc. SIGGRAPH ’00, pp.307–318, 2000.

[5] N. Fukushima, T. Yendo, T. Fujii, M. Tanimoto, “Real-time Arbitrary View Interpolation and Rendering System using Ray-Space,” Proc. of SPIE OpticsEast ITCom, vol. 6016, pp. 250-261, Jan. 2005.

[6] D. Scharstein and R. Szeliski, “A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms, International Journal of Computer Vision,” Vol. 47, Issue 1-3, pp. 7-42, Apr. -June 2002.

[7] Y. Boykov, O. Veksler and R. Zabih, “Fast Approximate Energy Minimization via Graph Cuts,” IEEE Trans. PAMI, Vol. 23, No. 11, Nov. 2001.

[8] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient Belief Propagation for Early Vision,” International Journal of Computer Vision, Vol. 70 , Issue 1, pp. 41 - 54, Oct. 2006

[9] H. Hirschmuller, “Stereo Processing by Semi-Global Matching and Mutual Information,” IEEE Trans. on PAMI, Vol. 30, Issue 2, pp. 328-341, Feb. 2008.

[10] G. Borgefors, “Distance transformations in digital images,” Computer Vision, Graphics and Image Processing, Vol. 3, No. 3, pp. 344 - 371, June 1986.

[11] P. F. Felzenszwalb and D. P. Huttenlocher, “Distance transforms of sampled functions,” Cornell Computing and Information Science Technical Report TR2004-1963, Sep. 2004.

[12] S. Birchfield and C. Tomasi. “Depth Discontinuities by Pixel-to-Pixel Stereo,” International Journal of Computer Vision, Vol. 35, No. 3, pp. 269-293, Dec. 1999.

[13] N. Fukushima, T. Yendo, T. Fujii, M. Tanimoto, “Free View-Point Image Generation Using Multi-Pass Dynamic Programming,” Proc. SPIE, vol. 6490, 64901F, Jan. 2007.

[14] J. Kim, K.M. Lee, B.T. Choi, and S.U. Lee. “A dense stereo matching using two-pass dynamic programming with generalized ground control points,” IEEE CVPR, Vol II: 1075-1082, 2005.