3D Res. **04**, 02(2013)1
10.1007/3DRes.02(2013)1

3DR EXPRESS

A Novel Ray-Space based View Generation Algorithm via Radon Transform

Lingfeng Xu • Ling Hou • Oscar C. Au • Wenxiu Sun • Xingyu Zhang • Yuanfang Guo

Received: 24 November 2012 / Revised: 05 January 2013 / Accepted: 18 February 2013

Abstract Ray-space interpolation is one of the key technologies to generate virtual viewpoint images, typically in epipolar plane images (EPI), to realize free viewpoint television (FTV). In this paper, a novel Radon transform based ray-space interpolation algorithm (RTI) is proposed to generate virtual viewpoint images in the EPI. In the proposed RTI, feature points of each EPI are first extracted to form the corresponding feature epipolar plane image (FEPI). Radon transform is then applied to each FEPI to detect the candidate interpolation direction set. Then corresponding pixels in neighboring real view rows for each pixel to be interpolated are found by some block matching based interpolation method, in which the smoothness property of the disparity field and the correlation among neighboring EPIs are explored. Possible occlusion regions are processed by the proposed one-sided interpolation. Finally, to solve the problem that the cameras are not equally spaced, a novel ray-space based spacing correction algorithm is proposed to correct the EPI such that pixels corresponding to the same scene point will form a straight line. Experimental results suggest that the proposed RTI and the spacing correction algorithm can achieve good performance. Moreover, compare with traditional methods, our proposed algorithm can generate good interpolation result even if apriori depth range is not known.

Keywords Multi-view, Radon transform, ray-space, camera spacing correction, epipolar plane image, free view generation

Lingfeng Xu (✉) • Ling Hou • Oscar C. Au • Wenxiu Sun • Xingyu Zhang • Yuanfang Guo
Department of Electronic and Computer Engineering,
The Hong Kong University of Science and Technology,
Clear Water Bay, Hong Kong
E-mail: lingfengxu@ust.hk

1. Introduction

In recent years, multi-view technology has been becoming popular as its applications such as three-dimensional television (3DTV) and free viewpoint television (FTV) give people the opportunity to see the scene from different viewpoints and provide customers with high interactivity[1]. Since cameras cannot be placed continuously in space due to its bulky size, only a finite number of views (called real views) can be captured. Generating arbitrary views between the captured real views is a problem to be solved in realizing 3DTV and FTV systems[2]. Many free view image generation algorithms have been proposed in the literature and can be classified as image domain based[3], ray-space based[4-6], surface light field based[7] and model based[8] methods. Among these methods, image domain based methods perform rendering with geometry compensation. Surface light field and model based methods apply rendering from approximate geometry. And ray-space based methods apply rendering without geometry. In particular, ray-space based methods are attractive as they require almost no scene information and can generate any viewpoint image without complicated analysis and rendering process. This paper is about ray-space based free view generation.

In general, there are two major approaches for ray-space interpolation. In the first approach, some possible interpolation choices (e.g. filters or vectors) are identified apriori and then the captured real view images are analyzed to find the best choice to do the interpolation. One such method proposed by T. Kobayashi is to achieve ray-space interpolation by adaptive filtering[9]. In this algorithm, a number of filters with respect to different directions are defined apriori in the *Epipolar Plane Images* (EPI). For a pixel to be interpolated the pixels in the neighboring real view rows in the ray-space are analyzed and the best filter is determined by minimizing the output variance of each filter. A similar method proposed by T. Fujii predefines a number of vectors instead of filters[10]. A selector determines the best vector based on the analysis of the neighborhood area, and the up-sampled block is replaced by the best vector. One problem of this kind of method is that it is hard to predefine filters or vectors that should work universally well in different situations.

This paper is about the second approach, in which no predefined filters or vectors are used. Instead, some searching is performed on the captured real view images to determine locally optimal interpolation direction to be followed by the corresponding linear interpolation. Two such methods, the *Pixel Matching based Interpolation* (PMI) and *Block Matching based Interpolation* (BMI) algorithms, are proposed by Tehrani[11]. For a pixel to be interpolated in PMI, a search region is defined in two neighboring real view rows in the EPI. Then the absolute difference between a pair of pixels (one from each real view row) for each possible interpolation direction is calculated and the best pixel pair with minimum absolute difference is found. On the other hand, for a pixel to be interpolated in BMI, a similar search region is defined in the two real view rows in the EPI. For each possible interpolation direction, instead of calculating the absolute difference between just a pair of pixels, BMI defines a 1-D neighborhood around the pixel in each of the two real view rows, and the 1-D neighborhood forms a 1-D block. It then calculates the mean square difference between the two 1-D blocks, effectively performing 1-D block matching, to find the best-matched pixel pair. In both PMI and BMI, the virtual view pixel is linearly interpolated from the best matched pixel pair. Typically, BMI is more robust and thus more desirable than PMI especially when noise exists. To improve the performance of BMI, Jiang proposed the *Multi-Epipolar Lines based ray-space Interpolation* (MELI), in which three or more neighboring real view rows (instead of only two) are used in the block matching[12]. However, a common problem in existing methods is that it is hard to preset a reliable candidate interpolation direction set for all situations. If the candidate set is too small, the true corresponding pixels may be out of range. And if the set is too large, there can be many false alarms of which the MSE may be small but are totally wrong.

In recent years, some methods have been proposed to improve the performance by considering the smoothness constraint on the interpolation directions of neighboring pixels. Zhang and Yang proposed some methods[13, 14] based on the assumption that the disparity field among adjacent views are smooth. They formulated the interpolation problem into an energy minimization problem which contains a matching cost term and a smoothness term, and then minimized the cost function by dynamic programming. However, the computation complexity was extremely high, compare with BMI, PMI and MELI. Another problem is that the smoothness term is hard to choose. If the term is too large, depth discontinuity across object boundaries is hard to preserve resulting in wrong interpolation direction. And, if it is small, interpolation direction can be erroneous in noisy regions and in regions with repetitive patterns.

In order to solve the problems in the existing methods, we propose a novel algorithm called *Radon Transform based Interpolation* (RTI) in this paper by considering the properties of EPI and Radon transform. The main advantage of RTI is that the size of the candidate interpolation direction set is moderate and a reliable candidate interpolation direction set can be obtained without prior knowledge of scene depth information. Meanwhile, an optimal interpolation direction detection algorithm is proposed to achieve improved accuracy over BMI by replacing the traditional 1D matching block by a 2D one, and by adding an adaptive smoothness constraint. A novel occlusion detection and interpolation method is also introduced. The rest of the paper is organized as follows: In Section 2, we review some basic properties of the ray-space representation, including the definition of EPI. In Section 3, the proposed Radon transform based ray-space interpolation (RTI) algorithm is introduced. In Section 4, a novel ray-space based camera spacing correction algorithm for multi-view images is proposed. Experimental results of the proposed RTI are shown in Section 5 and conclusions are given in Section 6. Some preliminary version of this paper is reported[15, 16].

2. Review of Ray-Space Representation

In ray-space representation, each ray in the 3D world is represented by a corresponding point in the ray-space domain. The propagation path of a ray going through space can be uniquely parameterized by the location of a point (x, y, z) and a direction (θ,ϕ) [17]. A function $f(\cdot)$ is introduced to describe the light intensity of the ray passing through any 3D location (x, y, z) at any angle (θ,ϕ) as:

$$f(x,y,z;\theta,\phi),\quad -\pi\le\theta<\pi,-\pi/2\le\phi<\pi/2 \tag{1}$$

In the ray-space representation, a pin-hole camera is assumed as an image acquisition device. For the camera whose optical center is at (x_0,y_0,z_0), the intensity of the rays going through the pin-hole in any orientation are given by:

$$f(x,y,z;\theta,\phi)\,|_{x=x_0,y=y_0,z=z_0} \tag{2}$$

Although the 5-D ray-space mentioned above is very general and can describe fully the view from any point and orientation in the space, it is often considered too complicated and impractical for real applications. Thus, the function is sometimes reduced to 4-D as $f(x,y;\theta,\phi)$ by considering only rays that arrive at a reference plane ($z=0$), where (x,y) denotes the intersection of the ray and the reference plane $z=0$, (θ,ϕ) is the direction of the ray, and $f(x,y;\theta,\phi)$ represents the intensity of the ray. Often, it is further simplified to 3D by considering one particular ϕ at a time[6].

In this paper, we will mainly be considering $\phi=0$. We assume all the cameras are parallel and are placed perpendicular to the reference plane ($z=0$). Moreover, all the optical centers are along a baseline ($y=0$) in the reference plane. If the cameras are not parallel, the captured multi-view images can be rectified first by using camera parameters[18,19] or homographic transformations[20,21]. Suppose there are N cameras with identical setting and they are equally spaced. Let I_i be the *multi-view image*, of size $W\times H$, captured by Camera i for $i=0,1,2,...,N-1$.

Camera 0 is on the far left and Camera $N-1$ is on the right.

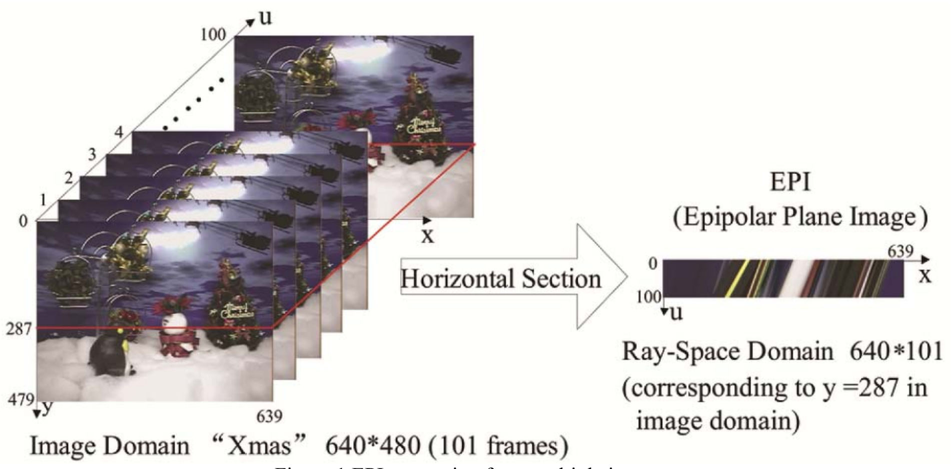


Figure 1 EPI generation from multiple images

An *epipolar plane image* (EPI) in the ray-space domain, E_j, of size $W \times N$ is constructed by collating the j^{th} rows of the N images[22], for $j = 0,1,2,...,H-1$. An example of the "Xmas" multiview test sequences is shown in Figure 1 with $N = 101$, $H = 480$ and $W = 640$. The EPI shown is E_{287} formed by combining row 287 of all images. The top row of E_{287} comes from the leftmost view ($i = 0$) while the bottom row is from the right ($i = 100$). An important property of EPI is that any scene point should appear in the same rows in consecutive multi-view images. They should thus appear in consecutive rows in the corresponding EPI forming a straight line whose slope is proportional to the depth of the scene point, since the cameras are co-linear, parallel, and equally-spaced. This can be proved easily from the projective camera model shown in Figure 2.

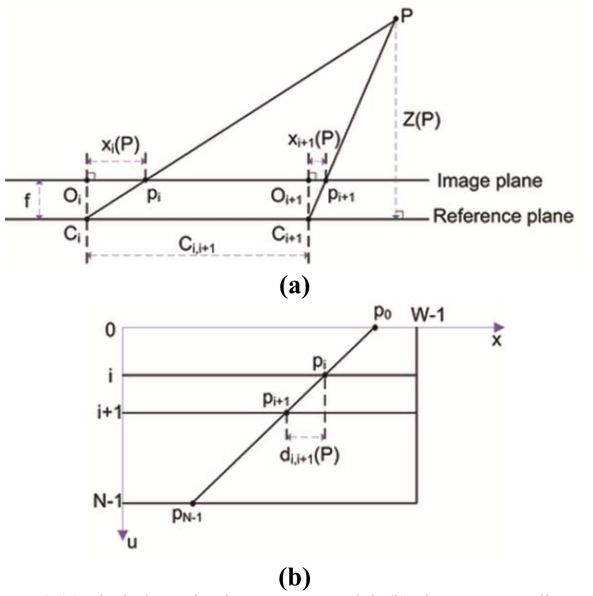


Figure 2 (a) Pin-hole projective camera model, (b)The corresponding EPI

In Figure 2(a), P is a *scene point* of interest at a depth of $Z(P)$. Without loss of generality, let $y = 0$ for P. All the cameras have identical focal length f and are placed equally along the baseline in the reference plane, with C_i being the optical center of Camera i, and O_i being where its optical axis intersects with the image plane. The *camera spacing* between C_i and C_{i+1} is $C_{i,i+1}$. The scene point appears as a point p_i in I_i at a horizontal displacement of x_i relative to O_i, where x_i can be positive or negative. In the local image coordinate in I_i, the coordinate of O_i is ($\frac{W-1}{2}$, $\frac{H-1}{2}$). Based on triangle similarity, we can get:

$$\frac{C_{i,i+1} - (x_i(P) - x_{i+1}(P))}{C_{i,i+1}} = \frac{Z(P) - f}{Z(P)} \quad (3)$$

which can be simplified to give

$$d_{i,i+1}(P) = x_i(P) - x_{i+1}(P) = \frac{f \cdot C_{i,i+1}}{Z(P)} \quad (4)$$

and $d_{i,i+1}(P)$ is called the *disparity* (or relative displacement in local coordinate) between the pixels p_i and p_{i+1}. As $Z(P)$ and focal length f are fixed, the disparity $d_{i,i+1}(P)$ depends only on the camera interval $C_{i,i+1}$. As the cameras are equally spaced, the $C_{i,i+1} = C$ is constant for all i and thus the disparity $d_{i,i+1}(P) = d(P)$ is constant for all i.

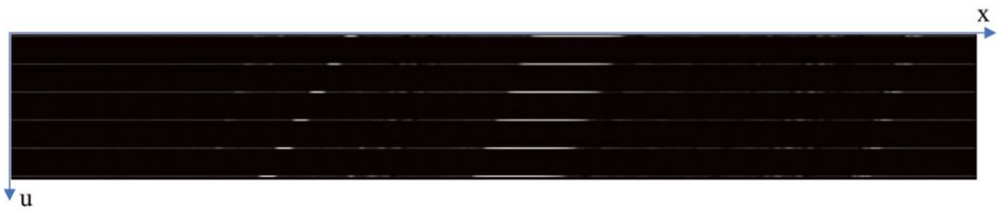
In the corresponding EPI shown in Figure 2(b), the scene point P appears as p_i in row i and p_{i+1} in row $i+1$. The horizontal distance between p_i and p_{i+1} is the disparity $d_{i,i+1}(P)$ which is constant for all i. The vertical distance between p_i and p_{i+1} is simply 1, constant for all i. Thus the slope between p_i and p_{i+1} is

$$s_{i,i+1}(P) = \frac{1}{d_{i,i+1}(P)} \quad (5)$$

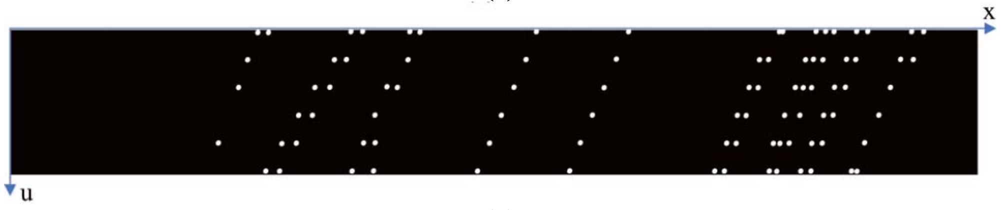
which is the same for all i. In other words, all the points p_i form a straight line in EPI with slope $s(P) = Z(P)/(fC)$. And all points in EPI belong to some straight line segments. And an EPI contains multiple straight lines each of which corresponds to one scene point. The problem of view synthesis is to generate any free viewpoint image which we will call "virtual view image" that would be captured by a virtual camera placed anywhere along the baseline between the left-most and right-most cameras. Without loss of generality, we model our problem as the generation of a virtual view image captured by the virtual camera at a displacement of ΔC relative to C_i, where $0 < \Delta C < C$. The advantage of this modeling is that, for a scene point P that appears in a particular EPI, it corresponds to a pixel in the virtual view image should also appear in the same EPI. And it should lie on the same straight line with slope $s(P)$. In other words, the problem of view synthesis can be viewed as a ray-space (i.e. EPI) interpolation problem such that the straight line structures in the EPI are maintained. In this problem, the major challenge is to identify accurately the line structures in the EPI. We will propose a novel method called RTI to address this problem in the next section.

3. Proposed Radon Transform based Ray-Space Interpolation (RTI)

A main task in ray-space interpolation is to find the line structure direction in each EPI for each pixel to be interpolated. While the direction may be obvious for some pixels, it can be confusing for some pixels as there may be many competing candidate directions, especially at regions with highly repetitive structures. In the proposed *Radon Transform based Interpolation* (RTI), our strategy is to identify a set of candidate interpolation directions and compute some distortion measure for each candidate direction in search of the optimal one with least distortion. Preliminary experiments suggest that the interpolation results can be highly sensitive to the "reliability" of the candidate interpolation direction set. If the candidate set is too large, the chance of finding a wrong direction is elevated. If it is too small, the correct direction may not be included. To achieve high reliability of the candidate interpolation direction set and the performance of ray-space interpolation, RTI extracts some robust features of an EPI and forms another feature image called *feature epipolar plane images* (FEPIs) and applies Radon transform on each FEPI. The Radon transform of FEPI is analyzed to obtain a reliable candidate interpolation direction set. Finally, an block-based matching algorithm based on BMI is proposed to find the optimal interpolation direction for each pixel to be interpolated.



(a)



(b)

Figure 3 (a) An example of a EPI (b) The corresponding FEPI.

3.1 Proposed RTI: Noise-suppressed FEPI Generation

Let $I_i(x,j)$, with $x=0,1,...,W-1$, be the j^{th} row of multi-view image I_i captured by Camera i. Then the j^{th} EPI, E_j, of size $W\times N$, is formed by combining the j^{th} row of all the N multi-view images such that i^{th} row of E_j satisfies

$$E_j(x,i)=I_i(x,j) \quad (6)$$

From each E_j, one FEPI, F_j, of size $W\times N$ is generated. FEPI is a kind of edge map of EPI taking on values of 0 or 1. While straight-forward edge detection can be done by applying high-pass filter and thresholding, it is sensitive to noise which may be generated during image capture and digitization. Instead, to suppress the noise, we apply a Gaussian filter to pre-smooth the multi-view images. We generate an intermediate edge map D_2 from which F_j is defined.

$$D_2(x,i)=\begin{cases}1, & \text{if } D_1(x,i)>T\\ 0, & otherwise\end{cases} \quad (7)$$

wher $D_1(x,i)=|E'_j(x,i)-E'_j(x-1,i)|$ is the horizontal gradient of E', the EPI obtained from the Gaussian filtered multi-view images. In our experiments, we find $T_1=\overline{D_1}+\sigma_{D_1}$ to give reasonable results, where $\overline{D_1}$ and σ_{D_1} are the mean and standard deviation of D_1. Often, edges in images can be blurry in real cases[23] especially after Gaussian smoothing and consequently the "candidate" feature pixels (those with $D_2(x,i)=1$) often appear consecutively at the genuine edge regions. On the other hand, false alarms, i.e. non-edge pixels with $D_2(x,i)=1$, tend to appear in isolation in noisy non-edge regions. Thus, we reject groups of *consecutive* candidate pixels with runlength shorter than a threshold T_2. For a group with runlength larger than or equal to T_2, the pixel in the group with the largest $D_1(x,i)$ is classified as a feature point with $F_j(x,i)=1$. Otherwise, $F_j(x,i)=0$. In our experiments, we find that $T_2=4$ gives good results. In FEPI, *feature points* are those pixels marked as 1. As discussed above, the feature points corresponding to the same scene point P should form a straight line, which we call a *feature line*, in FEPI.

A typical FEPI and its corresponding EPI are shown in Figure, from which we can see that FEPI often contain multiple feature lines. A smooth object would tend to have two parallel feature lines, one on each side of the object. An object with complicated textures and fine details can have many feature lines. Each feature line in the FEPI gives a candidate interpolation direction for the pixels to be interpolated in the EPI. If the feature line lies at the boundary between a foreground object and a background object, its slope would give a good interpolation direction for the foreground side. If both sides of a feature line have the same depth (distance from camera) such as a logo on the surface of a box, its slope gives a good interpolation direction for both sides. Unfortunately, some false candidate directions can be formed when points from different feature lines "accidentally" line up to form an almost-straight line. This is especially likely when neighboring feature lines are parallel and equally spaced.

3.2 Proposed RTI: Radon Transform of FEPI

Since the slope of each feature line in FEPI gives a candidate interpolation direction and Radon transform is a useful tool to detect straight lines, we apply *Radon transform* to each FEPI to detect feature lines in it. Radon transform is named after J. Radon who described a function in terms of its integral projections[24]. According to Figure 4, the *integral projection* of a digital image $g(x,y)$ at the angle θ and distance ρ is the summation of its intensity values along the straight line l whose points (x,y) satisfy:

$$\rho = x\cos\theta + y\sin\theta \tag{8}$$

The line l is perpendicular to a radial line at angle θ and is at a distance of ρ from the image center. If the image is our FEPI, the integral projection will be large if l is along a feature line. Otherwise, it should be small.

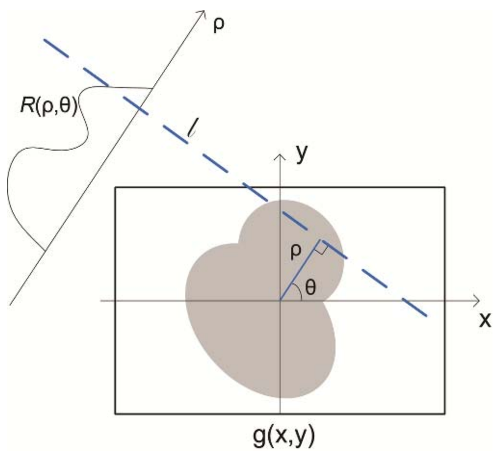
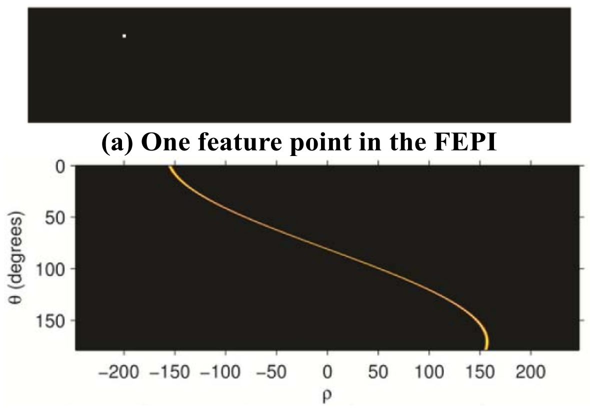


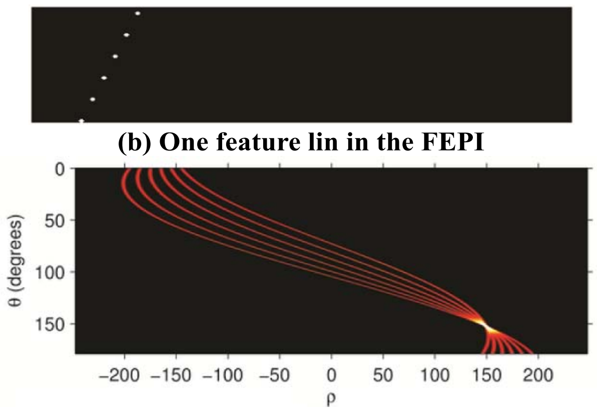
Figure 4 Radon transform

The Radon transform[25] of a 2D function $g(x,y)$ is defined as:

$$R(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\delta(\rho - x\cos\theta - y\sin\theta)dxdy \tag{9}$$

where $\delta(\cdot)$ is the Dirac delta function.





(c) Radon transfrom result corresponding to (a) **(d) Radon transform result corresponding to (b)**

Figure 5 Radon transform for the feature point and feature line

In our proposed RTI algorithm, we apply Radon transform to each FEPI and perform detection in the Radon domain. Note that each FEPI can be considered as a sum of "basic" FEPIs each being the indicator image of one feature point. As Radon transform is linear, the Radon transform of an FEPI with multiple feature points is equivalent to the sum of Radon transform of the corresponding basic FEPIs. Thus, before introducing our algorithm, it is useful to examine the Radon transform of such basic FEPI. Figure 5 (a) shows a typical basic FEPI. In this example, FEPI is of size 640×101 such that x ranges from -320 to 319, and u ranges from -50 to +50, and the feature point is at $(x,y)=(-216,-30)$. Figure 5 (c) is the corresponding Radon transform, with vertical and horizontal axis being θ and ρ respectively. The range of θ is from 0 to 180. The ρ is from -324 to +324 ($324=\sqrt{640^2+101^2}/2$). The Radon transform of a basic FEPI always contains a curve with magnitude 1 in Radon transform domain because, for every θ, only one value of ρ gives non-zero value which is 1.

For the basic FEPI:

$$F(x,y)=\begin{cases}1 & if\ x=x_0,\ y=y_0\\ 0 & Otherwise.\end{cases} \quad (10)$$

Its Radon transform is:

$$\begin{aligned}R(\rho,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}F(x,y)\delta(\rho-x\cos\theta-y\sin\theta)dxdy\\ &= F(x_0,y_0)\delta(\rho-x_0\cos\theta-y_0\sin\theta)\\ &= \begin{cases}1, & if\ \rho=x_0\cos\theta+y_0\sin\theta\\ 0, & Otherwise.\end{cases}\end{aligned} \quad (11)$$

On the curve $\rho=x_0\cos\theta+y_0\sin\theta$, we have $\rho=x_0$ for $\theta=0^\circ$, $\rho=y_0$ for $\theta=90^\circ$, and $\rho=-x_0$ for $\theta=180^\circ$.

For a more general case, in which multiple feature points form a straight line:

$$F(x,y)=\begin{cases}1 & f(x,y)\in A\\ 0 & Otherwise.\end{cases} \quad (12)$$

where, for some Δx and Δy,

$$A=\{(x_0,y_0),(x_0+\Delta x,y_0+\Delta y),\cdots,\\ (x_0+(N-1)\Delta x,y_0+(N-1)\Delta y)\}$$

Since each feature point corresponds to one curve in the Radon transform domain, there are N curves described by

$$\rho=x_0\cos\theta+y_0\sin\theta;$$
$$\rho=(x_0+\Delta x)\cos\theta+(y_0+\Delta y)\sin\theta;$$
$$.$$
$$\rho=(x_0+(N-1)\Delta x)\cos\theta+(y_0+(N-1)\Delta y)\sin\theta;$$

And these N curves will intersect at one point (ρ^*,θ^*) in the Radon transform domain that satisfies:

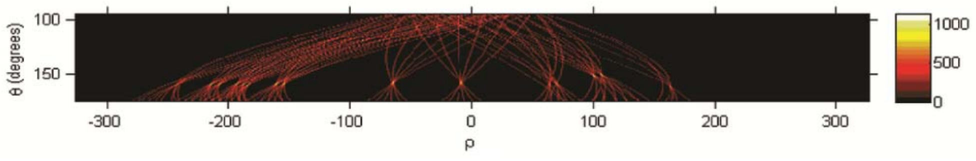
$$(x_0+k\Delta x)\cos\theta^*+(y_0+k\Delta y)\sin\theta^*=x_0\cos\theta^*+y_0\sin\theta^*$$

for $k=0,1,2,\cdots,N-1$. And

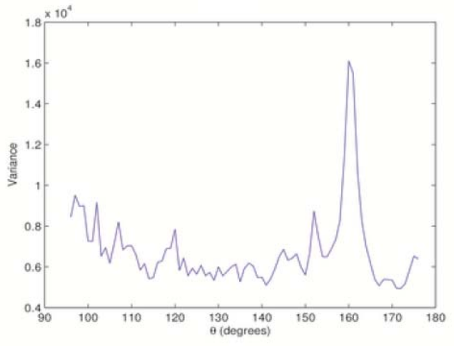
$$\theta^*=\arctan(-\Delta x/\Delta y) \quad (13)$$

$$\rho^*=x_0\cos\theta^*+y_0\sin\theta^* \quad (14)$$

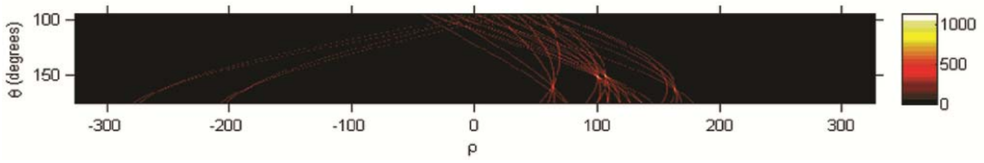
Figure 5(b) contains five feature points forming a feature line. Each feature point corresponds to one curve of magnitude 1 in Radon domain in Figure 5(d). As the five feature points form a feature line with some (ρ,θ)), all five curves will go through the point (ρ,θ) in Radon domain, and the Radon transform value at (ρ,θ) is 5, the sum of Radon transform values at (ρ,θ) of five curves, which is a local peak or maximum. A feature line in FEPI domain corresponds to a local peak in Radon domain. Therefore, a search for feature lines in FEPI domain is equivalent to a search for peaks in Radon domain.



(a)



(b)



(c)

Figure 6 (a)The Radon transform result of the 283rd FEPI of “Xmas" sequence(camera interval is 60mm). (b)Variance of ρ corresponding to θ. (c)The Radon transform result of FEPI in which parallel feature lines are removed

3.3 Proposed RTI: Radon Transform Based Candidate Direction Search

In the proposed RTI algorithm, we apply Radon transform to each FEPI and seek to search for possible feature lines and use those as candidate interpolation directions for the EPI. Firstly, we estimate the number of feature lines to be found. As shown in Figure 3(b), different rows of FEPI can have different numbers of feature points and each number is an estimate of the number of feature lines. In our experiments, we find that the median N_m of the row-wise feature point numbers is a reasonable estimate of the actual number of feature lines. We will seek to find N_m peaks (or feature lines) in the Radon transform domain of the FEPI.

The Radon transform of Figure 3(b) is shown in Figure 6 (a). As mentioned before, each peak (local maximum) in Radon domain corresponds to a possible feature line in the FEPI. In our experiments, we observe that false alarms (i.e. peaks that do not correspond to true feature lines[26] may occur in Radon domain. In each FEPI, there are often multiple parallel feature lines corresponding to scene points at the same depth. Sometimes the parallel lines are somewhat equally spaced. In such cases, the feature points from different feature lines can line up into a straight line in another angle resulting in a false alarm situation in the Radon domain.

To overcome the false alarms, we observe that the Radon transform value usually has the largest variance in the direction with the most number of parallel feature lines as shown in Figure 6(b). Such a dominant direction is typically very reliable and we choose it as a candidate interpolation direction. As the many parallel feature lines can cause significant ambiguity, we remove the feature points corresponding to this direction in the FEPI, and redo the Radon transform. From our experiments, we found that the removal of the dominant direction features can suppress the ambiguity very much. The followings are the details.

Firstly, we find the dominant angle $\hat{\theta}$ which has the largest variance:

$$\hat{\theta} = \arg\max_{\theta} (\sigma^2_{R(\theta)}) \tag{15}$$

Then we find the significant peaks, $(\rho_1, \hat{\theta}), (\rho_2, \hat{\theta}), \ldots, (\rho_{N_p}, \hat{\theta})$, at the dominant angle, with Radon transform values greater than $\zeta = \alpha \max_\rho \{R(\rho, \hat{\theta})\}$ for some $\alpha < 1$, such that $R(\rho_1, \hat{\theta}) \geq R(\rho_2, \hat{\theta}) \geq \ldots \geq R(\rho_{N_p}, \hat{\theta}) \geq \zeta$. Let N_p be the number of such significant peaks. To suppress false alarm, we remove all the feature points on the feature lines corresponding to these significant peaks. After these dominant direction feature points (with $\theta = \hat{\theta}$) are removed from the FEPI, Radon transform is applied again, as shown in Figure 6(c). Compared to Figure 6(a), a large number of Radon locations with large magnitude and $\theta \neq \hat{\theta}$ disappear in Figure 6(c). Finally, another $(N_m - N_p)$ largest peaks are extracted in Figure 6(c). For each candidate angle θ, the corresponding candidate slope s can be computed as $s = -\cot(\theta)$. In other words, the set of candidate directions corresponds to a set of candidate slopes.

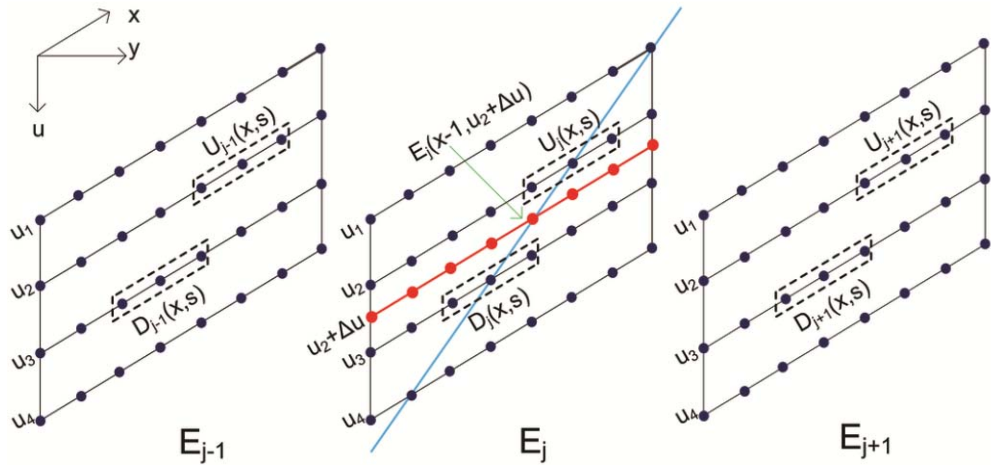


Figure 7 Proposed improved matching block searching method

3.4 Proposed RTI: Optimal Interpolation Direction Detection

In this section, for each pixel to be interpolated, a novel and robust method is proposed to choose the optimal interpolation direction from the candidate direction set. Four of the N cameras are shown in Figure 7 with four corresponding multi-view images I_i. Let u_i be the location of Camera i. (e.g. $u_1 = 1, u_2 = 2, u_3 = 3, u_4 = 4$.) Suppose we want to synthesize a virtual view between u_2 and u_3, at position $u_2 + \Delta u$ where $0 < \Delta u < (u_3 - u_2)$. (e.g. $\Delta u = 0.3$.) In the proposed RTI, we synthesize the virtual view by synthesizing the corresponding virtual row in each EPI. Without loss of generality, consider the j^{th} EPI, E_j.

The proposed RTI is an improvement of an existing ray-space interpolation method called block matching based interpolation (BMI). As a motivation of the proposed RTI, here we briefly describe BMI and its weakness. Consider a pixel to be interpolated at $E_j(x, u_2 + \Delta u)$. In BMI, a number of candidate directions are defined. The feature line l at slope s corresponds to a scene point P in the real world. The feature line l intersects row u_2 of E_j at location $x_2' = x + \Delta u / s$, and row u_3 at location $x_3' = x - (u_3 - u_2 - \Delta u)/s$. And $x_2 = round(x_2')$ and $x_3 = round(x_3')$. A 1D window U_j of width $2L+1$ is defined around x_2 in row u_2 of E_j. A similar window D_j of same width is defined around x_3 in row u_3 of E_j. In BMI, the mean square error (MSE) between U_j and D_j is computed for each candidate s and the s^* with minimum MSE is chosen. Linear interpolation between row u_2 and row u_3 along s^* is applied. One weakness of BMI is that the resulting s^* is often not accurate. Different L gives rise to different weakness of BMI. A small L implies the two windows U_j and D_j are short and probably are not sufficient to represent the local image characteristics. A large L can capture more local features, but there is a higher probability that the larger window contain more than one objects in which case the block matching does not work. In the proposed RTI, we seek to improve the accuracy of s^* by: (1) using relevant small windows in 3 adjacent EPI (E_{j-1} , E_j and E_{j+1}), (2) imposing an adaptive smoothness constraint on the s^* of adjacent pixels, and (3) special processing for occlusion cases. The small window in (1) ensures that there is little chance of including more than one object. The use of multiple EPI is to guarantee there is enough relevant information to represent the local image characteristics. The smoothness constraint in (2) ensures neighboring pixels in the virtual row have similar slope and thus similar depth. When occlusion occurs, it is unreasonable to apply BMI on row u_2 and row u_3 and thus we introduce special provision in (3).

3.4.1 Improving block matching by using 2-D blocks over neighboring EPIs

Recall that U_j is centered around x_2 in row u_2 of E_j , which is x_2 in row j of I_{u_2} . Similarly, D_j is centered around x_3 in row u_3 of E_j , which is x_3 in row j of I_{u_3} . Effectively BMI requires the 1-D neighborhood of size $(2L+1)\times 1$ in I_{u_2} to be similar to the corresponding 1-D neighborhood in I_{u_3} . In the proposed RTI, we improve BMI by replacing the 1-D blocks U_j and D_j by 2-D blocks. We require a small 2-D neighborhood in I_{u_2} to be similar to that in I_{u_3} . The small 2-D neighborhood (or block) we used in RTI is $(2L'+1)\times(2Q+1)$, where $L' < L$ and Q is small. The 2-D block in I_{u_2} is obtained by combining U_{j-Q} , ..., U_{j-1} , U_j , U_{j+1} , ..., U_{j+Q} , where U_k is centered around x_2 in row u_2 of E_k for $k = j-Q, ..., j+Q$. Similarly, the 2D block in I_{u_3} is obtained by combining D_{j-Q} , ..., D_{j-1} , D_j , D_{j+1} , ..., D_{j+Q} defined similarly. In our experiments, we find $Q=1$ is a good tradeoff between performance, computational complexity and memory cost. The optimal interpolation direction s^* is chosen from candidate slopes based on the criteria that will be introduced later.

3.4.2 Adding an adaptive constraint to the cost function to smooth the disparity field

It is well known that the disparity field should be essentially smooth. Thus the slopes of adjacent pixels in EPI should be similar. Therefore, in RTI, we impose an adaptive smoothness constraint in the cost function. Let the 2-D block of size $(2L'+1)\times(2Q+1)$ in I_{u_2} be represented as a row-ordered 1-D vector U of size $(2L'+1)(2Q+1)\times 1$. Similarly, let the 2D block in I_{u_3} be represented as a 1D vector D . As they depend on the position x and candidate slope s , we write them as $U(x,s)$ and $D(x,s)$. Let I_0 be a 1D vector with all elements being 1. Let m_U and m_D be the mean of the elements of U and D respectively. The optimal interpolation direction $s^*(x,u_2+\Delta u)$ for pixel $E_j(x,u_2+\Delta u)$ is the $s \in S_j$ that minimize the cost function:

$$C_{u_2,\Delta u}(x,s) = \left\| \begin{matrix}(U(x,s)-m_U(x,s)I_0) \\ -\ (D(x,s)-m_D(x,s)I_0)\end{matrix} \right\|_{l_2}^2 + \lambda \left| \frac{u_3-u_2}{s} - \frac{u_3-u_2}{s_{x-1}^*} \right| \qquad (16)$$

where $s_{x-1}^* = s^*(x-1,u_2+\Delta u)$ is the optimal slope for pixel $E_j(x-1,u_2+\Delta u)$, and S_j is the candidate direction set for E_j .

Although we assume the camera settings are identical in all cameras, there can be brightness difference between multi-view images I_{u_2} and I_{u_3} due to imperfect camera calibration, CCD noise, and camera position in real situations[27]. To account for such possible discrepancy, we subtract the mean from the 2D blocks and compute the MSE between the two mean-subtracted blocks in the first term in Eqn. 16. While λ can be set to be constant, we choose to adjust it adaptively as

$$\lambda = exp\{\psi - \left\| \begin{matrix}(U(x-1,s_{x-1}^*)-m_U(x-1,s_{x-1}^*)I_0) \\ -\ (D(x-1,s_{x-1}^*)-m_D(x-1,s_{x-1}^*)I_0)\end{matrix} \right\|_{l_2}^2 \}$$

If the MSE in the previous pixel (at $x-1$) is small, its optimal slope s_{x-1}^* tends to be more trustworthy and we prefer to use a larger λ to put more weight on the smoothness term. And when the previous MSE is large, we do not trust it as much and use a smaller weight for the smoothness term. The parameter ψ is used to control the smoothness strength. Compared with constant λ , our choice of λ can allow faster change in s when moving from occlusion region to dis-occlusion region. Meanwhile, it can also reduce the error permutation in the case that wrong interpolation direction are found the previous pixel. When the optimal direction s^* is found, we simply apply linear interpolation to synthesize the virtual pixel

$$E_j(x,u_2+\Delta u) = (1-\alpha)E_j(x_2',u_2) + \alpha E_j(x_3',u_3) \qquad (17)$$

where $\alpha = \dfrac{\Delta u}{u_3-u_2}$, $x_2' = x + \dfrac{\Delta u}{s^*}$ and

$x'_3 = x - \dfrac{(u_3 - u_2 - \Delta u)}{s^*}$. And $E_j(x'_2, u_2)$ and $E_j(x'_3, u_3)$ are estimated by linear interpolation

$$E_j(x'_2,u_2) \approx (1-\beta_1)E_j(\lfloor x'_2 \rfloor, u_2) + \beta_1 E_j(\lceil x'_2 \rceil, u_2) \quad (18)$$

$$E_j(x'_3,u_3) \approx \beta_2 E_j(\lfloor x'_3 \rfloor, u_3) + (1-\beta_2) E_j(\lceil x'_3 \rceil, u_3) \quad (19)$$

where $\beta_1 = \Delta u/s^* - \lfloor \Delta u/s^* \rfloor$, and

$\beta_2 = (u_3 - u_2 - \Delta u)/s^* - \lfloor (u_3 - u_2 - \Delta u)/s^* \rfloor$.

3.4.3 Special occlusion processing

If the virtual pixel to be interpolated is occluded on one side (e.g. row u_2) but dis-occluded on the other (e.g. row u_3), it does not make sense to apply BMI on rows u_2 and u_3 because they should not match and the cost $C_{u_2,\Delta u}(x,s)$ would be large even for the optimal s^*. Thus, when $C_{u_2,\Delta u}(x,s)$ is larger than some threshold T_{occ}, we apply the following special occlusion processing.

When occlusion occurs, the virtual pixel should be visible in multiple views on one side (e.g. both rows u_1 and u_2, or both rows u_3 and u_4). Thus we compute two one-side cost $C^-_{u_2,\Delta u}(x,s)$ and $C^+_{u_2,\Delta u}(x,s)$ for each candidate slope s

$$\begin{cases} C^-_{u_2,\Delta u}(x,s) = \left\| \begin{matrix} [U_2(x,s) - m_{U_2}(x,s)I_0] \\ - \quad [U(x,s) - m_U(x,s)I_0] \end{matrix} \right\|^2_{l_2} \\ C^+_{u_2,\Delta u}(x,s) = \left\| \begin{matrix} [D_2(x,s) - m_{D_2}(x,s)I_0] \\ - \quad [D(x,s) - m_D(x,s)I_0] \end{matrix} \right\|^2_{l_2} \end{cases} \quad (20)$$

where U_2 and D_2 correspond to the 2D block in rows u_1, and u_4 respectively, and m_{U_2} and m_{D_2} are the mean of U_2 and D_2 respectively. Out of the two one-sided optimal slope,

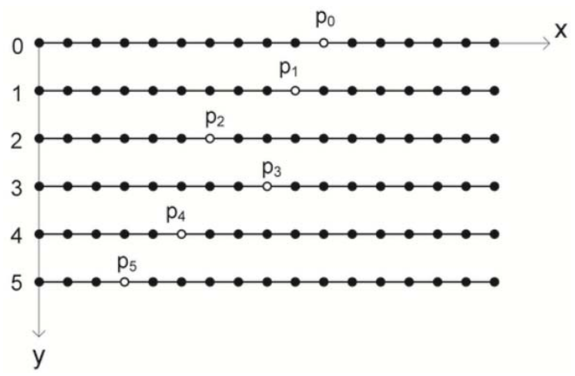
$$s^{-*} = \arg\min_{s \in S_j} C^-_{u_2,\Delta u}(x,s) \quad (21)$$

$$s^{+*} = \arg\min_{s \in S_j} C^+_{u_2,\Delta u}(x,s) \quad (22)$$

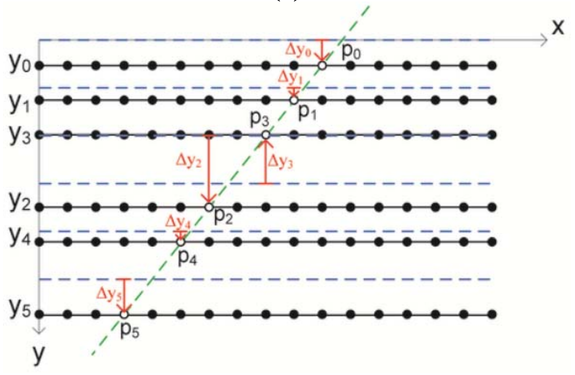
the optimal s^* in this occlusion situation is the one that gives smaller one-sided cost. The virtual pixel is synthesized as either $E_j(x'_2, u_2)$ or $E_j(x'_3, u_3)$, depending on whether s^{-*} or s^{+*} is chosen. Again, $E_j(x'_2, u_2)$ and $E_j(x'_3, u_3)$ as estimated by linear interpolation in Eqn. 18 and Eqn. 19.

4. Proposed Ray-space based Camera Spacing Correction

So far we have been assuming that the cameras are equally spaced with constant camera distance of C. However, in real situations, the cameras may not be equally spaced due to inaccurate camera set-up or environmental constraint. And if the cameras are not equally spaced or if the view order is unknown, pixels corresponding to the same scene point may not form a straight line in the EPI, as shown in the example in Figure 8(a). Without the straight lines, ourstrategy in the proposed RTI to interpolate a new view by preserving the straightline structures in EPIs would not work. In this section, we propose a correction method to ensure the planes are ordered and the unequal camera spacing is accounted for to ensure straight line structures are preserved in EPI. Similar to Section 3, we assume that all cameras have the same focal length f and are placed along a horizontal baseline on the reference plane $Z = 0$ and all the optical axes are perpendicular to the reference plane. Our method does not require the calibration parameters of the cameras, such as f, C_i, and camera intrinsic parameters.



(a)



(b)

Figure 8 Ray-space based spacing correction: (a)Equally-spaced EPI with non-equally-spaced cameras, (b)Space-corrected EPI

For at least one scene point P that appears in one of the EPI (say E_j), we assume that the locations of all corresponding pixels in the multi-view images and the corresponding points p_i in E_j are known. In real situations, the corresponding pixels can be marked manually or identified using some pixel matching algorithms. Let the location of p_i be (x_i, i).

When the cameras are equally spaced, adjacent rows in E_j are equally spaced with a distance of 1, which corresponds to the constant camera spacing C among the N cameras. When the cameras are not equally spaced, an EPI can still be formed and adjacent rows are still equally spaced at a distance of 1, as shown in Figure 8. (e.g. row 2 and row 3 have a vertical distance of 1 in E_j.) Note that in the example of Figure 8, view 2 and 3 have a wrong order and all of the views are not equally spaced. To account for the unequal camera spacing, we define an auxiliary vector $y=[y_0,y_1,...,y_{N-1}]^T$, such that each y_i reflects the "true" location of row i in E_j (and actually all EPIs, as row i of all EPIs corresponds to the location of Camera i). When the cameras are equally spaced, $y_i=i$ and we call this the nominal camera position. We define $u=[0,1,..., N-1]^T$ as vector of the nominal positions. When the cameras are not equally spaced, y_i can be any real number. We define $\Delta y_i = y_i - i$ which is the deviation of row i from its nominal location. Let $\Delta y=[\Delta y_0,\Delta y_1,...,\Delta y_{N-1}]^T$. If each row i is moved by Δy_i to form a Corrected EPI (CEPI) as shown in Figure 9, the feature points in the CEPI will line up and form a straight line. The camera spacing correction problem is now reduced to finding the optimal Δy with minimum magnitude such that the feature points are co-linear. Thus we can obtain the optimal y^* by solving the following convex optimization problem:

$$y^* = \arg\min_{y} \| y-u \|_{l_2}^2 \tag{23}$$

$$\textit{subject to} \quad y=\alpha x+\beta I_N = X\cdot a$$

where $x=[x_0,x_1,..,x_{N-1}]^T$, I_N is a vector of size $N\times 1$ with all elements being 1, α and β are the slope and y-intercept of the feature line, $X=[x\ I_N]$, and $a=[\alpha,\beta]^T$. Note that a is also a variable in the optimization.

This optimization guarantees the feature points of a scene point, say P_0, are co-linear. A property of y^* is that the feature points of any scene point P visible in E_j should also be co-linear in the CEPI for the following reason. For the feature points corresponding to any P, the slope of the line segment connecting p_i and p_{i+1} in CEPI can be computed using a generalization of Eqn. (4) and (5):

$$s_{i,i+1}(P)=\frac{(y_i-y_{i+1})\cdot Z(P)}{f\cdot C_{i,i+1}} \tag{24}$$

For the case of $P=P_0$, the feature points are co-linear and the slope $s_{i,i+1}(P_0)$ is equal for all i. Thus, there exists a κ such that $\kappa=(y_i-y_{i+1})/C_{i,i+1}$ for all i. For any other P, $s_{i,i+1}(P)=\kappa Z(P)/f$ which is the same for all i. In other words, the corresponding feature points are co-linear in the CEPIs.

We note that many y can cause the feature points to align into a straight line because, if y is a solution such that $y=\alpha x+\beta I_N$, then $y+\gamma I_N$ will also be a solution because $y+\gamma I_N = \alpha x+(\beta+\gamma)I_N$. In this paper, we want to find the y with minimum deviation, $\| y-u \|_{l_2}^2$, is unique since the cost function is convex.

In reality, the values of x_i may contain small errors due to the finite precision of locations (integer precision only) or human error during marking of feature point locations. There is no need to force such feature points to form a line. Instead, we only need the feature points to be close to some straight line. Assuming M available scene points leading to M feature lines in the EPIs, we relax the problem to

$$y^* = \arg\min_{y} \| y-u \|_{l_2}^2 \tag{25}$$

$$\textit{subject to} \quad \frac{1}{M}\sum_{k=0}^{M-1} \| y-X_k\cdot a_k \|_{l_2}^2 < \varepsilon$$

where α_k and β_k are the slope and y-intercept of the k^{th} feature line, and $a_k=[\alpha_k,\beta_k]^T$ and X_k are the corresponding quantities. Note that y is the same for all k. Using Lagrange multiplier, the optimal y^* can be obtained by minimizing the unconstrained cost function E,

$$E(y,a_0,a_1,\cdots,a_{M-1}) = \|y-u\|_{l_2}^2 + \frac{\lambda}{M}\sum_{k=1}^{M}\|y-X_k a_k\|_{l_2}^2 \tag{26}$$

For any y, we take the derivative of the cost function with respect to a_k

$$\frac{\partial E}{\partial a_k} = \frac{\partial}{\partial a_k}[\frac{\lambda}{M}(y-X_k a_k)^T(y-X_k a_k)]$$

$$= \frac{\lambda}{M}\frac{\partial}{\partial a_k}(y^T y - a_k^T X_k^T y - y^T X_k a_k + a_k^T X_k^T X_k a_k)$$

$$= -\frac{\lambda}{2M}(X_k^T y - X_k^T X_k a_k)$$

and set it to zero to obtain $a_k^* = (X_k^T X_k)^{-1} X_k^T y$. Substituting this into Eqn. (26), we get

$$E(y,a_0^*,a_1^*,\cdots,a_{M-1}^*) = \| y-u \|_{l_2}^2 + \frac{\lambda}{M}\sum_{k=1}^{M}\| y-A_k y \|_{l_2}^2 \tag{27}$$

where $I_{N\times N}$ is an identity matrix of size $N\times N$.

The proposed RTI can be extended easily for the camera spacing-corrected situation. We simply perform camera spacing correction before RTI. The only change to RTI is to replace the term u_3-u_2 in Eqn. (16) by y_3-y_2. The rest of RTI remains the same.

5 Experimental Results

5.1 Simulation Results for Proposed RTI

In our experiment, we compare the performance of the proposed RTI against some existing algorithms: BMI, PMI, and MELI, under the same test conditions. We conduct simulation on four test sequences "Xmas", "Rena", "Akko & Kayo" and "Ballroom", with the test conditions shown in Table 1. Both objective and subjective comparisons of the generated virtual viewpoint images among different methods are conducted. Two test cases are considered in the experiments of RTI, BMI, PMI and MELI. In Case 1, the range of depth (or equivalently, the range of interpolation direction) is given with s_1 and s_2 corresponding to the smallest and the largest search slope respectively. Therefore, the search angle range for PMI, BMI and MELI is from $\tan^{-1}(s_1)$ to $\tan^{-1}(s_2)$. And for RTI, the search angle range for Radon transform is from $90^\circ+\tan^{-1}(s_1)$ to

$90^\circ + \tan^{-1}(s_2)$. In Case 2, no prior search range is given. In this case we assume that for the same scene point, its corresponding pixels would appear in at least half of the real view rows such that we can choose $\tan^{-1}(s_1) = \tan^{-1}(0.5N/W)$, where W and N are the width and height of EPI, and $\tan^{-1}(s_2) = 90^\circ$. In both cases, the search step for the angle is 1°. The experiments are implemented in the platform of MATLAB R2012a using an Intel Core i3, 3.06-GHz PC with 8GB of memory.

Table 1 Test conditions for ray-space based interpolation

Test Sequences	**Camera Arrangement**	**EPI Configuration**	**Camera Intervals**
Xmas	101 cameras with 0.3cm spacing 1D/parallel	Extract 1 out of every 4 views uniformly	1.2cm
		Extract 1 out of every 10 views uniformly	3cm
		Extract 1 out of every 20 views uniformly	6cm
Rena	16 cameras with 3cm spacing 1D/parallel	Extract 1 out of every 2 views uniformly	6cm
Akko& Kayo	15 cameras, 2D array 3 rows × 5 columns H: 5cm spacing V: 20cm spacing	Each row treated as independent sequence For each row, extract 4 out of 5 views and interpolate the missing view	5cm/10cm (unequal spacing)
Ballroom	8 cameras with 20 cm spacing 1D/parallel	Extract 7 out of 8 views and interpolate the missing view	20cm/40cm (unequal spacing)

Table 2 Average candidate interpolation directions reduction of RTI compared with PMI, BMI and MELI

Test Sequences		**Candidate Interpolation Direction Reduction**	
		With Prior Depth Range	**Without Prior Depth Range**
Xmas	1.2cm	40%	96%
	3cm	70%	97%
	6cm	85%	97%
Ballroom		80%	92%
Akko & Kayo		75%	95%
Rena		87%	96%

Table 3 PSNR (in dB) and computing time (in sec/frame) of generated virtual views (computing time shown in bracket)

Test Sequences		**With Prior Depth Range**				**Without Prior Depth Range**			
		PMI	BMI	MELI	RTI	PMI	BMI	MELI	RTI
Xmas	1.2cm	35.6(1.9)	36.6(2.8)	38.2(4.4)	**42.7(3.9)**	24.5(5.2)	27.1(20.7)	30.9(51.2)	**42.2(4.9)**
	3cm	35.6(3.0)	36.2(4.9)	37.9(7.7)	**41.9(4.3)**	23.7(5.8)	26.5(23.8)	30.5(55.2)	**41.9(5.3)**
	6cm	29.6(3.9)	36.0(7.3)	37.4(11.9)	**41.7(4.5)**	23.9(6.5)	26.5(25.1)	30.4(58.9)	**41.5(5.7)**
Ballroom		20.3(4.2)	21.8(9.7)	22.2(14.1)	**29.0(6.0)**	18.8(7.4)	19.9(28.0)	21.1(61.9)	**26.0(7.7)**
Akko & Kayo		25.3(3.8)	30.9(7.1)	30.3(12.0)	**33.9(5.5)**	21.6(6.4)	24.8(24.9)	27.6(58.9)	**33.2(6.0)**
Rena		28.5(4.1)	32.7(8.4)	34.6(13.0)	**39.3(4.8)**	24.5(6.9)	24.8(26.2)	29.0(60.1)	**38.3(5.8)**
Average		29.1(3.0)	32.4(6.7)	33.4(10.5)	**38.1(4.8)**	22.8(6.4)	24.9(24.8)	28.3(57.7)	**37.2(5.9)**

Table 2 shows the average number of candidate interpolation directions in both Case 1 and Case 2. It can be seen that there are much fewer candidate directions in RTI than the other methods, especially in Case 2. Table 3 shows the average PSNR of the generated virtual viewpoint images compared with the original (ground truth) images, as well as the computing time of each algorithm. The proposed RTI can achieve significantly higher PSNR than PMI, BMI, and MELI. While other methods are hugely affected by the availability of prior knowledge of search range, achieving much lower PSNR when prior depth range is not know, the proposed RTI is not affected and maintains its very high PSNR in both cases. In Case 1, the proposed RTI is faster than BMI and MELI and slight slower than PMI. In Case 2, RTI is much faster than other algorithms because of the huge reduction of the candidate search directions.

Figure 9 and 10 show the synthesized virtual views by different methods in Case 1 and Case 2 respectively. The original ground truth images are also shown for comparison. The results of “Xmas" are those with camera interval of 6cm. Some of the defective regions are highlighted for ease of comparison. Subjectively, the results of RTI is significantly better than PMI, BMI and MELI, especially in Case 2, which agree with the objective results.

We do an experiment to study the two major tools in the proposed RTI: Radon transform based Candidate Direction Search (RCDS), and Optimal Interpolation Direction Detection (OIDD). In RTI, both RCDS and OIDD are on. In one case (which we call RCDS+BMI), we replace OIDD by BMI. In another case which we call OIDD, we turn off the RCDS and perform OIDD only. The average PSNR are shown in Figure 11. In Case 1 when depth range is available, both OIDD and RCDS+BMI can achieve 2.5dB PSNR gain over BMI. This suggests that both tools are capable achieving significant PSNR gain. When combined in RTI, the PSNR gain is increased to about 5dB which suggests that the advantage of OIDD and RCDS are complementary and additive. When no prior depth range is available in Case 2, both BMI and OIDD have significant PSNR drop compared with Case 1, because block matching can have errors easily in face of too many search directions. Both RTI and RCDS+BMI have only a small PSNR drop which confirms the advantage of RCDS to suppress unreliable search candidates before block matching.

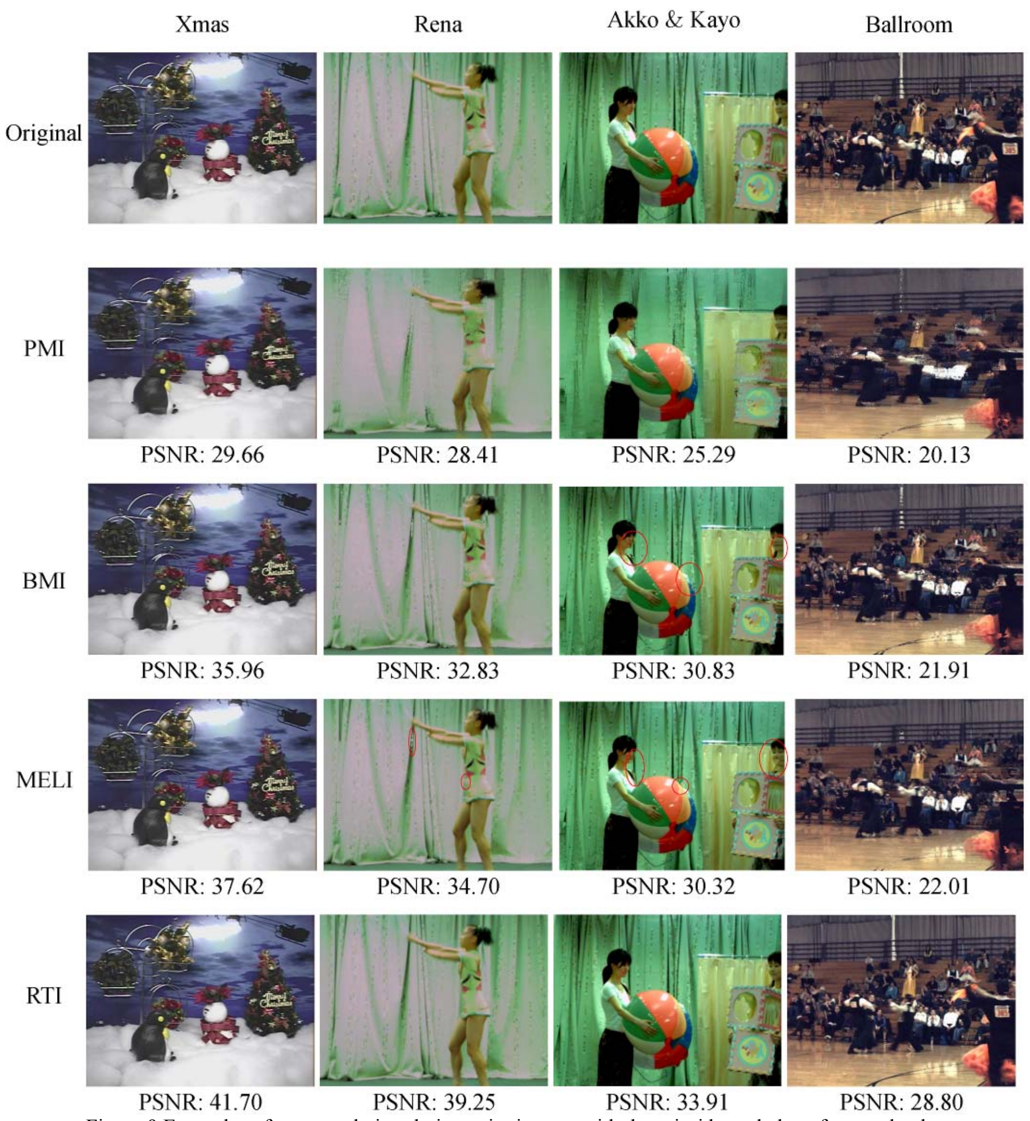


Figure 9 Examples of generated virtual viewpoint images with the priori knowledge of scene depth

5.2 Simulation Result for Camera Spacing Correction

We simulate our proposed spacing correction algorithm on “Xmas" which contains 101 views, each of 640×480. In our simulation, we choose $N=11$ unequally spaced views with the original (or ground truth) camera locations l_o = [1, 16, 18, 35, 37, 62, 51, 75, 80, 95, 99], and use them to form 480 EPIs each of size 640×11. In this experiment, we assume that we do not know the unequal spacing and simply assume the cameras are equally spaced. In each EPI, row i corresponds to a nominal camera location $(N-1)i$, for $i=0,1,...,10$. We simulate incorrect camera order by placing View 62 in front of View 51.

To measure the accuracy of the spacing correction algorithm, we define the Camera Spacing Error (CSE) for any corrected row position vector y as

$$CSE(y)=\|y-y_o\|_{l_1} \qquad (28)$$

where $y_o=l_o/(N-1)$ is the normalized original (ground truth) camera location. A small CSE suggests a good camera spacing correction algorithm.

In Figure 12, we show the simulation results of the proposed algorithm in Eqn. (26) for $M=1,2,...,10$ and $\lambda=1,2,...,50$. For each M, the curve is the average behavior of many experiments. For example, for $M=1$, we apply the algorithm to two EPIs corresponding to Row 287 and Row 319 of the views. A total of ten feature lines are marked manually. When $M=1$, there are ten ways to choose 1 feature line out of the ten. We compute the CSE for each case and the average is reported in Figure 12. For other $M>1$, all possible combination of M feature lines from the set of 10 are used to compute the average CSE. For each M, we compute CSE with respect to different λ which is from 1 to 50.

In Figure12, CSE appears to decrease as M increases, as expected because more data tend to give more reliable estimates. For any given M, small λ tends to give large CSE, because small λ does not emphasize enough the collinearity of the feature points along a feature line. The combination of $M=10$ and $\lambda=25$ appears to work well.

But even the combination of $M = 2$ (low complexity) and $\lambda = 25$ works quite well.

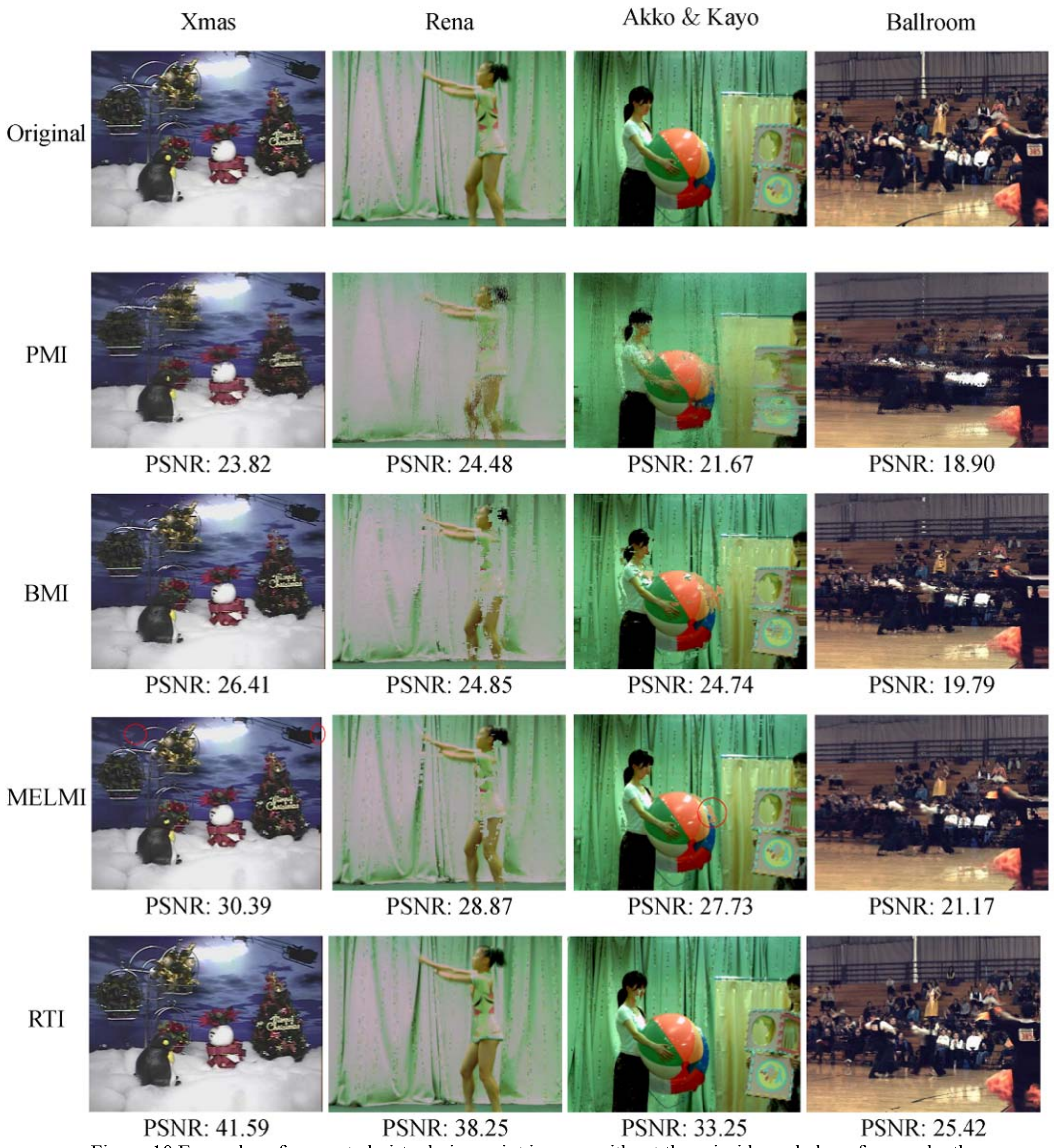


Figure 10 Examples of generated virtual viewpoint images without the priori knowledge of scene depth

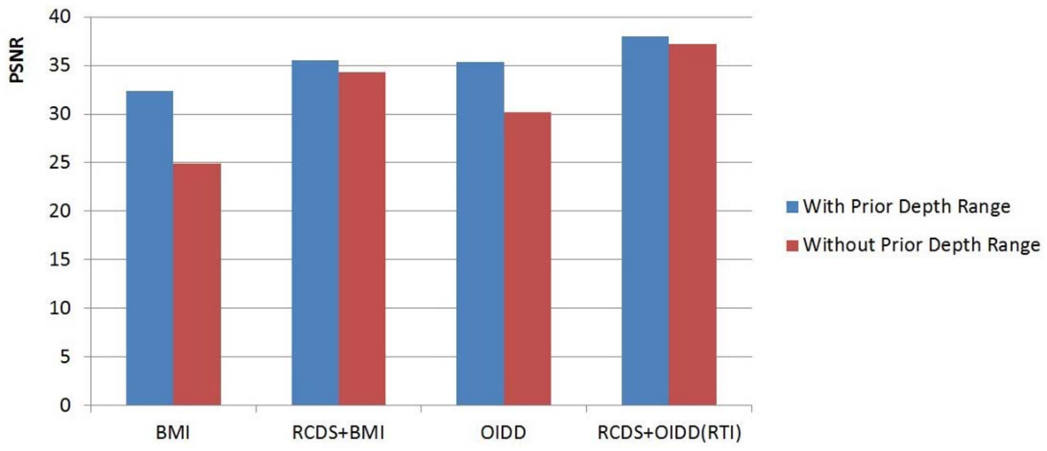


Figure 11 Comparison of two major tools in RTI: OIDD and RCDS

Figure 13(a) shows E_{287}, the EPI corresponding to Row 287 with equally spaced rows, before camera spacing correction. We apply our camera spacing correction with $M = 2$ and $\lambda = 25$. The two sets of feature points corresponding to $M = 2$ feature lines are marked manually as bright green points, which do not form straight lines before correction. Using the two marked feature lines, the proposed camera spacing correction algorithm is applied to the EPI resulting in Figure 13(b) with the optimal corrected row positions being:

$y^* = [0.1, 1.6, 1.8, 3.5, 3.7, 6.1, 5.1, 7.4, 8.0, 9.5, 9.8]$ with a low CSE of 0.027. The y^* corresponds to camera locations [1, 16, 18, 35, 37, 61, 51, 74, 80, 95, 98], which are very similar to the original locations l_o. As expected, the feature points form straight lines in the corrected EPI.

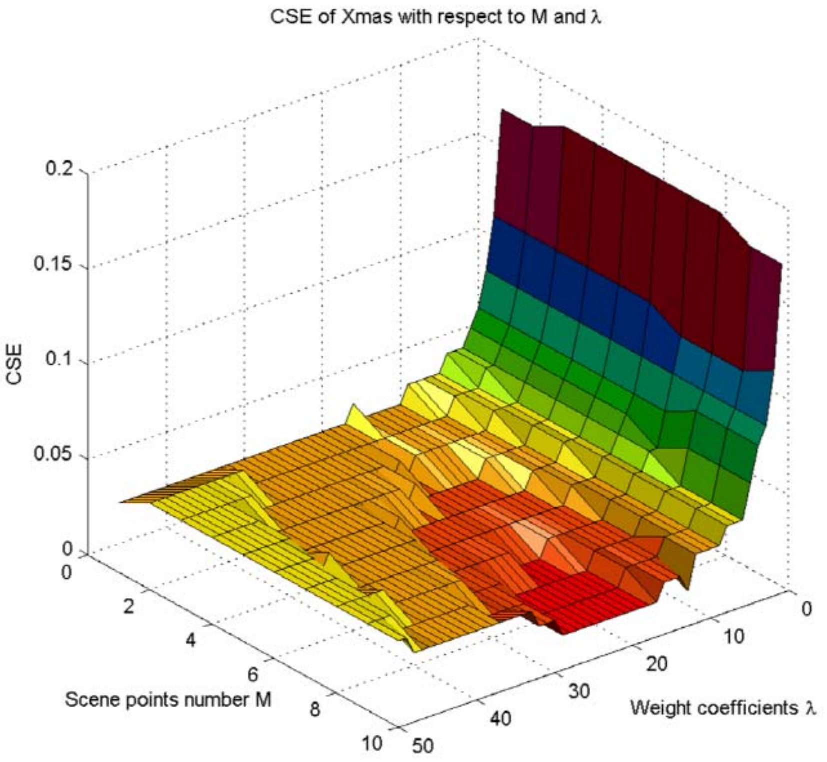


Figure 12 CSE of Xmas with respect to M and λ

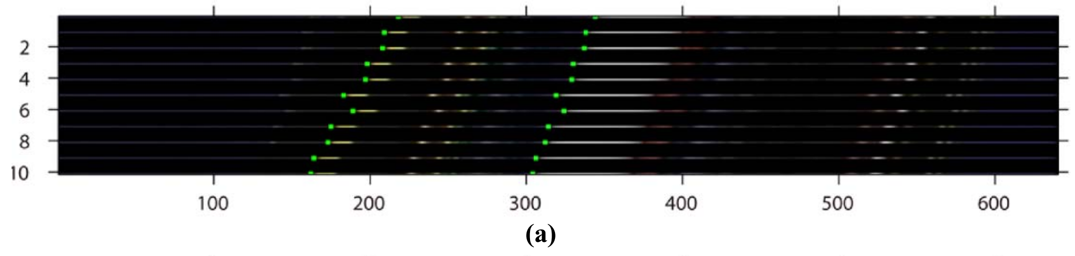
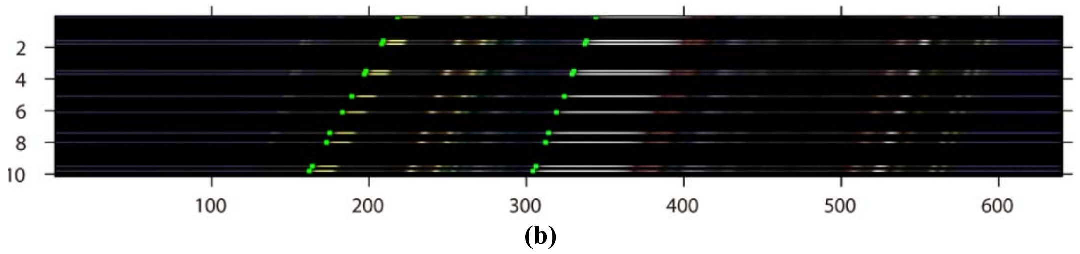



Figure 13 Results of ray-space based spacing correction. (a) EPI with incorrect camera order and distance, (b) EPI with camera spacing correction.

6. Conclusion

In this paper, a novel *Radon Transform Based Ray-Space Interpolation* (RTI) algorithm is proposed by considering the properties of both the EPI and the Radon transform. RTI not only reduces the size of the candidate interpolation direction set but also improves the reliability of the interpolation direction, especially in the case that no prior knowledge ofscene depth is available. Moreover, the smoothness property of the disparity field and correlations between neighboring EPIs are further explored to improve the performance of RTI. The occlusion problem is also alleviated by the proposed one-sided matching and interpolation. Meanwhile, in order to solve the problem that cameras are not equally spaced, a novel ray-space based camera spacing correction algorithm is proposed to adjust the EPIs such that pixels corresponding to the same scene point form a straight line in the EPI. From the simulation result we can see that virtual viewpoint images generated by RTI have much higher PSNR and visual quality than those generated by PMI, BMI and MELI. Moreover, RTI is more robust and practically not affected by the availability of prior scene depth information.

Acknowledgment

This work has been supported in part by the Research Grants Council (GRF Project no. 610210), and the Applied Science and Technology Research Institute (Project no. ASTR52-20A00210/11PN) of Hong Kong, China.

References

1. M. Tanimoto (2006) Overview of free viewpoint television, *Signal Processing Image Communication* **21**(6): 454-461.
2. D. Min, D. Kim, S.Y., K. Sohn (2009) 2D/3D freeview video generation for 3DTV system, *Signal Processing: Image Communication* **24**(1-2): 31-48.
3. W. Sun, L. Xu, O. C. Au, S. H. Chui, C. W. Kwok (2010) An overview of free view-point depth-image-based rendering (DIBR), *APSIPA Annual Summit and Conference.*
4. M. Levoy, P. Hanrahan (1996) Light field rendering, *Proceedings of the SIGGRAPH*, 31-42.
5. T. Fujii,, T. Kimoto, M. Tanimoto (1996) Ray space coding for 3D visual communication, *Proceedings of the Picture Coding Symposium*, 447-451.
6. T. Fujii, M. Tanimoto (2002) Free-viewpoint TV based on the ray-space representation, *Proceedings of the SPIE ITCom*, 175-189.
7. W. Chen, J. Bouguet, M. Chu, R. Grzeszczuk (2002) Light fieldmapping: efficient repre-sentation and hardware rendering of surface light fields, *ACM Transactions on Graphics* **21**(3): 447-456.
8. P. Heckbert (1989) Fundamentals of texture mapping and image warping, *Master's thesis, University of California, Berkeley.*
9. T. Kobayashi, T. Fujii, et al. (2000) Interpolation of ray-space data by adaptive filtering, *Proceedings of the SPIE*, 252-259.
10. A. Smolic , H. Kimata (2003) Report on 3DAV exploration, *ISO/IEC JTC1/SC29/WG11 N5878.*
11. M. P. Tehrani, T. Fujii, M. Tanimoto (2004) Offset block matching of multi-view images for ray-space interpolation, *The journal of the Institute of Image information and Television Engineers* **58**(4): 86-94.
12. G. Jiang, M.Yu, X. Ye (2005) New method of ray-space interpolation for free viewpoint video, *Proceedings of International Conference on Image Processing*, **2**: 1138-1141.
13. R. Yang (2010) A ray space interpolation algorithm based on improved dynamic programming, *International Conference on Signal Processing Systems* **1**: 26-30.
14. Q. Zhang, Y. Wu, P. An, Z. Zhang (2010) A novel ray space interpolation method based on hierarchical prediction, *International Conference on Audio Language and Image Processing (ICALIP)*, 1470-1774.
15. L. Hou, O. C. Au, X. Fan, L. Guo, M. Ma (2008) A novel Radon based ray-space interpolation algorithm, *IEEE Workshop on Multimedia Signal Processing*, 850-854.
16. L. Hou, O. C. Au, M. Ma, L. Guo, X. Fan (2008) Free view generation in ray-space via the Radon transform., *International Conference on Neural Networks and Signal Processing*, 671-675.
17. R. Bolles, H. Baker and D. Marimont (1987) Epipolar-plane image analysis: An approach to determining structure from motion, *International Journal of Computer Vision* **1**(1): 7-55
18. A. Fusiello, E. Trucco, A. Verri (2000) A compact algorithm for rectification of stereo pairs, *Machine Vision and Applications*, **12**(1): 16-22.
19. Y.S. Kang, C. Lee, Y.S. Ho (2008) An efficient rectification algorithm for multi-view images in parallel camera array, *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, 61-64.
20. C. Loop, Z. Zhang (1999) Computing rectifying homographies for stereo vision, *Proceedings of International Conference on Computer Vision and Pattern Recognition (CVPR)*, **1.**
21. A. Fusiello, L. Irsara (2008) Quasi-euclidean uncalibrated epipolar rectification, *Proceedings of International Conference on Pattern Recognition (ICPR)*), 1-4.
22. R. Hartley, A. Zisserman (2003) Multiple view geometry in computer vision, *Cambridge, UK: Cambridge University Press.*
23. W. Zhang, F. Bergholm (1997) Multi-scale blur estimation and edge type classification for scene analysis, *International Journal of Computer Vision*, **24**(3): 219-250.
24. M. Fiddy (1985) The Radon transform and some of its appli-cations, *Journal of Modern Optics*, **32**(1): 3-4.
25. P. Toft (1996) Radon transform: Theory and implementation, *Ph.D. Thesis: Danmarks Tekniske University.*
26. K. Khouzani, H. S., Zadeh (2006) Radon transform orientation estimation for rotation invariant texture analysis, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **27**(6): 1004-1008.
27. J. H. Kim, P. Lai, J. Lopez, A. Ortega, Y. Su, P. Yin, C. Gomila (2007) New coding tools for illumination and focus mismatch compensation in multi-view video coding, *IEEE Transaction on Circuits and Systems for Video Technology*, **17**(11):1519-153.