

# Illumination Normalization with Time-Dependent Intrinsic Images for Video Surveillance

Yasuyuki Matsushita, *Member, IEEE*, Ko Nishino, *Member, IEEE*,  
Katsushi Ikeuchi, *Fellow, IEEE*, and Masao Sakauchi, *Member, IEEE*

**Abstract**—Variation in illumination conditions caused by weather, time of day, etc., makes the task difficult when building video surveillance systems of real world scenes. Especially, cast shadows produce troublesome effects, typically for object tracking from a fixed viewpoint, since it yields appearance variations of objects depending on whether they are inside or outside the shadow. In this paper, we handle such appearance variations by removing shadows in the image sequence. This can be considered as a preprocessing stage which leads to robust video surveillance. To achieve this, we propose a framework based on the idea of intrinsic images. Unlike previous methods of deriving intrinsic images, we derive time-varying reflectance images and corresponding illumination images from a sequence of images instead of assuming a single reflectance image. Using obtained illumination images, we normalize the input image sequence in terms of incident lighting distribution to eliminate shadowing effects. We also propose an illumination normalization scheme which can potentially run in real time, utilizing the *illumination eigenspace*, which captures the illumination variation due to weather, time of day, etc., and a shadow interpolation method based on shadow hulls. This paper describes the theory of the framework with simulation results and shows its effectiveness with object tracking results on real scene data sets.

**Index Terms**—Intrinsic images, reflectance, shadow removal, illumination normalization, video surveillance, robust tracking.

## 1 INTRODUCTION

VIDEO surveillance systems involving object detection and tracking require robustness against illumination changes caused by variation of, for instance, weather conditions. Difficult situations arise not only by the change of illumination conditions, but also by the large cast shadows of surrounding structures, i.e., large buildings and trees. Since most visual tracking algorithms rely on the appearance of the target object, typically using color, texture, and feature points as cues, these shadows degrade the quality of tracking. In urban scenes, where building robust traffic monitoring systems is of special interest, it is usual to have large shadows cast by tall buildings surrounding the road, e.g., Fig. 2a. Building a robust video surveillance system under such an environment is a challenging task. To make the system insensitive to dramatic change of illumination conditions and robust against large static cast shadows, it would be valuable to cancel out those illumination effects from the image

sequence. Our goal is to “normalize” the input image sequence in terms of the distribution of incident lighting to remove illumination effects including shadow effects. We should note that our method does not consider shadows cast by moving objects but those cast by static objects such as buildings and trees. To achieve this goal, we propose an approach based on *intrinsic images*. Our method is composed of two parts as shown in Fig. 1.

The first part is the estimation of intrinsic images, which is an offline process, depicted in *A* of Fig. 1. In this part, first, the scene background image sequence is estimated to remove moving objects from the input image sequence. Using this background image sequence, we then derive intrinsic images using our estimation method which is extended from Weiss’s ML estimation method [21]. Using the estimated illumination image, which is a part of intrinsic images, we are able to robustly cancel out the illumination effects from input images of the same scene, enabling many vision algorithms such as tracking to run robustly. After the derivation, we construct a database using Principal Component Analysis (PCA), which we refer to as *illumination eigenspace*. This linear subspace captures the variation of lighting conditions in the illumination images. The subspace is used for the following direct estimation method.

The second part is a direct estimation of illumination images, shown in *B* of Fig. 1. Using the preconstructed illumination eigenspace, we estimate an illumination image directly from an input image. To obtain accurate illumination images, shadow interpolation using shadow hulls is accomplished.

• Y. Matsushita is with Microsoft Research Asia, 3/F, Beijing Sigma Center, No. 49, Zhichun Road, Hai Dian District, Beijing, China 100080.  
E-mail: yasumat@microsoft.com.

• K. Nishino is with the Department of Computer Science, Columbia University, 1214 Amsterdam Avenue, MC 0401 New York, NY 10027.  
E-mail: kon@cs.columbia.edu.

• K. Ikeuchi and M. Sakauchi are with the Institute of Industrial Science, the University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo 153-8505, Japan.  
E-mail: ki@cvt.iis.u-tokyo.ac.jp, sakauchi@sak.iis.u-tokyo.ac.jp.

Manuscript received 1 July 2003; revised 22 Mar. 2004; accepted 1 Apr. 2004.  
Recommended for acceptance by R. Basri.

For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0151-0703.

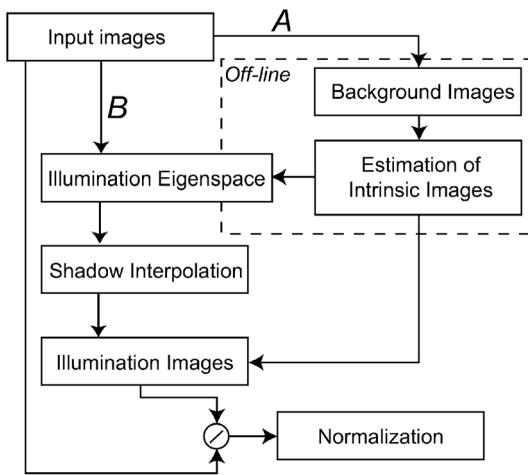


Fig. 1. System diagram for illumination-normalization.

In the remainder of this paper, we first overview related work in Section 1.1 and, in Section 2, we propose a method to derive time-varying reflectance images  $R(x, y, t)$  and corresponding illumination images  $L(x, y, t)$ . Derivation of the illumination-invariant images using the illumination images  $L(x, y, t)$  is described in Section 3. In Section 4, we propose a method to estimate  $R$  and  $L$  directly from an input image using PCA. In addition, to obtain more accurate illumination images, we use a shadow hull-based interpolation method, which is described in Section 4.1. Experimental results are described in Section 5. Finally, we conclude the paper in Section 6.

### 1.1 Related Work

Barrow and Tenenbaum proposed to consider every retinal image as a composition of a set of latent images, which they refer to as intrinsic images [7]. One type of the intrinsic images,  $R$ , contains the reflectance values of the scene, while the other type,  $L$ , contains the illumination intensities, and their relationship can be described by  $I = R \cdot L$ . Since illumination images  $L$  represent the distribution of incident lighting onto the scene while reflectance images  $R$  depict

the surface reflectance properties of the scene, this representation becomes useful to analyze and manipulate the reflectance/lighting properties of the captured scene.

While decomposing a single image into intrinsic images, namely, a reflectance image and an illumination image, remains a difficult problem [5], [7], [4], deriving intrinsic images from image sequences has seen great success. Recently, Weiss developed an ML estimation framework [21] to estimate a single reflectance image and multiple illumination images from a series of images captured from a fixed view point but under significant lighting condition variation. Finlayson et al. [6] proposed a similar approach to ours independently. They derive the scene texture edges from the lighting-invariant image, and by subtracting those edges from the raw input images, they successfully derive shadow-free images of the scene. We also take advantage of the fact that the reflectance image essentially models the scene texture in a manner invariant to lighting conditions. We accomplish edge substitution between the reflectance image and illumination images, enabling robust derivation of scene-texture-free illumination images. In a single image framework, Tappen et al. [17] proposed an approach to use learning-based shading classifier accompanied with chromaticity-based classification method. Categorization was done by assuming that differences in chromaticity values indicate reflectance edges and then classifying edges having similar chromaticity values by using a shading pattern classifier [13] which is prelearned using synthetic images. Although this method successfully separates shading and reflectance components, the shading classifier must account for the illumination direction, which is generally difficult to determine at each local point.

Several other works on shadow detection have been proposed. Deterministic model-based approaches to detect shadow regions are proposed by Kilger [14] and Koller et al. [3] that exploit gray level, local, and static features. In statistical approaches, Stauder et al. [9] and Jiang et al. [2] proposed a nonparametric approaches independently that use color, global, and dynamic features for enhancing object detection.

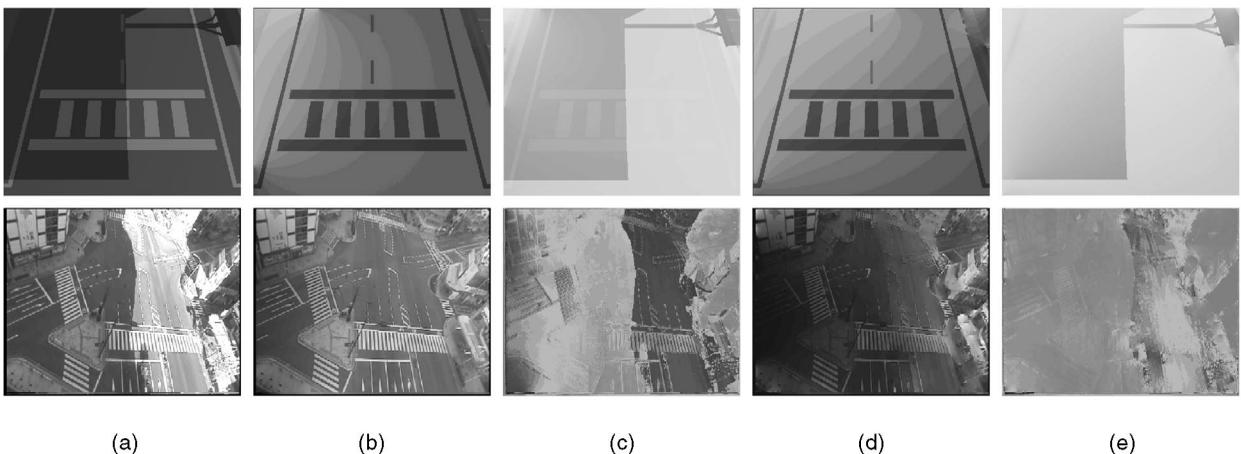


Fig. 2. (a) An input image  $i(x, y, t)$ , (b) reflectance image derived by ML estimation  $r_w(x, y)$ , (c) illumination image derived by ML estimation  $l_w(x, y, t)$ , (d) our time-varying reflectance image  $r(x, y, t)$ , and (e) our illumination image  $l(x, y, t)$ .

## 2 INTRINSIC IMAGE ESTIMATION

Our work is inspired by ML estimation method [21] of estimating intrinsic images from a sequence of images captured from a fixed viewpoint under varying illumination. In this method, natural image statistics [8] are used as priors to estimate a single reflectance image and multiple illumination images. Although the method assumes the scene to be Lambertian, it is still effective to extract the scene texture, not a reflectance image, from a scene which contains specular surfaces. In that case, however, the estimated reflectance image does not accurately depict surface reflectance property of the scene and some portion of the reflectance property will be included in illumination images. If we consider real world scenes, it is often difficult to expect the Lambertian assumption to hold. A typical example is white lines on a road surface, which show variable reflection with respect to illumination changes. We prepared a synthetic scene shown in Fig. 2a. The scene is composed of a road surface and pedestrian stripes that have different reflectance property from the road surface. With an assumption that the view point is fixed and the scene is roughly Lambertian, the ML estimation-based method derives the scene reflectance image by taking the median of filtered input image in log domain. Denoting log reflectance of the road surface and stripes with  $r_1$  and  $r_2$ , respectively, the reflectance difference at the boundary of the road surface and the stripes becomes  $|r_1 - r_2|$ . Since the ML estimation method assumes that the scene is Lambertian, i.e.,  $r_1$  and  $r_2$  are constants, the difference  $|r_1 - r_2|$  is also constant. However, if the Lambertian assumption does not hold, e.g.,  $r_2$  is varying,  $|r_1 - r_2|$  is no longer constant. In that case, the error  $\{|r_1 - r_2(t)| - \text{median}_t|r_1 - r_2(t)|\}$  appears in illumination images as a ghost of scene texture as shown in Fig. 2c.

To handle this problem, we propose to assume a set of time-varying reflectance images  $R(x, y, t)$  instead of a time-invariant reflectance image  $R(x, y)$ . To derive the time-varying reflectance images, we start with estimating a scene texture image using the ML estimation method [21]. We denote the scene texture image which is the reflectance image estimated by the ML estimation method and the illumination image with subscript  $w$ , i.e.,  $R_w$  and  $L_w$ , and our reflectance image and illumination image,  $R$  and  $L$ , respectively. Applying the ML estimation method, a single reflectance image  $R_w(x, y)$  and a set of illumination images  $L_w(x, y, t)$  are estimated. Our goal is to derive time-varying, i.e., lighting condition dependent, reflectance images  $R(x, y, t)$ , and corresponding illumination images  $L(x, y, t)$  that do not contain scene texture.

$$I(x, y, t) = R(x, y, t) \cdot L(x, y, t) \quad (1)$$

We use lower-case letters to denote variables in log domain, e.g.,  $r$  represents the logarithm of  $R$ . With  $n$ th derivative filters  $f_n$ , a filtered reflectance image  $r_{wn}$  is computed by taking median along the time axis of  $f_n \star i(x, y, t)$ , where  $\star$  represents convolution. We used two derivative filters, i.e.,  $f_0 = [0 \ 1 \ -1]$  and  $f_1 = [0 \ 1 \ -1]^T$ . With those filters, input images are decomposed into intrinsic images by Weiss's

method as described in (2). The method is based on the statistics of natural images [8].

$$\hat{r}_{wn}(x, y) = \text{median}_t\{f_n \star i(x, y, t)\}. \quad (2)$$

The filtered illumination images  $l_{wn}(x, y, t)$  are then computed by using estimated filtered reflectance image  $r_{wn}$ .

$$\hat{l}_{wn}(x, y, t) = f_n \star i(x, y, t) - \hat{r}_{wn}(x, y). \quad (3)$$

To be precise,  $l$  is computed by  $l = i - r$  in the unfiltered domain in Weiss's original work while we estimate  $l$  in the derivative domain for the following edge-based manipulation.

We use the output of the ML estimation method as initial values of our intrinsic image estimation. As mentioned above, the goal of our method is to derive time-dependent reflectance images  $R(x, y, t)$  and their corresponding illumination images  $L(x, y, t)$ . The basic idea of the method is to estimate time-varying reflectance components by cancelling the scene texture from initial illumination images. To factor out the scene textures from the illumination images and associate them with reflectance images, we use the texture edges of  $r_w$ . We take a straightforward approach to remove texture edges from  $l_w$  and derive illumination images  $l(x, y, t)$  with (4) and (5). These equations describe that if the magnitude of a pixel in the gradient of the texture image, i.e.,  $|r_{wn}(x, y)|$ , is larger than a threshold  $T$ , then this is an evidence of a texture edge. In that case, the texture edges that appear in the gradient of the illumination image are removed from the illumination image, and the removed edge is added to the time-varying reflectance image. Here, we assume that there is no scene texture edge which has not appeared in  $r_{wn}$ . Since the scene texture edges are the edges which stay at the same position, it is reasonable to assume that they are captured in  $r_{wn}$ .

$$l_n(x, y, t) = \begin{cases} 0 & \text{if } |r_{wn}(x, y)| > T \\ l_{wn}(x, y, t) & \text{otherwise,} \end{cases} \quad (4)$$

$$r_n(x, y, t) = \begin{cases} r_{wn}(x, y) + l_{wn}(x, y, t) & \text{if } |r_{wn}| > T \\ r_{wn}(x, y) & \text{otherwise,} \end{cases} \quad (5)$$

where  $T$  represents a threshold value. While we currently manually set the threshold value  $T$  used to detect texture edges in  $r_{wn}$ , we found the procedure is not so sensitive to the threshold as long as it covers texture edges well.

Since the operation is linear, the following equation is immediately confirmed.

$$f_n \star i(x, y, t) = r_{wn}(x, y) + l_{wn}(x, y, t) \\ = r_n(x, y, t) + l_n(x, y, t). \quad (6)$$

Finally, time-varying reflectance images  $r(x, y, t)$  and scene texture-free illumination images  $l(x, y, t)$  are recovered from filtered reflectance  $r_n$  and illumination images  $l_n$  through the following deconvolution process as done in [21].

$$(\hat{r}, \hat{l}) = g \star \left( \sum_n f_n^r \star (\hat{r}_n, \hat{l}_n) \right), \quad (7)$$

where  $f_n^r$  is the reversed filter of  $f_n$ , and  $g$  is the filter which satisfies the following equation:

$$g \star \left( \sum_n f_n^r \star f_n \right) = \delta. \quad (8)$$

Reconstruction of images from edges has been studied for a long time. The problem can be formulated as a boundary value problem, and an alternative technique that has been widely accepted is the multigrid method. Multigrid methods solve the boundary value problem efficiently by changing the problem into a set of linear algebraic equations. Multigrid methods can be used to reconstruct images from edges as an alternative to the method described above. Readers are referred to a good overview on multigrid solvers by Press et al. [20].

To demonstrate the effectiveness of our method for deriving time-dependent intrinsic images, we rendered a CG scene which contains cast shadows and surface patches with different reflectance properties, which is analogous to real road surfaces, e.g., pedestrian stripes. Fig. 2 shows a side-by-side comparison of the results of applying the ML estimation method and our method. The first row is the CG scene, where the scene has the property that the histogram of derivative-filtered output is sparse, which is the required property of the ML estimation-based decomposition method and also is the statistics usually found in natural images. As can be seen clearly, texture edges are successfully removed from our illumination image while they obviously remain in illumination image derived by the ML estimation method. Considering an illumination image to be an image which represents the distribution of incident lighting, our illumination image is much better since incident lighting has nothing to do with the scene reflectance properties.

### 3 SHADOW REMOVAL

Using the obtained scene illumination images by our method, the input image sequence can be normalized in terms of illumination.

To estimate the intrinsic images of the scene where video surveillance systems are to be applied, it is necessary to remove moving objects from the input image sequence because our method requires the scene to be static. Therefore, we first create background images in each short time range ( $\Delta T$ ) in the input image sequence, assuming that the illumination does not vary in that short time period. We simply use the median image of the short input sequence as the background image, but of course the more complicated methods would give the better background images [11]. The assumption here is that moving objects in the scene are not observed at the same point longer than the background in  $\Delta T$ . These background images  $B(x, y, t)$  are used for the estimation of intrinsic images. Using the estimation method described in the former section, each image in the background image sequence is decomposed into corresponding reflectance images  $R(x, y, t)$  and illumination images  $L(x, y, t)$ .

$$B(x, y, t) = R(x, y, t) \cdot L(x, y, t). \quad (9)$$

Once decomposed into intrinsic images, any image whose illumination condition is captured in the series of  $B(x, y, t)$  can be normalized with regards to its illumination condition by simply dividing the input image  $I(x, y, t)$  by its



(a)



(b)

Fig. 3. An input image  $I$  (a) and the illuminance-invariant image  $N$  (b).

corresponding estimated illumination image  $L(x, y, t)$ . Through the normalization, cast shadows are also removed from the input images.

Since the incident lighting effects are fully captured in illumination images  $L(x, y, t)$ , the normalization by dividing with  $L$  corresponds to removing the incident lighting distribution from the input image sequence. Let us denote the resulting illuminance-invariant image with  $N(x, y, t)$ , that can be derived by the following equation:

$$N(x, y, t) = I(x, y, t) / L(x, y, t). \quad (10)$$

Fig. 3 shows the result of our normalization method. Fig. 3a shows the input image  $I$  and Fig. 3b represents the illuminance-invariant image  $N$ . Note shadows of the buildings are removed in  $N$ .

Since we consider the time-dependent reflectance values, accurate normalization of incident lighting can be done compared to using illumination images that are derived by ML estimation method. Fig. 4 depicts the difference between using our illumination image  $L$  (Figs. 4a and 4c) and using illumination image  $L_w$  by ML estimation (Figs. 4b and 4d). As can be seen in the images, the pedestrian stripes appear over the trucks in every right-hand side image. On the other hand, when using our illumination images, the ghost of pedestrian stripes is almost vanished as seen in every left-hand side image. This is because our method handles reflectance

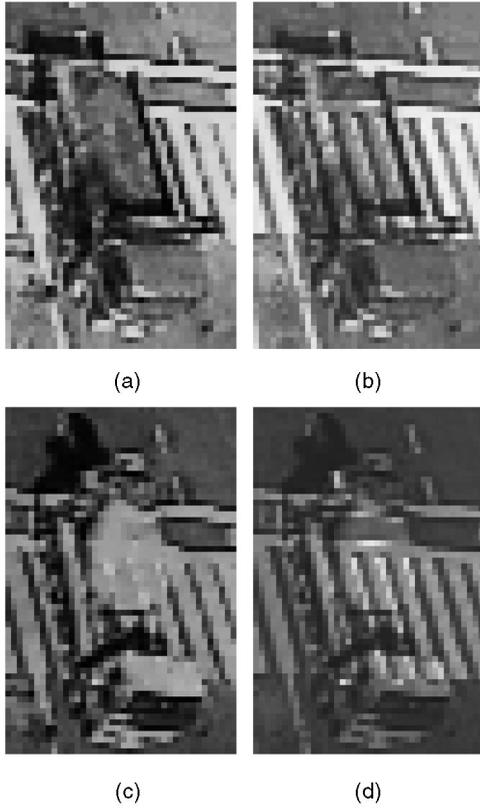


Fig. 4. The difference of the normalization results between using our illumination images  $L$  and illumination image  $L_w$  by ML estimation. The left images of each pair, (a) and (b), show the results using our illumination image  $L$ , and the right ones, (c) and (d), are the results using illumination image  $L_w$  derived by ML estimation.

variation properly while the reflectance values are fixed in the ML estimation method.

#### 4 ILLUMINATION EIGENSPACE FOR DIRECT ESTIMATION OF ILLUMINATION IMAGES

The intrinsic image estimation method described in the former section is fully offline since the method basically requires estimation of the background images and taking the median over the accumulated images to obtain the final intrinsic images. Computation of the intrinsic images using the method described in Section 2 is fast for typical frame-sizes of current videos. However, it is necessary to first estimate background images to adopt the method. Therefore, for our use the computation of the intrinsic images is done offline. However, realtime processing is required for practical use. In this section, we describe our approach to realtime derivation of illumination images for shadow removal. Our method first stores a lot of illumination images captured under different illumination conditions. Using stored illumination images, realtime estimation of illumination image from an input image is accomplished.

We propose *illumination eigenspace* to model variations of illumination images of the scene. The illumination eigenspace is an eigenspace into which only illumination effects are transformed. We use principal component analysis (PCA) to construct the illumination eigenspace of a target scene, in our case, the crossroad shown in Fig. 5. PCA is

widely used in signal processing, statistics, and neural computing. The reason why we employed PCA is that we assume the observed illumination images have Gaussian distribution. The basic idea in PCA is to find the basic components  $[s_1, s_2, \dots, s_n]$  that explain the maximum amount of variance possible by  $n$  linearly transformed components. Fig. 6 shows the hyperplane constructed by mapping illumination images onto the eigenspace visualized with the first three eigenvectors.

In our case, we mapped  $L_w(x, y, t)$  into the illumination eigenspace, instead of mapping  $L(x, y, t)$ . This is because, when given an input image, the reflectance image  $R_w(x, y)$  is useful to eliminate the scene texture by computing  $I(x, y, t)/R_w(x, y)$ , and the resulting image becomes  $L_w(x, y, t)$ . We keep the mapping between  $L_w(x, y, t)$  and  $L(x, y, t)$  to derive final  $L(x, y, t)$  estimates. First, an illumination space matrix is constructed by subtracting  $\bar{L}_w$ , which is the average of all  $L_w$ , i.e.,  $\bar{L}_w = \frac{1}{n} \sum_n L_w$ , from each  $L_w$  and stacked column-wise.

$$\mathbf{P} = \{L_{w_1} - \bar{L}_w, L_{w_2} - \bar{L}_w, \dots, L_{w_n} - \bar{L}_w\}. \quad (11)$$

$\mathbf{P}$  is an  $N \times M$  matrix, where  $N$  is the number of pixels in the illumination image and  $M$  is the number of illumination images  $L_w$ . We made the covariance matrix  $\mathbf{Q}$  of  $\mathbf{P}$  as follows:

$$\mathbf{Q} = \mathbf{P}\mathbf{P}^T. \quad (12)$$

Finally, the eigenvectors  $e_i$  and the corresponding eigenvalues  $\lambda_i$  of  $\mathbf{Q}$  are determined by solving,

$$\lambda_i e_i = \mathbf{Q}e_i. \quad (13)$$

To solve (13), we utilized Turk and Pentland's method [16], which is useful to compute the eigenvectors when the dimension of  $\mathbf{Q}$  is high. Fig. 6 shows the illumination eigenspace on which all the illumination image,  $L_w(x, y, t)$ , of 2,048 images from 120 days (7:00 - 15:00) are mapped.

Using the illumination eigenspace, direct estimation of an illumination image can be done given an input image which contains moving objects. We consider that the global similarity of the illumination image is measured by the distance weighted by the contribution ratio of eigenvectors (eigenvalues) in the illumination eigenspace. Thus, we first divide the input image by a reflectance image to get a pseudoillumination image  $L^*$  which includes dynamic objects:  $L^* = I(x, y, t)/R_w(x, y)$ . For the division, we use the scene texture image  $R_w$  as a reflectance image. Then, by using this pseudoillumination image as a query, best approximation of the corresponding illumination image  $\hat{L}$  is estimated from the illumination eigenspace based on the following distance metric,

$$\hat{L}_w = \arg \min_{L_{w_i}} \sum_j w_j \sqrt{(\mathcal{F}(L^*, j) - \mathcal{F}(L_{w_i}, j))^2}, \quad (14)$$

where  $\mathcal{F}$  is a projection function which maps an illumination image onto the  $j$ th eigenvector, and  $w_j = \lambda_j / \sum_{\Omega} \lambda_i$  where  $\lambda_i$ s are the eigenvalues. The term  $\mathcal{F}(L_{w_i})$  is premapped onto the illumination eigenspace, each  $L_{w_i}$  is a point in the eigenspace. To achieve the minimization of (14), we used nearest neighbor (NN) search in the illumination

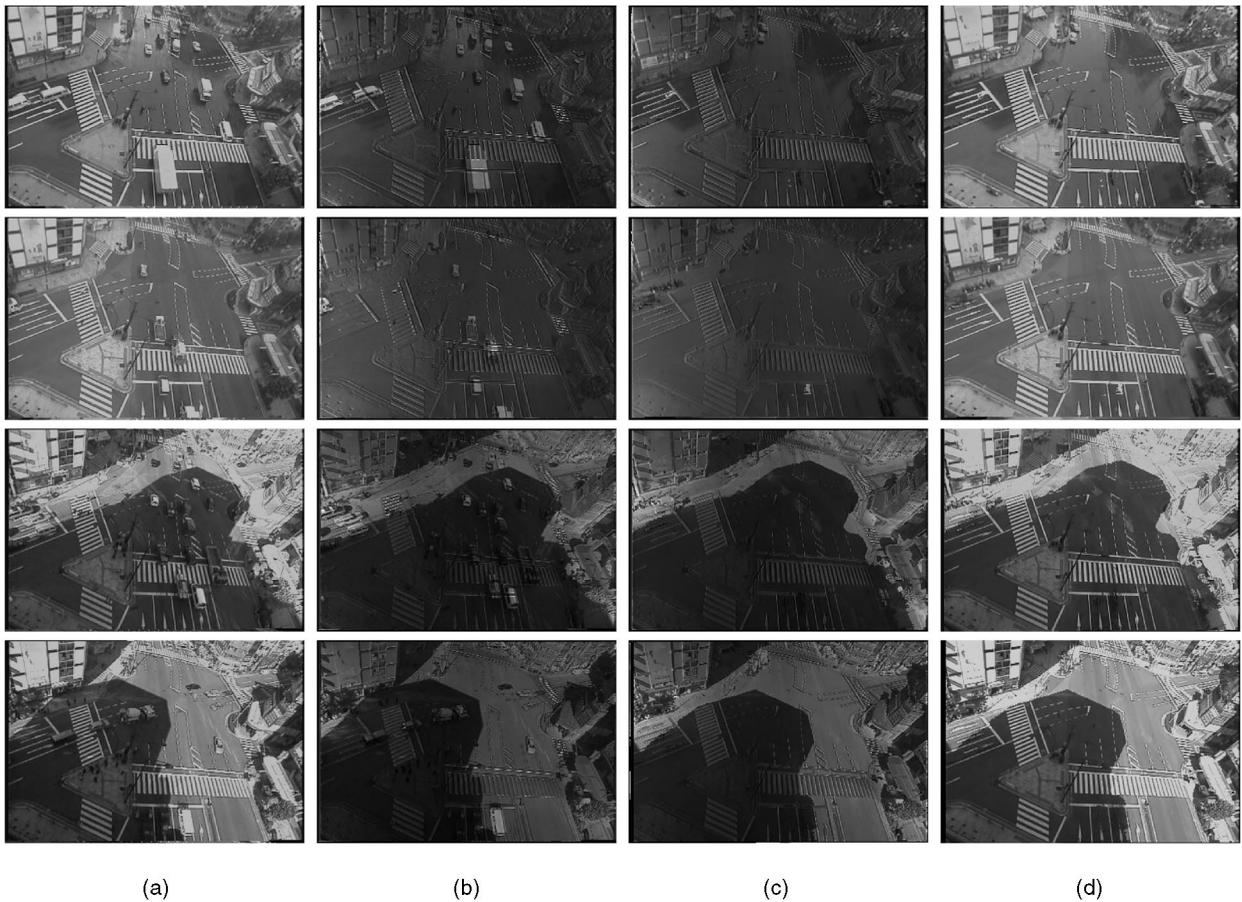


Fig. 5. Direct estimation of intrinsic images. Each row shows different weather conditions. (a) An input image  $I$ , (b) the pseudoillumination image  $L^*$ , (c) the estimated illumination image  $\hat{L}_w$  by the NN search in the illumination eigenspace, and (d) the corresponding background image  $B$  to (c).

eigenspace with the assumption that the distribution of the pseudoillumination image is also Gaussian. We also assume that the moving objects are small enough compared to the image size and the error of  $R_w$  is small. For NN search, the SR-tree method [18] is used which is known to be fast

especially for high-dimensional and nonuniform data structures such as natural images. Finally, the illumination image  $L(x, y, t)$  is derived using the mapping table from  $\hat{L}_w$  to  $L$ . The number of stored images for this experiment was 2,048 and the contribution ratio was 84.5 percent at 13 dimensions, 90.0 percent at 23 dimensions, and 99.0 percent at 120 dimensions. We chose to use 99.0 percent of eigenratio for this experiment. The compression ratio was approximately 17:1, and the disk space needed to store the subspace was about 32 MBytes when the image size is  $320 \times 243$ .

Results of illumination image search is shown in Fig. 5. In this figure, starting with the left-hand side column, the first column shows input images  $I$ , the second column shows pseudoillumination images  $L^*$ , and the third column corresponds to estimated illumination images  $\hat{L}_w$ . The right end column shows the background images which correspond to the estimated illumination images. The NN search in PCA is reasonably robust to estimate the most similar illumination image  $L_w$  from the pseudoillumination image  $L^*$ . However, since the sampling of the illumination images is sparse, there are slight differences in the shadow shapes. It is possible to acquire the exact illumination image  $L$  when the database is dense enough, but it is not easy to prepare such a database. To solve this problem, we propose an approach to compute intermediate illumination images by interpolating shadow regions using geometry estimated

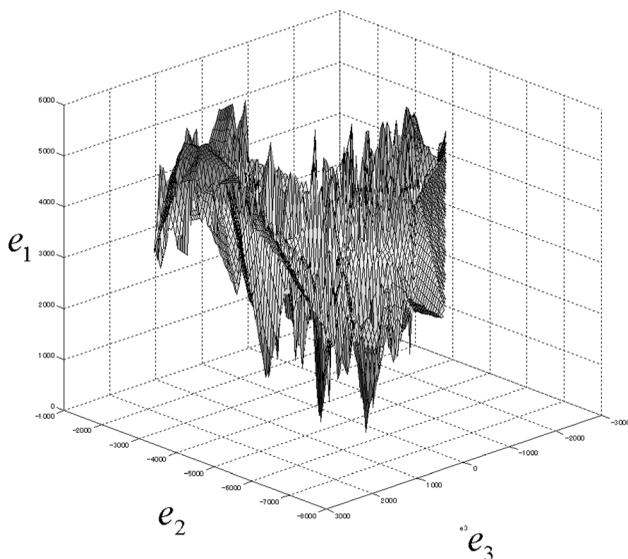


Fig. 6. Illumination eigenspace constructed using 2,048 images from 120 days data of a crossroad.

TABLE 1

Dimension of the Illumination Eigenspace, Contribution Ratio, and NN Search Cost

Dimension	13	23	48	120
Contribution ratio(%)	84.5	90.0	95.0	99.0
NN Search time( $\mu s$ )	6.7	6.8	7.9	12.0

from sampled cast shadow regions and sunlight angles. The details are described in Section 4.1.

As for the computational cost, the average time of the NN search is shown in Table 1 with MIPS R12000 300MHz, when the number of stored illumination images is 2,048, the image size is  $360 \times 243$  and the number of output search results is 5. Since the input image is obtained at the interval of  $33ms$  (at 30 frames/sec), the estimation time is fast enough for realtime processing.

#### 4.1 Shadow Interpolation Using Shadow Hulls

Unfortunately, it is difficult to store all illumination images under every possible illumination condition. NN search in the illumination eigenspace gives good results, however, it is often the case that they are slightly different from the true illumination image due to the limitation of the number of stored illumination images. Therefore, we propose to interpolate NN search results to generate the final estimate of the illumination image. We assume global intensity changes are linear as long as they are densely sampled, but the motion of cast shadows cannot be represented by linear intensity interpolation. To achieve the interpolation of the shadow motion, we propose to use *Shadow Hull* s. The idea of shadow hull is the same as Visual Hull (VH) except using cast shadow silhouette instead of object's silhouette and illumination source as a view point in VH. VH, also known as Shape-From-Silhouette (SFS) [15], [1], is a popular 3D reconstruction method which estimates the shape of the object from multiple silhouette images. These traditional approaches basically assume that the object is static. Recently, extending the traditional SFS formulation to handle the shape of a rigidly moving object over time and a dynamic articulated object are well investigated [12], [10].

Our purpose is to reconstruct the rough shape of an object (such as a building) which gives *enough* information to compute the cast shadow regions given illumination directions. Even though parts of many objects cannot be represented by an SFS-based technique, such as concavities, it does not matter for our method because our purpose is not reconstructing the 3D scene geometry but generating intermediate shadow regions. To accomplish shadow hull reconstruction, we use estimated shadow region maps  $S$ , sunlight angles which are immediately computed from the time-stamp, and calibration parameters. Assuming that the scene is roughly planar, we pitch the shadow volume using the shadow map  $S$  and illumination direction in the world coordinate system. Taking the intersection of different shadow volumes pitched using different shadow maps, the shadow hull can be estimated. The resulting hull is not necessarily precise, but it gives enough scene geometry to compute cast shadow region between sampled illumination

conditions. Also, we do not expect this method to work well if the sampling of the illumination variation is not sufficiently dense.

To compute the intermediate illumination images, we first compute the shadow region map  $S$  for each stored illumination image. Since an illumination image depicts the intensity distribution of reflected lighting from the scene, we can assume that low intensity pixels in the illumination image represent shadowed area. By thresholding, shadow regions  $S$  are derived from the illumination images. To automatically determine the threshold for shadowed area determination, we adopt the clustering technique of Otsu [19]. The threshold is computed from maximizing the between-class scatter by minimizing within-class variances. In our case, we classify pixels into two classes, shadowed and lit, assuming shaded pixels can be categorized into either of them. We first create an intensity histogram for each illumination image to obtain the probability density function  $p(i)$  where  $i$  indicates an intensity value. Assuming that shadowed pixels have relatively lower intensity than lit pixels, we define the cumulative probability functions  $P$  for shadowed ( $P_s$ ) and lit ( $P_l$ ) area using a threshold value  $T$ .

$$P_s(T) = \sum_{i=i_{min}}^T p(i), \quad P_l(T) = \sum_{i=T}^{i_{max}} p(i). \quad (15)$$

In the same manner, we define the mean of the shadowed ( $\mu_s$ ) and lit ( $\mu_l$ ) area as functions of the threshold  $T$  as follows:

$$\mu_s(T) = \sum_{i=i_{min}}^T ip(i), \quad \mu_l(T) = \sum_{i=T}^{i_{max}} ip(i). \quad (16)$$

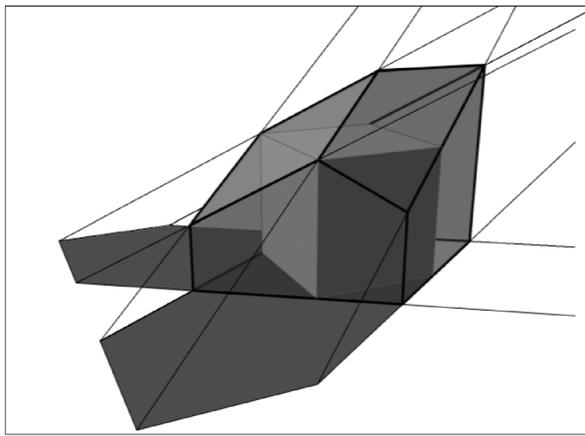
Finally, the optimum threshold value  $T_{opt}$  can be obtained by the following equation:

$$T_{opt} = \arg \max_T \{P_s(T) \cdot P_l(T) \cdot (\mu_s(T) - \mu_l(T))^2\}. \quad (17)$$

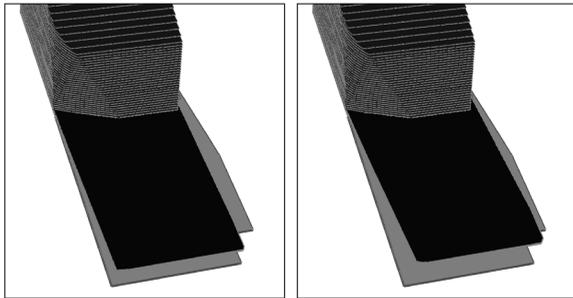
Using  $T_{opt}$ , we compute shadowed area for each illumination image and store the shadowed area in shadow map  $S$ . The shadow map  $S$  is a Boolean map which stores 1 if the pixel is shadowed, otherwise 0.

Shadow regions are then mapped onto the world coordinate, and shadow volumes are computed using shadow regions associated with sunlight angles. By taking the intersection of shadow volumes in the 3D space, we get the rough geometry of objects casting shadows, which has enough information for computing intermediate cast shadows (Fig. 7a). Fig. 7b shows the results of shadow interpolation in a CG scene using an estimated shadow hull. The dark regions show the interpolated shadow regions, while the lighter regions represent the sampled shadow regions.

Shadow interpolation using shadow hulls is useful to estimate the intermediate shadow shapes between sampled lighting conditions. Fig. 8 shows the interpolated result of the real world scene. The left-hand column represents the estimated shadow regions, while the right-hand column shows the corresponding illumination images. The top and bottom row represent the images under sampled illumination conditions, and the middle row depicts the interpolated



(a)



(b)

Fig. 7. Interpolation of cast shadow using a shadow hull. (a) Computing shadow hulls using shadow regions associated with sunlight angles. (b) Result of shadow interpolation.

result. With the assumption that illumination angle is monotonously changing, intermediate shadow regions can be approximately computed using a shadow hull pitched by only two neighboring shadow map samples.

For computational efficiency, we adopt a multiple planar slice representation of shadow hull as shown in Fig. 9. For each  $z$  along the vertical axis, slices parallel to the  $x$ - $y$  plane are assumed. Cast shadows on the world plane are then projected onto these slices using the illumination direction to obtain the region where a shadow volume penetrates each slice. Finally, the intersections of the projected shadow regions are taken as a shadow hull region on the slice. These intersections can be computed with simple Boolean operations. The reason why a set of 2D representation is adopted instead of 3D representation is that our objective is to generate intermediate shadow regions and, for that purpose, dense recovery of the shadow hull is not required. The sampling rate along the  $z$ -axis is currently manually set, however, we have confirmed that as long as it is not too sparse virtually the same result is obtained. As described above, intermediate shadow shapes can be computed with some projection and intersections computations that are very efficient.

Once we obtain the intermediate shadow map  $S_{int}(x, y)$ , we can compute the intermediate illumination image  $L_{int}$  by interpolating the first  $k$  illumination images  $L_k$  that are

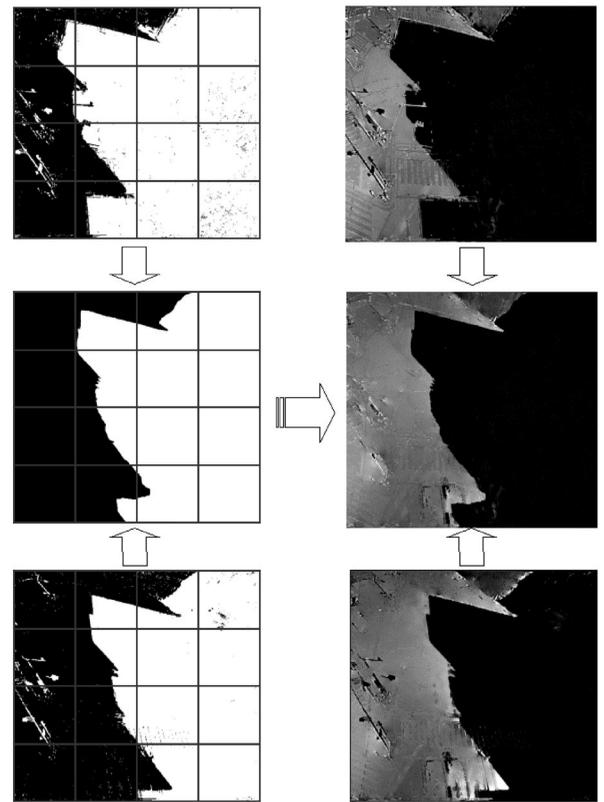


Fig. 8. Shadow hull-based shadow interpolation. Figures in the top and bottom rows are shadow regions and sampled illumination images. The middle row shows the interpolated shadow region. Note that the interpolated result describes only the area of the shadow and is used to compute the intermediate illumination image. The grid is overlaid for better visualization.

computed by NN search. With the  $k$ th illumination image obtained by NN search  $L_k$  and  $w_k$  which is the weighting factor, the distance from  $L_w^*$  to  $L_{w_k}$  (See Section 4) in the illumination eigenspace, the intermediate illumination image  $L_{int}$  can be computed as follows:

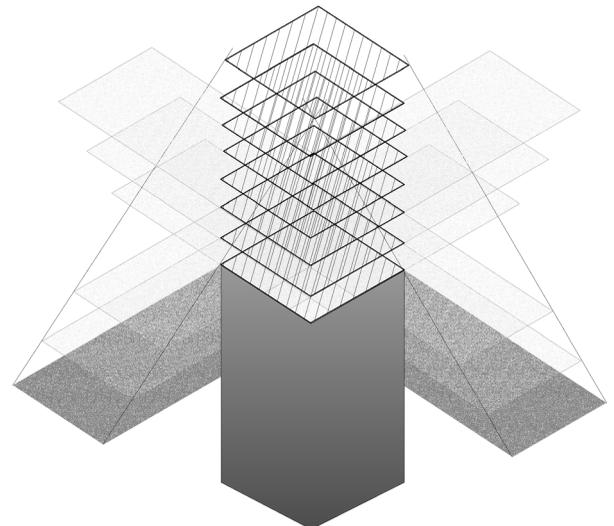


Fig. 9. Illustration of shadow hull construction. To efficiently represent the shadow hull, a set of 2D slices of the shadow hull along the vertical axis is used instead of a volumetric representation.

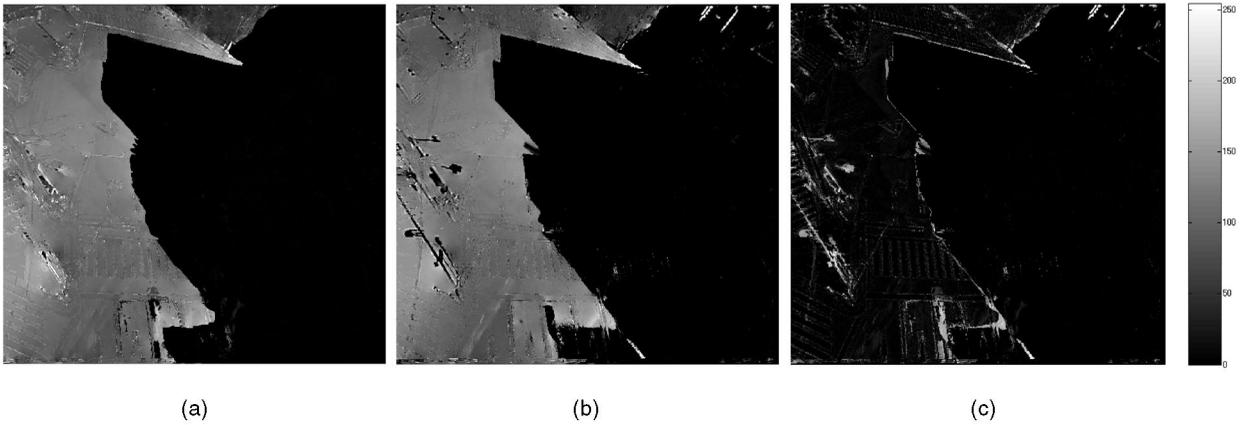


Fig. 10. Comparison with the ground truth. (a) Interpolated result using our method, (b) the ground truth, (c) result of image differencing between (a) and (b).

$$L_{int}(x, y) = S_{int}(x, y) \frac{\sum_k w_k S_k(x, y) L_k(x, y)}{\sum_k w_k S_k(x, y)} + (1 - S_{int}(x, y)) \frac{\sum_k w_k (1 - S_k(x, y)) L_k(x, y)}{\sum_k w_k (1 - S_k(x, y))}, \quad (18)$$

where  $S_k(x, y)$  is a map which holds 1 if the pixel  $(x, y)$  in  $L_k$  is inside the shadow region, otherwise 0. As shown in (18), pixels in shadow and pixels outside the shadow are separately interpolated. Practically, sometimes it is possible to observe that  $\sum_k w_k (1 - S_k(x, y)) = 0$  or  $\sum_k w_k S_k(x, y) = 0$ . In that case, the term whose denominator equals zero is treated as zero because the term does not contain any information.

The resulting intermediate illumination image is shown in middle right in Fig. 8. Fig. 10 shows the comparison between the result of our method and the ground truth. We can notice the slight difference between them from Fig. 10c; however, it shows a globally correct shadow shape which is useful to remove shadow effects from the input image.

To evaluate the effectiveness of our shadow interpolation method, we compared the results with simple image interpolation as shown in Fig. 11. Both our result and image interpolation result are computed from two neighboring illumination images. As we can clearly see in the figure, our

method generates quite similar cast shadow shape to the ground truth while the simple interpolation yields a big ghost at the shadow transition area. To quantify the difference from the ground truth, we evaluated the absolute intensity difference per pixel. Table 2 shows the average of the absolute difference in 12 sets of interpolation results. In our shadow interpolation method, the error is much smaller compared to image interpolation.

## 5 EXPERIMENTAL RESULTS

We evaluated our shadow elimination method by object tracking based on block matching using two different matching functions. One is using sum of squared differences (SSD) as a matching function, and the other is using normalized correlation function (NCF) instead of SSD. NCF is considered to be robust to illumination effects because the matching score is largely independent of linear variations of shading on objects. We chose the block matching algorithm for actual tracking to show even the simplest and widely used tracking method can achieve good results after utilizing our illumination normalization preprocess. The block matching method based on SSD is accomplished by

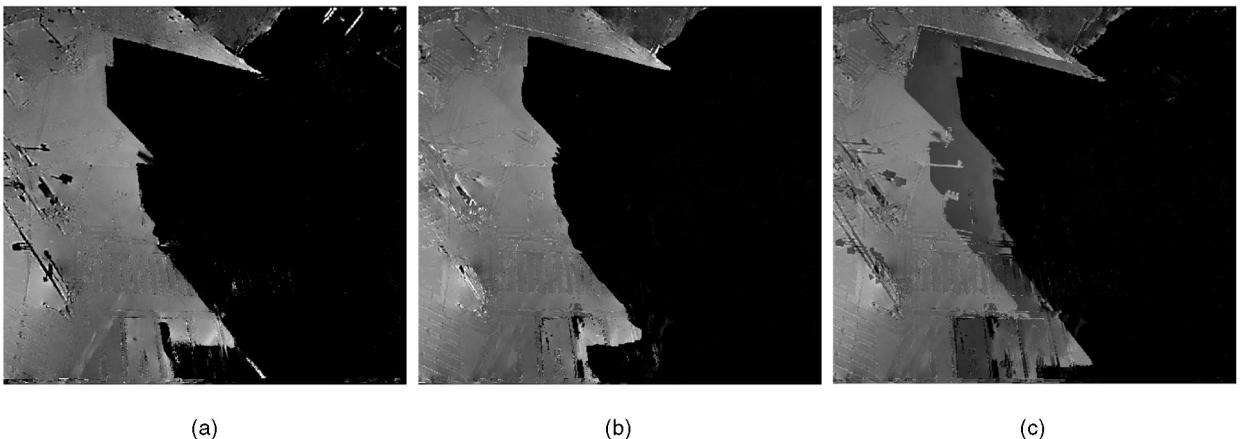


Fig. 11. Comparison of the ground truth, our interpolation result, and image interpolation result. (a) The ground truth, (b) interpolation result using our method, and (c) the result using simple image interpolation. Both (b) and (c) are computed from two illumination images.

TABLE 2  
Absolute Differences of Pixel Intensity from the Ground Truth

	Our method	Image Interpolation
Whole image	0.4328	0.6839
Area of shadow transition	12.6969	67.6341

The top column shows the mean difference across the whole image and the second column shows the mean difference only at the shadow transition regions.

pursuing the most similar window in the neighboring frame evaluated by (19).

$$SSD(x, y) = \min_{\forall i, j} \left\{ \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left( f_t(x+m, y+n) - f_{t-1}(x+m+i, y+n+j) \right)^2 \right\}. \quad (19)$$

For the case of using NCF, the most similar window is found by evaluating (20).

$$NCF(x, y) = \max_{\forall i, j} \frac{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f_t(x+m, y+n) f_{t-1}(x+m+i, y+n+j)}{\sqrt{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f_{t-1}(x+m+i, y+n+j)^2}}. \quad (20)$$

For the experiments, we chose image sequences containing frames of vehicles crossing boundaries of cast shadows since we focus especially on the advantage of our shadow elimination. The parameter set of vehicle tracking for each sequence is totally even, i.e., the same initial window position,  $10 \times 10$  pixels of window, 10 pixels for maximal search distance, and  $10fps$  of frame-rate.

The result is shown in Fig. 12. In Fig. 12, the time axis is represented from top to bottom. The first column of each pair and the second column of that represent results of the block matching-based tracking applied to the original image sequence and image sequence with our preprocessing of illuminance normalization, respectively. In the original image sequence in Figs. 12a, 12b, and 12c, we get typical erroneous results where the block matching fails at shadow boundaries, because of the large intensity variation between inside and outside the shadow. On the other hand, after proper illumination normalization using our method, we get successful results.

We accomplished the tracking experiments over 502 vehicles in 11 sequences under different lighting conditions. Since we cannot have the ground truth, we carefully evaluated the tracking results. The results are shown in Table 3 and Table 4. In the tables,  $O_{correct}$ ,  $O_{error}$ ,  $N_{correct}$ , and  $N_{error}$  mean:

- $O_{correct}$ : Count of the correct tracking results on the original input image sequences.
- $O_{error}$ : Error count of the tracking results on the original input image sequences.
- $N_{correct}$ : Count of the correct tracking results on the preprocessed image sequences using our method.
- $N_{error}$ : Error count of the tracking results on the preprocessed image sequences using our method.

The success rate using the block matching method based on SSD over the original input image sequences was 55.6 percent, while with normalized input it improved to 69.3 percent as shown in Table 3. The effectiveness of our method is clearly confirmed by the result that 45.3 percent of originally failed results were rescued by our method. On the other hand, 11.5 percent got worse after applying our method. This poor effect happens typically when the

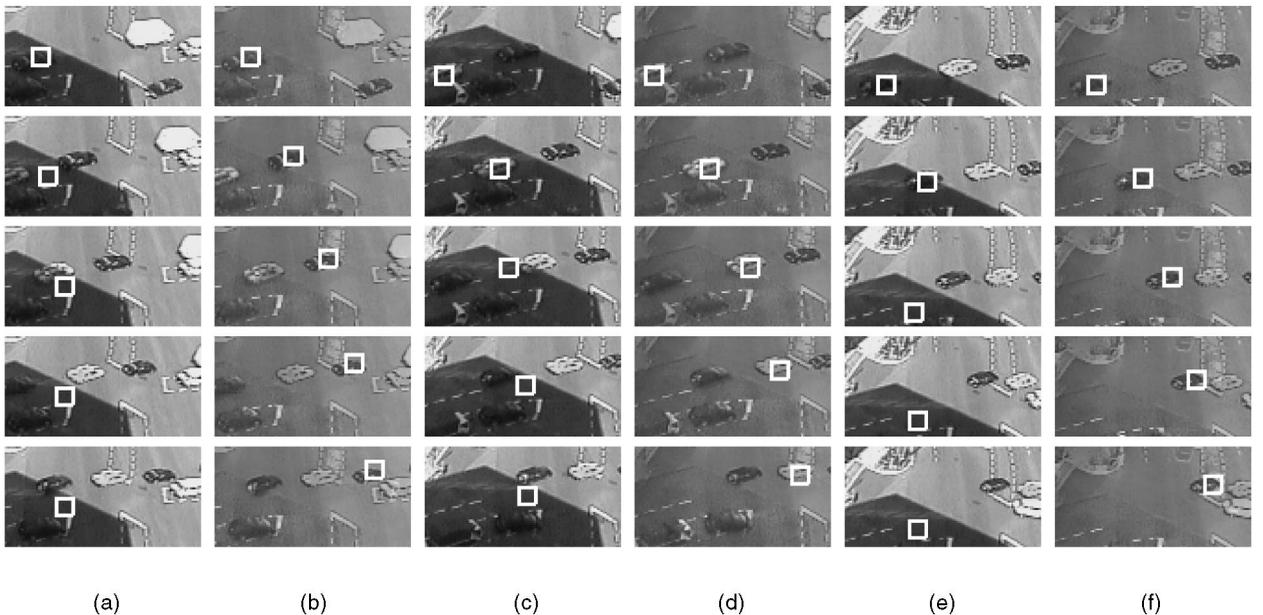


Fig. 12. Results of tracking based on block matching. Along row from top to bottom it shows the frame sequence. The first column of each pair, (a), (b), and (c), shows the tracking result over the original image sequence, and the second column of each pair, (d), (e), and (f), shows the corresponding result after our preprocessing.

TABLE 3

Tracking Result over 502 Sequences Using a Block Matching Method Based on SSD

	$O_{\text{correct}}$	$O_{\text{error}}$	sum
$N_{\text{correct}}$	247	101	348(69.3%)
$N_{\text{error}}$	32	122	154(30.7%)
sum	279(55.6%)	223(44.4%)	502

shadow-edge cast on the vehicle surface largely differs from the shadow-edge in the illumination image. It happens because our method currently can handle only 2D shadows on the image plane, but the actual shadow is cast three-dimensionally on the scene. When the gap of the shadow-edge position is large, the error of the normalization gets large, as a result, the block matching fails. Our system currently does not handle this problem since the error rate is small compared to the improved correct rate, but we are investigating on handling cast shadows three-dimensionally.

Table 4 shows the tracking result using NCF. NCF basically is not sensitive to overall illumination variations, however, it is vulnerable to the case where a shadow boundary is observed inside the window. Suppressing such cases, the tracking error rate reduces from 32 percent to 25.2 percent using our illumination normalization technique. The improvement in the tracking accuracy is not very significant using NCF, however, it is confirmed that our illumination normalization method raises the accuracy of SSD-based tracking to the same level of that of NCF-based tracking. Also, it is important to note that our preprocess does not affect the NCF-based method.

From the experimental results, we confirmed that our method significantly improves the accuracy of tracking using SSD-based block matching. The SSD-based method is the simplest method in the class of tracking methods that do not handle the appearance variation caused by illumination effects. Actually, many rich matching algorithms use SSD-based matching as their core component due to its simplicity for implementation and low computational cost. Therefore, we consider that our preprocess method can be used with this class of methods to improve the accuracy of the matching. As for the methods that are naturally robust against the appearance variations caused by illumination effects such as NCF-based methods, our method may not contribute much to improve the accuracy. However, we have confirmed that our method does not affect the NCF-based method at all and even slightly improves the accuracy.

## 6 CONCLUSIONS

We have described a framework for normalizing illumination effects of real world scenes, which can be effectively used as a preprocess for robust video surveillance. We believe it provides a firm basis to improve existing monitoring systems. We started with a previous method to derive intrinsic images from image sequences, and

TABLE 4

Tracking Result over 535 Sequences Using a Block Matching Method Based on NCF

	$O_{\text{correct}}$	$O_{\text{error}}$	sum
$N_{\text{correct}}$	326	74	400(74.8%)
$N_{\text{error}}$	38	97	135(25.2%)
sum	364(68.0%)	171(32.0%)	535

extended the method to properly handle surfaces with nonrigid reflectance properties. This is accomplished by modifying pseudoderivative illumination images with regards to the scene texture edges that can be derived from the pseudoreflectance image estimated through ML estimation algorithm. The proposed method is remarkably robust and does not require the information of scene geometry, camera parameter, and lighting condition at all, but requires the camera to be fixed and several lighting conditions to be observed. As a key component of our framework, we proposed to utilize *illumination eigenspace*, a preconstructed database which captures the illumination variation of the target scene, to directly estimate illumination images for elimination of lighting effects of the scene, including elimination of cast shadows. As for the intermediate illumination images that cannot be represented by linear combinations of sampled illumination images, we proposed to use shadow hulls for interpolating cast shadow regions using sunlight angles and estimated camera parameters.

The effectiveness of the proposed method was shown by comparing the tracking results between the original image sequence and the image sequence with our method used as a preprocess. Since our method is used as a preprocessing stage, we believe this method can be applied to many video surveillance systems to increase the robustness against lighting variations. The SSD-based matching method is widely used as the core component of many tracking algorithms and have been already integrated in many systems. We have shown that our method can be used to increase the reliability of such SSD-based matching methods. We also have shown that our preprocessing method does not contribute much to the matching algorithms such as NCF-based methods that are naturally robust against appearance variation caused by illumination. However, our method does not negatively affect NCF-based tracking results and it even slightly improves the accuracy. Therefore, we conclude that our preprocessing method is effective for the class of matching methods that do not handle the appearance variation caused by illumination by themselves which we strongly believe has significant use.

In addition to the illumination estimation using illumination eigenspace, we have investigated direct estimation of illumination images corresponding to real scene images using the illumination eigenspace and shadow interpolation based on shadow hulls. Though our current implementation of the shadow interpolation in research code is not fast enough for realtime processing, we believe the framework has the potential to be processed in realtime.

## REFERENCES

- [1] A. Laurentini, "The Visual Hull Concept for Silhouette-Based Image Understanding," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, no. 2, pp. 150-162, Feb. 1994.
- [2] C. Jiang and M.O. Ward, "Shadow Identification," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pp. 606-612, 1992.
- [3] D. Koller, K. Daniilidis, and H.H. Nagel, "Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes," *Int'l J. Computer Vision*, vol. 10, pp. 257-281, 1993.
- [4] E.H. Adelson and A.P. Pentland, "The Perception of Shading and Reflectance," *Perception as Bayesian Inference*, pp. 409-423, 1996.
- [5] E.H. Land, "The Retinex Theory of Color Vision," *Scientific Am.*, vol. 237G, no. 6, pp. 108-128, Dec. 1977.
- [6] G.D. Finlayson, S.D. Hordley, and M.S. Drew, "Removing Shadows from Images," *Proc. European Conf. Computer Vision*, vol. 4, pp. 823-836, 2002.
- [7] H.G. Barrow and J.M. Tenenbaum, "Recovering Intrinsic Scene Characteristics from Images," *Computer Vision Systems*, pp. 3-26, 1978.
- [8] J. Huang and D. Mumford, "Statistics of Natural Images and Models," *Proc. Conf. Computer Vision and Pattern Recognition*, pp. 541-547, 1999.
- [9] J. Stauder, R. Mech, and J. Ostermann, "Detection of Moving Cast Shadows for Object Segmentation," *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 65-76, Mar. 1999.
- [10] K. Grauman, G. Shakhnarovich, and T. Darrell, "A Bayesian Approach to Image-Based Visual Hull Reconstruction," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 187-194, 2003.
- [11] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and Practice of Background Maintenance," *Proc. Int'l Conf. Computer Vision*, pp. 255-261, 1999.
- [12] K.M. Cheung, S. Baker, and T. Kanade, "Shape-from-Silhouette of Articulated Objects and Its Use for Human Body Kinematics Estimation and Motion Capture," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 77-84, 2003.
- [13] M. Bell and W.T. Freeman, "Learning Local Evidence for Shading and Reflection," *Proc. Int'l Conf. Computer Vision*, vol. 1, pp. 670-677, 2001.
- [14] M. Kilger, "A Shadow Handler in a Video-Based Real-Time Traffic Monitoring System," *Proc. IEEE Workshop Applications of Computer Vision*, pp. 11-18, 1992.
- [15] M. Potmesil, "Generating Octree Models of 3D Objects from Their Silhouettes in a Sequence of Images," *Computer Vision Graphics and Image Processing*, vol. 40, pp. 1-20, 1987.
- [16] M. Turk and A. Pentland, "Eigenfaces for Recognition," *The J. Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.
- [17] M.F. Tappen, W.T. Freeman, and E.H. Adelson, "Recovering Intrinsic Images from a Single Image," *Advances in Neural Information Processing Systems 15*, MIT Press, 2002.
- [18] N. Katayama and S. Satoh, "The SR-Tree: An Index Structure for High-Dimensional Nearest Neighbor Queries," *Proc. 1997 ACM SIGMOD Int'l Conf. Management of Data*, pp. 369-380, 1997.
- [19] N. Otsu, "A Threshold Selection Method from Gray Level Histograms," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 9, pp. 62-66, 1979.
- [20] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, second ed. Cambridge Univ. Press, 1992.
- [21] Y. Weiss, "Deriving Intrinsic Images from Image Sequences," *Proc. Ninth IEEE Int'l Conf. Computer Vision*, pp. 68-75, July 2001



**Yasuyuki Matsushita** received the BEng, MEng, and PhD degrees in electrical engineering from the University of Tokyo in 1998, 2000, and 2003, respectively. Currently, he is an associate researcher at Microsoft Research Asia. His research interests include photometric methods in computer vision, image-based rendering, and modeling from images. He is a member of the IEEE.



**Ko Nishino** received the BE and ME degrees from The University of Tokyo in 1997 and 1999, respectively. He received the PhD degree in information science from The University of Tokyo in 2002. Since 2002, he has been a postdoctoral research scientist at Columbia University. His research interests span computer vision and computer graphics. He has published several papers on photometric and geometric problems in scene and object modeling that include physics-based vision, image-based modeling, and rendering. He is a member of the IEEE and ACM.



**Katsushi Ikeuchi** received the BE degree from Kyoto University in 1973 and the PhD degree from the University of Tokyo in 1978. After working at the MIT AI Laboratory for three years, ETL for five years, and CMU Robotics Institute for 10 years, he joined the University of Tokyo in 1996, and is currently a full professor. His research interest spans computer vision, robotics, and computer graphics. In these research fields, he has received several awards, including the David Marr Prize in computational vision for the paper "Shape from Interreflection," and IEEE R&A K-S Fu memorial best transaction paper award for the paper "Toward Automatic Robot Instruction from Perception—Mapping Human Grasps to Manipulator Grasps." In addition, in 1992, his paper, "Numerical Shape from Shading and Occluding Boundaries," was selected as one of the most influential papers to have appeared in the *Artificial Intelligence Journal* within the past 10 years. His IEEE activities include general chair, IROS95, ITSC00, IV01; program chair, CVPR96, ICCV03; Associate Editor, IEEE TRA, IEEE TPAMI; distinguished lecture SPS (2000-2002), RAS (2004-2006). Dr. Ikeuchi was elected as an IEEE fellow in 1998. He is the EIC of the *International Journal of Computer Vision*.



**Masao Sakauchi** received the BSci degree in electrical engineering from the University of Tokyo in 1969 and the MS and PhD degrees in electronics engineering from the University of Tokyo in 1971 and 1975, respectively. He is a deputy director general of the National Institute of Informatics, Japan, and a professor in the Institute of Industrial Science at the University of Tokyo. He has acted as the general chairman and program chairman of 10 international conferences and workshops, including the IEEE International Workshop on Machine Vision and Machine Intelligence (1987), IAPR and IEEE International Conference on Document Analysis and Recognition (ICDAR '93) (1993), IEEE International Conference of Multimedia Processing and Systems (IEEE Multimedia 96) (1996), and Intelligent Transportation Systems Conference ITSC '99 (1999). He has authored more than 340 refereed papers in the research fields of multimedia databases, multimedia systems, image processing and understanding, spatial data structures, geographical information systems, and fault tolerant computing.

► For more information on this or any other computing topic, please visit our Digital Library at [www.computer.org/publications/dlib](http://www.computer.org/publications/dlib).