Target-color Learning and Its Detection for Non-stationary Scenes by Nearest Neighbor Classification in the Spatio-Color Space

Norimichi Ukita
Graduate School of Information Science, Nara Institute of Science and Technology
ukita@ieee.org

Abstract

We propose a method for detecting foreground objects in non-stationary scenes. The method can (1) detect arbitrary foreground objects without any prior knowledge of them, (2) identify background pixels under various changes in a background scene, and (3) detect minor difference between the background and target colors. Online detection is realized by the nearest neighbor classifier in the 5D xy-YUV space (the spatio-color space), consisting of the x and y coordinates of an image and Y, U, and V colors, which holds rectified training data of background colors and automatically learned target colors. We conducted experiments to confirm the effectiveness of our method.

1. Introduction

Object detection is one of the most fundamental techniques used in building various vision systems. Background subtraction is effective in exactly extracting the regions of arbitrary objects from incoming images. Earlier studies on background subtraction have discussed the problems below:

Problem 1: Varying illumination Miss-detection of the variations in background colors due to the changes in the shadows and lighting intensity and color.

Problem 2: Non-stationary background objects Miss-detection of swaying objects (e.g., leaves and flags), such as a swaying curtain.

Problem 3: Similar colors Miss-discrimination between similar background and foreground colors.

In [1], to cope with problem 1, illumination components in an observed image is estimated by the eigenspace method using a large amount of background images. Foreground objects can be detected by comparing the observed image with the image generated from the illumination eigenspace. Similar to [1], many algorithms probabilistically model background images taken in advance. For example, whole images are modeled by Principle Component Analysis[2] and the median template and standard deviation of the normalized correlation from the median are calculated for each pixel block in an image[3]. However, these methods cannot exactly detect the detailed boundaries of objects because an approximate representation of the background is employed.

In [4], on the other hand, the distribution of gray values represented by the Gaussian mixture model in each background pixel is updated according to the incoming images. This background model is useful to cope with problems 1 and 2. However, since the background model is updated gradually, the detection result becomes unreliable when sudden changes in illumination happens. In [5], the problem caused by sudden changes in illumination is solved by automatically retaining a representative set of background models and switching between them. Even with the background representations in [4, 5], however, performance deterioration due to approximation is unavoidable. Any newer similar approach (e.g., [6, 7]) has the same limitation.

The above discussion suggests that it is difficult to solve problem 3 based on background subtraction. This is because (1) approximate representation of background pixels results in detection errors and (2) only learned background colors are referred for distinguishing between foreground and background pixels. That is, comparing the colors of the foreground and background objects without approximate representation makes it easier to solve problem 3.

This idea has been realized in [8]. To exactly represent arbitrary distributions of background and target colors, non-processed colors observed in training images are recorded in a classification space (3D YUV color space). Target color detection is implemented by the nearest neighbor classification in the 3D YUV space. Moreover, the major drawback of the nearest neighbor classification, namely computational complexity, is overcome with a look-up table.

However, all methods for target color detection including [8] exhibit the following problems:

Problem A: Lack of spatial information Color values of every pixel are recorded in a color space and processed independent of their coordinates. However, the spatial similarity of colors is important for efficiency of color representation. Furthermore, the crowds of training background colors in a color space produce the overlap between colors of different objects. This leads to a deterioration in the accuracy of classification.

Problem B: Non-automatic processing Since target colors must be entered manually in the classification space in advance, it is not applicable to an autonomous system that can detect any object.

The discussions in Sec. 1 can be summarized as follows:

- Background subtraction maintains the appearance information of each pixel and enables detection of unknown arbitrary objects. When a target object overlaps with a similar colored background object, the accuracy of detection declines.
- Color detection is capable of discriminating between similar colors. It requires careful and manual color learning and representation in advance and does not cover spatial information.
- The nearest neighbor classifier can processes unapproximate values observed in the images. This results in improving the accuracy of both background subtraction and color detection.

As described above, (1) background subtraction can compensate the deficit in color detection and vice versa, and (2) the nearest neighbor classifier can improve the accuracy of both the methods. In this paper, therefore, we propose an object detection method by integrating background subtraction and color detection with the nearest neighbor classifier.

2. Spatio-Color Space

Our objective is to find a solution to problems 1, 2, 3, A, and B mentioned in Sec. 1. Information of arbitrary objects has to be automatically detected and collected. The colors of a target object can be easily collected when the target is in front of a background object whose color is quite different from that of the target. By comparing the colors of the background and target objects using the nearest neighbor classifier, the accuracy of object detection can be improved. In addition, the nearest neighbor classifier can concatenate adjacent pixels in the classification space.

On the basis of the above concepts, we employ the spatio-color space for learning and classification:

- A set of background images that include various changes in illumination and non-stationary background objects (left upper part in Fig. 1) is acquired in advance. All the color values (YUV values) in the background set are regarded as the training data.
- To represent the adjacency relationships of pixels, the YUV values at a particular xy position in the background image are recorded in a corresponding position in the xy-YUV space (right part in Fig. 1).
- To learn target colors of the foreground objects, a region considered to be the foreground object ('Foreground region' in Fig. 1) is detected based on the distance between incoming and background pixels in the classification space. The color values of the detected pixels are recorded into the classification space and employed for the nearest neighbor classification.

Employing spatial and color information for image representation is general idea; for example, [9] has employed it for image segmentation. In this paper, however, we apply it to background modeling and target detection.

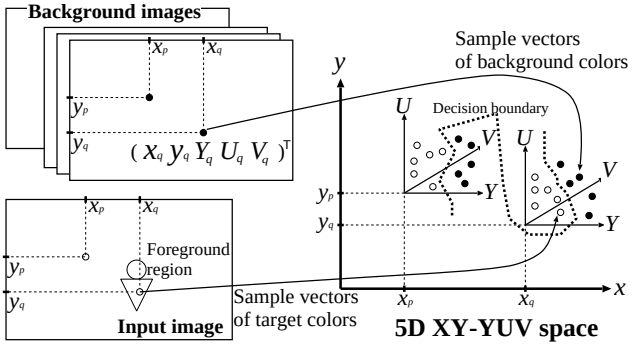


Figure 1: Color classification in the 5D xy-YUV space.

3. Color Learning and Detection

3.1. Background Model Generation

To guarantee that arbitrary objects can be detected under changes in illumination and non-stationary background objects, background images that include all possible changes should be collected. It is, however, difficult to observe all changes in advance, including gradual changes in illumination in time of day, shadows cast by moving clouds, and reflections of foreground objects onto a background. Background maintenance methods[4, 5] cope with this problem by updating the background model according to pixel values in the previous few frames. Although the update of the background model is not realized in this paper, our method can avoid detection errors through the following steps:

Detection only based on the background information Since the training images of a background are possibly incomplete, training values are rectified (described later) and the criterion for detection is designed such that regions that can certainly be regarded as foreground objects are detected (See Sec. 3.2).

Detection using the nearest neighbor classifier After target colors are learned, the nearest neighbor classifier enables discrimination between the background and the targets robustly against errors and variations in observed images (See Sec. 3.3).

First, some background images are captured while illumination conditions and other changes in a background are varied. The entire set of xy-YUV values in the captured images are then recorded into the xy-YUV classification space. Here, we should determine the sampling rates for the xy-space and YUV-color axes because a large classification space leads to high computational costs. We, therefore, determine the sampling rate for each axis taking into account the trade-off between the discrimination performance and the computational cost. In an example shown in Fig. 2, (1) the x and y values are resampled one-b-th to obtain a set of $b \times b$ YUV colors (denoted by S_S) at a xy position in the

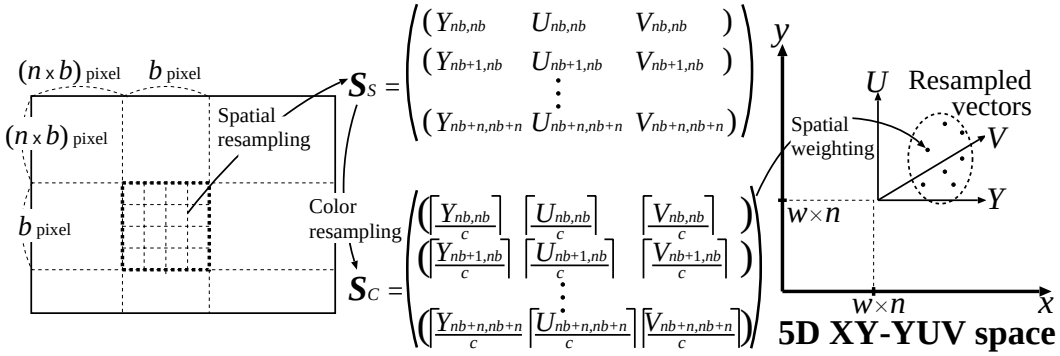


Figure 2: Spatio-color resampling from the observed image to the 5D xy-YUV space.

classification space, and (2) a set of reduced YUV colors (denoted by S_S) are obtained by resampling the Y, U, and V values one-c-th.

The xy-YUV classification space consists of different kinds of axes, namely the xy space and the YUV color space. This may possibly exert an adverse influence on the result of discrimination. We, therefore, assign different weights to the xy axes and the YUV axes. In an example shown in Fig. 2, the YUV colors extracted from the $(x = n, y = n)$-th block in the image are recorded into the $(x = wn, y = wn)$-th block in the classification space by increasing a unit length of the x and y axes by n. If this weight is quite large, the xy-YUV space is identical to a group of independent YUV space for each xy coordinates. Note that the resampling process adjusts the size of the classification space, but the image size is not reduced. That is, for example, the nearest neighbor classification is executed 640×480 times for an image with 640 $\times$ 480 pixels.

In the YUV representation, Y values in the same pixel change largely even in a fixed illumination. This change happens due to camera noise. To cope with this problem, Y values that are certainly regarded as background colors are registered as background values for interpolation. The Gaussian mixture model is used to evaluate the observation probabilities of Y values. With the weight variable w_i, which is proportional to the number of training data in each Gaussian, the mean vector $\boldsymbol{\mu}_i$, and the covariance matrix $\boldsymbol{\Sigma}_i$ of the i-th Gaussian in the mixture, the probability of observing $\boldsymbol{C}_p$ is represented by a mixture of K-Gaussians:

$$P(\boldsymbol{C}_p) = \sum_{i=1}^{K} w_i \, q\left(\boldsymbol{C}_p; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right), \qquad (1)$$

where $q\left(\boldsymbol{C}_p; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)$ denotes the Gaussian probability density function defined by the following equation:

$$q\left(\boldsymbol{C}_p; \boldsymbol{\mu}, \boldsymbol{\Sigma}\right) = \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{C}_p - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{C}_p - \boldsymbol{\mu})}$$

Note that a probability threshold for interpolation must be high such that only Y values that are certainly regarded as background colors are interpolated.

Similar to the interpolation process, background YUV values with quite low observation probabilities must be removed. This is because these background values suppress target-color learning around them and then the performance of discrimination between foreground and background colors decreases in general cases. Note that this outlier elimination process is implemented for YUV vectors while the interpolation process is executed only for Y values.

3.2. Detecting and Learning Initial Target Colors

Object detection has to be implemented only based on the learned images of a background when no target information is stored in the classification space.

First, each xy-YUV value in an observed image is projected into the classification space in accordance with the rule used in learning the xy-YUV values of a background. Each projected xy-YUV value is classified into "background" or "others" by the nearest neighbor classifier. In this paper, only two classes exist, namely a background and a target. That is, the colors in all foreground objects are regarded as target colors even if multiple objects are observed.

Every xy-YUV value is classified as a background pixel by the nearest neighbor classifier if no target color is registered. To detect object regions without the nearest neighbor classification, a xy-YUV value, whose distance to the nearest neighbor point is larger than a threshold Th_d, has to be determined. The threshold Th_d should be adjusted depending on the extent of dispersion of the color values at each position in the background images. We evaluate the extent of dispersion by representing the YUV values at each block with the Gaussian mixture model as well as the method proposed in [4]. In [4], the background model (Eq. 1) is utilized for object detection under the following conditions:

- The number of Gaussian components is small because the mixture of Gaussians at every capturing timing should be updated in real time in order to track the changes in a scene.
- If an incoming pixel is within 2.5 standard deviations

of any K Gaussian distribution, the pixel is considered to be a background pixel.

Our method, on the other hand, employs the mixture of Gaussians as follows:

- A large K is used to represent a complicated distribution of background colors as exactly as possible because our method collects the static data of background appearances prior to monitoring a scene.
- Since the background model is not updated, the threshold Th_d can be determined in advance. We determine Th_d so that it is identical to the maximum distance from the boundary of 3.0 standard deviations of $P(\boldsymbol{C}_p)$ to the nearest point of the training background values; 3.0 standard deviations of $P(\boldsymbol{C}_p)$ is considered to be adequately large to avoid miss-detection.

Suppose $(x_p, y_p, Y_p, U_p, V_p)^\top$ is newly detected as a foreground value by implementing the process mentioned above. The YUV value (Y_p, U_p, V_p) should then be registered for all xy coordinates in the classification space in order to distinguish (Y_p, U_p, V_p) as a target value wherever (Y_p, U_p, V_p) is observed in an image. However, $(x_q, y_q, Y_p, U_p, V_p)^\top$ may possibly be registered as a background value at (x_q, y_q). If $(x_q, y_q, Y_p, U_p, V_p)^\top$ is reregistered as a target value, the image pixel corresponding to (x_q, y_q) in the classification space is always detected incorrectly. To avoid this problem, we employ the following procedure for registering target colors:

Step 1 When (Y_t, U_t, V_t) is newly detected as a foreground value, the nearest neighbor classification is executed for all $\{(x_i, y_i, Y_t, U_t, V_t)^\top | i \in S\}$, where S is a set of all xy coordinates in the classification space. Here, the nearest neighbor classification used in our method not only classifies an incoming value but also calculates the distance between the incoming value and the nearest neighbor training value.

Step 2 If the distance to the nearest neighbor training value is larger than a threshold Th_l, it can be considered that $(x_i, y_i, Y_t, U_t, V_t)^\top$ does not overlap with the distribution of training background values. $(x_i, y_i, Y_t, U_t, V_t)^\top$ is then registered as a target value.

In the above mentioned procedure, the pixel in a target object, whose color value is the same as the background color, is regarded as a background pixel. Intrinsically, identical colors cannot be distinguished only on the basis of their color values. Therefore we consider that such deficit in the detected pixels should be rectified by the higher-level processing implemented after simple discrimination of background and foreground pixels.

3.3. Detection and Learning with the Nearest Neighbor Classifier

After target colors are registered, the nearest neighbor classifier may detect the xy-YUV value $(x_p, y_p, Y_p, U_p, V_p)^\top$ that is regarded as a target color. Under the assumption that the training data of background colors is sufficiently registered to represent all the changes in a background scene, the result of target color detection using the nearest neighbor classifier (V_1 in the left illustration of Fig. 3) is reliable even if the training data of target colors is insufficient. The pixel (x_p, y_p) can therefore be detected as a target pixel. On the other hand, the result of background color detection using the nearest neighbor classifier (V_2 in the left illustration of Fig. 3.) is unreliable.

This problem can be solved as more training data of target colors is accumulated. Figure 3 illustrates an example. While V_1 is considered to be a target value, V_2 is erroneously regarded as a background value when the amount of the training data on target colors is insufficient (the left illustration of Fig. 3). Both V_1 and V_2 are classified as target values after sufficient target colors are registered (the right illustration of Fig. 3).

Even if a xy-YUV value $(x_p, y_p, Y_p, U_p, V_p)^\top$ is newly regarded as a target value with the nearest neighbor classifier, the xy-YUV value is not unconditionally registered as a target. This is because miss-detection across the entire observed image may possibly occurs if a xy-YUV value ,whose position is close to training background values, is registered as a target value, as mentioned in the last section. Therefore, steps 1 and 2 introduced in the last section must be executed in order to check whether or not (Y_p, U_p, V_p) should be registered as a target at each xy position in the classification space.

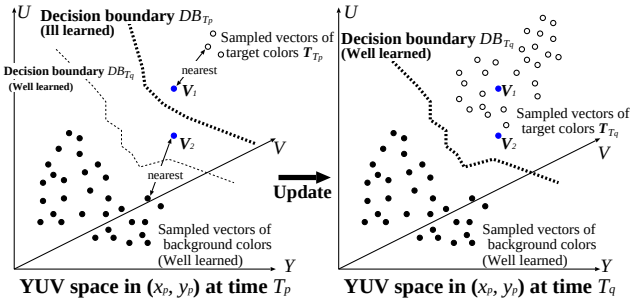


Figure 3: Change in the nearest neighbor class depending on the training data of target colors. This figure illustrates the 3D YUV space of (x_p, y_p) in the 5D xy-YUV space.

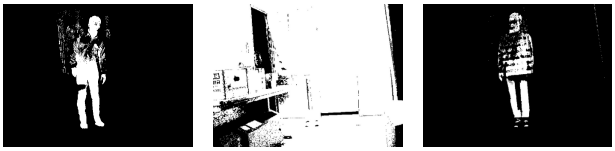
Table 1: Results of detection and color learning depending on the nearest neighbor class and the distance to the nearest neighbor point: an *italicized word* indicates the classification result of the nearest neighbor classifier.

	Detection result / Learning colors or not	
Distance \ *Result*	*Background*	*Target*
$D \leq Th_b$	Background	Target / Non-learning
$D > Th_b$	Target /Learning	Target / Learning

Figure 4: Ex.1: Experimental environment.

(a) With light (b) Right after lights-out (c) Without light

Figure 5: Ex.1: Test images for performance comparison.

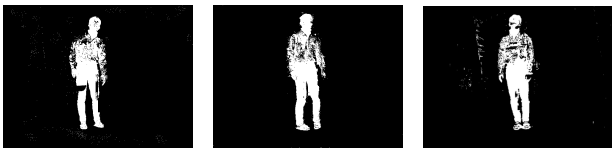


(a) With light (b) Right after lights-out (c) Without light

Figure 6: Ex.1: Results of Gaussian Mixture[4].

(a) With light (b) Right after lights-out (c) Without light

Figure 7: Ex.1: Results of Wallflower[5].



(a) With light (b) Right after lights-out (c) Without light

Figure 8: Ex.1: Results of our method.

4. Experimental Results

We conducted experiments using a Pentium 2.4GHz PC and SONY IEEE1394 camera (DFW-VL500). The size of the observed image was 640 $\times$ 480 pixel. Figure 4 shows the experimental environment with the following difficulties:

Problem 1: Varying illumination The left and right images in Fig. 4 show the images observed with and without light, respectively.

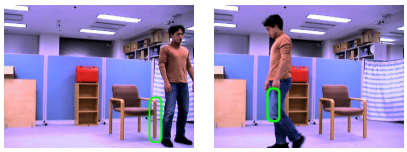
Problem 2: Non-stationary background objects A curtain observed in the left upper part of the image was swaying in the wind.

Problem 3: Similar colors The colors of the curtain are similar to those of the shirt of the target person.

For a comparative evaluation, we evaluated the Gaussian Mixture[4] and Wallflower[5]. Three images in Fig. 5 show the test images used for comparison.

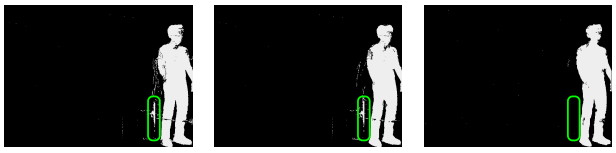
The Gaussian mixture model[4] could suppress miss-detection in the swaying curtain as shown in Fig. 6 (a). However, in both bright and dark conditions, similar colors of the background and target objects could not be well discriminated. Moreover, a sudden light change resulted in severe miss-detection across the entire image because the background model could not be updated according to the sudden change in lighting (Fig. 6 (b)).

Figure 9: Ex.2: Experimental environments.



(a) Cast shadows (b) Similar colors

Figure 10: Ex.2: Test images for performance comparison.



(a) Gaussian Mixture (b) Wallflower (c) Our method

Figure 11: Ex.2: Comparison: Identifying a background object with cast shadows.

(a) Gaussian Mixture (b) Wallflower (c) Our method

Figure 12: Ex.2: Comparison: Discerning similar colors.

Wallflower[5] produced better results as shown in Fig. 7. In particular, miss-detection was suppressed almost completely in the image observed immediately after the light was switched off. However, the problem regarding discrimination between similar colors was not solved.

We display the experimental results of our method in Figure 8. The fast nearest neighbor classification in the xy-YUV space is implemented by employing Approximate Nearest Neighbor[10] and an efficient caching technique. The sampling rates of the xy axes and the YUV axes were $\frac{1}{8}$ and $\frac{1}{2}$, respectively. The xy and YUV axes were adjusted such that the unit lengths of the xy and YUV axes were in the ratio of 2:1. With these resources and parameters, our method captured and processed images at 5$\sim$9 fps depending on the cache hit ratio. On the other hand, Gaussian mixture and Wallflower worked at 16 fps and 19 fps on average. However, processing speed increases as a CPU power improves and our method will possibly be able to work at video rate in a few years.

Initially, five background images were captured under each illumination condition shown in Fig. 4. The entire set of xy-YUV values in these were registered as background

Table 2: Performance of the three methods: the number of false negative and positive pixels are indicated.

Method	Error type	Observation scene: Scene1: light	Scene1: lights-out	Scene1: no light	Scene1: total	Scene2: light	Scene2: lights-out	Scene2: no light	Scene2: total	Total
Gaussian	false neg.	11347	2092	17695	12745	14347	3043	16451	13633	13189
Mixture	false pos.	2859	220021	2044	33522	3340	181456	4634	29339	31435
	false neg.	9823	13768	11865	11261	9576	11170	13058	11296	11278
Wallflower	false pos.	2786	3432	1427	2295	2478	4570	2379	2734	2514
Our	false neg.	4386	6167	7122	5813	4445	7347	7039	5971	5892
method	false pos.	2313	2923	3029	2707	2652	2223	2661	2594	2650

values into the xy-YUV space. Figure 8 shows the results obtained after sufficient learning; learning target values was continued for about five seconds at each illumination condition. It can be seen that the proposed method produced the best results in these experiments.

We also conducted experiments in a different scene. Figure 9 shows four lighting conditions of the background scene. In each of these lighting conditions, five images were captured. The two images in Fig. 10 show the test images for comparison. The left and right images were used to check whether or not each method could identify background pixels robustly against a shadow cast on a background object (rectangles indicate the shadow cast on a chair by the target person) and distinguish between background and target pixels whose colors were very similar (blue wall and jeans), respectively. It must be noted that our method could identify shaded background pixels using the nearest neighbor classifier even if they were not included in the reference background images. The results shown in Figures 11 and 12 confirm the effectiveness of our method.

We evaluated the performance of three methods. Two image sequences of 40 seconds were observed in the scenes shown in Figures 4 and 9, respectively. Seven images at interval of five seconds, including three kinds of the lighting conditions (i.e., with light, lights-out, and without light), were then extracted from each image sequence and selected as test images. The detected results were compared with the true target regions given by hand. Table 2 lists the results of performance comparison. By assessing every factor, we can confirm that our method produced the best results.

5. Concluding Remarks

We proposed a real-time object detection method for non-stationary scenes. In this method, the nearest neighbor classifier is effective in the 5D space comprising the x and y axes of an image and YUV axes of the color representation. The nearest neighbor classification in the 5D classification space enables not only the identification of a background scene under sudden and extreme variations in illumination but also the detection of an arbitrary object whose colors are similar to those of the background pixels.

We are examining the following aspects:

- Target color learning for each observed object.
- Appropriate selection of multiple classification spaces, each of which is optimized for respective illumination conditions.

This study is supported by PRESTO program of JST and the Grant-in-Aid for Scientific Research (No.15700157).

References

[1] Y. Matsushita, K. Nishino, K. Ikeuchi and M. Sakauchi: "Illumination Normalization with Time-dependent Intrinsic Images for Video Surveillance," In *Proc. of CVPR2003*, Vol.1, pp.3–10, 2003.

[2] N. M. Oliver, B. Rosaria, and A. Pentland: "A Bayesian Computer Vision System for Modeling Human Interactions," IEEE Transaction on *PAMI*, Vol.22, No.8, pp.831–843, 2000.

[3] T. Matsuyama, T. Ohya, and H. Habe: "Background subtraction for non-stationary scenes," In *Proc. of ACCV 2000*, pp.662–667, 2000.

[4] C. Stauffer and E. Grimson: "Adaptive background mixture models for real-time tracking," In *Proc. of CVPR99*, Vol.2, pp.246–252, 1999.

[5] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers: "Wallflower: Principles and Practice of Background Maintenance," In *Proc. of ICCV99*, pp.255–261, 1999.

[6] J. Zhong and S. Sclaroff: "Segmenting Foreground Objects from a Dynamic Textured Background via a Robust Kalman Filter," in *Proc. of ICCV03*, pp.44–50.

[7] A. Mittal and N. Paragios: "Motion-Based Background Subtraction using Adaptive Kernel Density Estimation," in *Proc. of CVPR04*, Vol.2, pp.302–309, 2004.

[8] T. Wada: "Color-target Detection Based on Nearest Neighbor Classifier," IPSJ Transaction on Computer Vision and Image Media, Vol.44, No.SIG17, pp.126–135, 2003.

[9] D. Comaniciu and P. Meer: "Mean Shift Analysis and Applications," in *Proc. of ICCV'99*, pp.1197–1203, 1999.

[10] S. Arya, D. M. Mount, N. Netanyahu, R. Silverman, and A. Y. Wu: "An optimal algorithm for approximate nearest neighbor searching in fixed dimensions," In *Proc. of 5th ACM-SIAM Symposium Discrete Algorithms*, pp.573–582, 1994.