平成２５年度　博士学位論文

Computational complexity reduction and performance improvement for object detection
(物体検出における計算量低減と精度向上に関する研究)

鳥取大学大学院工学研究科
博士後期課程情報エレクトロニクス専攻

臼井　温

Abstract

Vision is a basic sense in human beings, and a lot of the information we receive comes mainly from our ability to visualize our surroundings. This is mainly true for robots as well. Therefore, visual technology has many applications in such areas as robotics, manufacturing, and security. Robots and home appliances that can understand their surroundings based on visual information can be very useful for human users. They can provide rich opportunities through their use of visual information such as a user-friendly human interface.

However, image recognition issue is not easy to achieve. There are many inherent problems such as luminance changes, complex backgrounds, rotational changes of the target objects, and occlusion problems. Many researchers have been working stridently to solve these problems.

Considering computational resources, computational processing power has increased in recent years, allowing computers to deal with huge amounts of information. The various methods of object detection are proposed that assumes by using huge computational power. However, such methods are impractical for embedded applications such as robots and home appliances, because we need to implement into small-sized hardware. Many works have assumed the implementation of generic processors. We cannot implement generic processors into smaller robots or home appliances.

There are many techniques and methods suitable for implementation into smaller hardware. Furthermore, hardware implementation is difficult to achieve for some algorithms. In this thesis, we therefore focus on object recognition algorithms suitable for hardware implementation. In order to solve these problems of conventional method, we proposed a reduction method of the *SIFT* feature points for object detection and a human-pose tracking method for body parts detection, and we achieved the following results, respectively.

First, the *SIFT* feature based object detection algorithm is described. While a *Haar-like* feature is a powerful method, it is not robust to changes in rotation or scale. The *SIFT* feature based method is used to overcome this problem because of its robustness to changes in size and rotation. However, in principle, the *SIFT* point based method requires large amount of memory and computational power. A *SIFT* feature produces a thousand feature points and requires 128 dimensional vectors per point. Moreover, the method requires a lot of computational resources during the matching process. Straightforward implementation in hardware is not practical. Therefore, we propose a reduction of the *SIFT* feature point method before performing a matching step. The evaluation results show that this method can reduce the amount of memory and processing without sacrificing accuracy. In addition, we propose an effective matching method for texture-less objects. For example, a complex objects produce rich feature points

indicates rich feature information. However, some objects in indoor scenes are texture-less. For example, home appliances tend to have square shape and little texture. The *SIFT* feature based method is difficult to apply to such cases. We therefore propose a new matching method using a pre-trained confidence table. The evaluation results by using 2432 samples show that the proposed method improved the recognition performance for texture-less objects.

Second, a human pose tracking method is described. Considering the applications of embedded object detection, human recognition is a very important task for such products. While this type of tracking seems to be a different area of research, home appliances and robots are used by human users, and human recognition is important for such products. In this thesis, we focus on a tracking problem of golf swing, which tracks the grip of a golf player from an uncontrolled golf swing video by using a monocular camera. Generally speaking, particle filter based methods are widely used for such tracking problem. However, tracking a part of a person in difficult uncontrolled and complex background is not easy. A novel method that combines both global and local features was proposed. We therefore combined human pose estimation, and tracking using a pictorial structure model (*PSM*) for tracking human pose motion. The evaluation results show that the combination of the global and local features outperformed the conventional method based on particle filter. When we compared an unconstrained conventional particle filter with the proposed method, the proposed method showed a better tracking performance for all samples compared to conventional particle filter.

Contents

1. Introduction

Vision is a basic sense in human beings, and we receive much of the information around us mainly through our sense of vision. Therefore, object recognition technology has the possibility to allow the realization of robot with a smarter sense of vision. Recognition technology has many applications in such field as robotics, manufacturing, and security. Robots and home appliances that can understand their circumstances through image information can be useful for human users. They can provide a lot of opportunities through their use of visual information, such as user-friendly human interface.

However, object recognition is not an easy task to achieve. There are many inherent problems such as changes in luminance, complex backgrounds, changes in the appearance of the target objects, and occlusion problems. Many object recognition systems find objects from an original image. Luminance changes affect the recognition performance because many such systems use pre-trained models. Complex backgrounds also pose difficult problems. Separating the background and foreground (target object image) is a difficult problem and is still considered as unresolved. Appearance changes are also difficult to deal with. The appearance of an object differs with its changes in rotation and scale. To recognize the changes appearance of an object, the detection algorithm should use different appearance models. This problem is still unresolved, as is the occlusion problem. An occlusion occurs when one object is hidden by another object pass between it and the camera. As a result, the appearance of the target object varies widely. While many researchers have been working to solve these problems, many difficult problems still remain.

Meanwhile, recent years have seen dramatic increases in computational processing power, allowing huge amounts of information to be treated using modern computers. The various methods of object detection proposed thus far have assumed the need for a significant amount of computational power. For example, some methods process different parts of an original image simultaneously. Such algorithms are easy to implement using the latest multi-core processors or GPU. However, these methods are still difficult to use in a real environment. Generic processors are designed for use in personal computers. They require a large power supply, complex peripheral circuitry and heat sink systems. In other words, they are impractical for implementation in small robots. For a practical use of object detection, we need to implement such algorithms into smaller hardware. For such occasions, we need to tackle such issues as hardware-oriented algorithms, a reduction of computational resources, and pipeline implementation. In this thesis, we focus on an object-recognition architecture that can be implemented in small hardware. We describe two conventional object-recognition methods in the following section.

1.1. Haar like based object detection method

In this section, we describe a sign recognition algorithm implemented in an FPGA [1]. This algorithm can be used for indoor miniature robots. The pipeline architecture can process video streams in real-time and reduce the working memory. Many papers have focused on the design and implementation of the hardware for real time object detection [2][3]. Nair *et al.* implemented an embedded system for human detection on an FPGA [4]. This system detects people at a speed of 2.5 frames per second. However, it requires a large size SDRAM memory. Cho *et al.* implemented a face detection system [5] in a Xilinx Vertex-5 FPGA using a scalable architecture. However, this system also uses a large amount of memory, requiring 41BRAMs (Block RAM) for 320x240 (QVGA) resolution images, which corresponds to 1.4Mbit of memory on a Virtex 5 device. Through the use of a pipelined architecture, however, the system does not require a large amount of frame-size memory.

An outstanding face detection algorithm proposed by Viola and Jones [6][7][8] is used as a base algorithm. Compared to a conventional algorithm, it can process images rapidly with high accuracy. An overview of the Viola-Jones face detector algorithm is described below. The detector process uses three key algorithms.

The first key algorithm is a feature extraction method called an *integral image*. This allows for very fast feature extraction. An integral image is similar a Haar-basis function. The value of a dual-rectangle feature is defined as the difference between the sums of the pixels in two adjacent regions. Viola and Jones paper presented a very fast integral image computation method with computational advantage over previous methods because, like most object detection systems, the detector scans the input at many different scales.

A second key algorithm is a learning algorithm based on *AdaBoost*, which is used to compute different features at many different times. In an image sub-window, the total number of features is extremely large. To ensure a fast classification, we do not need a large majority of the features, and can focus on only the important critical ones. Viola and Jones adopted a variant of *AdaBoost* to select the features used to train the classifier. The *AdaBoost* algorithm can boost the classification performance of a simple learning algorithm, and is a combination of weak classification functions for forming a stronger classifier. Here, a simple learning algorithm is called a weak learner. Such a learner is considered weak because we do not expect the best classification function for classifying the training data perfectly. For boosting a weak learner, the examples are re-weighted during the rounding step in the training process. The final strong classifier is a weighted combination of weak classifiers. The *AdaBoost* procedure can be easily interpreted as a greedy feature selection. It is an effective procedure for searching out a small number of good features from a large number of candidate features.

The third key algorithm has a cascade classifier structure. Such a structure increases the

processing speed of the detector by focusing on the important regions of an image. Boosted classifiers that reject many of the negative sub-windows can be constructed. A cascade structure is constructed through the training of the classifiers using *AdaBoost*. Starting with a strong two-feature classifier, an effective detector can be gained by tuning the strong classifier threshold to minimize false negatives. The initial *AdaBoost* threshold is calculated to minimize the low error rate of the training data. A lower threshold yields a higher detection and higher number of false positives. The cascade structure reflects the fact that, in any single image, an overwhelming majority of the sub-windows are negative. Negative images should be rejected at the earliest stage possible.

Haar-like features represent differences in the average intensities between adjacent rectangular regions, and are based on Haar basis functions. It can extract texture information without depending on the absolute intensities or color values. More specifically, adjunct regions have the same size and shape and are horizontally or vertically adjacent. They are not affected too much by noise because the haar-like feature is based on integral value.

Figure 1.1.1 shows a block diagram of the hardware design, implemented on the Altera Cyclone III FPGA [9] using SystemVerilog [10] and Verilog-HDL. A fixed point hardware implementation was developed based on floating point software. Since the software uses a floating point, testing was required to convert the software implementation to work with fixed-point operations.

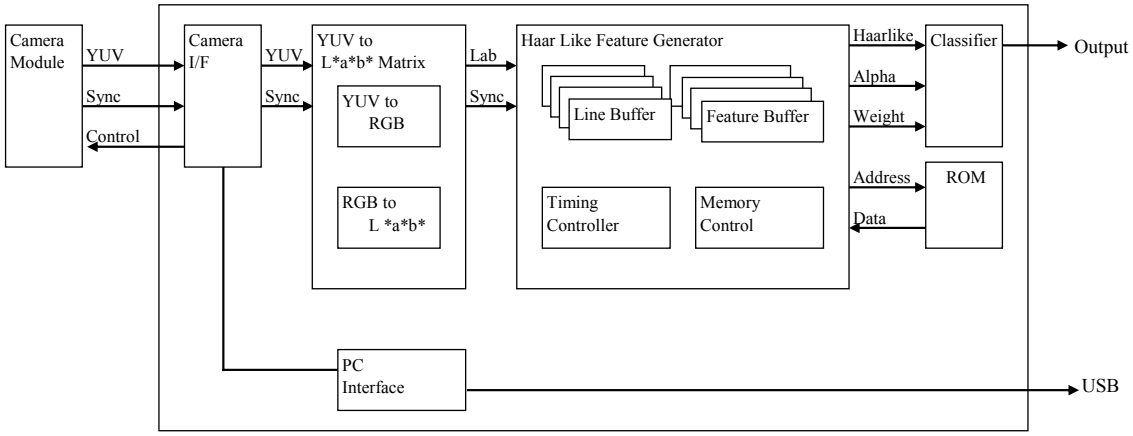
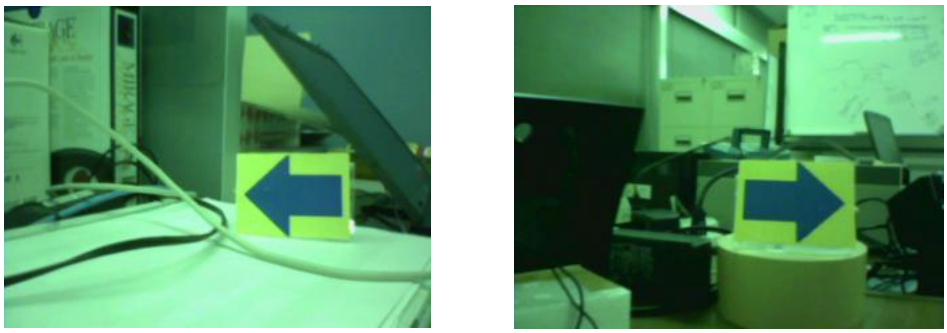


Fig. 1.1.1 Block diagram of a hardware implementation on a Haar-like based system.

Image data from the camera module are sent to the camera interface module, which converts the camera data from a serial form into a parallel form. Data from the camera interface then lead to a matrix and are converted into an L*a*b* color space. The converted data are then sent to the Haar-like feature generator and *AdaBoost* classifier. The parameters of the Haar-like features and trained *AdaBoost* classifiers are programmed in the ROM. The key feature of this design is a pipelined architecture. All modules in the design synchronize with video signal.

Therefore, a Haar-like feature calculator needs to address the control of the buffer memories. Figure 1.1.2 shows sample images taken from a camera.



(a) Left arrow „LT“ (b) Right arrow „RT“

Fig. 1.1.2 Sample images from onboard camera

Table 1.1 Hardware implementation details

Input data format	YUV 4:2:2 (640 x 480 pixels)
Processing time	< 1frame
Number of classifiers	256 (max)

Table 1.1 shows the hardware implementation details. The FPGA design is intended for practical use, and we therefore implemented the design suitable for a camera module of a mobile phone. Moreover, the design is pipelined and synchronized to vertical sync signal from a camera module. The MC5VC camera module provided by Konica Minolta for mobile phone applications is used in this work. The basic picture control function is processed inside the camera module. This module can connect directly to an FPGA device, without an interface circuit. The frame rate varies according to the optical luminance. The frame rate decreases in a dark environment. In contrast, in a bright environment, the frame rate increases. Many conventional systems uses frame buffer memory to synchronize the vertical sync timing. In contrast, the proposed system designed to synchronize with a vertical sync pulse without frame memory.

1.2. Human detection using HOG and random trees

A human detection method has many applications for surveillance, human interaction, and human behavior analyses such as a customer behavior analysis in a shopping centre. Dalal and Triggs recently proposed a method based on Histogram of Oriented Gradients (*HOG*) [11] features with *SVM* [12] for human detection, and achieved a high performance for the standing posture of a person. The appearance of a local object can be described based on a distribution of intensity gradients. Hou *et al.* proposed a multi-pose framework [13] with vector boosting,

which is a hierarchical tree of a detector cascade. Moreover, human detection in a cluttered background [17] and with occlusion [18] are also proposed. However, human detection is still a difficult problem because of the various changes in human appearance. Multi-pose detection can be considered a multi-class classification problem. For such problems, hierarchical-structure based approaches are widely used. A *Randomized Tree* is a tree structure method for multi-class recognition, which uses an ensemble of detection trees in which each decision tree outputs the likelihood of each class. The classification is based on a sum of the likelihoods for all classes from all decision trees.

Human detection algorithm based on a hierarchical tree structure [16] is introduced in this section. A *Randomized Tree* [14][15] is suitable for parallel processing because each decision tree is independent. In addition, it is robust to noise in the training samples.

As shown in Fig. 1.1.3, a random tree consists of multiple decision trees with branch nodes and terminating leaves. When recognizing individual classes, each leaf has a probability distribution for each class. Branching at each node is based on a split function.

The training procedure consists of three steps: creating subsets, generating nodes, and partitioning the created subsets. In this method, a subset is a randomly selected data-set of the training samples. Nodes are made based on a pre-defined split function. The node generation process is repeated until the number of training samples reaches a pre-determined depth of the tree, or when the training samples comprise only a single class. Classification is performed by computing the probability of each class based on the following steps.

The input image reaches a single leaf node in each decision tree. The probability distributions of each leaf nodes are then accumulated for each class, and the average probability of trees are computed. Finally, the class with the highest average probability is computed for the classifier output.

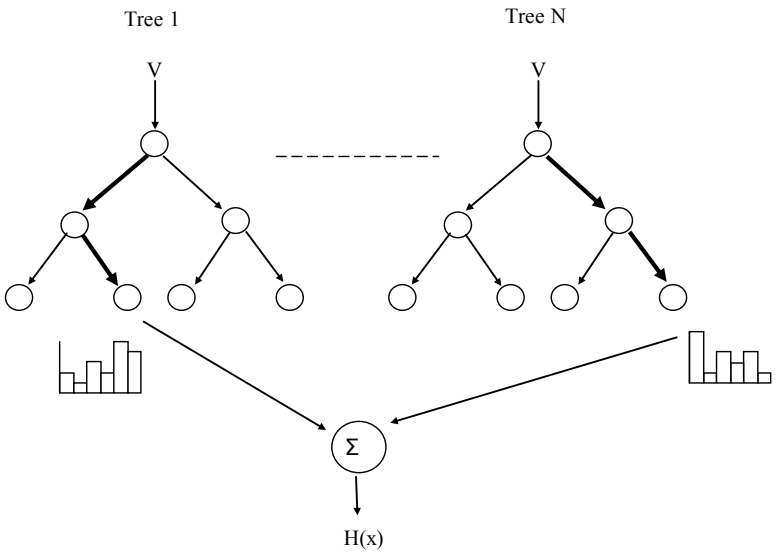


Fig. 1.1.3 Random tree architecture

1.3. Thesis Organization

In this thesis, object detection methods which suits for hardware-implementation is described. To achieve an object detection system for smaller hardware, low computational cost is major issue. We begin with *SIFT* feature based object recognition method. There are three major components.

First, a hardware oriented S*IFT* feature based object detection algorithm is described. *SIFT* point based object detection methods are widely used. However, they require a high computational cost. We propose a reduction of the *SIFT* feature points.

Second, a matching method for texture-less objects is proposed. For texture-less objects, the matching method is difficult to adopt. To deal with this subject, we propose a confidence based matching method.

Finally, a human-pose tracking method is described. Human body parts are important objects for user friendly applications such as robots and home appliances.

In the final part of this introduction, a brief tour of the contents is described. The rest of this thesis is organized as follows.

In chapter 2, we introduce feature-point based object recognition. We then describe two methods for improving the object recognition performance. In chapter 3, we detail the human-pose tracking problem. Finally, some concluding remarks are provided in chapter 4.

2. Feature−point based object recognition

2.1. Point-based matching

In computer vision, a window-scan based approach is widely used. This method computes features inside the image of the scanning window to features and compares the similarity with a trained model. The entire image inside the scanning window is used for the computation. However, this is not suitable for embedded systems because of the computational costs incurred.

In this section, we focus on a feature-point based object recognition method. In this method, we extract the feature points from the original image. We compute the features from the appearance information around the feature points. The detection process is conducted by matching the features with those of the trained models. If many similar feature pairs exist, the object detector is fired. This method is robust to occlusions because not all feature points are needed for matching. Moreover, the method is faster than a scanning window based method because there is no need for scanning.

One drawback of this method, however, is its weakness on a cluttered background because it does not include a pre-process for distinguishing background. In addition, it is difficult to classify texture-less objects because they produce low numbers of feature points. We can adopt many kinds of features. *SIFT* was an original proposal and is very popular owing to its robustness to rotation and scale invariants. However, *SIFT* features requires high computational costs. To deal with this, many features such as *SURF*, and *PCA-SIFT* have been proposed.

In this section, we first briefly describe about *SIFT* features and a basic matching method. Two different computationally reduced point-based matching methods are also proposed: a reduction method for the number of *SIFT* feature [19], and a cost effective computational matching method [20].

2.2. SIFT feature reduction method

2.2.1. Introduction

This section focuses on obtaining an efficient feature recognition method. Specifically, a particular feature-point matching method using the descriptors for a limited amount of computational resources is considered. Research on robust local descriptors continues to be an active area of computer vision. A *scale invariant feature transform* (*SIFT*) [21] is probably the most commonly used among the evaluated descriptors and has been proven to be the most discriminative. *SIFT* relies on extracting the scale-invariant keypoints using a Difference of Gaussian (DoG) operator. The descriptor part is based on computing the magnitude and

orientation of the gradient images, which is based on a histogram of the gradient orientation computed for the surrounding regions centered on the extracted keypoints.

However, the high dimensionality of the *SIFT* descriptor is a significant drawback, especially for online or large-scale dataset applications. For example, for a typical outdoor scene, *SIFT* usually produces several-hundred local features, which yield a large, high-dimensional feature space that needs to be searched, indexed, and matched.

Several researchers have addressed the problem of dimensionality reduction for feature descriptors. For example, Herbert Bay *et al.* proposed an approach called *SURF* [22] that combines a Hessian matrix-based measure for the detector and Haar-wavelet responses for the descriptor, resulting in a 64-dimension feature representation. *PCA-SIFT* [23], proposed by Yan Ke *et al.*, reduces the dimensionality of the descriptor to 36, while obtaining a performance equal to the original *SIFT*. The key to *PCA-SIFT* is to apply the standard principal component analysis (*PCA*) technique to the gradient patches extracted around the local features, thus yielding a compact feature representation. However, *PCA-SIFT* requires an offline stage to train and estimate the covariance matrix used for a *PCA* projection, which typically requires the system to collect and train from a large, diverse collection of images prior to use.

Another approach for a computational reduction of the feature descriptors is data compression. Vijay Chandrasekhar *et al.* proposed a low-bit rate descriptor called Compressed Histogram of Gradients (*CHOG*) [24]. The main idea behind *CHOG* is the representation of the gradient histogram as a tree structure that can be efficiently compressed. The method can reduce the bit rate of the descriptors using the Huffman and Gagie trees. The matching process of *CHOG* is based on a compressed domain, and thus requires a pre-computed distance look-up table. Nitser *et al.* proposed a recognition scheme for a large number of objects called a Scalable Vocabulary Tree (*SVT*)) [25]. The main basis for an *SVT* is a hierarchical k-mean clustering of the sample feature descriptors. An *SVT* can treat a large number of objects. However, the data size of *SVT* tends to be quite large.

Chandrasekhar *et al.* also proposed a lossy, transform-based feature compression approach called Transform Coding [26], using a codec consisting of a Karhunen-Loeve Transform(*KLT*), scalar quantization, and entropy coding, and can efficiently compress the descriptors.

2.2.2. Matching method

This section describes the process used in an image matching method. The main idea is to reduce similar feature points. The image matching process occurs in three stages:

First, the descriptors are computed using the *SIFT* method, which is described later.

Second, similar feature points are removed. This process reduces the number of features as well as computational costs.

Third, matching between the reduced feature points and database feature point is performed

using *approximate kd-tree (ANN)* method [27]. The comparison process reduces the number of matching feature pairs to a minimum.

2.2.3. SIFT overview

The *SIFT* descriptor proposed by Mikolajczyk is widely used for many problems including object recognition, image matching, or stereo correspondence. The features are invariant to image scaling and rotation, and partially invariant to changes in illumination. Large numbers of features can be extracted from the algorithm. Moreover, the features are highly distinctive, which allows a single feature to be matched with a large feature database.

The key stages of computations used to generate *SIFT* features are as follows:

1. Scale-space extrema detection

   At the first step of the computation searches interest points at all scales and image locations are searched by using a DoG function.

2. Keypoint localization:

   Keypoints are selected based on the stability criterion measures at each candidate of interest point candidate.

3. Orientation assignment:

   One or more orientations are assigned to each keypoint by computing the local image gradient directions. Future operations are performed for robustness with the changes in orientation, scale, and location for each feature.

4. Keypoint descriptor:

   The local image gradients are measured at the selected scale in the region around each keypoint. The *SIFT* keypoint descriptor is based on the histogram of gradients.

2.2.4. SIFT details

2.2.5. Scale-space extrema detection

The first stage of keypoint detection is to identify the locations and scales that can be assigned under different views of the same object. To efficiently detect stable keypoint locations, the DoG function is used. This function can be computed from the difference of two nearby scales separated by a constant multiplicative factor, k:

$$D(x,y) = (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \\ = L(x,y,k\sigma) - L(x,y,\sigma) \tag{2.1}$$

Here, $L(x,y,\sigma)$ is the scale space of an image computed from the convolution of a variable-scale Gaussian, as shown in Fig. 2.2.1.

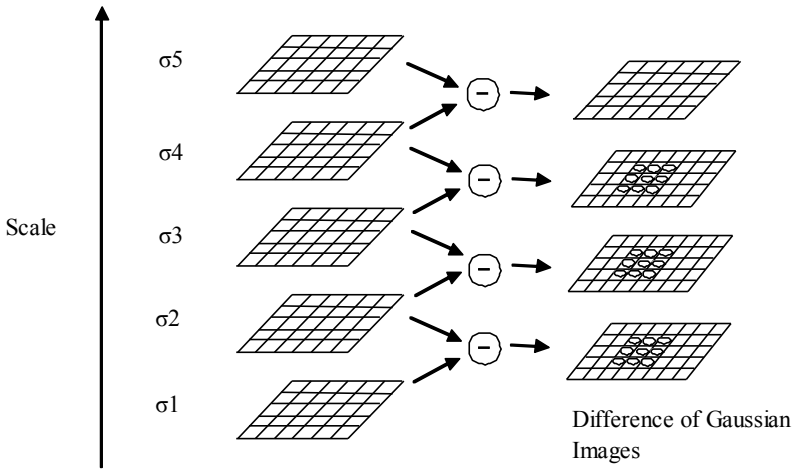


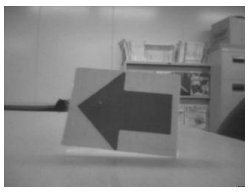
Figure 2.2.1 Difference of Gaussian images

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y), \tag{2.2}$$

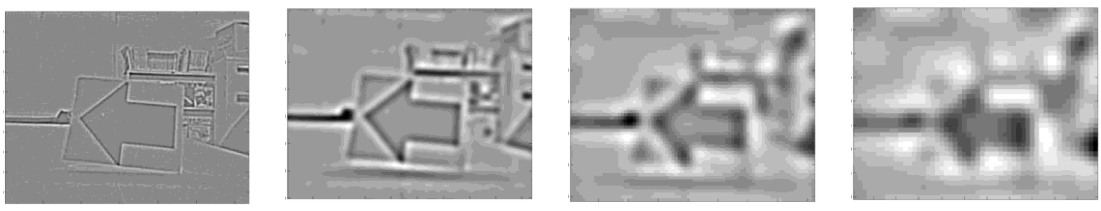
where * is the convolution in x and y, $I(x,y)$ is an input image, and

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad . \tag{2.3}$$

Figure 2.2.2 shows Gaussian-filtered image samples.



(a) Original Image



σ_1 σ_2 σ_3 σ_4

(b)Gaussian Filtered Image

Figure 2.2.2 Gaussian-filtered images

2.2.6. Local extrema detection

Local extrema tend to produce many points in a local area. In *SIFT* algorithm, they produce local extreme from comparing its neighbors. To detect the local maxima and minima, each sample point is compared with its eight neighbors in the current image, and nine neighbors in the above and below scale, as shown in Fig. 2.2.3. The keypoint is selected only if it is larger than each of these neighbors. The computational cost of this process is reasonably low because most of the simple points are eliminated.

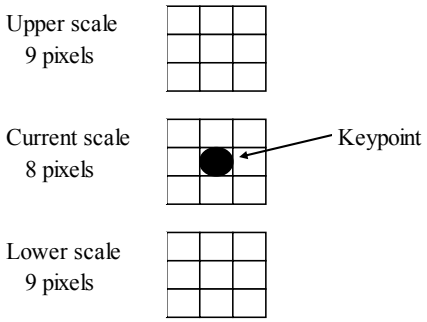


Figure 2.2.3 Local extrema search sample

2.2.7. Orientation assignment

The keypoint scale is used to select the Gaussian-smoothed image L, with the closest scale. All computations are therefore performed in a scale-invariant manner. In addition, to achieve invariance to the image rotation, the keypoint descriptor can be represented relative to the change in orientation.

For each image, $L(x,y)$, at this scale, the gradient magnitude, $m(x,y)$, and orientation $\theta(x,y)$, are computed by using the pixel differences:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{2.4}$$

$$\vartheta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y))) \tag{2.5}$$

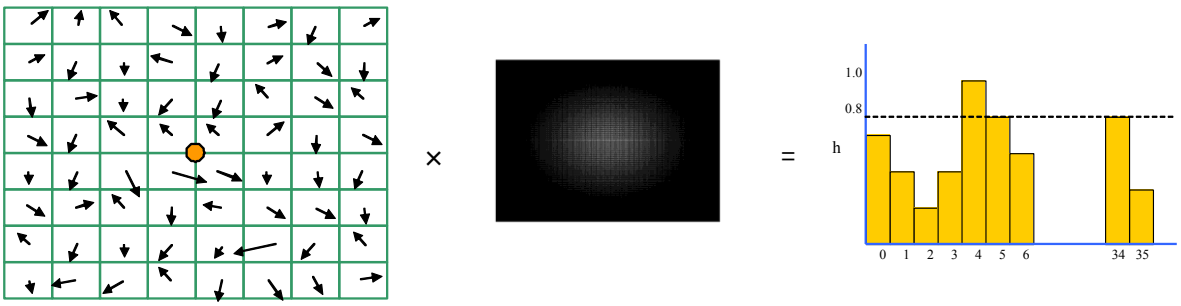
An orientation histogram is computed from the gradient orientations of the sample image patch within a region around the keypoint. The orientation histogram has 36 bins covering the 360° range of orientations. Each sample added to the histogram is weighted using a Gaussian window with an σ value 1.5-times that of the keypoint scale.

Peaks in the orientation histogram correspond to the local directions of the local gradients. The highest peak in the histogram is detected, and any other local peak over 80% of the highest peak is also

used to create the keypoint.

2.2.8. Keypoint descriptor

Figure 2.2.4 shows the process flow used to compute the descriptors. Image patches detected around the keypoints are first divided into different 4-by-4 square grid configuration cells. The *SIFT* descriptor is calculated as a function of the gradient histograms, provided that such histograms are available for each cell and that the d_x, d_y values are sorted into sufficiently fine bins. Let $P_{Dx,Dy}(d_x, d_y)$ be the normalized joint (x,y)-gradient histogram for each cell. The gradient within a cell may be weighted using a Gaussian window prior to the computation of the descriptor. The Gaussian window is applied for filtering, which reduces the effect of the cells that far from the keypoint.



Gird configuration cells　　Gaussian window　　Orientation histogram

Figure 2.2.4 Orientation histogram calculation procedure for SIFT descriptor

The *SIFT* components of a cell are

$$I(i) = \sum_{(dx,dy)\in\Omega_i} \sqrt{dx^2 + dy^2}\, P_{DxDy}(dx, dy) \tag{2.6}$$

$$\Omega_i = \left\{ (dx, dy) \,|\, \frac{(i-1)\pi}{4} \le \tan^{-1}\frac{dy}{dx} \le \frac{i\pi}{4}, i = 1 \ldots 8 \right\}. \tag{2.7}$$

The final *SIFT* descriptor is a 4-by-4 array of the histograms, each with eight orientation bins, that captures the rough shape of the oriented image. In addition, the keypoint descriptor is robust for changes in rotation and scale, and illumination.

2.2.9. Feature reduction

This section describes the details of the proposed feature reduction method. Figure 2.2.5 shows an overview of the feature reduction method. The main idea is to reduce the number of features that are near the feature value. The matching method includes the BBF, which is used to

compare feature descriptor and create feature pair. The *SIFT* descriptors usually produce several-hundred local features. However, some of them have a similar value, which implies that they do not contribute to the final matching result. Thus, it is reasonable to remove similar feature points in advance.

The *Histogram Intersection Kernel*[28] measures the degree of similarity between two histograms. Since *SIFT* descriptors consist of 4-by-4-by-8 histograms, they can be considered a group of histograms.

In addition, the *Histogram Intersection Kernel* is practical for low computation implementation and it can be defined as follows:

$$K(A,B) = \frac{1}{D}\sum_{i=1}^{m} \min\{a_i, b_i\} \quad , \tag{2.8}$$

where A and B denote the histograms, and D is the total number of bins. It is assumed that both histograms consist of m bins such that the i-th bin for (i = 1, ..., m) is denoted as a_i and b_i for the histograms A and B, respectively.

If descriptor pairs have histogram intersection larger than a given threshold, they can be considered as be significantly similar. In this case, one of the descriptor pairs is removed.

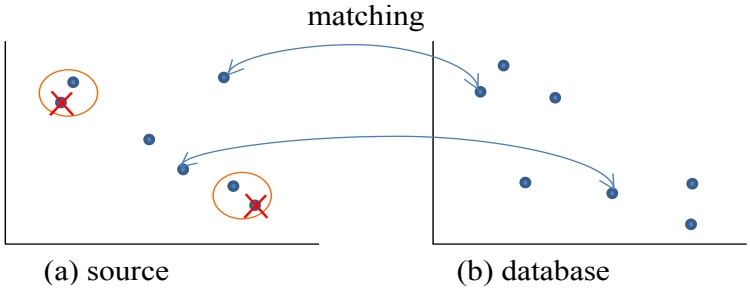


Figure 2.2.5 Overview of feature reduction comparing each source descriptor and removing similar descriptors in the source feature data.

To adopt a method for measuring the similarity that presents the lowest computational cost, the performance and processing time of the *Histogram Intersection Kernel*[28] are examined and compared to those of the *KL distance*[29] and the *Bhattacharyya* distance[30]. The receiver operating characteristic (ROC) curves[31] as an indicator of accuracy are shown in Fig. 2.2.6. All of these methods provide similar characteristics.

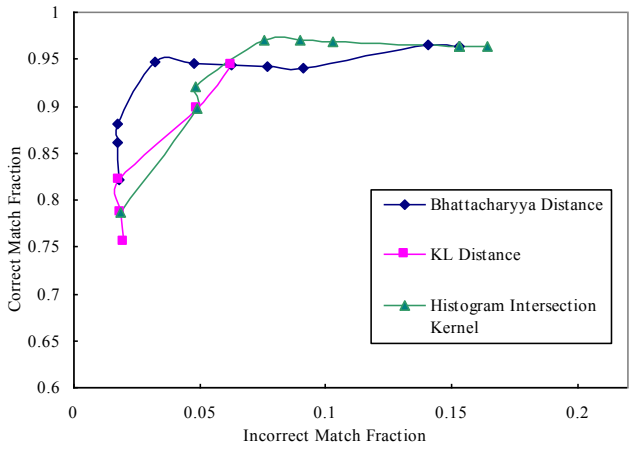


Figure 2.2.6 ROC curves for various distance measurement methods.

In addition to accuracy, the runtime of the reduction process is also important. We evaluated the process runtime. The runtime of the reduction process as a computational cost versus the number of features is shown in Fig. 2.2.7. The evaluation was conducted using a 1.4GHz Intel Core 2 Duo computer on a Linux system. The runtime of each method is almost linear in terms of the number of feature points, and the *Histogram Intersection Kernel* provides a faster processing time. *KL distance* and *Bhattahcarrya* distance require almost the same amount of processing time. Thus, the *histogram intersection kernel* is selected as the similarity criterion.

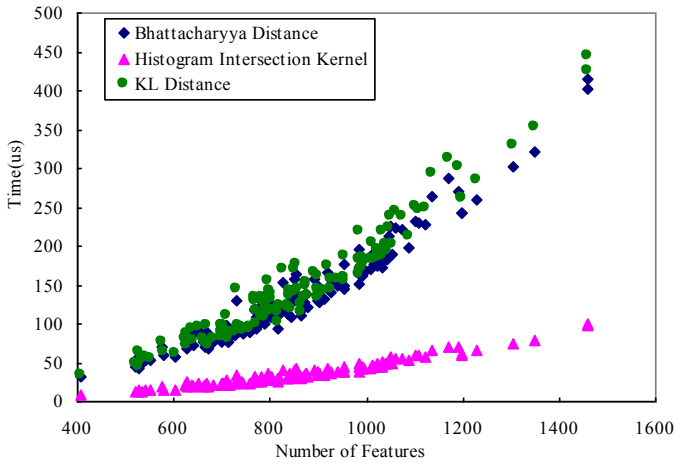


Figure 2.2.7 Comparison of the processing times

2.2.10. Experimental results

We compared the performances at the different distances described in the previous section, as used for the matching of the *SIFT* descriptors. The method proposed by Winder and Brown [32] was used for the evaluation.

Figure 2.2.8 shows sample images of the Trevi Fountain dataset [32], which can be used for

evaluating the descriptor performance for accurate matching and non-matching information. This dataset consists of 64-by-64 grayscale pixel image patches and matching information.

A matching algorithm calculates the distances between descriptors. Since a descriptor of image A should have at most one correct match in image A', the simplest criterion is selected to match the descriptors. That is, if the distance between a of A and a' of A' is smaller than threshold τ, a source descriptor a is counted as matched with its nearest neighbor a'. Next, if the number of matched descriptors is larger than a predefined value, images A and A' are classified into the same class.

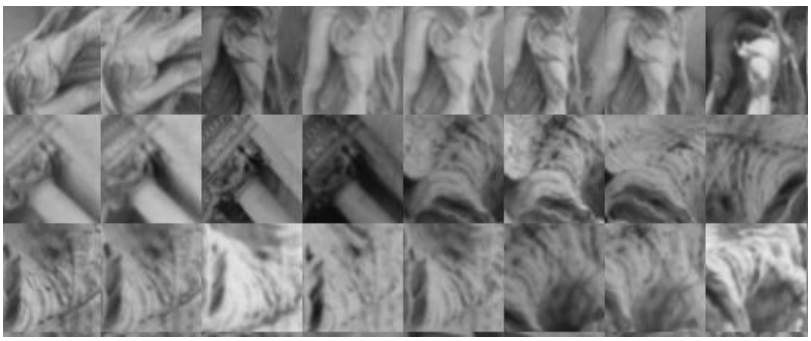


Figure 2.2.8 Sample images of the Trevi Fountain dataset.

The performances of the different distances are compared on a set of 800 images. Each image A of the database is compared to image A'.

The matching is processed with a spatial tolerance of the positional relation of A and A'. As a result, incorrect matching may occur. An incorrect match is counted as a false positive, and a correct match is counted as a true positive.

Table.2.1 True positive and false positive rates.

		True Class	
		p	n
Hypothesized Class	P	True Positives	False Positives
	N	False Negatives	True Negatives

True positive rate = True Positives / (True Positives + False Negatives)

False positive rate = False Positives / (False Positives + True Negatives)

For the evaluation, we use the ROC curves shown in Table 2.1. The true and false positive rates are calculated according to this table. The label of the dataset determines the hypothesized class of P and N. The true class of p and n is the matching result. Figure 2.2.9

shows the ROC curves[31] of the original *SIFT* and reduced *SIFT,* whose threshold τ is 18 for the Trevi Fountain dataset. Table 2.2 shows the results of the reduction. The data size can be reduced to 91.7%. The performance of the reduced data matches that of the original in this case.

Here, another dataset, ZuBuD dataset [33] was used for further evaluation. The dataset contains Zürich buildings taken from different positions and matching information. Figure 2.2.10 shows some sample images of the dataset.

Figure 2.2.11 shows the ROC curve results for the ZuBuD database. Table 2.3 shows that the data size can be reduced. The reduction of the features varies between 60 and 80% depending on the threshold. Therefore, the results show that the value of the reduction threshold value slightly affects the performance of the descriptor. Furthermore, the ROC performance of the proposed method with *SIFT* and a reduction in the number of features was compared to that of the original SURF. Figure 2.2.10 shows that the proposed method performs as well as SURF.

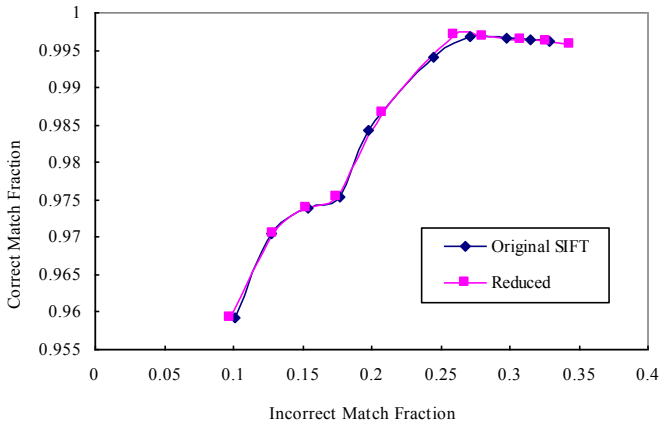


Figure 2.2.9 ROC curves of the original SIFT and reduced SIFT (Th = 18).

Table 2.2
Data reduction ratio using the Trevi fountain database
(Original data size: 155,433 bytes).

Reduced features (bytes)	Ratio (%)
142,476	91.7

Figure 2.2.10 Sample images of the ZuBuD dataset

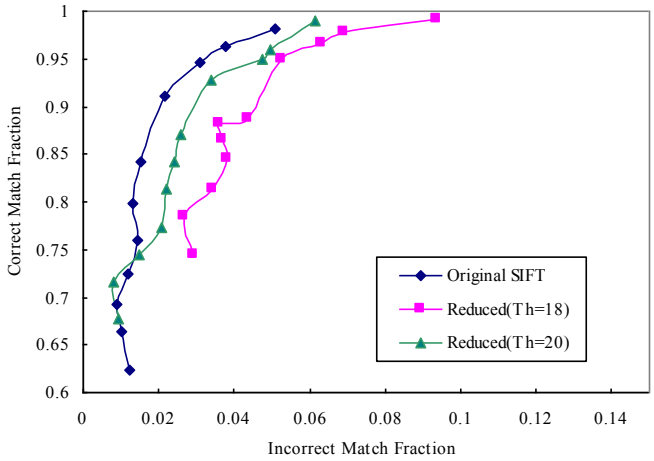


Figure 2.2.11 ROC curves of reduced SIFT (ZuBud database).

Table 2.3 Data reduction ratio using the ZuBuD database

(Original data size: 644,937 bytes).

Threshold	Reduced features (bytes)	Ratio (%)
18	394,787	61.2
20	494,288	76.6

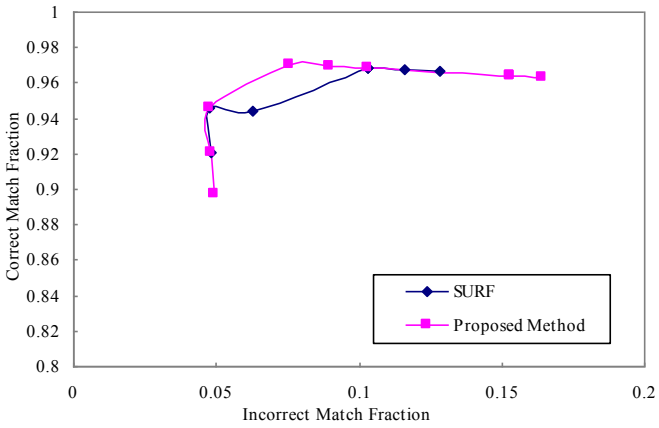


Figure 2.2.12 Comparison between SURF and proposed method.

2.2.11. Conclusion of SIFT feature-reduction method

In this section, a new feature reduction method for *SIFT* feature-point based object recognition is described. *SIFT* feature-matching based object recognition is widely used. However, the high dimensionality of the *SIFT* descriptor is a significant drawback, and the matching process of a *SIFT* descriptor requires high computational costs. To overcome this problem, the *SIFT* feature-reduction method is proposed. The conclusions of this section are outlined below.

The SIFT feature-point reduction method is proposed. A conventional point-based matching method requires high computational costs and is difficult to implement on a small hardware system. In the proposed method, to reduce the computational costs, the number of the future point is reduced before matching. We search the query feature space and remove similar *SIFT* descriptors. The evaluation results show that using the Zubud dataset, a data size reduction of 9% can be achieved without sacrificing the recognition accuracy.

The proposed method uses the distance function for measuring the *SIFT* feature distance. We compared three distance functions. The evaluation shows that *Histogram Intersection Kernel* is the fastest method compared to *Bhattacharrya* distance and KL distance.

The proposed method is suitable for processing because it is independent of each feature descriptor. Implementing the proposed method using multi-threaded programming, or implementing it into hardware will result in a further improvement of the processing time.

2.3. SIFT feature-matching based on confidence LUT

2.3.1. Introduction

The ability to recognize objects from a single image has many possible applications. Home appliances with an intelligent vision system can be useful applications for users.

For example, a remote control system equipped with an image recognition system is such user friendly application. A small monocular camera module and an image recognition algorithm are embedded into a remote control system to fulfill the primary purpose of an image recognition system, which is obtaining information on the target object in front of the camera. In general, remote control devices are dedicated to particular systems. For example, a general television remote control can be used for only a specified television. However, if a universal remote control unit is used, users can access numerous home devices using a single remote control unit. In most cases, however, the controller's target device must be selected in advance. In contrast, our proposed remote control system, which is equipped with an image recognition function, can automatically recognize which device a user wants to control. Since most users spontaneously and unconsciously point remote controllers at the devices they wish to operate, this system can provide a more user-friendly interface. In addition, conventional remote control systems require multiple selection buttons to operate multiple devices. The system does not need multiple selection buttons and can handle numerous devices without physical limitations.

Adapting an image recognition algorithm to a remote control unit is an extremely challenging task. In general, the computational resources of remote control devices are strictly limited. However, the application requires a much higher standard of recognition performance.

Point based object-recognition [34][35][36][37] based methods have been widely proposed. Figure 2.2.1 shows a general framework for such methods. The general framework for object recognition follows three steps: generating the feature points, matching the points to model, and estimating the positions using information associated with the matched model features. *SIFT* features [21] are commonly used to represent image features.

Gordon and Lowe [34] proposed a method for accurate camera tracking that uses trained scene models and *SIFT* features. Collet *et al.* [42] extended Gordon and Lowe's method to improve the efficiency of recognizing multiple instances of an object. They used *RANSAC* [43] and *Mean Shift* [44] clustering to simulate object instances. Hsiao *et al.* presented a 3D recognition framework [36] that retains the ambiguity in the feature matching. Their method utilizes an affine transformation and hierarchical matching for improved performance.

However, these point-based approaches are difficult to adapt to the home appliance recognition problem. Figure 2.12 shows an example of the *SIFT* features of various objects, including home electrical devices and coffee cans. An analysis of a TV set, DVD player, and air conditioner

shows that these devices produce a small number of *SIFT* feature points when compared to image of coffee and soymilk cans. Electrical home appliances tend to have simple external shapes. That is, there are few *SIFT* features available in these images to recognize, and it is therefore difficult to reach a preset threshold for proper identification.

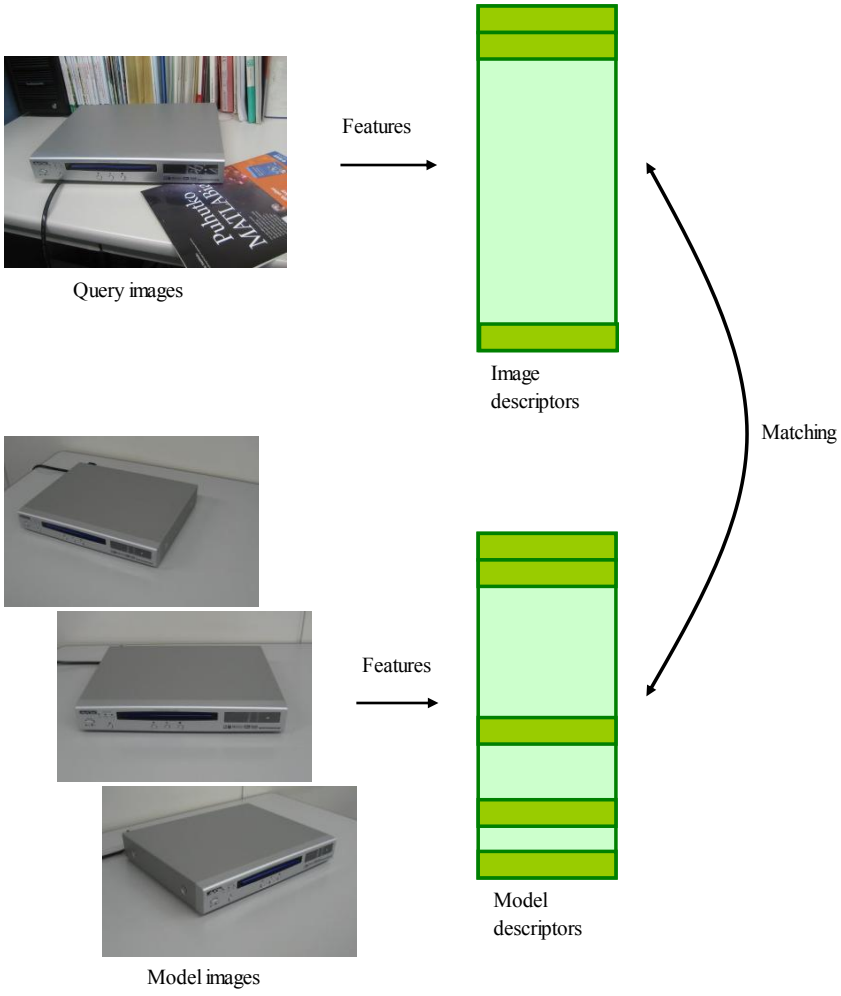


Figure 2.3.1 SIFT feature-point based recognition framework. Features are extracted from a target image and matched to a set of model descriptors. Model descriptors are trained using numerous different viewed model images.

2.3.2. Algorithm

The proposed matching method framework is based on the general point-based object recognition systems shown in Fig. 2.3.1. Features are extracted from an input image and then matched to the model descriptors. For a point-based image-recognition task, one common technique is the use of hierarchical matching using vocabulary trees [45], in which a visual word is assigned to each image feature.

The proposed algorithm is based on Lowe's *BBF* [21], which compares the distance between one point and its closest neighbor with the distance to its second closest neighbor. Figure 2.3.3 shows some sample matches. However, this method is particularly difficult to apply to objects with a small number of *SIFT* features. Figure 2.3.4 shows the result of *SIFT* feature extraction for different objects. General home appliances with simple external shapes normally produce low feature counts. The proposed matching method uses confidence-based criterion whereby the confidence is computed in advance through the use of a labeled training dataset. This method was inspired by the *Real AdaBoost* algorithm[46][47], but is quite different in two ways. First, the proposed method does not use Haar-like features, and second, it does not apply to weak learners.

Figure 2.3.2 *SIFT* feature detector results for (a) a soymilk can, (b) coffee can and home appliance,(c) TV set, (d) DVD player, and (e) an air conditioner with simple shape producing a small number of SIFT feature counts compared to the coffee can.

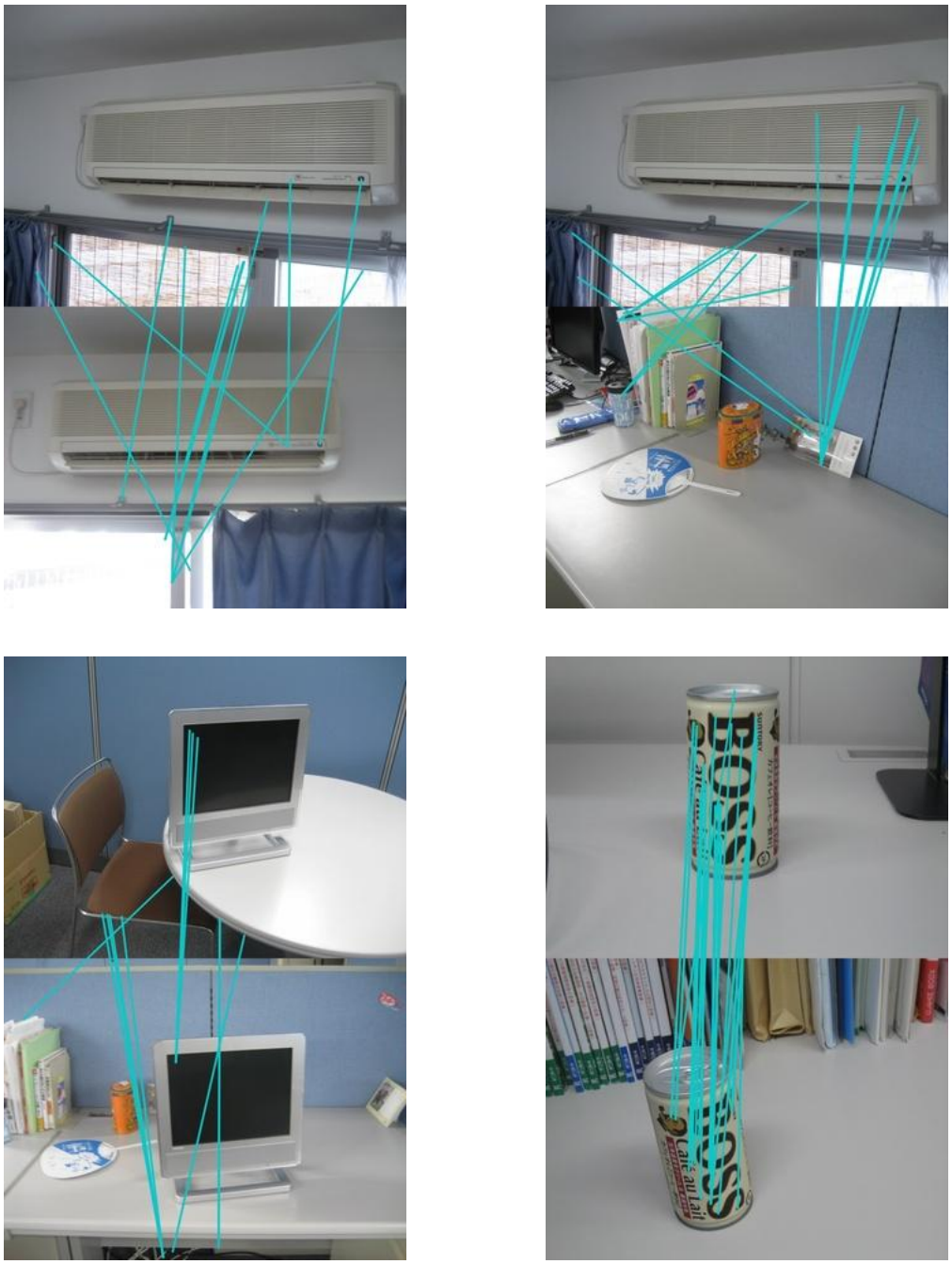


Figure 2.3.3 *SIFT* feature matching samples.

Matching results when making pairs with similar *SIFT* feature points. The method outperforms other good results for objects producing many *SIFT* features such as a coffee can (right bottom). However, it has a poor performance for many types of home appliances with few feature points.

2.3.3. Best bin first

The Best Bin First (*BBF*)[48] method proposed by Beis *et al.* is constructed around an approximate search technique that locates the nearest neighbor for a large fraction of queries, and a very close neighbor in the remaining cases. The BBF search algorithm is a modified version of a *k-d tree*[49], and can be described as follows.

It begins with a dataset of N points. The data space is then split on the dimension i', where the data exhibit the greatest variance. A division is then made at the median value of m of the data in this dimension, and thus an equal number of points fall to one side or the other.

An internal node is then created to store i and m, and the process repeats with both data nodes. This process creates a binary tree with a depth of $d=log2N$.

To search the nearest neighbor tree to query point q, a backtracking, branch, and bound search algorithm is used. The tree is first searched to locate the bin that contains the query point. This requires only d scalar comparisons, and in general, the located point from the bin provides a good approximation to the nearest neighbor. In the backtracking stage, whole branches of the binary tree can be pruned if the region of space that they represent is further from the query point than the distance between q and its nearest neighbor. The search process terminates when all unsearched branches have been pruned.

This search process is very effective in low dimensional spaces, but in higher dimensions, there are many more bins to explore, and the performance rapidly degrades. However, extra prolonged extra search efforts can be avoided by limiting the number of leaf nodes.

2.3.4. Affine SIFT

Affine SIFT [50][51] simulates all distortions caused by variations to the camera's optical axis direction, after which the *SIFT* method is applied. An affine transformation A can be decomposed as

$$A = \lambda R(\psi)\begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} R(\phi) \tag{2.9}$$

using singular value decomposition (SVD), where R(ψ), and R(ϕ) are rotation matrices, λ>0 and $t\geq 1$. λ corresponds to a camera zoom and R(ψ) corresponds to a planar rotation. Since *SIFT* features are both scale and rotation invariant, we can ignore both the zoom and planar rotation parameters. The remaining parameters in the decomposition correspond to the camera viewpoint. More specifically, *Affine SIFT* simulates three parameters: scale, longitude and the latitude angles of the camera.

The *Affine SIFT* procedure is described as follows:

1. An image is transformed by simulating all affine distortions caused by changes in the camera optical axis orientation from a frontal position. The longitude φ and latitude θ are

parameters that affect the distortions.

2. An anti-aliasing filter is applied in the direction of *x*, specifically, convolution using a Gaussian with a standard deviation of $c\sqrt{t^2-1}$. A value of c=0.8 was chosen by Lowe for the *SIFT* method.
3. The filtered image is sub-sampled to simulate distortion.

   Tilt parameter t is denoted as $t=\left|\frac{1}{\cos\theta}\right|$.

   Images that undergo φ rotations followed by a tilt of t can be simulated by using the operation $u(x,y)\rightarrow u'(tx,y)$.
4. These rotations and tilts are performed for a small number of latitude and longitude angles.
5. The affine simulated images are compared using *SIFT* as the similarity invariant matching algorithm.

Fig. 4.3.4 shows sample affine-transformed images.

Figure 2.3.4 Affine SIFT transformed sample images

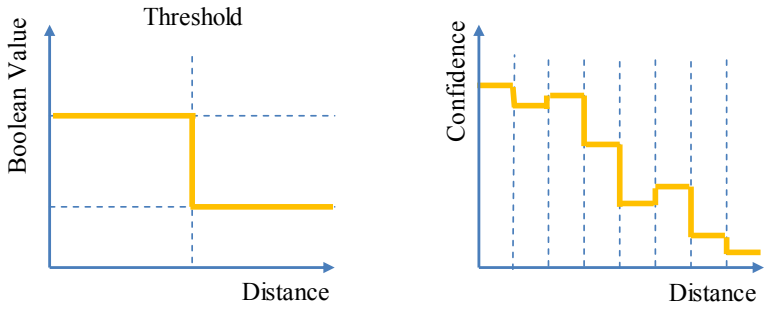
2.3.5. Confidence-based matching

This section shows details of the confidence-based matching method. First, the BBF algorithm is used to search for the nearest descriptor. Equation 2.10 shows the conventional Lowe's decision criterion.

$$d_o > d_1 \cdot T_{RATIO} \tag{2.10}$$

Here d_0 denotes the Euclidean distance from a query feature to its closest neighbor. d_1 denotes the distance to the second closest neighbor, and T_{RATIO} is the threshold value, which was chosen as 0.49 by Lowe. The decision making process is constructed for all detected *SIFT* feature points, after which the total matched point number is compared to the threshold value. If the matched point number is higher than the threshold, the query image is classified as matched. However, as shown Fig. 4.3.2, general usage home appliances that have simple external shapes normally produce low feature counts. This makes it difficult to set a threshold

value.



(a)binary decision (b)LUT based decision

Figure 2.3.5 Binary decision and LUT based decision

2.3.6. Details of the proposed training method

In general, simple-shaped objects produce significantly fewer *SIFT* feature counts. As shown in Fig. 2.2.1, soy milk and coffee cans produce many *SIFT* feature points compared to TV sets. Home appliances normally have simple shapes, and thus result in very low *SIFT* feature counts. Minor decision errors will affect the final decision. In our proposed method, the decision-making criterion is replaced with a confidence-based look-up table (LUT), as shown in Fig. 2.3.5.

The proposed method uses confidence-based decision-making where the Euclidean distance between the query feature to its nearest neighbor and the confidence value are closely related. The primary idea is to use a confidence LUT instead of binary decision-making.

An LUT is trained using labeled training data. In this method, the LUT should be appropriately trained to ensure it gains an efficient performance. A confidence LUT is trained as follows:

First, a labeled image pair is defined. Positive image samples denote images that contain recognition target objects, while negative image samples denote images that do not.

We let $d(x)$ denote the Euclidean distance between the query feature and the nearest neighbor using the BBF method.

$$bin_j = \left[(j-1)d_{\max}/n, jd_{\max}/n\right),\ j = 1,\ldots,n \tag{2.11}$$

d_{max} denotes the threshold distance.

$$W_l^j = P(w_i \in X_j, y_i = l) = \sum_{i:x_i \in X_j \wedge y_i = l} D_t(i) \tag{2.12}$$

If $d(x) \in bin_j$, then $h(x) = \frac{1}{2} \ln\left(\frac{\overline{W}_{+1}^{j} + \varepsilon}{\overline{W}_{-1}^{j} + \varepsilon} \right)$,

(2.13)

where

$$\overline{W}_{l}^{j} = P(d(x) \in bin_j, y = l), l = \pm 1, j = 1,...,n \quad . \tag{2.14}$$

Here, $W+$ and $W-$ denote histograms for a positive and negative sample, respectively, and ε is a constant.

The characteristic function is given by

$$B_n^j(u) = \begin{bmatrix} 1 & u \in [j-1/n, j/n) \\ 0 & u \notin [j-1/n, j/n) \end{bmatrix}, j = 1,...,n \tag{2.15}$$

The LUT size is set to 64 experimentally. Equation (2.11) shows the confidence LUT table as:

$$h_{LUT}(x) = \frac{1}{2} \ln\left(\frac{\overline{W}_{+1}^{j} + \varepsilon}{\overline{W}_{-1}^{j} + \varepsilon} \right) B_n^j(d(x)) \tag{2.16}$$

The final value is either1 or 0, specifying matching or non-matching, respectively. This value is calculated using Equation (2.12).

$$s(x) = \begin{cases} 1 & \sum h_{LUT}(x) > 0 \\ 0 & otherwise \end{cases} \tag{2.17}$$

2.3.7. Experimental results

We performed a number of experiments using the home appliance images shown in Fig. 2.3.6: a TV set, two types of DVD players, an air conditioner, and a coffee can. The images used for the experiments are 640 pixels × 480 pixels in size. Table 4.3 shows the details of the sample image data used for the evaluation.

The sample data were randomly separated into either training data or test dataset. Figure 2.3.7 shows two images of a trained LUT of a DVD player. The normalized Euclidean distance of the feature point corresponds to the LUT input. The threshold distance, *dmax* is 120000 experimentally.

Figure 2.3.8 shows the performance comparison results using ROC curves. The results show that the LUT method can improve the true positive rates for the TV, DVD-1, DVD-2 and air conditioner. For the coffee can images, the proposed method was not fully effective.

(a) TV

(b) DVD player 1

(c) DVD player 2

(d) Air conditioner

(e) Coffee can

Figure 2.3.6 Sample objects used for the evaluation

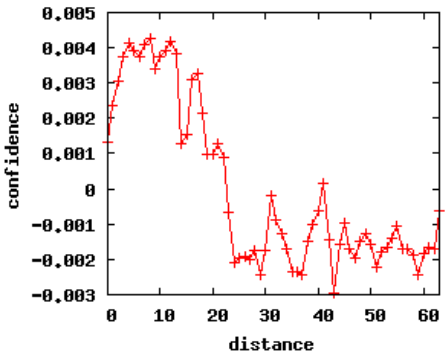
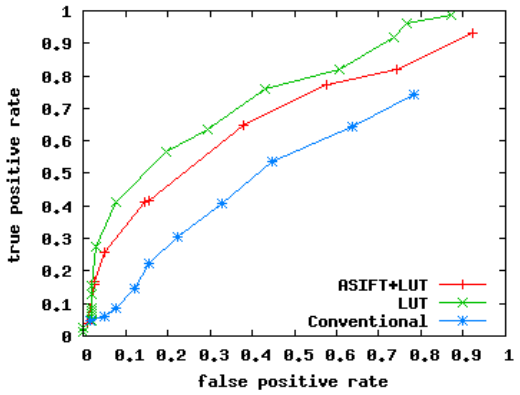
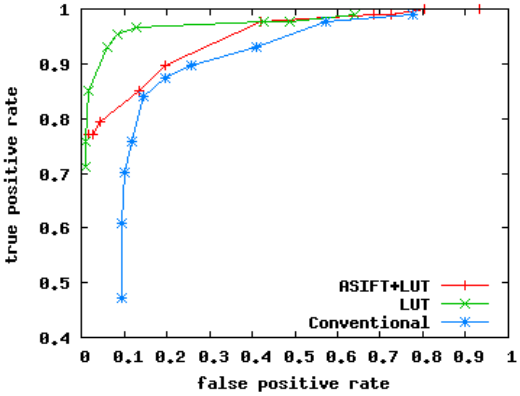


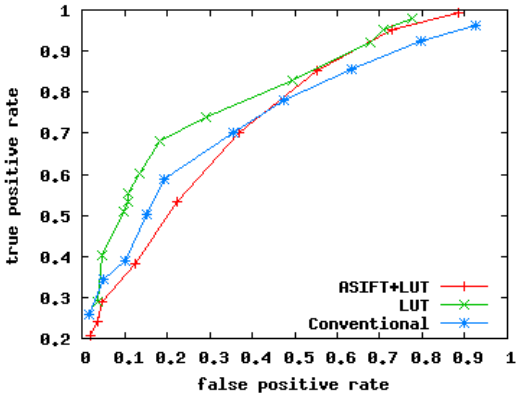
Figure 2.3.7 LUT of DVD player 2. X axis is feature distance and Y axis is computed confidence. Note that the distance is normalized to a LUT bin size of 64.



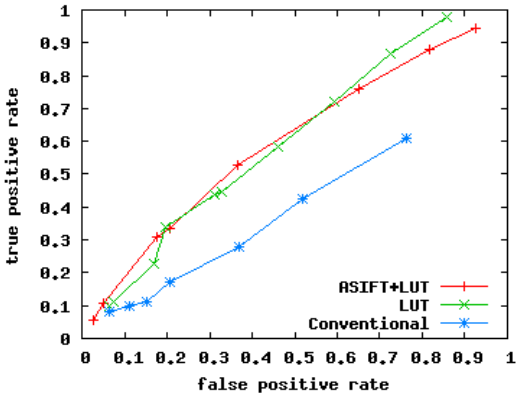
(a) TV



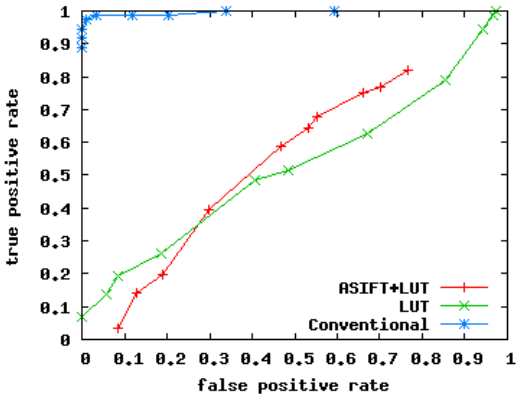
(b) DVD1



(c) DVD2



(d) Air conditioner



(e) Coffee can

Figure 2.3.8 Performance comparison of a conventional method, an LUT-based method, and LUT+Affine SIFT applied using the sample datasets.

Table 2.4 Number of thst samples used for the evaluation

Objects	Positive	Negative
TV set	403	232
DVD player 1	172	232
DVD player 2	278	232
Air conditioner	279	232
Coffee can	140	232

2.3.8. Conclusion of feature point based object recognition

In this section, a confidence-based feature-matching method is described. *SIFT* feature-based matching is widely used. However, a *SIFT* descriptor usually produces low feature points when applied to texture-less objects. It is not easy to adopt a feature-matching method to texture-less objects because of its low feature count. The *SIFT* point-based object-recognition method is based on the BBF method, which compares the distances between one feature point and its closest neighbor with the distances to the second closest neighbor. To overcome this problem, we replaced a binary-based decision method with an LUT-based method. The conclusions of this section are outlined below.

We replaced a binary decision with a pre-trained LUT-based method. The evaluation results by using 2432 samples show that the proposed method can improves the recognition performance for texture-less objects. However, it cannot improve the performance of texture-full objects.

In addition, the proposed method can be considered suitable for hardware implementation, because the pre-trained LUT does not require a computational process in the classification stage.

2.4. Conclusion of feature-point based object recognition.

In this section, we focus on *SIFT* feature matching based object recognition. This method utilizes object detection relying on matched pairs of *SIFT* features between query and database *SIFT* feature points. Two *SIFT* feature-point based object-recognition related methods are proposed in this section.

First, a computational cost reduction method is proposed. After extracting the feature point, object detection is performed using a number of matched *SIFT* feature pairs. To overcome this problem, the *SIFT* feature-point based matching method has a drawback in terms of its computational costs. Similar *SIFT* descriptors in the query image are removed in advance. The evaluation results show that this method can reduce the amount of memory and processing without sacrificing accuracy.

Second, we propose a new matching method for low feature counts. A *SIFT* descriptor usually produces a few points when applied to texture-less objects. That is, for texture-less objects, the *SIFT* descriptor-based matching method is difficult to adopt. To deal with this problem, we replaced a binary based decision method with an LUT-based method. This confidence LUT-based matching method can improve the recognition performance when applying *SIFT* feature matching to texture-less objects.

3. Human pose tracking

3.1. Introduction

Visualization of a human posture is a beneficial field of research. For example, for sport training, sedulous training is essential to attain greater sports proficiency. It is not easy for athletes to obtain a complete view of their posture on their own. A skilled trainer can provide the guidance on an athlete's posture, but it is not easy to explain verbally.

Many researchers have recently researched human-pose estimation in the field of computer vision. As an example, the visualization of a ballet dance using Gaussian Process Latent Variable Models (GPLVM) was proposed [56]. The method does not rely on an image, and uses information from a motion capture sensor.

Kinect [57] is widely used for pose estimation. This system captures image data using depth data obtained through a combination of an RGB camera, IR projector, and IR camera. In addition to an image data processor for a camera module, *Kinect* also contains a human pose estimator. The human-pose classification algorithm is based on a random forest method.

Urtasan *et al.* proposed a dynamic 3D reconstruction model [58] for tracking of the golf swing. In this method, Club tracker [59] is used for tracking a golf club, WSL tracker [60] is used to track the body joints of the knees, ankles, and wrists.

In this section, a method for estimating and tracking of a golf swings in a video sequence is proposed.

This tracking method [61][62] combines the global-pose estimation, a *pictorial structure model* (PSM)[63], and a particle filter. The proposed system estimates the tracking of the golf player's grip from a monocular camera video using the original image, and does not require another sensor.

3.2. Particle filter

Particle Filter is a very popular algorithm for the tracking problem. Kitagawa originally proposed the method as Monte-Carlo filter [64] in 1995. In the area of computer vision research, Condensation algorithm[65] was proposed for contour tracking. This algorithm is based on a manner analogous to *Particle Filter*. There are many problems to be tackled for robust tracking including lost tracking and modeling the changes in appearance and motion. Many researchers have attempted to tackle these problems. For changes in appearance, Zhou *et al.* presented an approach [66] that uses appearance-adaptive models in a particle filter to realize robust visual tracking.

Another approach is the use of a combination of off-line trained discriminative observers with

different life spans [67]. For lost tracking, a memory-based particle filter [68] method that stores the past history of the estimated target states is used. In this section, we treat the tracking of a golf swing. For visual tracking of human body parts, an algorithm [69] using a limb-tracking system is proposed based on an accumulated 2D model.

A particle filter is a Monte Carlo approximation approach, and is commonly used for tracking applications [70]. The posterior probability can be defined as

$$p(x_t \mid y_{1:t}) = k \; p(y_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}) p(x_{t-1} \mid y_{1:t-1}) dx_{t-1} \tag{3.1}$$

,where x_t is the processing state at time t, y_t is the observation, $y_{1:t}$ indicates the entire observation through time t, $p(x_t \mid x_{t-1})$ is a process dynamic distribution process, $p(y_t \mid x_t)$ is the observation of the likelihood distribution, and k is the normalizing factor.

At each time step, the particle filter updates each particle according to the previous particle set, and n particles x_{t-1} are sampled from the current particle set.

1. Updated particle sets are generated by sampling from the proposed distribution.
2. The weights of each particle are calculated.
3. The tracking position of the target is estimated by calculating the mean position of the particles.

In this section, we track the grip position of a golfer using a particle filter. HSV color histogram is used as an observation model. For a similarity measure, we adopt the *Bhattacharrya* distance [30] as

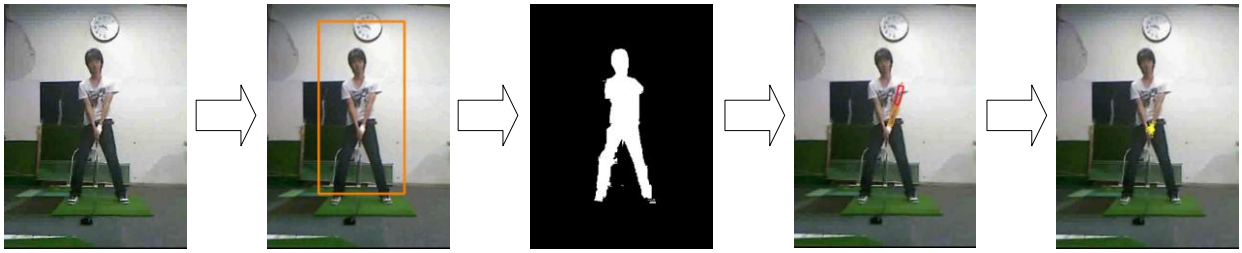
$$b_{dist} = -\ln(\sum \sqrt{u(x)v(x)}) \quad , \tag{3.2}$$

where $u(x)$ and $v(x)$ are the color histogram of the initial and tracking frames, respectively.

3.3. Proposed method

Figure 3.3.1 shows the flow of the proposed algorithm. The main idea is to combine a conventional particle filter with the global-pose estimation. The conventional particle filter depends on the local information. Therefore, it is affected background clutters too much. In addition, lost track recovery is difficult. We focus on a combination of both global and local features and estimate the trajectory of the golfer's grip as follows:

1. Position estimation is performed.
2. For each frame, the pose of the golfer is estimated using a silhouette feature-based estimator.
3. The left arm of the golfer is tracked by using the PSM.
4. The grip of the golfer is tracked using a constrained particle filter.



(a)Original Image (b)Person Detection (c)Pose Estimation (d)PSM Tracking (e)Tracking of Swing

Figure 3.3.1 Process flow of the proposed algorithm

3.3.1. Position estimation of a golfer

As the first step, we need to estimate the position of the golf player in the video. In the proposed method, we combine the global-pose estimation and local grip tracking for an accurate tracking of the golf swing. As a pre-process of the pose estimation, cropping around the are of the person is essential, because estimation algorithm assumes that a golf player exists in the region of interest (ROI) box. Therefore, precise position estimation is required to achieve accurate pose estimation. The position and size of the golfer differ for the sample images as shown in Fig. 3.3.2. We assume that an input video stream contains the image of the golf player. For the pose estimation, we need to crop the foreground regions around the golf player.

We use the first frame of the video sequence as the target frame for the position estimation. Strictly speaking, because the proposed system is intended for a whole video sequence, the estimation process should be conducted for every frame. Considering a golf swing movement, the arms, torso, and head move, while the rest of the player's body dose not. Therefore, we assume that the position of the golfer in a video stream does not change during the golf swing. In the proposed system, the position-estimation process uses the image of the first frame. The detected position is applied for every following frame during the position-estimation process.

In addition, using the first frame has one more advantage. Using the first frame of the golfer, we can predict the player's posture. When the golf player starts swing, the player may enter a position called an *address*. This posture is common for many golf players. This means that the variety of shapes in the detection target (person) can be reduced. For this issue, one common difficulty comes from the variety of golfer position. Simple-shaped models may not achieve good results because of the changes in position. In the proposed system, we can expect the shape of the person in the first frame to be in an address posture. The human-detection system can assume the shape of the player. Therefore, we use this shape for the golfer pose estimation.

Figure 3.3.2 Different size and positions of golf players for different samples

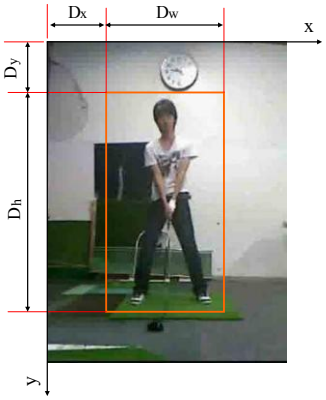


Figure 3.3.3 Position estimation results

The overall architecture of the position estimation is based on a combination of *HOG* features and the *AdaBoost* method. As described above, we use the first frame of the video. The classifier is trained using the training images in advance.

Position estimation is performed using the following steps.

1. The crop detection window is created using a raster scan.
2. The *HOG* features of the detection window are calculated.
3. The position with the highest score is estimated after all scans are finished.

3.3.2. Pose estimation

The framework of the pose-estimation method is based on the *silhouette feature* [71] and *HOG* features [12]. A golf swing can be divided in eight pose according to the orientation of the golf club. Figure 3.3.3 shows typical silhouette image samples. The pose of the golf player can be estimated as follows:

1. Silhouette images are calculated by using the *Grabcut* method.
2. The *HOG* features of the silhouette images are computed.
3. The *AdaBoost* classifier is applied.

(Preprocessing)

A filtering process is performed to avoid an incorrect segmentation caused by a small noise such as a complex background or clothes wrinkles. However, the edge information should be preserved because it contains important features for segmentation. A conventional low-pass filter eliminates such edges. Therefore, a *bilateral filter* [72][73] is applied as a preprocessing. This filter can preserve the edge details edges and remove noise simultaneously. The parameters of the *bilateral filter* are set as d=5, σ_1=35, and σ_2=5 experimentally. Here, d is the diameter of each pixel neighborhood, σ_1 is the color sigma in the color space, and σ_2 is the space sigma in the coordinate space, respectively.

We then crop a rectangular area around the golf player. The silhouette features need to be calculated using the image region around the player to avoid any background influence. The position and size of the golf player are assigned by hand in advance in the first image frame, and the pose estimator applies this location information to each flame. However, this approach does not work well on actual scenes. We found that the performance had noticeably declines when the player's arms are extended in a horizontal direction (P2 or P6 in Fig. 3.3.4). Because the position of the cropping box is estimated by observing the images in the initial frame, most of the player's arms are out of the cropping box when in these positions. Therefore, the size of the cropping box is extended by ten percent in horizontal direction.

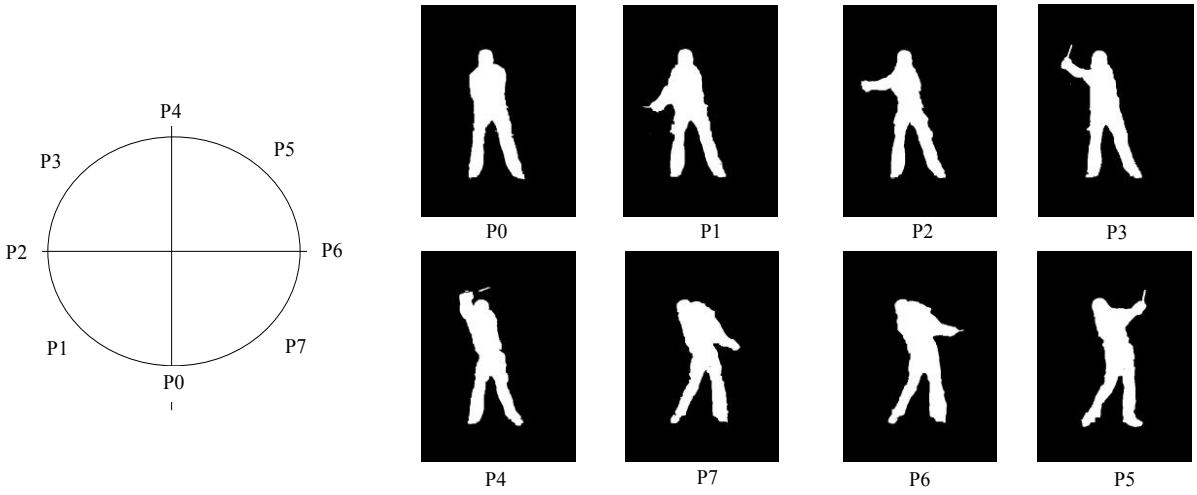


Figure 3.3.4 Typical silhouette image samples

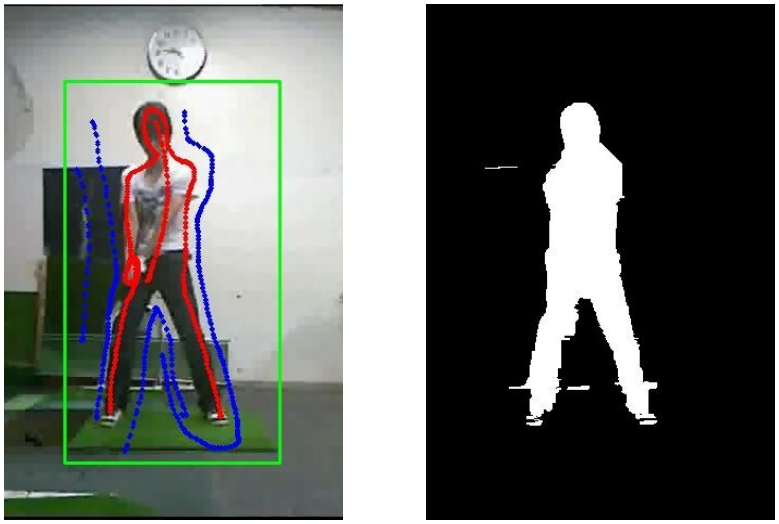
(Grabcut)

To estimate a human pose, foreground and background separation is an important task. To get obtain a good pose-estimation performance, a highly precise foreground separation is essential. The hand position of the players affects the classification results of the pose estimator. For example, as shown in Fig. 3.3.4, the silhouettes of classes P6 and P7 are very similar, and the difference seems to be in the position of the golfer's arms. We can see a similar result in P2 and P3. For the P0 class, the arms of the players appear to be missing because the player is in

"address" position, and the player's arms are covered by the torso. The proposed design is intended for use on a real scene. Therefore, this problem should noted.

We used *Grabcut* [74], which is an effective segmentation method using a energy minimization. An energy function is defined so that its minimum should correspond to good segmentation. This algorithm combines hard segmentation from iterative graph-cut optimization with boarder matting. This an effective method and is widely used.

In our method, *Grabcut* is chosen to segment the background and foreground of a golf player to generate silhouette features. Figure 3.3.5 (b) shows the segmentation results from this method. Note that this method requires both foreground and background information.



(a) Original image (b) Segmentation results

Figure 3.3.5 Segmentation results by using the *Grabcut* method. (a) original image with user inputs with a green colored rectangle (region of interest) , red (foreground) , and blue (background) , and (b) Human silhouette.

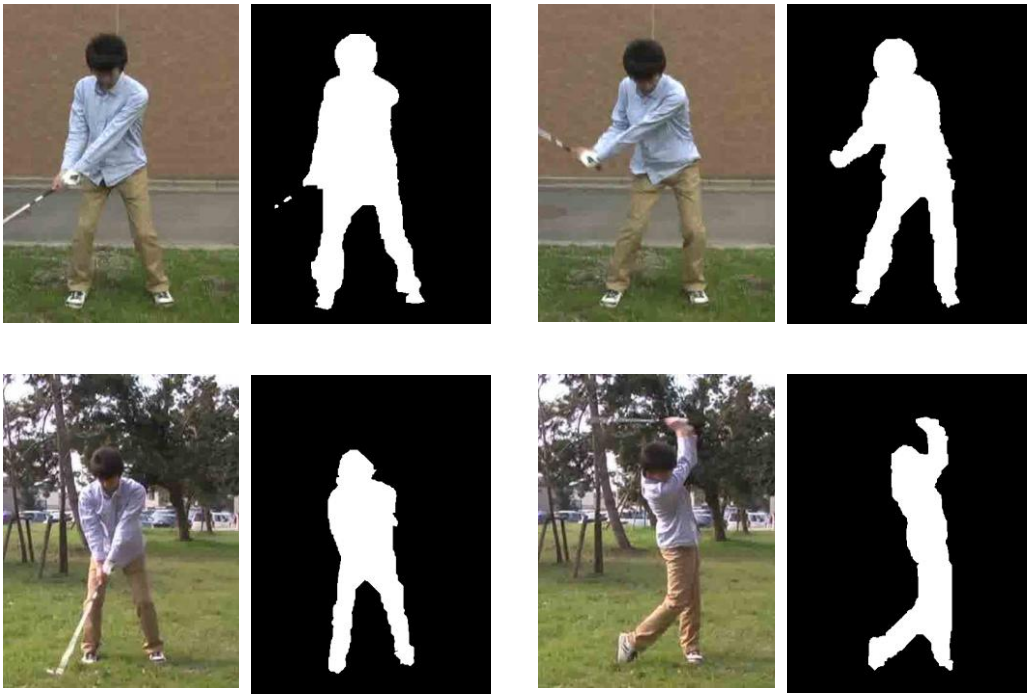


Figure 3.3.6 Segmentation samples

(HOG)

Histogram of Gradients (*HOG*) [12] achieves outperforming performance compared to other features used in human detection. *HOG* is widely used for such applications as object [40] and human detection [12][56][39], as well as human-pose estimation [38]. Qiang *et al.* integrated the cascade-of-detectors approach with *HOG* for a fast and accurate human detection method [74]. For pose detection, a method in combination with randomized trees was proposed [38]. For this method, they grow an ensemble of random trees generated based on randomly sampled *HOG* features.

HOG features are a histogram of gradient orientations of the intensity in local regions that can describe the shapes of an objects. They can provide a dense overlapping description of the image regions. The basic idea here is evaluating a well-normalized local histogram of the image gradient orientations in a dense grid. Similar features are increasingly used. The appearance and shape of a local object can be characterized using the distribution of local intensity gradients, even without precise knowledge of the corresponding gradient positions. Position information is not used in *HOG*. In practice, this is implemented by dividing the original image into small image patches called "cells", and the local histogram gradient directions of each cell are accumulated. For better robustness to changes in illumination, a contrast that normalizes the local responses is useful. This can be done using larger spatial regions, called "blocks". We refer to the normalized descriptor blocks as *HOG* descriptors.

A feature is a histogram of adjacent pixel gradients for the local regions. The magnitude and

gradient orientation are calculated to compute the *HOG* features from the intensity of the pixels.

We therefore represent the set of histograms of the magnitude in cell region c ($p \times q$ pixels) as

$$Vc = \{vc(0), vc(1), \ldots, vc(bin)\} \quad , \tag{3.3}$$

where $vc(n)$ is the histogram of cell region c and bin is the histogram bin. As shown in Fig. 3.3.7, the histogram is normalized by each block region ($r \times r$ cells) to extract the features as

$$v'c(n) = \frac{vc(n)}{\sqrt{\sum_{k=1}^{r \times r \times bin} vc(k)^2}} \tag{3.4}$$

After normalization, histogram $V'c$ can be represented as

$$V'c = \{v'c(0), v'c(1), \ldots, v'c(bin)\} \tag{3.5}$$

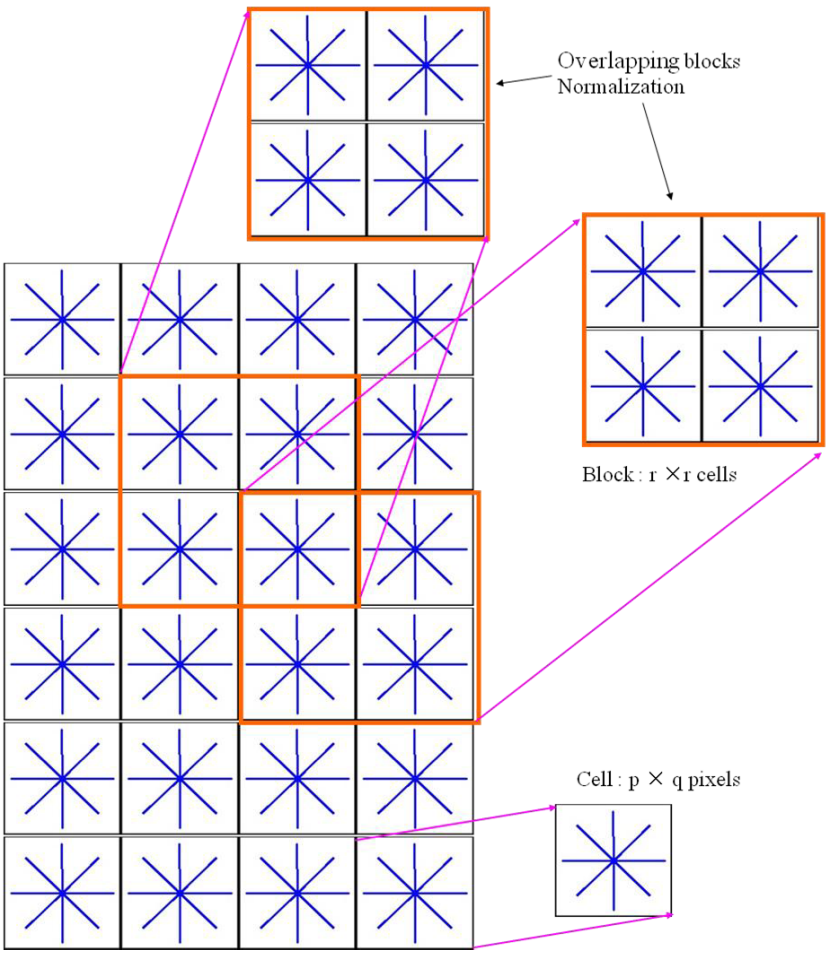


Figure 3.3.7 Normalization scheme of HOG features

(AdaBoost)

The *AdaBoost* algorithm [47] is a classifying method combining multiple weak learners. It is adopted to select *HOG* features from the feature pool and to combine them into a strong classifier. The combined strong learner $H(x)$ can be defined as

$$H(x) = sign\left[\sum \alpha_t h_t(x)\right], \tag{3.6}$$

where α_t are the learned weights of the weak learners, and $h_t(x)$ are the weak learners, respectively. Training of the *AdaBoost* classifier is used to determine the weight, α_t.

(Class Label)

We define the class labels to adopt the pose estimation according to the orientation of the golfer's club. Figure 3.3.4 shows typical silhouette image samples. We divide the golf swing orientation into eight classes as P0 - P7. For instance, the address position corresponds to P0, and the position in which the golfer raises the club at the highest corresponds to P4. During the golf swing, the following transition of classes is expected as follows: P0 - P1 - P2 - P3 - P2 - P1 - P0 - P7- P6 - P5 - P4. Here, P0 is the initial address posture, and P2 is the top position.

3.3.3. PSM

The pictorial structure model (*PSM*) [63][74][76] is an efficient framework for a part-based appearance recognition of an object. It is commonly used for object recognition and human-pose estimation. The basic idea is to represent an object based on a collection of parts arranged in a deformable configuration. The appearance of each part is modeled separately, and the deformable configuration is described using the spring-like connections of the parts. These models allow for descriptions of visual appearance. The model is used for human bodies and faces.

A pictorial structure model for an object is given by a collection of parts with connections. A general way to represent such a model is in terms of an undirected graph:

$$G = (V, E) \quad , \tag{3.7}$$

where the vertices $V = \{v_1, \ldots, v_n\}$ corresponds to the n parts. There is an edge $(v_i, v_j) \in E$ for each pair of connected parts, v_i and v_j. An object instance is given by the configuration $L = (l_1, \ldots, l_n)$, where each l_i specifies the location of part v_i. The matching problem of a pictorial structure to an image can be defined as an energy function minimization problem. The cost of a given configuration depends on two elements: how well each part matches the image data at its location, and how well the relative locations of the parts meet the deformable model. Given an image, a matching of the model to an image can be naturally defined as

$$L^* = \arg\min_L \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i, v_j) \in E} d_{ij}(l_i, l_j) \right) , \tag{3.8}$$

where $m_i(l_i)$ is a function measuring the degree of mismatch when part v_i is placed at location l_i in the image, and $d_{ij}(l_i, l_j)$ is a function measuring the degree of deformation of the model when part v_i is placed at location l_i, and part v_j is placed at location l_j.

An optimal match is a configuration that minimizes the sum of the match costs m_i for each part, and the deformation costs d_{ij} for connected pairs of parts. This energy function is simple and makes intuitive sense.

In the proposed system, the *PSM* is used to track the left arm of the golfer. Based on the general pictorial structure idea, as shown in Fig. 3.3.8, the player's arm is represented as a joint configuration of its parts, i.e. the upper arm and lower arms. In addition, S_x and S_y are the positions of the player's shoulder, θ_1 is the absolute orientation of the upper arm, θ_2 is the absolute orientation of the lower arm, S_h is the length of the arm, S_w is the width of the arm, and w_1 and w_2 are the arm positions.

Here, the score function can be defined as

$$S = b_0 + b_1 \tag{3.9}$$

$$b_n = \sum I'(x, y) \tag{3.10}$$

$$I'(x, y) = (I(x, y) - \mu)^T \sum^{-1} (I(x, y) - \mu)) , \tag{3.11}$$

where b_0 and b_1 are the score functions of the upper and lower arms, respectively. $I(x,y)$ is the RGB color of the position (x,y), and μ is the mean color of the target area, respectively. This score measures the similarity between the tracking arm target and the golfer's left arm using the color information.

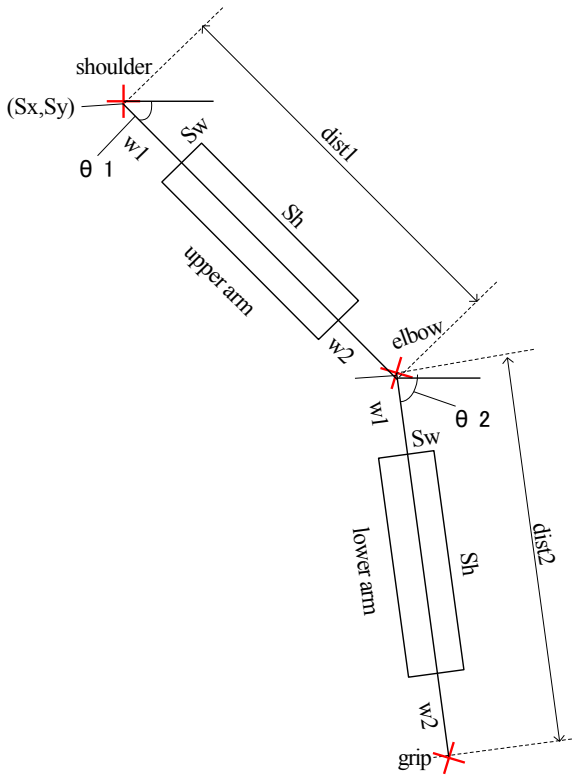


Figure 3.3.8 Arm model.

The posture and position of the golf player is modeled using two jointed rectangles.

(Constrained PSM tracking)

The purpose of *PSM* is to track the player's arm precisely. We combine *PSM* with the pose estimator to accurately track the player's arm. Here, the base position of the *PSM* model is defined using the shoulder position (S_x,S_y). In addition, the golf player's shoulder point during a golf swing is not fixed.

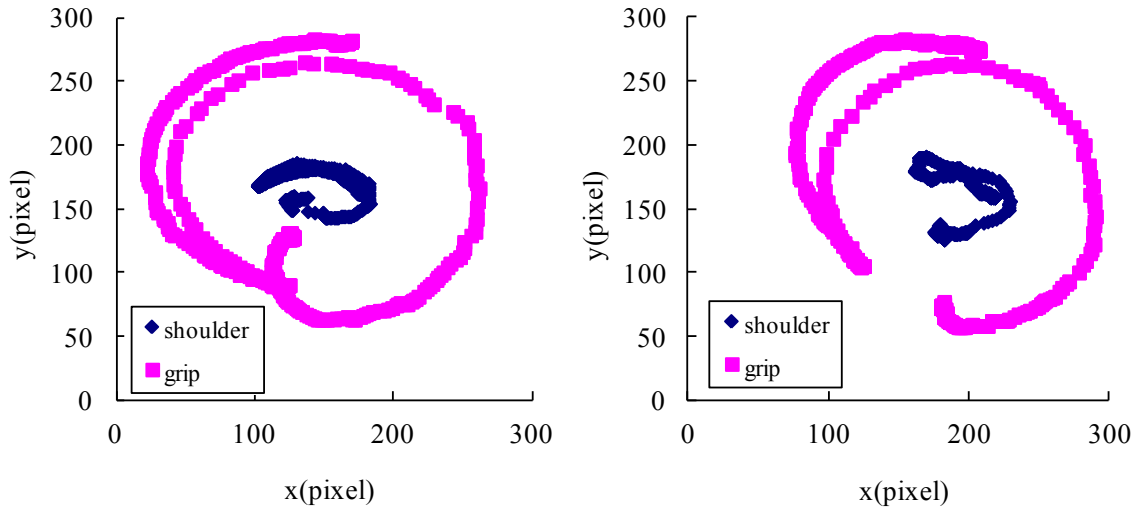
Figure 3.3.9 shows a plot of the golf player's shoulder and grip position movements from certain video images. In this figure, the x-y position scale is at the pixel level. According to the movement in grip position, the shoulder position is also moves in a circular fashion. The golf player's movement is not only in a simple circular motion, but also in a complex one. Because this motion seems to be difficult to model, we use a look-up-table approach.

We estimate the shoulder position using a look-up table (LUT) created based on the mean position from a video of a sample swing because the shoulder position rotates during the golf swing. Therefore, LUTs $f_x(l)$, and $f_y(l)$ are built using manually sampled data. The variable l indicates the estimated class labels P0 - P7. The LUT stores the normalized shoulder position which is not influenced by the size or position of the player. S_x,S_y can be defined as

$$S_x = f_x(l)\cdot D_w + D_x \tag{3.12}$$

$$S_y = f_y(l)\cdot D_h + D_y \quad , \tag{3.13}$$

where D_x ,D_y indicates the top-left location of the golf player area, and D_w, and D_h are the width and height of the player, respectively. To avoid a lost tracking, θ_1 is constrained. The angle θ_2 is maintained with θ_1 because we assume that the lower and upper arm rotations are synchronized.



(a) Player A (b) Player B

Figure 3.3.9 Shoulder position plots of a golf swing.

(red: grip position, blue: shoulder position)

3.3.4. Constrained particle filter

Recovering from a lost tracking using *Particle Filter* is not easy. One reason for this is that *Particle Filter* relies on the local feature information. Both local and global features are combined to tackle this problem. As mentioned above, course tracking can be conducted using *PSM*. The tracked target is the grip, and it is assumed that the grip is an extension of the player's arm. We therefore replace some particles in the extension of the estimated arm. Some particles are dismissed during the transition step of the particle filtering process. Here, we replace the low-weight particles using the new estimated position. When the tracking is operating normally, these replaced particles are dismissed during the transition step. Conversely, these replaced particles are selected for a lost tracking.

We opted for a Sequential Importance Re-sampling (SIR) particle filter [78].

Here, we approximate the grip motion using linear motion. The system model can be defined as:

$$X_t - X_0 = A_2(X_{t-2} - X_0) + A_1(X_{t-1} - X_0) + B_0\omega_t \quad , \tag{3.14}$$

where X_t is the position at time t, ω_t is system noise, and B_0,A_1, and A_2 are constants.

As a likelihood function, we use the color similarity as follows:

$$L = \exp\left(\sum \sqrt{p(k)q(k)}\right) , \tag{3.15}$$

where $p(k)$ and $q(k)$ are the color histogram of the initial frame and the local patch of each particle, respectively, and k is the histogram bin. For a color similarity, we used the *Bhattacharrya* distance [30].

Here, the following steps are used for $p(k)$ and $q(k)$ calculations.

(1) A 16 pixel x 16 pixel area is clipped around the initial grip tracking area of the golfers, as shown in Fig. 3.3.10.

(2) Conversion into an HSV color space [79].

$$V_H = \begin{cases} 60\dfrac{G-B}{\max(R,G,B)-\min(R,G,B)} & (if\ \max(R,G,B)=R) \\ 60\dfrac{B-R}{\max(R,G,B)-\min(R,G,B)}+120 & (if\ \max(R,G,B)=G) \\ 60\dfrac{R-G}{\max(R,G,B)-\min(R,G,B)}+240 & (if\ \max(R,G,B)=B) \end{cases} \tag{3.16}$$

$$V_S = \frac{\max(R,G,B)-\min(R,G,B)}{\max(R,G,B)} \tag{3.17}$$

$$V = \max(R,G,B) \tag{3.18}$$

(3) The color histogram is calculated.

We divide each hue, color, and saturation into ten steps. To choose the grayscale information, the grayscale part is added to the color histograms as follows.

If $V_s < S_{TH}$, or $V_V < V_{TH}$, then

$$bin = NH \cdot NS + BIN_V , \tag{3.19}$$

else

$$bin = BIN_S \cdot NH + BIN_h , \tag{3.20}$$

where NH=10, NS=10, NV=10, $BIN_h = V_h \cdot NH/360$, $BIN_S = Vs \cdot NS$, $BIN_V = V_V \cdot NV$, S_{TH}=0.1, and V_{TH}=0.1.

The dimension of the calculated histograms is $NH \times NS + NV$=110.

(4) The histogram is normalized to calculate the Bhattacharrya distance.

$$\begin{aligned} p(k) &= p'(k) / \sum p'(k) \\ q(k) &= q'(k) / \sum q'(k) \end{aligned} \tag{3.21}$$

Here, $p(k)$ and $q(k)$ are normalized histograms, and $p'(k)$ and q'(k) are the histogram of the initial frame and patch histogram of the particles, respectively.

Figure 3.3.10 Tracking target area

3.4. Evaluation of the algorithm

Next, we demonstrate that the proposed method can work on the tracking of a golf swing. The evaluation consists of two parts: the performance of pose estimator and the tracking performance.

3.4.1. Evaluation of the pose-estimator

The pose-estimator was evaluated using an uncontrolled video taken by a consumer video camera. The video was divided into training samples and test samples. The background scenery, player position, and frame numbers are differ from sample to sample. The movie is 512 pixel x 384 pixels in resolution. The models were trained as mentioned above, and we evaluated them based on separate test data.

Table 3.1 shows a confusion matrix. The label “test class” corresponds to the input class, and the “classifier result” label corresponds to the classification results. For example, 129 samples of “P2” labeled samples were correctly classified into the "P2" class. Five "P2" labeled samples were misclassified as "P1" class. In this case, a perfect classification is very hard to achieve because the golf swing has a rotational movement. We divided the rotation into eight classes. A precious division was impossible, however. We therefore adopted a tolerance for

misclassification, which allows for adjunct misclassifications. In Table 3.1, the blue labeled cells are correct classifications. In addition, we accepted adjunct correction (green cell of Table 3.1).

Table 3.2 shows the performance rate along the criteria. All matching rates were over 94 percent. This rate is sufficient for a pose-estimator used in golf swing tracking, and is quite a good performance. Note that this evaluation is conducted under the supposition that the golf player is within the detection window.

Table 3.1 Confusion matrix of the pose estimator

		Classifier result							
Test class		P0	P1	P2	P3	P4	P5	P6	P7
	P0	76	38	0	1	0	1	2	14
	P1	28	131	15	0	0	1	0	6
	P2	7	5	129	55	0	1	2	0
	P3	1	0	6	113	19	0	0	2
	P4	4	2	2	21	209	11	2	3
	P5	2	3	1	1	47	133	14	2
	P6	0	0	0	0	1	24	125	8
	P7	6	0	1	0	2	1	13	133

Table 3.2 Pose-estimation matching rate

	Matches	Samples	Rate(%)
P0	128	132	97.0%
P1	174	181	96.1%
P2	189	199	95.0%
P3	138	141	97.9%
P4	241	254	94.9%
P5	194	203	95.6%
P6	157	138	99.4%
P7	152	156	97.4%
Average			96.7%

3.4.2. Tracking-performance evaluation

We evaluated the tracking performance for a video capturing an uncontrolled swing movie. We used golf swing videos from an Internet website [80]. These videos were not intended for research purposes, and they contain cluttered backgrounds, shadows and wrinkles.

We used six sample movies of different golf players. Initially, we compared the tracking error of the proposed system with the particle filter. Eqn. 3.1 shows the evaluation criteria.

$$\frac{1}{N}\sum_{i=0}^{N}\left\|d_{\exp}(i)-d_{track}(i)\right\| \quad , \tag{3.22}$$

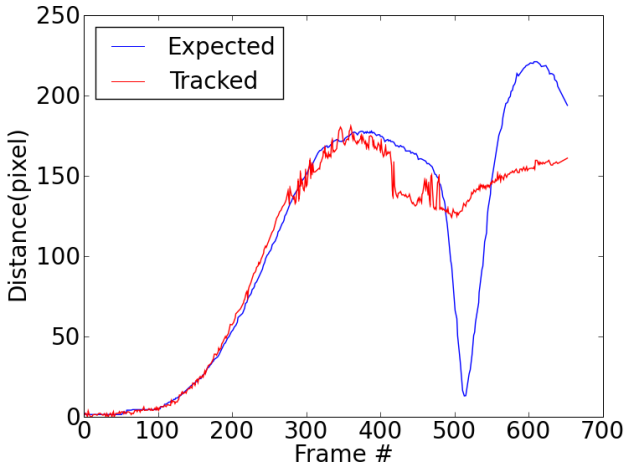
where $d_{exp}(i)$ and $d_{track}(i)$ are the *Euclidean* distances from the origin to the true position and tracking point, respectively. The true points were manually plotted. N is the frame number of the sample video. Table 3.3 shows the tracking errors. The proposed method improves the tracking errors. For all the sample players, the result indicates that the proposed method outperforms the conventional particle filter.

Table 3.3 Tracking errors

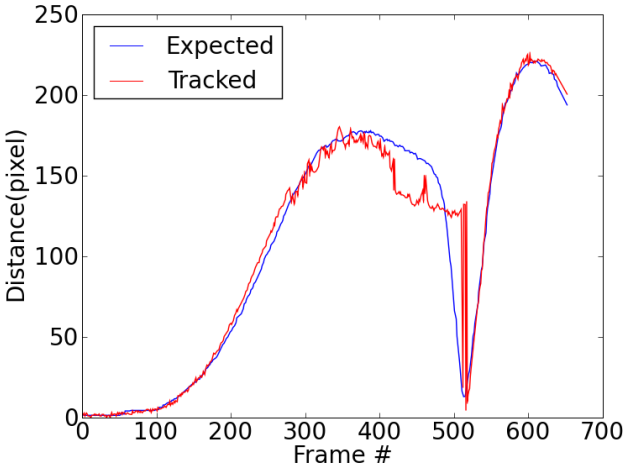
	Error (pixels)	
	Particle filter	Proposed method
Player A	18.2	8.4
Player B	49.1	15.3
Player C	38.7	33.3
Player D	65.8	25.3
Player E	22.4	9.1
Player F	7.7	7.5
Average	33.66	16.48

Figure 3.4.1 shows the result of the tracking-performance evaluation. The expected position is manually plotted. The X axis of the graph is the frame number, and the Y axis is the Euclidean distance from the left-top base point.

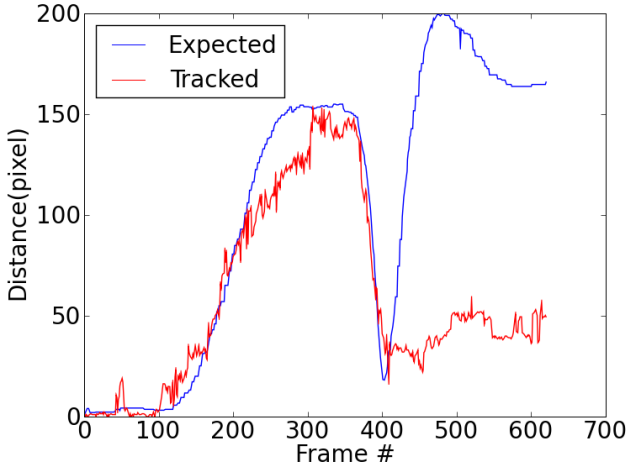
In general, the head speed varies during the swing movement. In particular, the head speed is increasing during a downswing motion. In many cases, the particle filter lost the tracking during a down swing. On the other hand, the proposed method could recover the lost tracking. (frame #400 of Fig. 3.4.1 (c) and (d), and frame #340 of Fig. 3.4.1 (k) and (l)). In Fig. 3.4.1(f), the tracking is not stable even when using the proposed method. In this case, an occlusion between the player's head and grip occurred, resulting a lost tracking.



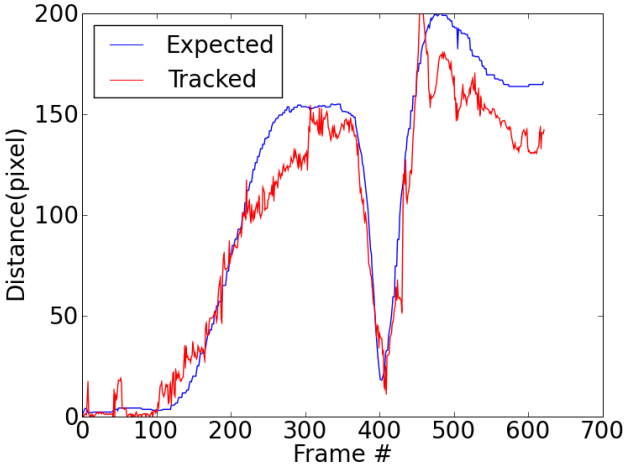
(a) Particle filter (player A)



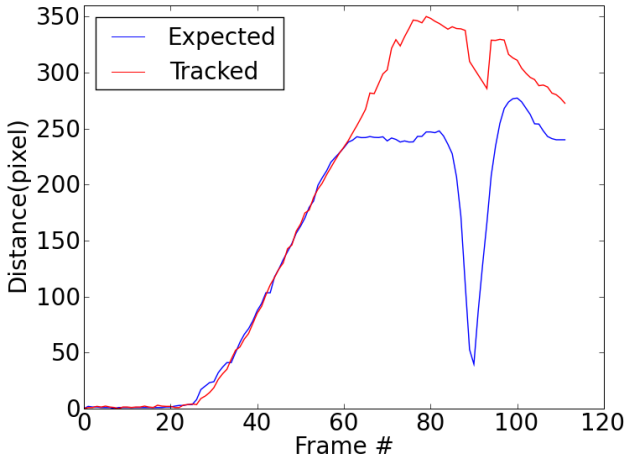
(b) Proposed method (player A)



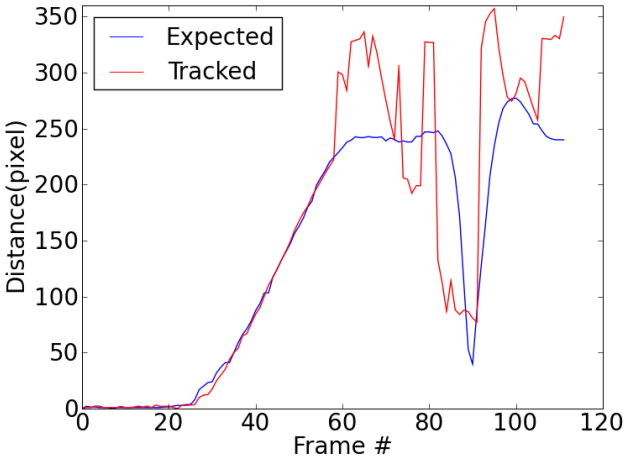
(c) Particle filter (player B)



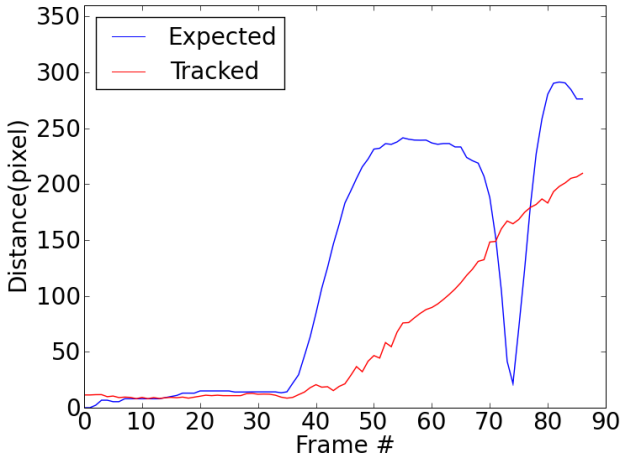
(d) Proposed method (player B)



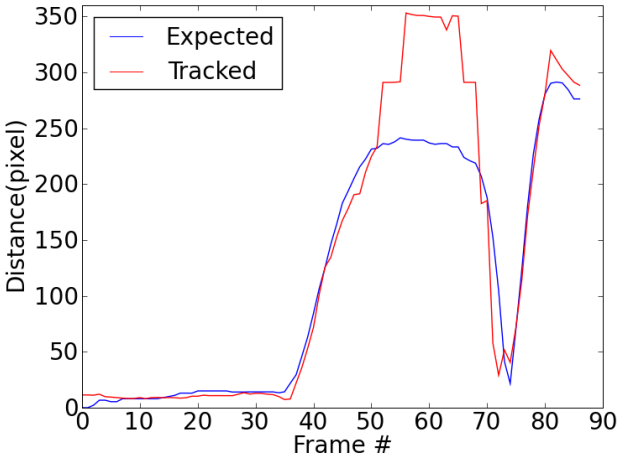
(e) Particle filter (player C)



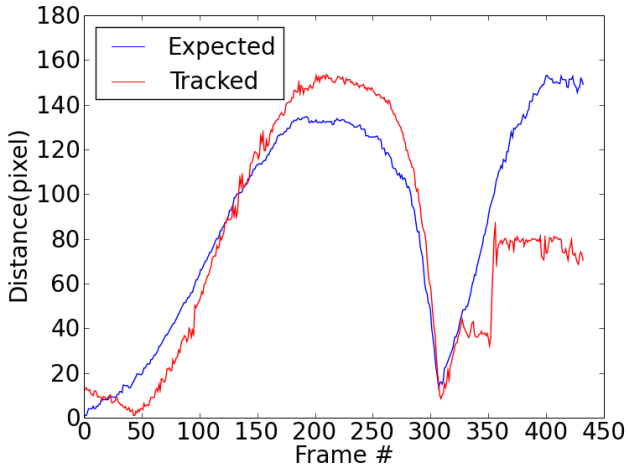
(f) Proposed method (player C)



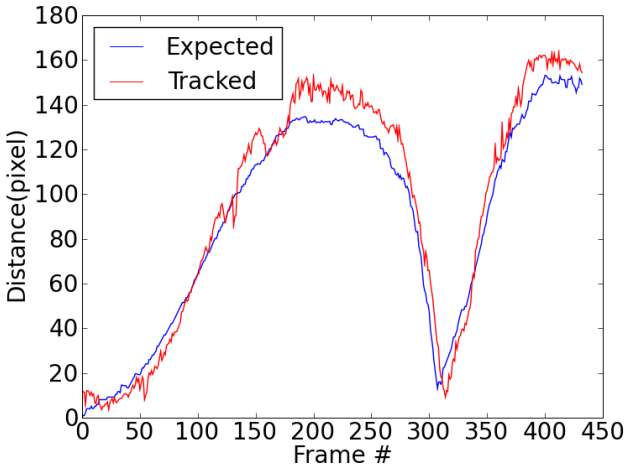
(g) Particle filter (player D)



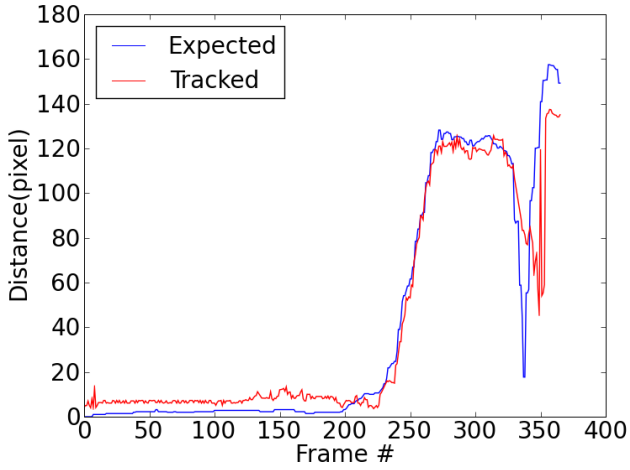
(h) Proposed method (player D)



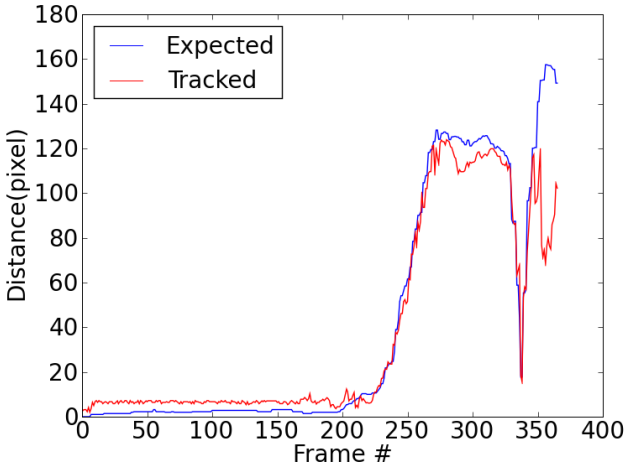
(i) Particle filter (player E)



(j) Proposed method (player E)



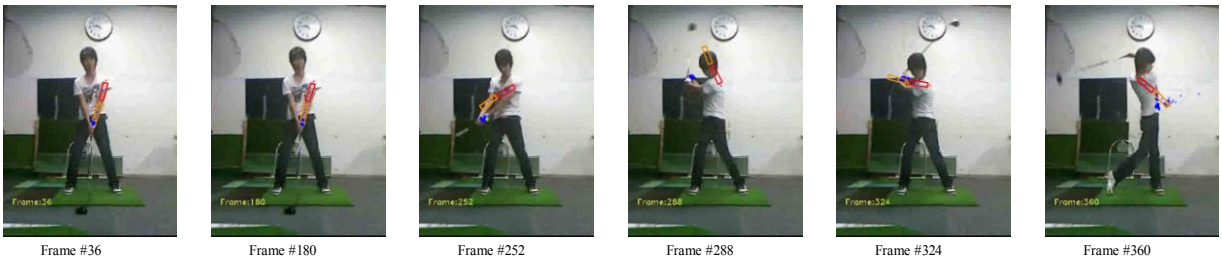
(k) Particle filter (player F)



(l) Proposed method (player F)

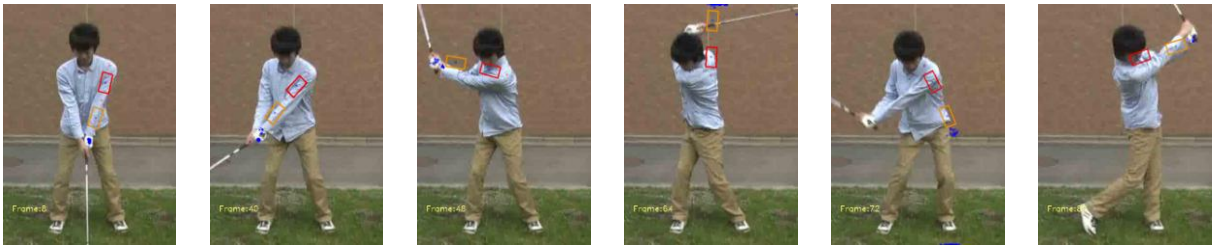
Figure 3.4.1 Tracking-performance comparison.

Figure 3.4.2 shows additional sample images. In Frame #72 in Fig. 3.4.2 (b), a PSM tracking error occurred which was caused by a color similarity with the complex background. Figure 3.4.3 shows the moving paths of the tracking result. In many samples, the experimental results demonstrated a high tracking recovery performance, compared to a conventional particle filter.



Frame #36 Frame #180 Frame #252 Frame #288 Frame #324 Frame #360

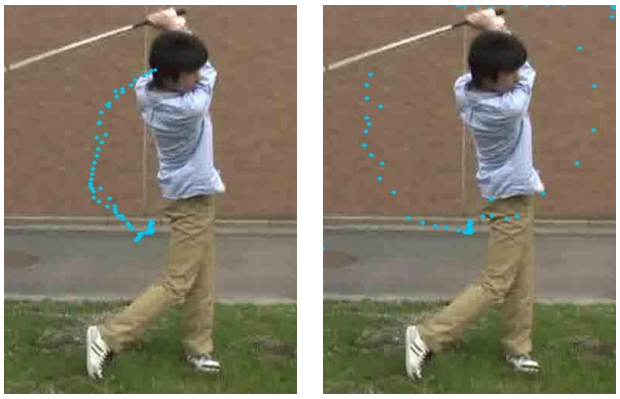
(a) Indoor scene



Frame #8 Frame #40 Frame #48 Frame #64 Frame #72 Frame #80

(b) Outdoor scene

Figure 3.4.2 Experimental results of swing tracking (red rectangle: upper arm, orange rectangle: lower arm, blue dots: particle filter)



(a) Particle Filter (b) Proposed Method

Figure 3.4.3 Estimation of player C"s swing

3.5. Conclusion of human-pose tracking

In this section, the proposed human-pose tracking method was described. We focused on a tracking problem of a golf swing. Tracking the grip area of a golf player from a video showing an uncontrolled golf swing using a monocular camera is not easy. Particle filter based methods are widely used for tracking problems. However, tracking one part of a person in a difficult uncontrolled and complex background is not easy, because the change in appearance of the person and complex background are difficult to resolve. In this section, a novel method that combines both global and local features was proposed. The conclusions of this section are outlined below.

The proposed method combines both global and local features. As global features, a human detection and pose estimation method is used. As local features, a constrained particle filter is used. The constraint comes from the global feature, which is the estimated arm location of the golf player. The proposed method consists of four steps: Human detection, Pose estimation and arm tracking using the *PSM*, and grip tracking using a particle filter. Human detection is based on *HOG* and *AdaBoost*. The pose estimation is based on a similar architecture as the human detector. We divide a golf swing motion into eight classes. Arm tracking is then performed by using the *PSM*, which is based on an appearance model. Next, a constrained particle filter is used for tracking the grip position of the golf player.

The pose estimator was evaluated by using an uncontrolled video sequence taken by a general video camera. The evaluation shows over a 94% accuracy in pose-class estimation.

The final tracking performance was evaluated by using uncontrolled video sequences. We compared an unconstrained conventional particle filter with the proposed method. The proposed method shows a better tracking performance than the unconstrained particle filter for all samples.

4. Conclusion

In this thesis, object recognition methods suitable for implementation in small hardware were discussed. Robots or home appliances that can comprehend their circumstances through image information will be helpful to users. They can provide many services using visual information. Such information will provide a user-friendly human interface. For such applications, it is necessary to implement an object-recognition method into smaller hardware, but is not an easy task. There are many problems to be tackled, such as changes in luminance, complex backgrounds, changes in the appearance of the target objects, and occlusions.

Meanwhile, recent years have seen dramatic increases in computational processing power. Therefore, some methods have been proposed for object recognition, which assume huge amounts of computational resources and memory. However, such solutions are impractical, that is, they are difficult to implement for small robots and home appliances owing to their high computational cost and only a few practical methods for such applications have proposed.

In this thesis, in order to solve these problems of conventional method, we proposed a reduction method of the *SIFT* feature points for object detection and a human-pose tracking method using PSM for body parts detection, and we achieved the following results, respectively.

1. A *SIFT* feature based object detection algorithm was described. This method is based on a similarity count of *SIFT* feature points of the query and database images. The conventional method is robust for changes in orientation and scale, and it is widely used. However, it requires a large amount of computational resources because *SIFT* features produces a large amount of feature points and require 128 dimensional vectors per point. Therefore, a straightforward implementation into smaller hardware is not practical. Therefore, in the thesis, we proposed a reduction of the *SIFT* feature points before matching. Proposed method removes similar *SIFT* descriptor pairs in the query image in advance. The criteria of similarity are based on the location of the *SIFT* keypoint and similarity of the descriptor. The evaluation results show that this method can reduce the amount of memory and processing without sacrificing accuracy.

In addition, an effective matching method using a pre-trained confidence table is proposed. A *SIFT* descriptor usually produces a few points when applied to texture-less objects. That is, for texture-less objects, the matching method is difficult to adopt. To deal with this problem, we replaced a binary based decision method with a LUT-based method. In the proposed method, two feature pair is picked up using conventional BBF. Then, decision making is performed by accumulating confidence score by LUT, which is trained by a log likelihood of histograms of positive and negative training sample datasets. The confidence LUT-based matching method can improve the recognition performance when applying *SIFT* feature matching to texture-less objects. The evaluation results by using 2432 samples show that the proposed method improved

the recognition performance for texture-less objects.

2. A human-pose tracking method was described. Considering the applications of embedded object detection, human recognition is a very important task for such products. However, human recognition remains an unsolved research area in computer vision. A person has a wide variety of appearance changes. In this thesis, we focused on a tracking problem of golf swing, which tracks the grip of a golf player from an uncontrolled golf swing video by using a monocular camera. Generally speaking, particle filter based methods are widely used for the tracking problem. However, tracking a part of a person in difficult uncontrolled and complex background is not easy. A novel method that combines both global and local features was therefore proposed. A pose estimation is used as the global features and color histogram is used as a local feature.

The proposed method consists of four steps: a) Human detection, b) Pose estimation, c) Arm tracking using the *PSM*, and d) Grip tracking using a particle filter. As a first step, we find human region by using human detection based on *HOG* and *AdaBoost*. Then, a pose estimation, which estimates player's posture is performed. After that, arm tracking is performed by using the *PSM*, which is based on parts-based appearance model. Finally, a constrained particle filter is used for tracking the grip position of a golf player. The pose estimator was evaluated by using an uncontrolled video sequence taken by a general video camera. The evaluation shows over a 94% accuracy in pose-class estimation. And the final tracking performance of golf swing was evaluated by using uncontrolled video sequences. The evaluation results show that the combination of the global and local features outperformed the conventional method based on particle filter. When we compared an unconstrained conventional particle filter with the proposed method, the proposed method showed a better tracking performance for all samples.

My future plan is a combination of the proposed methods and a realization of smaller hardware-implementation. The implementation of a complete object-recognition system oriented for smaller hardware remains unresolved issues, which includes a hardware oriented feature detector and a multi-class classifier. The straightforward implementation using conventional methods is impractical for smaller hardware because they require high computational cost and a large amount of memory. Therefore, in order to realize the complete object-recognition system, we would require the hardware oriented feature detector and the multi-class classifier.

5. References

[1] Yutaka Usui and Katsuya Kondo, “Hardware Implementation of Adaboost-based Sign Recognition for Miniature Robot,” Proc. of Asia Pacific Signal and Information Processing Association Annual Summit and Conf. (APSIPA-ASC), DVD-ROM, pp. 490-493, 2009.

[2] Yuehua Shi, Feng Zhao, and Zhong Zhang, “Hardware Implementation of Adaboost Algorithm and Verification,” Advanced Information Networking and Applications - Workshops, Int'l Conf. on, pp. 343-346, 2008.

[3] Hung-Chih Lai, M.Savvides, and Tsuhan Chen, “Proposed FPGA Hardware Architecture for High Frame Rate (>100 fps) Face Detection using Feature Cascade Classifiers,” Biometrics: Theory, Applications, and Systems (BTAS) First IEEE Int'l Conf. on, pp. 1-6, 2007.

[4] Vinod Nair, Pierre-Olivier Laprise, and James J.Clark, “An FPGA-Based People Detection System,” EURASIP Journal on Applied Signal Processing, Vol. 2005, No. 7, pp. 1047-1061, 2005.

[5] Junguk Cho, Shahnam Mirzaei, Jason Oberg, and Ryan Kastner, “FPGA-based Face Detection System using Haar Classifiers,” Proc. the ACM/SIGDA Int'l symposium on Field Programmable Gate Arrays, pp. 103-112, 2009.

[6] Paul Viola, and Michael J.Jones, “Rapid Object Detection using A Boosted Cascade of Simple Features,” Proc. of IEEE Computer Vision and Pattern Recognition (CVPR), pp. 511-518, 2001.

[7] Paul Viola, and Michael J.Jones, “Robust Real-time Object Detection,” Int’l Journal of Computer Vision (IJCV), Vol. 57, No. 2, pp. 137-154, 2001.

[8] Rainer Lienhart and Jochen Maydt, “An Extended Set of Haar-Like Features for Rapid Object Detection,” Proc. Int'l, Conf. on Image Processing (ICIP), Vol. 1, pp. 900-903, 2002.

[9] Altera Cyclone-III FPGA datasheet, http://www.altera.com/devices/fpga/cyclone3/cy3-index.jsp

[10] SystemVerilog, http://www.systemverilog.org/

[11] Navneet Dalal and Bill Triggs, “Histgrams of Oriented Gradients for Human Detection,” Proc. Int’l Conf. on Computer Vision & Pattern Recognition (CVPR), Vol. 1, pp. 886-893, 2005.

[12] Navneet Dalal and Bill Triggs, “Histgrams of Oriented Gradients for Human Detection,” Proc. Int’l Conf. on Computer Vision & Pattern Recognition (CVPR), Vol. 1, pp. 886-893, 2005.

[13] C.Hou , H.Ai and S.Lao, “Multiview Pedestrian Detection based on Vector Boosting,” Asian Conf. of Computer Vision (ACCV), Vol. 1, pp. 210-219, 2007.

[14] Anna Bocsh, Andrew Zisserman, and Muoz Xavier, “Image Classification using Random Forests and Ferns,” IEEE Int'l Conf. Computer Vision(ICCV), pp.1-8, 2007.

[15] Gregory Rogez, Jonathan Rihan, Srikumar Ramalingam, Carlos Orrite, and Philip H.S.Torr, “Randomized Trees for Human Pose Detection,” Proc. Int’l Conf. on Computer Vision and Pattern Recognition(CVPR), pp.1-8, 2008.

[16] Takayoshi Yamashita, Yuji Yamauchi, and Hironobu Fujiyoshi, “Human Detection for Multiple Pose by Boosted Randomized Trees,” First Asian Conf. on Pattern Recognition (ACPR), pp.229-233, 2011.

[17] B.Leibe, E.Seeman and B.Schiele, “Pedestrian Detection in Crowded Scenes,” IEEE Conf. on Computer Vision and Pattern Recognition (ICCV), Vol. 1, pp.878-885, 2005.

[18] B.Wu and R.Nevatia, “Detection and Tracking of Multiple, Partially Occuluded Humans by Bayesian Combination of Edgelet based Part Detectors,” Int’l Journal of Computer Vision(IJCV), Vol. 75, pp.247-266, 2007.

[19] Yutaka Usui and Katsuya Kondo, “The SIFT Image Feature Reduction Method using the Histogram Intersection Kernel,” Proc. of Int'l Symposium on Intelligent Signal Processing and Communication Systems (ISPACS) CD-ROM, pp. 517-520, 2009.

[20] Yutaka Usui and Katsuya Kondo, “3D Object Recognition Based on Confidence LUT of SIFT Feature Distance,” Proc. of the World Congress on Nature and Biologically Inspired Computing (NaBIC), pp. 300-304, 2010.

[21] David G.Lowe, “Distinctive Image Features from Scale-invariant Keypoints,” Int’l Journal of Computer Vision (IJCV), Vol. 60, No.2, pp. 91-110, 2004.

[22] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, “SURF: Speed Up Robust Features,” Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346-359, 2008.

[23] Yan Ke and Rahul Sukthankar, “PCA-SIFT: A More Distinctive Representation for Local Image Descriptors,” Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Vol. 2, pp. 506-513, 2004.

[24] Vijay Chandrasekhar, Gabriel Takacs, David Chen, Sam Tsai, Radek Grzeszczuk and Bernd Girod, “CHoG: Compressed Histogram of Gradients,” Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 2504-2511, 2009.

[25] D.Nister and H.Stewenius, “Scalable Recognition with A Vocabulary Tree,” Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Vol. 2, pp. 2161-2168, 2006.

[26] Vijay Chandrasekhar, Gabriel Takacs, David Chen, Sam S.Tsai, Jatinder Singh, and Bernd Girod, “Transform Coding of Feature Descriptors,” Proc. of SPIE, Visual Communications and Image Processing, Vol.7257, 725710, 2009.

[27] Sunil Arya, David M.Mount, Nathan S.Netanyahu, Ruth Silverman and Angela Y.Wu, “An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions,” Journal of the ACM, Vol. 45, No. 6, pp. 891-923, 1998.

[28] Annalisa Barla, Francesca Odone and Aalessandro Verri, “Histogram Intersection Kernel for Image Classification,” Proc. Int’l Conf. on Image Processing (ICIP), Vol. 3, pp. 513-516, 2003.

[29] Kullback, Solomon, and Richard A. Leibler. “On Information and Sufficiency,” The Annals of Mathematical Statistics Vol. 22, No.1, pp. 79-86, 1951.

[30] A.Bhattacharrya, “On a Measure of Divergence between Two Statistical Populations Defined by Their Probability Distributions,” Bulletin of the Calcutta Mathematical Society, Vol. 35, pp. 99-109, 1943.

[31] Tom Fawcett, “An Introduction to ROC Analysis,” Pattern Recognition Letters, Elsevier, Vol. 27, pp. 861-874, 2007.

[32] Simon Winder and Matthew Brown, “Learning Local Image Descriptors,” Proc. Int’l Conf. on Computer Vision and Pattern Recognition (CVPR), pp.1-8, 2007.

[33] Hao Shao, Tomáš Svoboda, and Luc Van Gool, “Zubud-zurich Building Database for Image-based Recognition,” in Technical Report, 260 (Zurich, Switzerland), 2003.

[34] Irnya Gordon and David G.Lowe, “What and where : 3D Object Recognition with Accurate Pose,” In Toward category-level object recognition, Springer Berlin Heidelberg, pp. 67-82, 2006.

[35] Jean Ponce, Martial Hebert, Cordelia Schmid, and Andrew Zisserman (eds.), “Toward category-level object recognition,” Vol. 4170, Springer, 2007.

[36] Edward Hsiao, Alvaro Collet and Martial Hebert, “Making Specific Features Less Discriminative to Improve Point-Based 3D Object Recognition,” Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 2653-2660, 2010.

[37] Alvaro Collet, Dmitry Berenson, Siddhartha S.Srinivasa, and Dave Ferguson, “Object Recognition and Full Pose Registration from a Single Image for Robotic Manipulation,” IEEE Int'l Conf. on Robotics and Automation (ICRA), pp. 48-55, 2009.

[38] Gregory Rogez, Jonathan Rihan, Srikumar Ramalingam, Carlos Orrite and Philip H.S.Torr, “Randomized Trees for Human Pose Detection,” Proc. Computer Vision and Pattern Recognition (CVPR), pp. 1-8, 2008.

[39] Xiaoyu Wang, Tony Han and Shuicheng Yan, “An HOG-LBP Human Detector with Partial Occlusion Handling,” Proc. Int'l Conf. on Computer Vision (ICCV), pp. 32-39, 2009.

[40] Ivab Laptev, “Improving Object Detection with Boosted Histograms,” Proc. Image and Vision Computing Archive, Vol. 27, Issue 5, pp. 535-544, 2009.

[41] Yi Yang, and Deva Ramanan, “Articulated Pose Estimation using Flexible Mixtures of Parts,” Proc. Computer Vision and Pattern Recognition (CVPR), 2011.

[42] A.Collet, D.Berenson, S.S.Srinivasa and D.Ferguson, “Object Recognition and Full Pose Registration from A Single Image For Robotic Manipulation,” Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA), pp. 48-55, 2009.

[43] Martin A.Fischler and Robert C.Bolles, “Random Sample Consensus: A Paradigm for Model Fitting with Application to Image Analysis and Automated Catrography,” Proc. Communications of the ACM, Vol. 24, Issue 6, pp. 381-395, 1981.

[44] Comaniciu Dorin and Peter Meer, “Mean Shift: A Robust Approach Toward Feature Space Analysis,” IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI), Vol. 24, pp. 603-609, 2002.

[45] David Nistér and Henrik Stewénius, “Scalable Recognition with A Vocabulary Tree,” Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 2161-2168, 2006.

[46] Bo Wu, Haizhou Ai, Chang Huang and Shihong Lao, “Fast Rotation Invariant Multi-View Face Detection Based on Real Adaboost,” Proc. IEEE Int'l Conf. on Automatic Face and Gesture Recognition (FGR), pp. 79-84, 2004.

[47] Robert E. Schapire, and Yoram Singer, “Improved Boosting Algorithms using Confidence-Rated Predictions,” Machine Learning, Vol. 37, pp. 297-336, 1999.

[48] Jeffrey S. Beis and David G. Lowe, “Shape Indexing using Approximate Nearest-Neighbor Search in High-Dimensional Spaces,” Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1000-1006, 1997.

[49] Jerome H.Friedmanm, John Louis Bentley and Raphael Ari Finkel, “An Algorithm for Finding Best Matches in Logarithmic Expected Time,” ACM Trans. on Mathematical Software (TOMS), No. 3, pp. 209-226, 1977.

[50] Jean-Michel Morel and Guoshen Yu, “ASIFT: A New Framework for Fully Affine Invariant Image Comparison,” SIAM Journal on Imaging Sciences, Vol. 2, No. 2, pp. 438-469, 2009.

[51] Guoshen Yu and Jean-Michiel Morel, “A Fully Affine Invariant Image Comparison Method,” Proc. IEEE Int'l Conf. on Acoustics, Speech, and Signal Processing (ICASSP), pp. 1597-1600, 2009.

[52] Vandergheynst Pierre, Raphael Ortiz, and Alexandre Alahi, “FREAK: Fast Retina Keypoint,” Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 510-517, 2012.

[53] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski, “ORB: an Efficient Alternative to SIFT or SURF,” Proc. Int'l Conf. on Computer Vision (ICCV), pp. 2564-2571, 2011.

[54] Stefan Leutenegger, Margarita Chli, and Roland Siegwart, “BRISK: Binary Robust Invariant Scalable Keypoints,” IEEE Int'l Conf. on Computer Vision (ICCV), pp. 2548-2555, 2011.

[55] Mitsuru Ambai and Yuichi Yoshida, “CARD: Compact and Real-time Descriptors,” Proc. IEEE Int'l Conf. on Computer Vision (ICCV), pp. 97-104, 2011.

[56] Jesus M.Rincon, Dimitrios Makris, Carlos O.Uruñuela, and Jean-Christophe Nebel, “Tracking Human Position and Lower Body Parts Using Kalman and Particle Filters Constrained by Human Biomechanics,” IEEE Trans. on Systems, Man, and Cybernetics, Vol. 41, Issue 1, pp. 26-37, 2011.

[57] Jamie Shotton, Andrew Fitxgibbon, and Mat Cook, “Real-time Human Pose Recognition in Parts from Single Depth Images,” Proc. IEEE Int'l. Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1297-1304, 2011.

[58] Raquel Urtasun, David J Fleet, and Pascal Fua, “Monocular 3-D Tracking of the Golf Swing,” Proc. IEEE Int'l. Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 932-938, 2005.

[59] Vincent Lepetit, Ali Shahrokni, and Pascal Fua, “Robust Data Association For Online Applications,” Proc. IEEE Int'l. Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 281-288, 2003.

[60] Allan D.Jepson, David J.Fleet, and Thomas F.El-Maraghi, “Robust Online Appearance Models for Visual Tracking,” IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI), Vol. 25, No. 10, pp. 1296-1311, 2003.

[61] Yutaka Usui and Katsuya Kondo, “The Golf Swing Tracking Method by using Combination of Pose Estimation and Local Features,” Proc. Int'l Technical Conf. on Circuit/Systems, Computers and Communications (ITC-CSCC), CD-ROM, F-T2-03, 2012.

[62] Yutaka Usui and Katsuya Kondo, “Tracking of a Golf Swing by using Particle Filter with Silhouette Features and Local Features,” Trans. of the Society of Instrument and Control Engineers (SICE), No. 49, Vol. 4, pp. 440-448, 2013. (in Japanese).

[63] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele, “Pictorial Structures Revisited: People Detection and Articulated Pose Estimation,” Proc. IEEE Int'l. Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1014-1021, 2009.

[64] Genshiro Kitagawa, “Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models,” Journal of Computational and Graphical Statistics, Vol. 5, No. 1, pp. 1-25, 1996.

[65] Michael Isard and Andrew Blake, “CONDENSATION - Conditional Density Propagation for Visual Tracking,” Proc. Int'l Journal of Computer Vision (IJCV), Vol. 29, No. 1, pp. 5-29, 1998.

[66] Shaohua Zhou, Rama Chellappa, and Babacj Moghaddam, “Visual Tracking and Recognition using Appearance-Adaptive Models in Particle Filters,” IEEE Trans. on Image Processing, No.13. pp. 1491-1506, 2004.

[67] Yuan Li, Haizhou Ai, Takayoshi Yamashita, Shihong Lao, and Masato Kawade, “Tracking in Low Frame Rate Video: A Cascade Particle Filter with Discriminative Observers of Different Life Spans,” IEEE Trans. on, Pattern Analysis and Machine Intelligence (PAMI), No. 30, Vol. 10, pp. 1728-1740, 2008.

[68] Dan Mikami, Kazuhiro Otsuka, and Junji Yamato, “Memory-based Particle Filter for Face Pose Tracking Robust under Complex Dynamics,” Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 999-1006, 2009.

[69] Katja Nummiaro, Esther Koller-Meier, Luc Van Gool, “An Adaptive Color-based Particle Filter,” Image Vision Computing, Elsevier, No. 21, pp. 99-110, 2003.

[70] Rob Hess and Alan Fern, “Discriminatively Trained Particle Filters for Complex Multi-object Tracking,” Proc. IEEE Int'l. Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 240-247, 2009.

[71] Ankur Agarwal and Bill Triggs, “Recovering 3D Human Pose From Monocular Images,” IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI), Vol. 28, pp. 44-58, 2006.

[72] Carlo Tomasi, “Bilateral Filtering for Gray and Color Images,” Proc. IEEE Int'l. Conf. on Computer Vision (ICCV), pp. 839-846, 1998.

[73] Ben Weiss, “Fast Median and Bilateral Filtering,” Proc. of ACM SIGGRAPH, Vol. 25, Issue 3, pp. 519-526, 2006.

[74] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake, “GrabCut: Interactive Foreground Extraction Using Iterated Graph Cuts,” ACM Trans. on Graphics, Vol. 23, pp. 309-314, 2004.

[75] Pedro F.Felzenszwalb and Daniel Huttenlocher, “Pictorial Structures for Object Recognition,” Proc. IEEE Int'l. Journal of Computer Vision (IJCV), Vol. 61, pp. 55-79, 2005.

[76] Pedro F.Felzenszwalb and Daniel Huttenlocher, “Efficient Matching of Pictorial Structures”, IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). Vol. 2, pp. 66-73, 2000.

[77] Qiang Zhu, Shai Avidan, Mei-Chen Yeh, and Kwang-Ting Cheng, “Fast Human Detection using a Cascade of Histograms of Oriented Gradients,” Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Vol. 2, pp. 1491-1498, 2006.

[78] N.Gordon, D.Salmond, and A.Smith, “Novel Approach to Nonlinear/non-Gaussian Bayesian State Estimation, Radar and Signal Processing,” IEE Proc. F, Vol. 140, pp. 107-113, 1993.

[79] Rolf G.Kuehni , “Color Space and Its Divisions: Color Order from Antiquity to the Present,” New York: Wiley, 2003.

[80] Golf　Digest Online, http://news.golfdigest.co.jp/tournament/players/swing.htm

Acknowledgements

First, I would want to express my deepest thanks to my academic supervisor, Professor Katsuya Kondo. I am grateful for his continuous motivation, encouragement and support of my Ph.D research. This thesis would not have been completed without his commitment.

In addition to my academic supervisor, I would like to thank the rest of my thesis committee: Professor Yoshio Itoh and Professor Shigang Li, for their encouragement, insightful comments and questions.

I am also grateful to Mr. Mitsuji Yoshida, president of Raytron, who gave me the chance to study at the graduate school of Tottori University and supported this study.

Furthermore, I would also like to acknowledge the students in the system design laboratory, Mr. Kensaku Kumada and Mr. Takuya Okamoto, for their support of my study.

Most of all, I'd like to thank my parents for their support.

学位論文の章と研究業績の対応

1. 学術雑誌発表論文

	著者・論文題目・発表機関	本文
1	臼井温,近藤克哉, “シルエット特徴と局所特徴を用いたパーティクルフィルタによるゴルフスイング軌跡推定” 計測自動制御学会論文誌, 第 49 巻,4 号, pp. 440-448, 2012.	第 3 章

2. 国際会議発表論文

	著者・論文題目・発表機関	本文
1	Yutaka Usui and Katsuya Kondo, “The SIFT Image Feature Reduction Method using the Histogram Intersection Kernel,” Proceedings of International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS2009), CD-ROM, pp. 517-520, 2009.	第 2 章
2	Yutaka Usui and Katsuya Kondo, “3D Object Recognition Based on Confidence LUT of SIFT Feature Distance,” Proceedings of the World Congress on Nature and Biologically Inspired Computing (NaBIC2010), CD-ROM, pp. 300-304, 2010.	第 2 章
3	Yutaka Usui and Katsuya Kondo, “The Golf Swing Tracking Method by using Combination of Pose Estimation and Local Features,” Proceedings of International Technical Conf. on Circuit/Systems, Computers and Communications (ITC-CSCC2012) , CD-ROM, F-T2-043, 2012.	第 3 章

研究業績

1. 学術雑誌発表論文(査読付き)

臼井温,近藤克哉,"シルエット特徴と局所特徴を用いたパーティクルフィルタによるゴルフスイング軌跡推定",計測自動制御学会論文誌, 第 49 巻, 4 号, pp. 440-448, 2012.

2. 国際会議発表論文(査読付き)

Yutaka Usui and Katsuya Kondo, “The SIFT Image Feature Reduction Method using the Histogram Intersection Kernel”, Proc. of Int'l Symposium on Intelligent Signal Processing and Communication Systems(ISPACS2009), CD-ROM, pp. 517-520,2009.

Yutaka Usui and Katsuya Kondo, “3D Object Recognition Based on Confidence LUT of SIFT Feature Distance”, Proc. of the World Congress on Nature and Biologically Inspired Computing (NaBIC2010), CD-ROM, pp. 300-304, 2010.

Yutaka Usui and Katsuya Kondo, “Hardware Implementation of Adaboost-based Sign Recognition for Miniature Robot”, Proc. of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference(APSIPA-ASC2009), CD-ROM, pp. 490-493, 2009.

Yutaka Usui and Katsuya Kondo, “The Golf Swing Tracking Method by Using Combination of Pose Estimation and Local Features”, Proc. of Int'l Technical Conf. on Circuit/Systems, Computers and Communications(ITC-CSCC2012), CD-ROM, F-T2-03, 2012.

3. 研究会等

臼井 温, 近藤 克哉,"小型組み込み用途 Adaboost を用いた標識認識 FPGA の設計", 電子情報通信学会技術研究報告, Vol 108, No. 334, pp. 157-160, 2008.

臼井 温,宮崎善行, 近藤 克哉,"形状特徴照合による家電種類画像認識システム", 電子情報通信学会 2009 年総合大会講演論文, CD-ROM, AS-2-2, 2009.

臼井 温, 近藤 克哉,"FPGA を用いた小型自律ロボット向けビジョンセンサシステムの設計", 第 11 回 DSPS 教育者会議予稿集, pp. 81-82, 2009.

臼井 温, 近藤 克哉,"LAB を用いた小規模ハードウェア向け顔認識手法の検討", 電子情報通信学会技術研究報告 Vol. 109, No. 447, pp. 49-52, 2010.

熊田 健作, 臼井 温, 近藤克哉, "Kinect センサとパーティクルフィルタによるゴルフスイング時のグリップ追跡", 電子情報通信学会技術研究報告", Vol. 112, No. 78, pp. 65-70, 2012.