Title	Informative Sequential Patch Selection for Image Retrieval
Author(s)	申，志豪
Citation	
Issue Date	2017-09
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/14796
Rights	
Description	Supervisor:丁　洛榮，情報科学研究科，修士

Informative Sequential Patch Selection for Image Retrieval

Shen Zhihao

School of Information Science
Japan Advanced Institute of Science and Technology
September, 2017

Master's Thesis

Informative Sequential Patch Selection for Image Retrieval

1510208 Shen Zhihao

Supervisor: Professor Nak-Young Chong

Main Examiner: Professor Nak-Young Chong

Examiners: Professor Kazunori Kotani

Associate Professor Atsuo Yoshitaka

School of Information Science

Japan Advanced Institute of Science and Technology

August, 2017

Informative Sequential Patch Selection for Image Retrieval

SHEN Zhihao (1510208)

School of Information Science

Japan Advanced Institute of Science and Technology

Abstract —To quickly and efficiently analyze a large-scale environment by the camera with limited field-of-view, intelligent systems should sequentially select the optimal field-of-view to observe important and informative parts of the area. Especially in the image retrieval tasks, small observations could be sequentially selected to improve the performance of image retrieval with less computational costs than whole observations at once and the enhanced retrieval performance could be used to select the next best-view again in a cyclic process. In this paper, we have investigated the effects of selected image patches, which might be either overlapped with a certain ratio or on-overlapped with previous observations, in this cyclic process. The adaptive patch selection algorithm is also described as follows: (1) A current observation is decided by its own information gain model which is designed by a similarity value between current observed information and training data set. (2) After then, the system will update the information gain model by discarding the irrelevant training data with the current observation. During this process, we have shown that an informative patch, even though a part of selected patch is already observed at previous steps, can enhance the retrieval accuracy and it has a better performance than an independent observation method. Experimental results also have shown that the model selects the informative patches around the important contents to retrieve the target images such as the sky, building and so on.

Key Words — Image retrieval, Small field-of-view, the Next best view, Sequential Selection, Visual attention

Contents

1 Introduction

1.1 Background

Nowadays, intelligent robots are applied to a perform variety of tasks such as: searching, exploring or rescue tasks (shows in Fig. 1.1-1.3), and they make decisions and adjust their actions on the basis of information captured from the different environments. However, the robots need to perform their tasks always with some limitations and restrictions, such as operating time, battery capacity, and/or limited sensing coverage.

Fig. 1.1 Searching Task

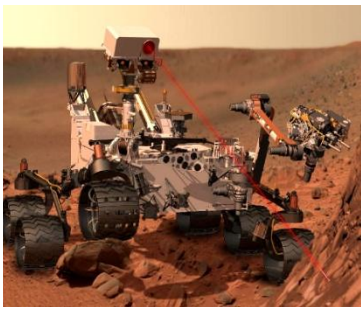


Fig. 1.2 Exploring Task

Fig. 1.3 Rescue Task

There are some researches that have been done for solving robot navigation problems with limited time or battery capacity [1] [2]. The studies, such as viewpoint planning [3] (shows in Fig. 1.4) and saliency-based visual attention [4] (shows in Fig. 1.5), are good method to select an informative view point, nonetheless, the proposed researches are assuming fully access to the target image at once. On the other hand, it takes high computational cost, if the robots try to analyze a large-scale environment at once. Such large amount of information is superfluous and can easily lead to a significant increasing of time and space requirements for processing the data [5].

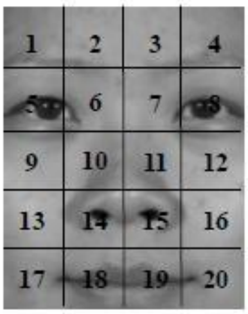


Fig. 1.4 Viewpoint planning

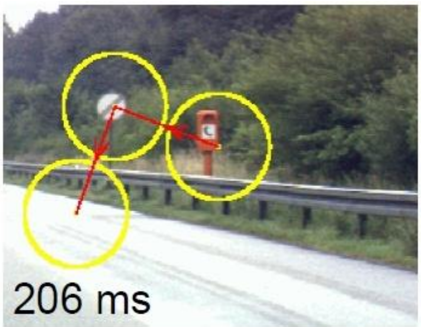


Fig. 1.5 Saliency-based Visual Attention

For avoiding processing such high-dimensional data, the robots are equipped with a small field-of-view camera and capture small image patches from large-scale environment sequentially. Therefore, the robots can access to the target environment by sequentially selecting view images depending on the size of small field-of-view camera. On the other hand, with limited information input, it is difficult for the robots to recognize a large environment. This is why the robots need to decide where to take the image patch at each time. This problem can be explained with the human visual perception that how human decided where to look [6]. Assuming that there is a large picture that is overlaid with small tiled shutters, and humans can uncover one shutter at each time. Humans will decide the view point based on their knowledge that where can provide important information to recognize the image. And then, they compare the partial information to their knowledge and remove the target irrelevant memory for the next best view point selection, Fig. 1.6. The concept that a robot with limited sensing coverage sequentially selects observation from a large-scale environment is similar with human eye movement when human move their eyes purposefully to gain the sensory information.

Usually, we move our eyes purposefully to gain the sensory information to complete a task. It is important that how to make our eyes focus on some attentions. If we want to figure out where are, especially when our eyes with the limited field of vision, it may not be easy to find out only with the initial point of regard. Then we choose the next attention point until the environment can be recognized. We can also use the way of human eye movement to robot navigation problems.

Fig. 1.6 Memory based view point selection

Many robotic path planning applications can be applied to different uncertain environments. Such as, searching and rescuing the survivors from an unknown location of the disaster struck environment. With a limited number of constraints, robots are available for monitoring the environments, so it is essential to make path planning to the next most "informative" location. Assuming that the robot with small field-of-view camera for a large-scale environment classification. The robot can obtain the area information partially at each step by selecting proper fixations.

The above issue that robots with small field-of-view sequentially select view point and manage the attention path to recognize a large environment can be solved as sequential feature selection problem [7], such as efficient sequential feature selection

[5], adaptive floating search methods [8], and L2,1-Norm regularized discriminative feature selection [9].

1.2 Limitation of the Attention-Path Planning Algorithm

Attention-path planning algorithm [10] was proposed to enable the robot to sequentially access to a part of the target environment with a limited field of view (the concept of attention path planning as shown in Fig. 1.6). It is similar to the human visual perception [11] that visual stimulus information combines with prior knowledge and task goals to plan an eye movement. The Attention-path planning algorithm mainly contains two components: (1) observation selection based on the informativeness of each fixation. (2) at each step, the robot selects an informative patch from the target environment based on its prior knowledge and updates its prior-knowledge based on currently selected observations. With this cyclic process, a robot can classify the target environment without accessing the whole environment. However, each image patch that the robot observed from the target environment is fixed in specific position and fully independent of each other.

In practice, some informative features can be omitted or only observed once with the independently defined patches. It is necessary to place the patches adaptively to enhance the retrieval performance. In this research, we proposed a method by considering the partially overlapped patches to select observation more adaptively. All blocks are partially overlapped by their neighbors. And the size of the block images is the same case of the small field-of-view camera on the robot. All image data are represented by feature vectors. The fixation selection means to select a crucial position on the target environment depending on the whole prior knowledge. It is an arduous work to manually labeling a large set of training data. We try to solve this problem in the unsupervised scenarios. The fixation is selected based on the concept that best preserve the informativeness of the fixation derived from the whole prior knowledge. A simple greedy algorithm is used to guarantee that the selected fixation is the most informative position under the current state.

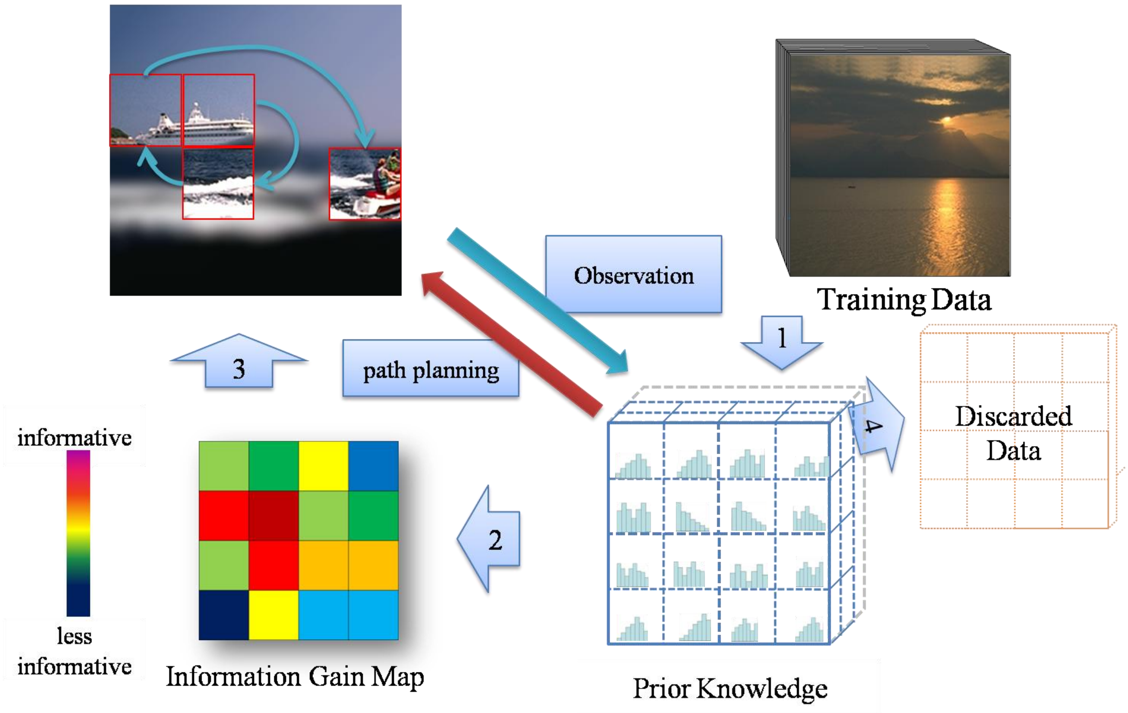


Fig. 1.7 The concept of attention path planning: 1. Dataset is divided by several small patches and each patch is represented by the local feature vector that stores in a memory as a prior knowledge, 2. Informativeness of each patch, 3. Best patch selection, and 4. Prior knowledge update by discarding target-irrelevant training data set. From steps 2 to 4 are repeated to correctly retrieve the target image.

2 Problem Statement

(a) (b)

Fig. 2.1 Image patch selection with different methods. (a) block images without overlap, (b) block images with overlap.

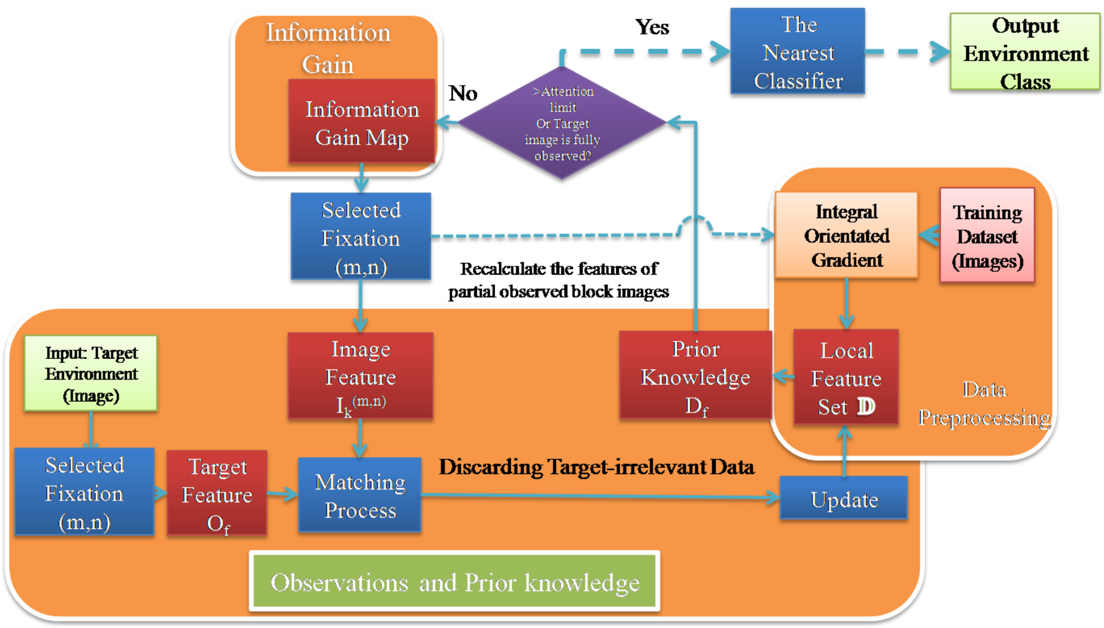


Fig. 2.2 Overall architecture of sequential patch selection for image retrieval

In the attention-path planning algorithm, all images are divided into non-overlapped blocks. It becomes less informative when a valuable feature is divided into different patches, which can be seen in Fig. 2.1(a). Therefore, in this research, we propose that the target image is divided into rectangular blocks according to the coverage of the robot's camera and each block is 50% overlapped by their neighbors, which can be seen in Fig. 2.1(b).

Hence, there are more observation selection options than the non-overlapped condition. The framework of the adaptive observation selection is presented in Fig. 2.2. There are three components of the framework are illuminated in the following.

2.1 Data Preprocessing

Fig. 2.3 ***LabelMe* (*urban and natural scene categories*)**: (Class1) coast/beach, (Class2) open country, (Class3) forest, (Class4) mountain, (Class5) highway, (Class6) street, (Class7) city center, and (Class8) tall building.

The number of training dataset [12] is 2080 (size: 256×256 pixels) and the dataset is categorized into 8 classes (8 classes $\times 260$ images). We assume that the field-of-view of the camera is limited to 64×64 pixels. We apply the partially overlapped partition method: one image is divided by several parts of patches from left most top position to right most bottom position with the flexible size of the window (field of view of robot's camera) $a \times b$ pixels. Then, the window (view point) is sliding with a certain interval:

$$a \times (1 - overlap\_rate\ )$$

or/ and

$$b \times (1 - overlap\_rate)$$

along the horizontal and vertical image axes.

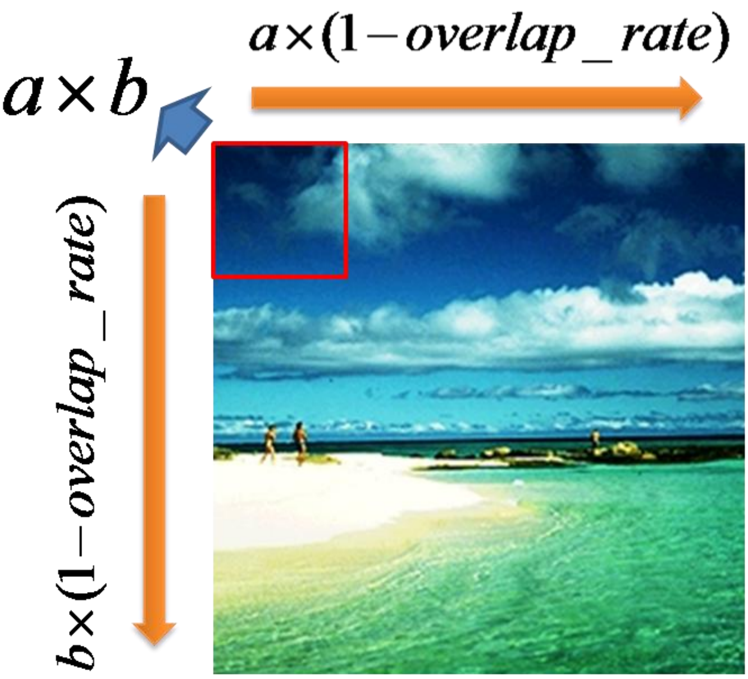


Fig. 2.4 How to extract image patch from image

The overlapping ratio and the window size are defined by us. In the following experiment, we set different overlapping ratio with fixed window size. The attached figures are considering the overlapping ratio is 0, which means non-overlap.

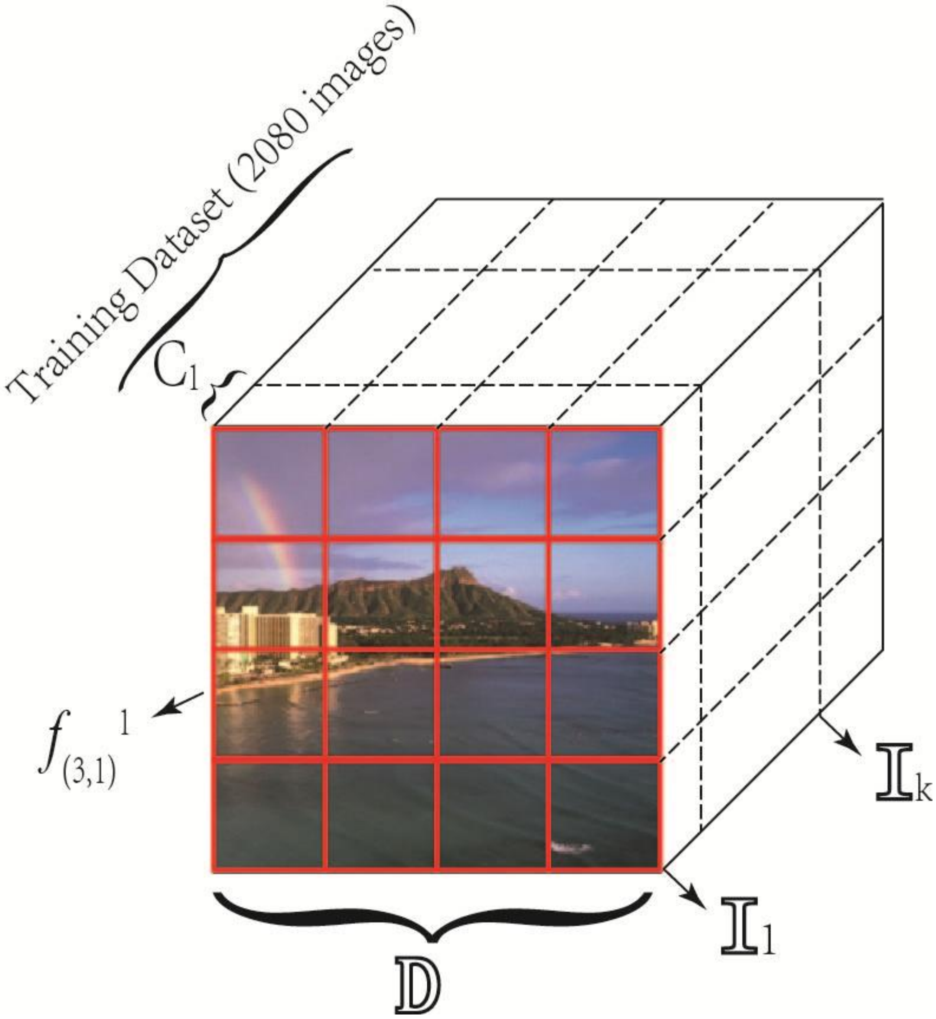


Fig. 2.5 Training dataset structure: ($\mathbb{D}$) the local feature set, (C_1) class 1, ($\mathbb{I}_k$) local feature set of the *k*-th image, ($f_{(m,n)}^{k}$): the local feature; superscript (^{k}): the image ID number; subscript ($_{m,n}$): position in the *m-th* row and the *n-th* column

All images which are used as training data categorized into the given training class set, $\mathbb{C} = \{c_1, c_2, c_3, \cdots\}$. We extract the local features of the image patches by dividing images with overlap. There is a training data set of all images $\mathbb{D} = \{\mathbb{I}_1, \mathbb{I}_2,$

$\cdots$, $\mathbb{I}_n\}$ and every image contains local features such as $\mathbb{I}_k = \{ f^{k}_{(1,1)}, \dots, f^{k}_{(r,c)}\}$, where all images are divided into r row and c column blocks. $f_{(m,\, n)}$ represents the feature of the block image in the position *(m, n)* and the fixation is represented with a coordinate *(m, n)*. The prior knowledge will be updated after each selection by discarding a number of target-irrelevant images for the next selection. The structure of the training dataset is shown in Fig. 2.5.

2.2 Observations and Prior Knowledge

If the fixation is decided, then, the local feature $f_k^{(m,n)}$ of the image patch in the position *(m, n)* will be combined with the feature set

$$I_k^{(m,n)} = \left[I_k, f_k^{(m,n)}\right] \text{ ------------------------------ } (1)$$

Note that I_k is the feature vector of the k-th image in the training dataset and I_k is generated by combining the features of the block images of the positions that have been selected already, it can be seen in Fig. 2.7.

$D_f = \{\dots, I_k\}$ is considered as the prior knowledge, and it contains the images that are related to the target image. While the observation position is decided, the target feature in the position *(m, n)* is denoted $o_{(m,n)}$. The target feature set

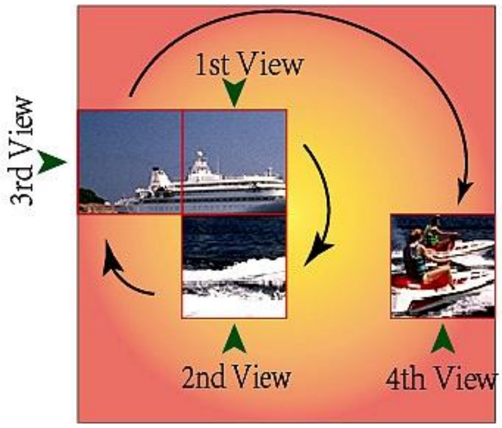
$$O_{l+1}^{(m,n)} = \left[O_l, o_{(m,n)}\right] \text{ ------------------------------ } (2)$$

combines all local features that are sequentially observed from the target image, shown in Fig. 2.6.

The prior knowledge will be updated by discarding target irrelevant images according to similarity value between the observed image patches and the prior knowledge, namely, a small similarity value is considered that the images of the prior knowledge are irrelevant to the target image. The similarity value is calculated by using Cosine Similarity (CS) [13], shown in the following:

$$Similarity = CS(O_l, I_k) = \frac{O_l \cdot I_k^T}{\|O_l\| \times \|I_k\|} \quad \text{--------------------------- (3)}$$

Where O_l is the feature vector of the target image that is observed with small image patches l-times. And I_k is the feature vector of the k-th image of the prior knowledge.



1st view: $O_1^{(2.2)} = [o_{(2.2)}];$

2nd view: $O_2^{(3,2)} = [O_1^{(2.2)}, o_{(3,2)}];$

3rd view: $O_3^{(2,1)} = [O_2^{(3.2)}, o_{(2,1)}];$

4th view: $O_4^{(3,4)} = [O_3^{(2.1)}, o_{(3,4)}];$

Fig. 2.6 To sequentially select observations from target image

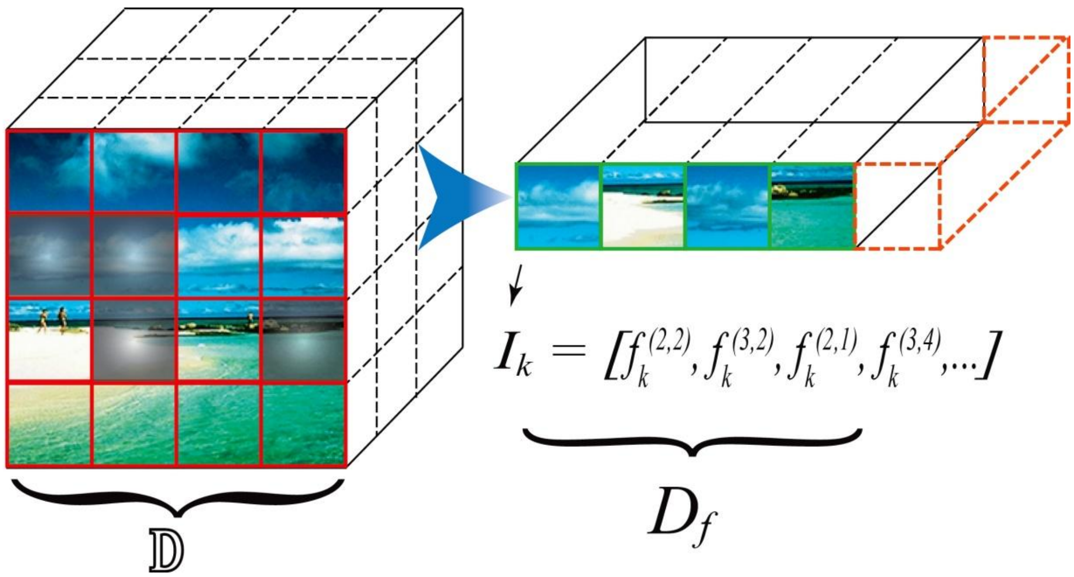


Fig. 2.7 Structure of the prior knowledge

In Fig. 2.7, the gray block means that the image patch is observed. And the image feature of the prior knowledge is generated by combining all feature vectors of image patches of the training dataset, in where it has been selected as a view position.

2.3 Information Gain

In the observation selection processing, the system selects the most informative fixation to make an observation from the target image. The informative fixation is measured by calculating the dissimilarity value across the prior knowledge. It is inspired by human visual perception. In humans cognitive concept, they have to make a quick decision with limited information from the environment, that is why they bring back their memory and decided their view position based on the concept that the position shows the significant dissimilarity across the whole memory.

The function to calculate similarity is presented in the previous section. The CS is the abbreviation of cosine similarity, therefore, the dissimilarity is defined as:

$$Dissimilarity = 1 - CS(A, B) \text{ ------------------------------ } (4)$$

We make an information gain map to measure a fixation is informative or not. The information gain of each fixation can be calculated as:

$$I_k^{(m,n)} = \left[I_k, f_{(m,n)}^k\right] \Longrightarrow I_k^{(m,n)} \in D_{f_{(m,n)}} \text{ -------------------- } (5)$$

$$G_{(m,n)} = E\left[1 - CS(\omega, D_{f_{(m,n)}})\right] \text{ ------------------------------ } (6)$$

At each iteration, the information gain is the average of the dissimilarity between ω and all image features. ω is the mean feature vector of all of the image features $I_k^{(m,n)}$ of the prior knowledge. So, ω can be calculated like:

$$\omega = E\left[D_{f_{(m,n)}}\right] \text{ ------------------------------ } (7)$$

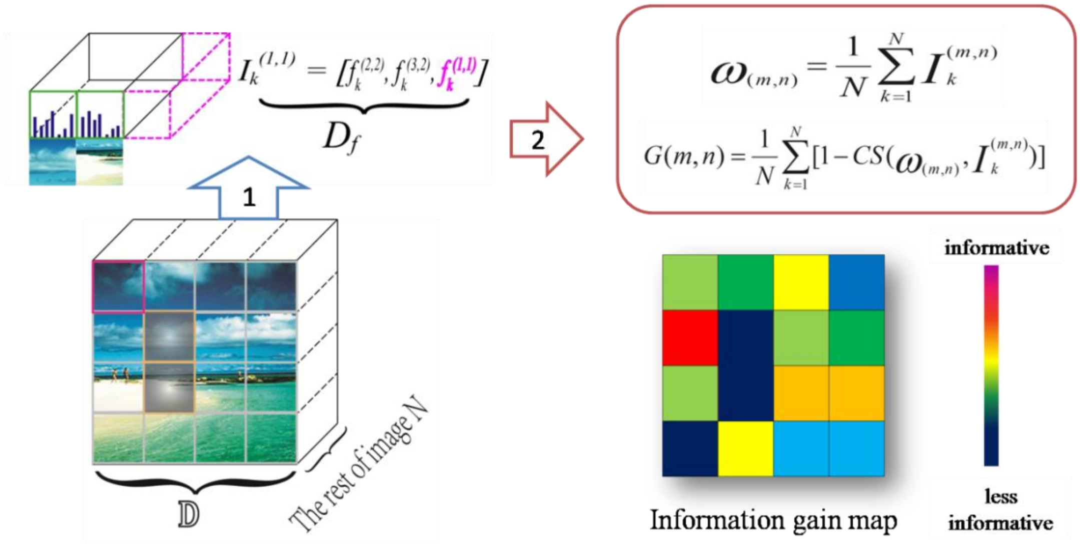


Fig. 2.8 Procedure of generating an information gain map

The Fig. 2.8 shows the procedure of generating an information gain map. The system already selected two fixations; the third fixation will be selected depending on the information gain value of each fixation. First of all, the system will calculate the information gain in the first row and the first column where is marked by the pink block. The new image feature will be generated by combining the local feature in the first row and the first column to the current image feature. Secondly, the mean feature ω will be calculated. The information gain is the average of the dissimilarity between ω and all image features. At last, by using such process, the information gain map will be generated, and the next fixation will be selected according to the map that the fixation shows the most informative.

The information gain map will be generated at each iteration by updating the prior knowledge. In the final experiment result, it also shows that the less informative fixations do not need to be observed, and the system can achieve a good performance by only using part of the information from the target image.

2.4 Fixed Field-of-View

While we applied partially partition method, there are two different kinds of way to extract observation from target environment, fixed field of view and variable field of view.

In this work, we assume that the robot can observe a partial area in fixed size. The target image is partially accessed at every time steps by observing the most informative fixation. If the overlapped area contains important image content, usually we need to keep such important features within the current observation. Therefore, some areas of the target image can be observed several times. As shown in Fig. 2.9, supposing that the block image in the green box is the selected as the best view point and the areas within the orange box are partially overlapped with its neighbors. If the next observation is selected among these eight block images, some partial information will be used again in the following observations.

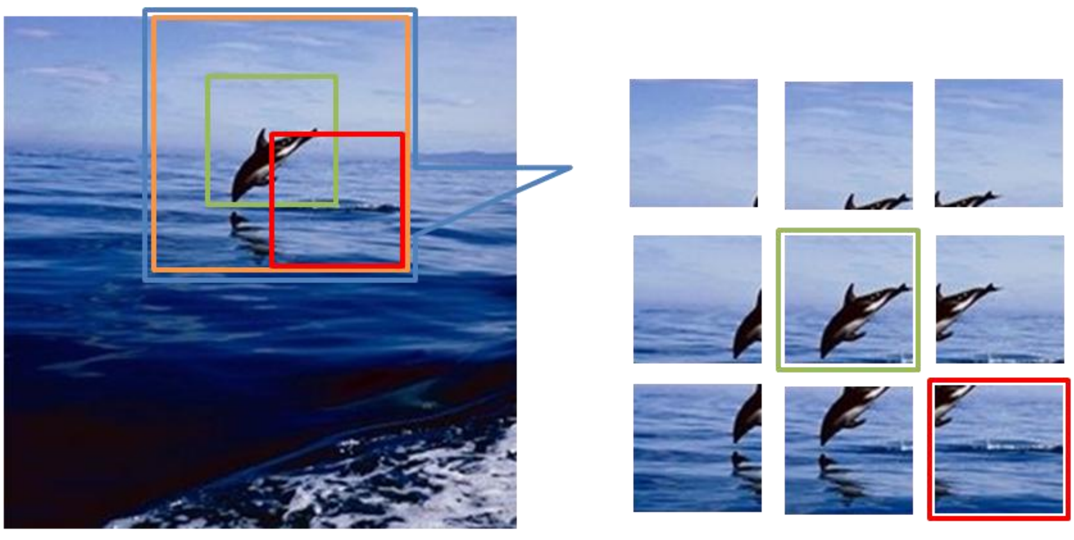


Fig. 2.9. Fixed field of view

The Fig. 2.9 shows the first observation in the green block is partially overlapped by the second observation that in the red block. In this condition, the system makes observations that sometimes partially overlap previous observations.

2.5 Variable Field-of-View

In the fixed field of view condition, the system makes observations that sometimes partially overlap previous observations. In this case, the partial area will be removed for the next observation selection.

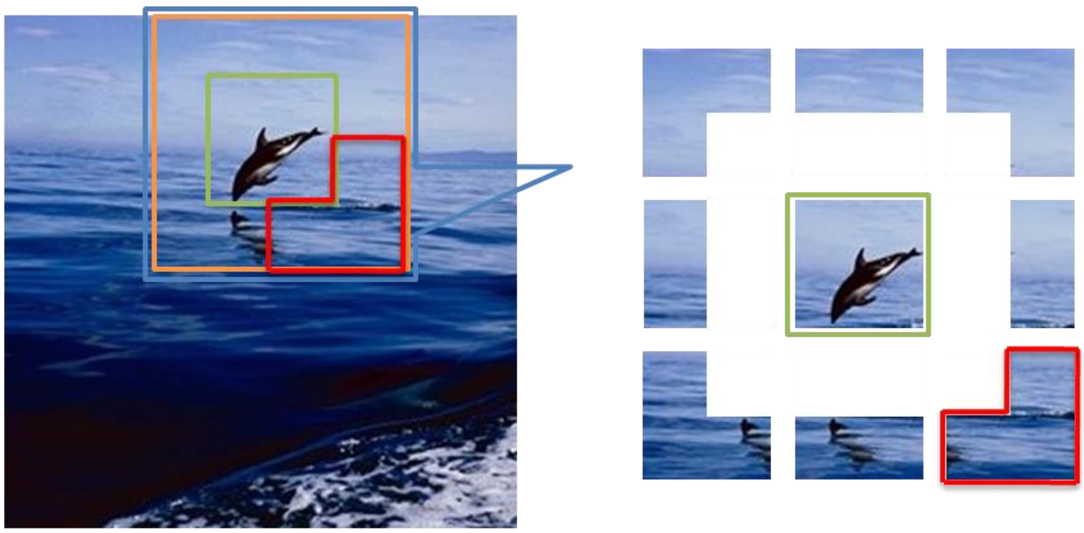


Fig. 2.10 Variable field of view

To reduce the feature vectors from a previously observed area, a previously observed area could be discarded and the rest of area is only used to calculate the feature vectors. As shown in Fig. 2.10, the green block is selected as informative image patch and it is used to extract the feature vectors. After then, the partial area of the block images that was not observed, like the block image in the blue concave polygon, need to recalculate its feature vector. So the training data is updated in two aspects: discarding dissimilar images from training data and recalculating the feature vectors of the block images that are partially observed in the previous step. There is non-overlapped area among the observations that are selected at each iteration. The observations that are selected at each iteration are independent of each other. The whole framework of this environment classification method with variable field of view is described in Fig. 2.2.

3 Experimental Verification

3.1 The Experimental Information

Dataset:

LabelMe: urban and natural scene categories [12]

Contains:

2080 images, 8 classes (8 classes $\times$260 images)

Image size: 256$\times$256 pixels

Window size: 64 $\times$64 pixels

Evaluation Method

10-fold cross-validation

Experiment Parameters

30 images remain at last

Overlapping ratio: 0%, 25%, 50%, and 75%

3.2 Program Design

3.2.1 Definition of *k*

Based on the parameters that we set above, the system updates the prior knowledge in each iteration; finally, there will be 30 images that remain in the prior knowledge. The rest of images are used as the nearest classifier [14].

Each time, the system needs to decide how many images have to be removed. In non-overlap case, all observations are independent to each other. It is easy to decide to remove a constant number of images in non-overlap case. However, it can not be a constant number when the system is processing the partially overlapped condition.

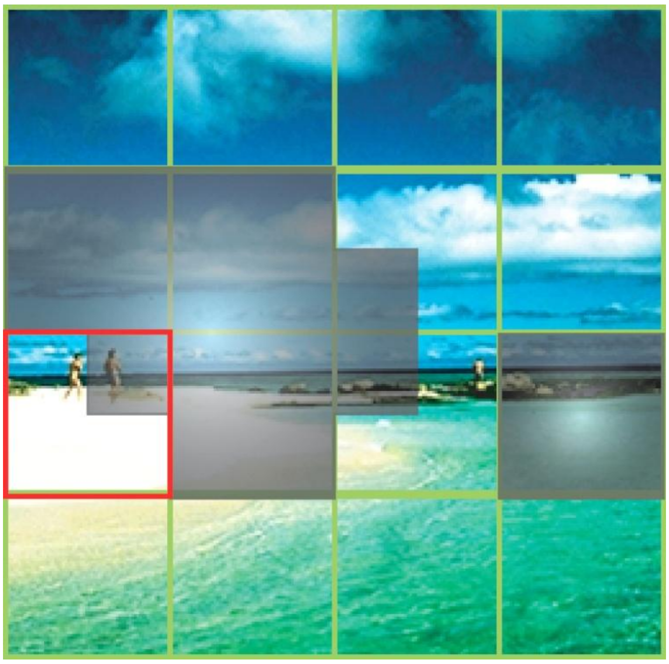


Fig. 3.1 The observation from target environment

Here is one way to solve this problem, in the Fig. 3.1, the gray area has been observed in the previous step. And the red block shows the current observation. It is obvious that the current observation partially overlaps previous observation.

- The number of images that are remaining in the prior knowledge at last is ***30***;
- The number of images that are remaining in the prior knowledge at the current step is ***N***;
- The limits we set is ***l***;
- ***S*** is how many times the system makes observations from environment so far;
- The current observation rate is $\boldsymbol{C_{rate}}$;
- The $\boldsymbol{rest_{rate}}$ is the area that was not observed yet.

Therefore, based on the above information, ***k*** images will be removed for updating the prior knowledge. ***k*** can be calculated as:

$$If\ \left(\frac{rest_{rate}}{C_{rate}}\right) > (l - S)$$

$$k = \frac{(N-30)}{l-S} \text{ -- } (8)$$

$$else$$

$$\mathrm{k} = (\mathrm{N} - 30) \times \frac{\mathrm{C_{rate}}}{\mathrm{rest_{\,rate}}} \text{ ---------------------------------- } (9)$$

Based on this method, the system can adjust how many images need to be discarded automatically, and guarantees there will be 30 images remaining at last.

3.2.2 Local Feature Extraction

It is important to find a good feature detection method to represent the partial image that observed from the target environment and all block images in the training dataset. There are many feature detection methods bases on visual features such as edges, contours, corners, and regions for solving the problem in computer vision and image processing [15]. A high-performance feature detector should show robustness to changing image conditions. In this research, we aim to classify the natural images, one of whose primitive feature is edges. Histogram of orientation gradient (HOG) [16] is a well-known feature descriptor which can be used to extract local features from block images. Initially, each block image will be calculated HOG feature vector. Then, the feature vector is saved in the prior knowledge. Based on the current prior knowledge, the most informative position which means the features in this position best presents the data diversity across the whole prior knowledge is selected as the observation fixation.

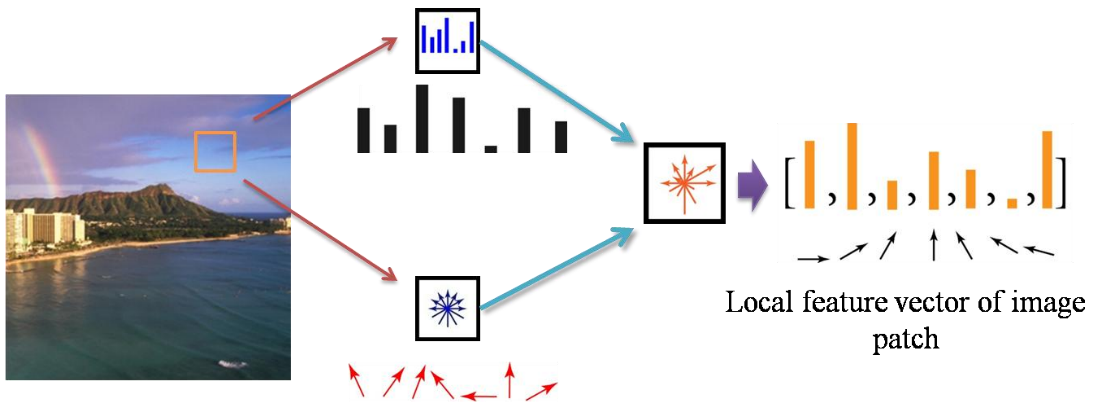


Fig. 3.2. Extracting the local feature from small image patch

The data preprocessing is illustrated in Fig. 3.2. We can see from above, firstly, the training data are all images. We divide each image into small image patches that are same as the small field-of-view camera. We calculate the feature vector of each image patch by HOG calculator. Note that the feature vector sometimes needs to be recalculated in the variable field-of-view case. The image patches are calculated by the HOG feature descriptor, the program is attached in Appendix 1.

We finished the first experiment in non-overlap case, 50% overlap case in fixed field of view, and 50% overlap case in variable field of view. The local feature is extracted by using the method that is described above.

We use the retrieval accuracy to indicate the performances of different approaches are good or not. As shown in Fig. 3.3, one folder is used as test data set, and the rest of folders are used as the prior knowledge. If the test data belongs to class 2, and the system makes the decision that "the target image belongs to class 2", the decision is correct. If the system makes the decision that the target image does not belong to class 2, which means the test image is not correctly classified.

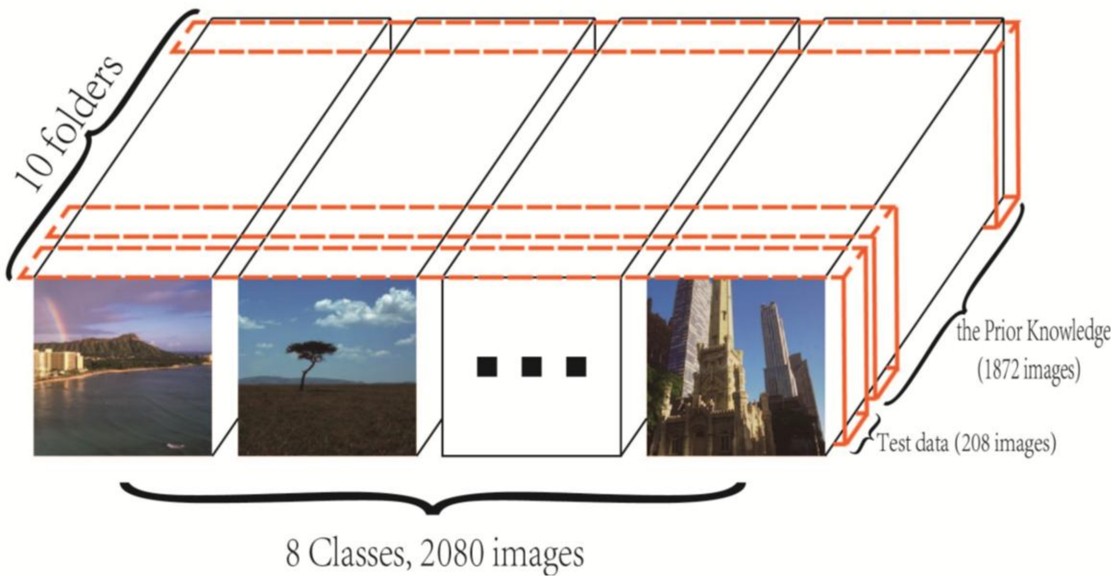


Fig. 3.3 10-fold cross-validation

We test all folders and count up the number of images that are correctly classified.

Therefore, the retrieval accuracy is calculated by:

$$accuracy = \frac{the\ number\ \ of\ currectly\ \ classified\ \ images}{the\ number\ \ of\ all\ images\ \ (2080)} \qquad \text{-----------} \ (10)$$

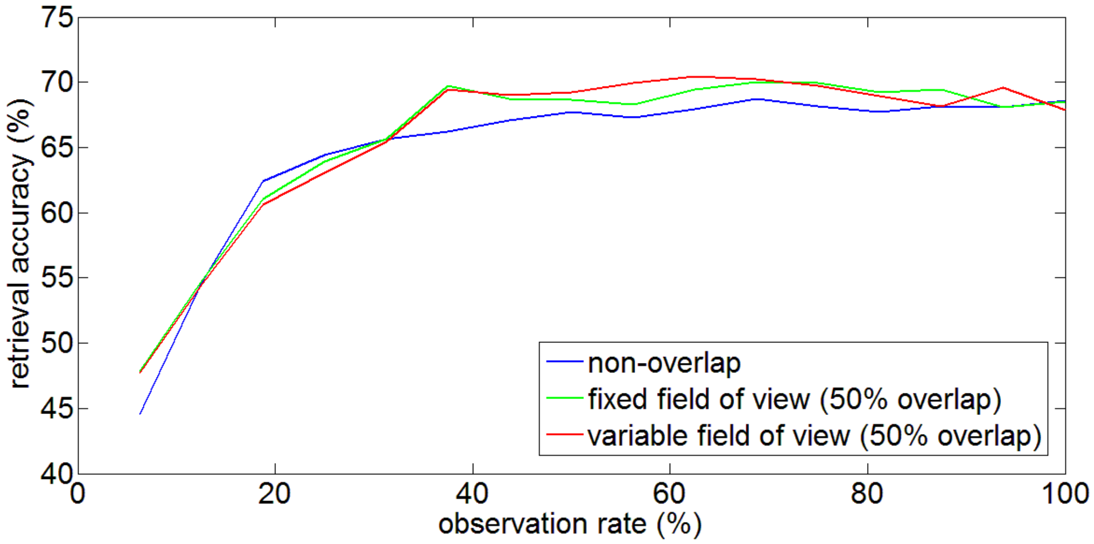


Fig. 3.4 The result of the first experiment

This experiment result shows the partially overlapped conditions are better than the non-overlap case. There are three parts I want to introduce more details. Firstly, the initial point, the overlapped cases show better performance than the non-overlap case. By using partially overlapped partition method, the system can access a more informative position than the non-overlap case. While the observation rate within 15% and 30%, non-overlap case shows the highest accuracy. While the observation rate is higher than 30%, partially overlapped cases are better than the non-overlap case.

It is inconvenient if we use the above method to extract the local feature from an image. It is difficult to extract the local feature vector from such irregular rectangular. Therefore, we propose a method to solve this problem.

3.3 Integral Oriented Gradient

It is necessary to recalculate the local features when the target image is observed sequentially with variable field of view. On the one hand, the boundary information of the small image patches will be neglected, it takes a lot of time in the local feature re-calculating process, especially, when the training data is a large quantity and the overlap rate is high. There are some researches have been done for decreasing the computational complexity, like integral histogram [17] [18]. The local features can be easily calculated by applying the integral histogram. However, if the partial view is not a regular rectangular, it is also complex to calculate the feature vectors.

Every image is represented by two matrices, an orientation matrix, and a gradient matrix. And two matrices have the same size of the image to store the orientation and gradient values at the same position. When a partial image patch is added into the current observation, the new feature vector is generated by extracting information of the same partial area in the orientation matrix and gradient matrix. An image pixel mask is used to indicate either pixel has been observed or not. With this method, partial image patches are used to easily calculate the feature vectors even if the observation is irregular. The program is attached in Appendix 2.

The Fig. 3.5 shows how the system extracts the local feature from images by using integral oriented gradient. Two matrices are used as the integral oriented gradient. The system calculates all local feature vectors of every image in each fixation.

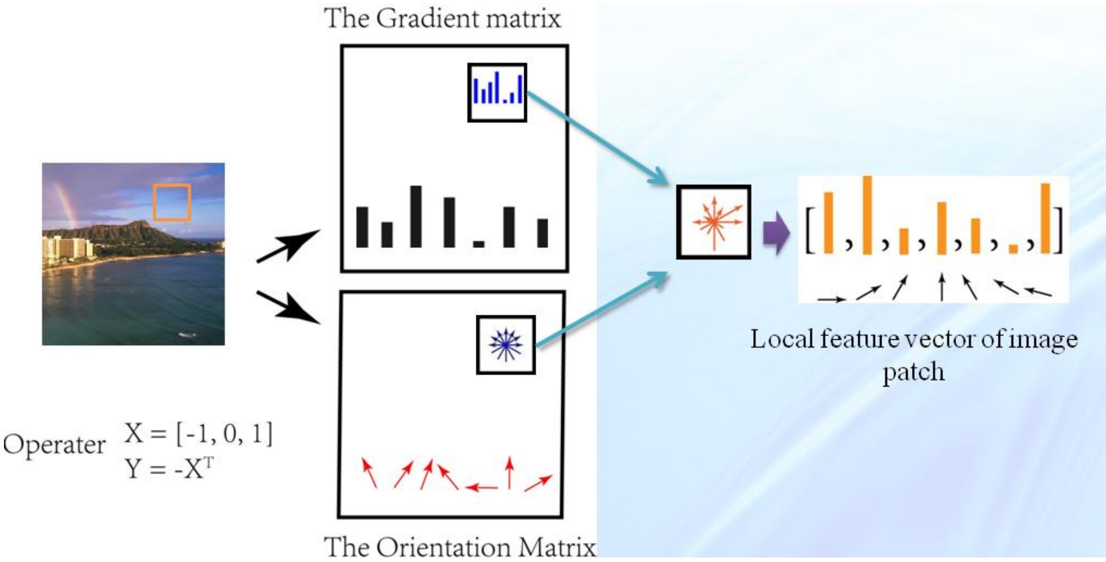


Fig. 3.5. Extracting local feature by integral oriented gradient

On the other hand, when we processing an irregular image patch, a pixel mask is used to indicate whether the pixel is observed or not. So the system extracts the part of useful information, shows in Fig. 3.5.

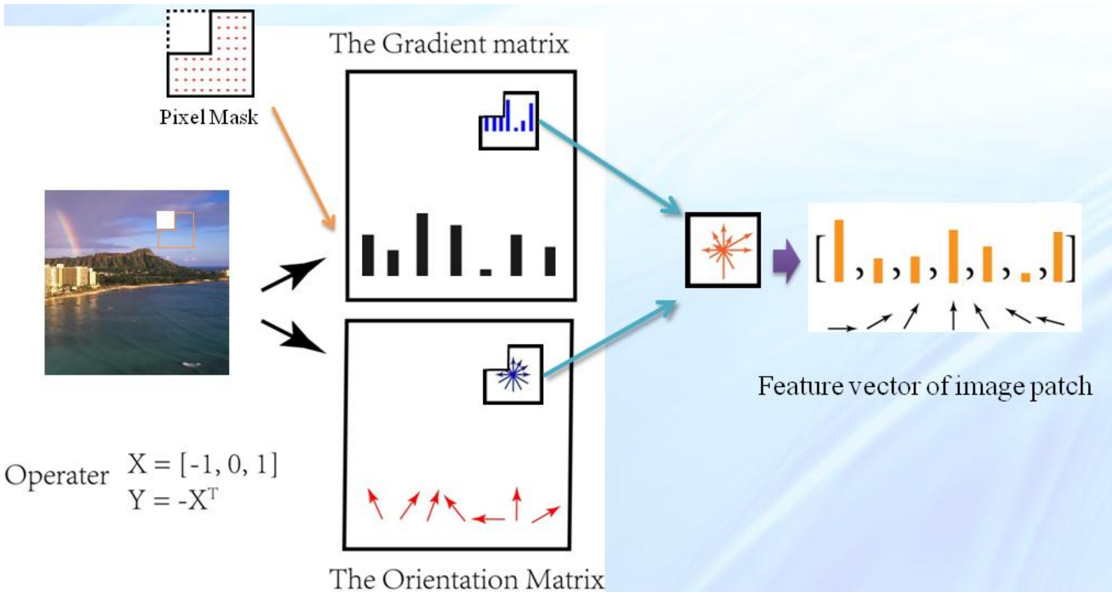


Fig. 3.6. Extracting local feature from irregular image patch

The following experiment result also shows the image retrieval performance is different with different feature extraction method. In other words, if we apply other feature descriptors, the performance may also change. However, in this research, we do not focus on verifying what kind of feature descriptor is the best one for us. When we apply the integral oriented gradient to extract local feature, the data preprocessing can be illustrated in Fig. 3.6.

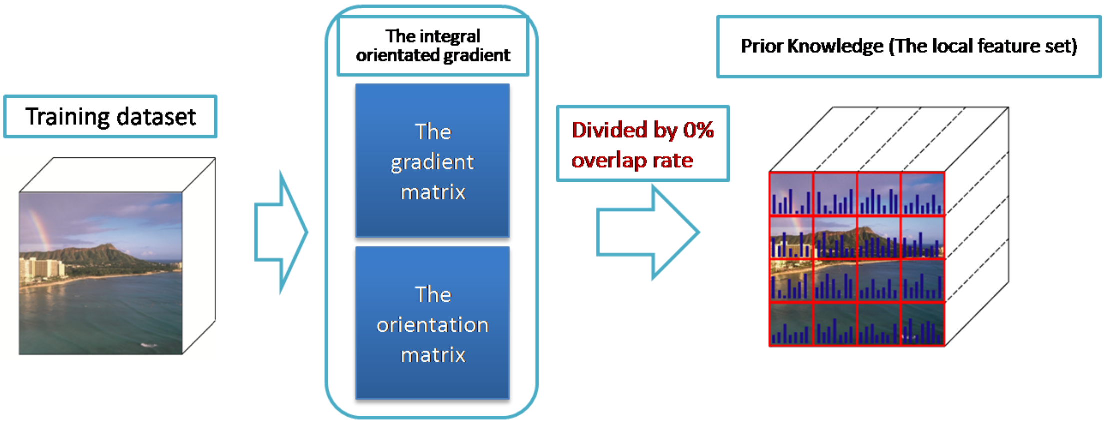


Fig 3.7 Data preprocessing

In the system, the original image is represented by two matrices, especially, the system can efficiently extract local feature from irregular image patch in the variable field-of-view case, the program shows in Appendix 3.

3.4 Experiment Result

Classification decision is evaluated with the simple nearest neighbor algorithm [14]. The total number of the remained training data is 30 images, which means that the decision is made from 30 images by counting the number of the training data remained in each class. The result is generated by limiting the number of observations.

Note that the classification processing does not stop until the selection times equals to observation limits or the target image is fully observed. With 100% attention rate, the Fig. 3.8 shows the experiment results of accuracy of:

- Non-overlap (blue line) is 72.64%;
- 25% overlap in fixed field-of-view (green solid line) is 72.03%;
- 25% overlap in variable field-of-view (green dashed line) is 71.4%;
- 50% overlap in fixed field-of-view (red solid line) is 73.07%;
- 50% overlap in variable field-of-view (red dashed line) is 68.79%;
- 75% overlap in fixed field-of-view (cyan solid line) is 72.46%;
- Base line (gray dashed line) is 73.41%;

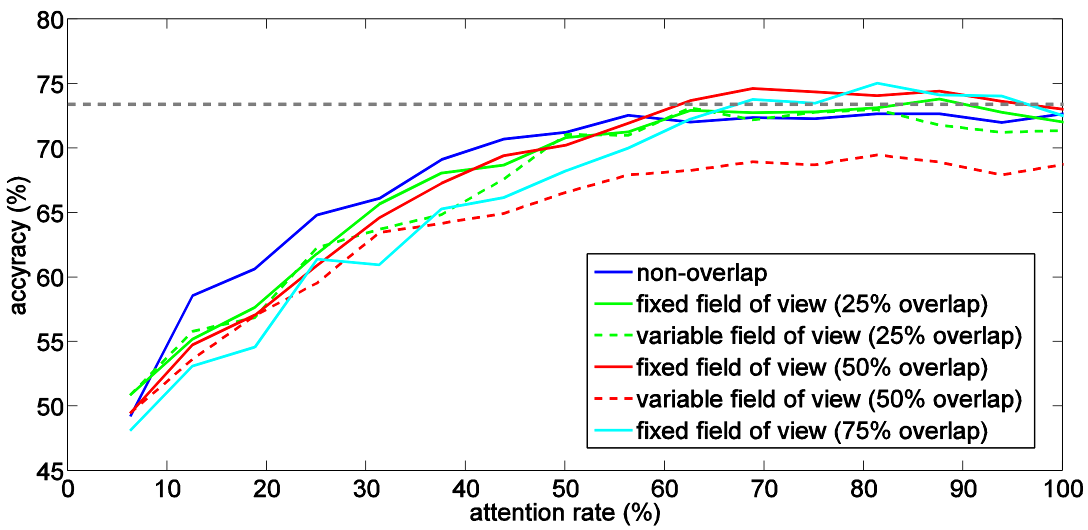


Fig. 3.8 Comparison of image retrieval accuracy between different selection approaches. Gray dashed-line presents the baseline accuracy using HOG feature with fully observed image.

The system combines all local features to an image feature and discards dissimilar data according to the similarity value between the target image and each image feature. The system will make the decision at once and generates the gray dashed base line. We can easily see the 50% overlap in fixed field-of-view can achieve the same and better performance as the base line, however, only with 60% attention rate. The 50% overlap in fixed field-of-view can achieve the highest accuracy is 74.64%. The 75% overlap in fixed field-of-view can achieve the highest accuracy is 75.04%, however, in this case, the system needs to take more observations than 50% overlap, which can be seen from Fig. 3.8.

Appendix 4 shows the Matlab code of sequential patch selection for image retrieval. In the variable field-of-view case, the local feature in some fixation needs to be recalculated, the code is marked in green. Therefore, it only needs to remove the marked code in the fixed field-of-view case.

When the attention rate is less than 60%, the non-overlap case can achieve the best performance. While the attention rate is higher than 60%, the performance of 25% overlap cases is better than the non-overlap case. However, it is obvious that the 50% overlap case shows the best performance in Fig. 7. Based on the experiment result, the gray dashed line is a base line of combining all feature vectors in 50% overlap case. It is not necessary to observe the entire environment, by sequentially selecting observations; the system can achieve same performance in the 50% overlap case just with 60% attention rate. It also shows that 75% overlap in fixed field-of-view can achieve an even higher accuracy, however, while the attention rate is around 80% attention rate.

According to the experiment results and the notion of observation selection algorithm, because one image is divided into more blocks in the overlap condition, the system can select a better fixation to make the first view than the non-overlap case. The observation is selected based on the concept that the informative position best preserves the dissimilarity across the whole training data. Without enough information input, namely, if the attention rate is less than 60%, the system needs to make its decision on the basis of variant local information that observed from the large environment. The information may be used many times in the overlap case. However, after discarding enough redundant training data with sufficient information input, the remaining training data are so similar. It proves that though some information is used many times, this information is still important for classifying the most similar data from the similar training data.

Fig. 3.9 (a), (b), (c) and (d) show the observation selection sequence of non-overlap, 25% overlap in fixed field of view, 25% overlap in variable field of view and 50% overlap case respectively. The 50% attention rate is achieved at step 8 of non-overlap case (shown in Fig. 9 (a) step 8), step 11 of 25% overlap case (shown in Fig. 9 (b) step 11) and step 14 of 50% overlap case (shown in Fig. 9 (d) step 14). It achieves 48.44% attention rate in step 11 of the variable field of view with 25% overlap (shown in Fig. 9 (c) step 11). While the attention rate is less than 50%, the system tries to discarding dissimilar images from training data set quickly, it takes 11 steps in 25% overlap and 14 steps in 50% overlap. The red dot is the center position of attentions, it shows that the differences among non-overlap case and 25% overlap cases are not so obvious, expect more observations are extracted from the target image. In the 50% overlap case, as the red dot shows, more detailed information is accessed by the system. After discarding plant of images, the images of training data set are so similar to each other that it is difficult to classify which class the target image belongs to. So, when the attention rate is higher than 50%, it can be seen that the system takes more views around the building’s boundary in overlap case. The boundary information is used repeatedly in several observations, which promotes the system can carefully classify the images of target class from the rest of training data.

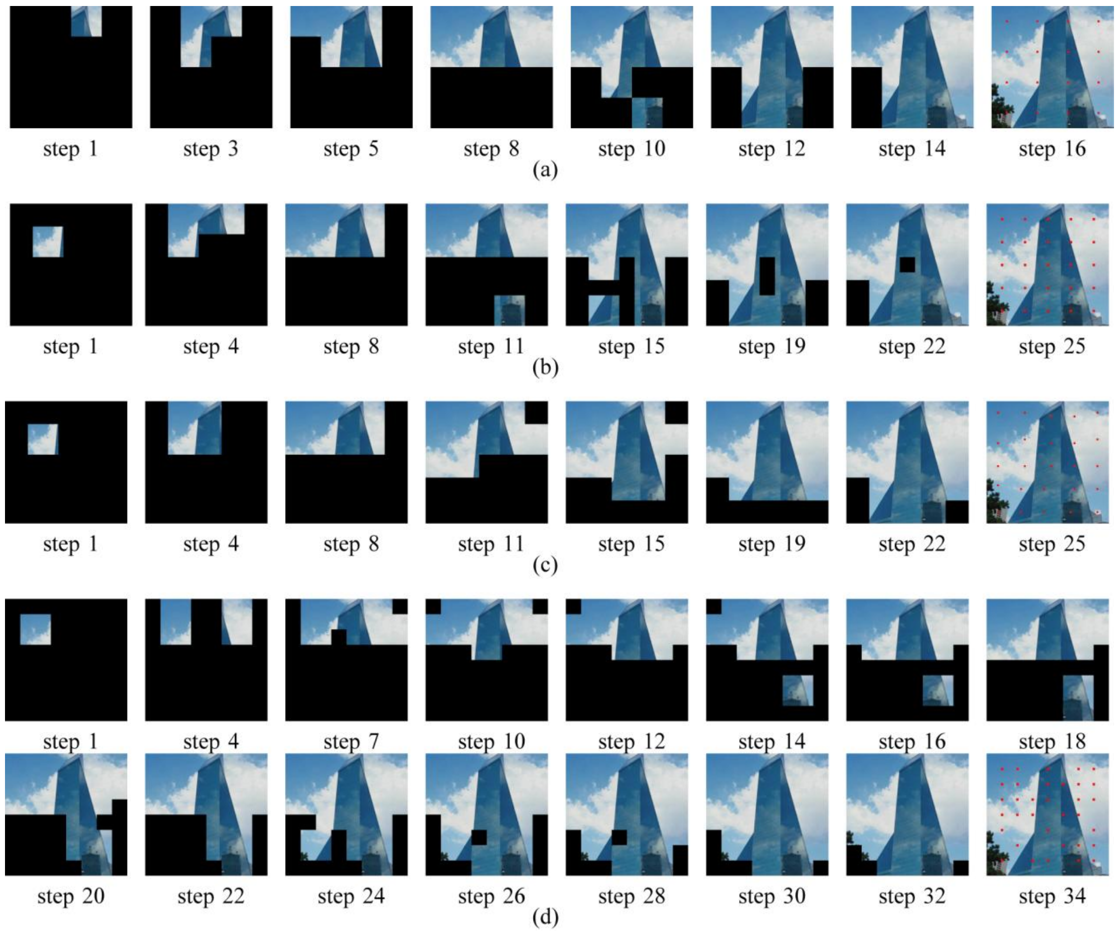


Fig. 3.9 Examples of sequential patch selection with different approaches. (a) attention sequences with non-overlap case, (b) attention sequences with 25% overlap case (fixed field of view), (c) attention sequences with 25% overlap case (variable field of view), (d) attention sequences with 50% overlap case(fixed field of view) .

We can notice that the accuracy will decrease when the image is fully observed, shows in Fig. 3.8. And the answer can be found in Fig. 3.9, we can see the useful information of an image is not 100%. In non-overlap case, all observations are independent to each other; therefore, the system can not access more information about the target objects. In the partially overlapped condition, the system can access to more information that is related to the target objects indeed, shows in Fig. 3.9, however, it also observes the useless information many times. Apparently, it is double-edged sword when we apply partially overlapped partition method.

3.5 Correlation Analysis

In this section, we want to investigate why 50% overlap in fixed field-of-view shows the best performance. We analyze the correlativity among all small patches.

Firstly, a random image is selected and divided into small patches with the different overlapping ratio from 0%, 5% and 10% to 95% (20 cases). Then, the local feature is used to represent the image patch. We calculate the correlation value between every two local features and normalize all correlation values. The Fig. 3.10 shows the method that we used to calculate the correlation value in this research.

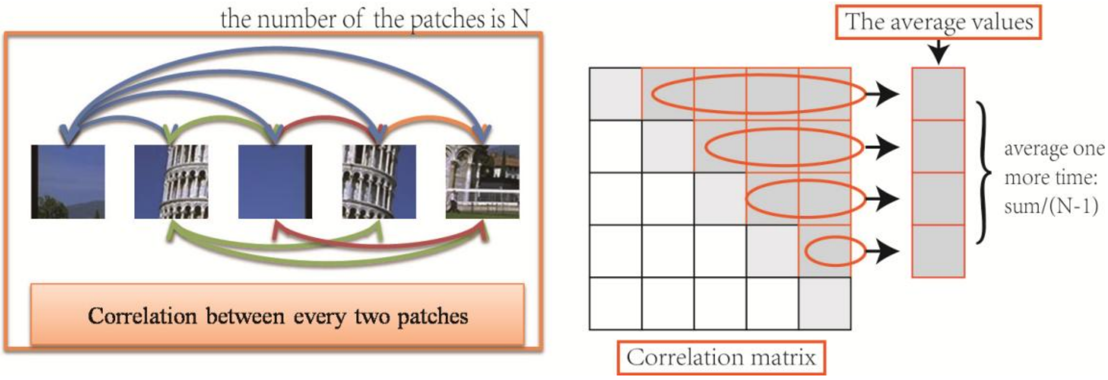


Fig. 3.10 Correlation analysis

We calculate the correlation value between every two patches, the number of the image patches is N. Then, we can generate an N×N correlation matrix. The elements of the first row are the correlation value between the first image patch and other patches. We sum the average value of each row and divide by (N-1). We calculate all correlation values of the different overlapping cases which are shown in Fig. 3.11.

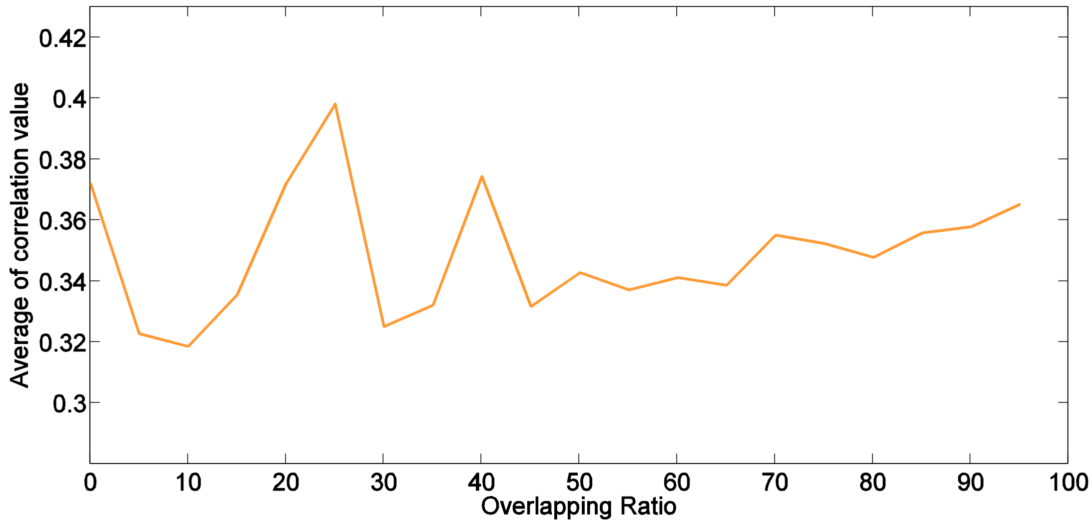


Fig. 3.11 the correlation value of the different overlapping case

The Fig. 3.12 shows that some image patches only contain a part of image information, which is described as the first kind of patches. The second kind of patches means that the patch does not contain any black area.

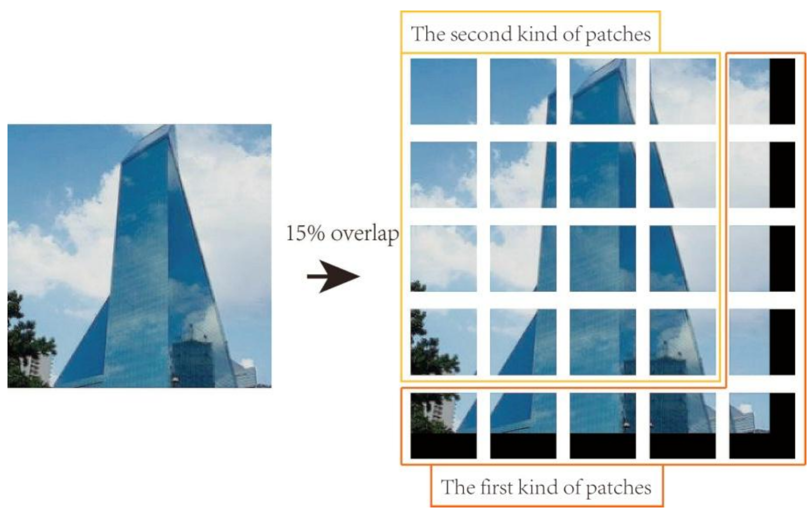


Fig. 3.12 Image patches with 15% overlap

Table. 3.1 the number of each kind of patch

Overlapping ratio (%)	0	5-20	25	30-35	40	45	50	55	60	65	70	75	80	85	90	95
The number of the patches (the first kind)	0	9	0	11	0	13	0	15	17	19	21	0	33	41	63	121
The number of the patches (the second kind)	16	25	25	36	49	49	49	64	81	100	121	169	289	441	1024	3721

The Table. 3.1 shows the number of each kind of patch. We can find when the overlapping ratio is less than 45%, there are too many patches that only contain a part of image information. This is why the trend is undulate. So, the trend is going down and slightly up as the overlapping ratio increases.

The trend shows that it achieves a small correlation value when the overlapping ratio is around 50%. The small correlation value means all patches are not related to each other, which means each patch contains quite new information. When the overlapping ratio is around 50%, each patch contains some information that is related to previous patches and also some new information. This is why the experiment when the overlapping ratio is around 50% can achieve the highest accuracy.

4 Summary

4.1 Conclusion

In this study, we proposed a partially overlapped partition method for adaptive observation selection in the large scale environment retrieval system. The new partition method provides a flexible observation selection and the system attains an important image patch to correctly retrieve the target image. Experimental results have shown that a partial observation could have enough information to efficiently retrieve the target image without the entire observations. It means that the proposed model can efficiently and effectively extract an informative area only and discard a meaningless region to enhance the image retrieval performance.

Firstly, based on this algorithm, it provides the system more chance to capture more observations around a large object and a small object with one observation. The detailed information is really helpful for classifying images in the prior knowledge with high similarity. More importantly, the system can achieve a same or better performance, even without fully accessing to the target image.

Secondly, in this study, the system divided the images of prior knowledge fixed in 25%, 50%, and 75% overlapping ratio. This research provides the robots a way to smoothly move their camera like humans and capture target relevant information sequentially from target environment. The robots' camera and head movement are more human-like.

4.2 Future works

The proposed model will be applied to solve different tasks such as image classification with limited sensing coverage and next best view selection for the mobile robot by considering the three-dimensional action spaces. For the improvements, an adaptive method will be investigated to automatically adjust overlapping ratio and the window size during the image retrieval tasks. It will be accomplished by the following step.

Firstly, the image data is processed by computing the correlation among each patch and figure out the informative area and useless information area. The correlation value is used gave a reward to the system, it can be used to improve the system when the system generates a robust information gain map. On the other hand, the correlation value that is calculated between current observation and previous observations from target environment is used to indicate the current observation is related to our target environment or not. Trying more overlapping ratio and combine various experiment together, which provides a smooth camera movement.

Secondly, even based on the current research, the system doesn't need to fully observe the target environment and can achieve a good performance. And after applying correlation function to extract the informative area, the robots can make an efficient and effective image patch selection, and automatically adjust the overlapping ratio during the task.

Furthermore, I also think the reinforcement learning approach can be applied in this research. This two system will be compared and find out the advantages and disadvantages of each of them. After completing all elements, then, we prepare experimental procedures, scenarios, instructions, and environment as the way to evaluate performance of our system.

5 Acknowledgements

It is so lucky that I can study at JAIST for two years. I met many hospitable, conscientious and learned professors, and also got acquainted with many agreeable, distinguished and lovely friends. Here and now, I would like to extend my sincere thanks to all those who have helped me.

Firstly, I am deeply grateful to Professor Nak-Young Chong, for their gentle encouragement and instruction for my research. I could not finish this work without his support. I also want to thank and Assistant Professor Sungmoon Jeong for his valuable comments and advises which are greatly helpful for me to accomplish my research and thesis. Then thanks to the teachers and professors who have taught me over the past two years of study. Professor Hiroyuki Iida kindly answered my questions and directed me to finish my minor research. I received extensive knowledge in a different field.

I am very grateful to my lovely friends who have offered me great help to compose my thesis. Hosun Lee, Dang Thi Le Quyen, Zhang Lu, Liu Xin, Wu Chengbo, Huang Jingliang, Wang Zhaoqi, Gao Yan, Chen Yu, Ou Wei, Yang Zhengguo, Wu Chuyao, Zhao Rui, Li Ke and Wang Zhenyu helped me a lot and always cheer me up when I feel dispirited. I am very grateful to be their friend.

Finally, I want to say the deepest thanks to my family for the unconditional love and continuous support. They always teach me right from wrong since I was born. I will do my best to pay them back for their support and encouragement throughout my life.

6 References

[1]. Singh, A. Krause and W.J. Kaiser, "Nonmyopic Adaptive Information Path Planning for Multiple Robots", Proc. Intl. Joint Conf. on Artificial Intelligence, 1843-1850

[2]. G.A. Hollinger, B. Englot, F.S. Hover, U. Mitra and G.S. Sukhatme, "Active planning for underwater inspection and the benefit of adaptivity", Intl. Jour. Of Robotics Research 32(1):3-18, 2013

[3]. Y. Su, S. Shan, X. Chen and W. Gao, "Hierarchical ensemble of global and local classifiers for face recognition", IEEE Trans. on Image Processing, 20(11):1885-1896, 2009

[4]. L. Itti, C. Koch and E. Niebur, "A model of saliencybased visual attention for rapid scene analysis", IEEE Trans. on Pattern Analysis and Machine Intelligence, 18(8):1254-1259, 1998

[5]. Nannan Gu, Mingyu Fan, Liang Du, Dongchun Ren, "Efficient sequential feature selection based on adaptive eigenspace model", Neurocomputing 161: 199-209 (2015), http://dx.doi.org/10.1016/j.neucom.2015.02.043

[6]. L.W. Renninger, P. Verghese and J. Coughlan, "Where to look next? Eye movements reduce local uncertainty", Jour. of Vision, 7(3):1-17, 2007

[7]. T. Rückstieß, C. Osendorfer and P. van der Smagt, "Sequential Feature Selection for Classification", AI 2011: Advances in Artificial Intelligence, Springer, pp.132-141, 2011

[8]. Somol, P., Pudil, P., Novovicova, J., Paclik, P.: "Adaptive Floation Search Methods in Feature Selection", Pattern Recognition Letters 20, 1157-1163(1999)

[9]. Yi Yang, Hengtao Shen, Zhigang Ma, Zi Huang, Xiaofang Zhou.: "L2, 1-Norm Regularized Discriminative Feature Selection for Unsupervised Learning", In:

Proceedings of the AI, pp. 1589-1594(2011)

[10]. Hosun Lee, Sungmoon Jeong, Nak Young Chong, “Unsupervised Learning Approach to Attention-path Planning for Large-scale Environment Classification”, Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.14471452, 2014

[11]. J. Najemnik and W.S. Geisler, “Optimal eye movement strategies in visual search”, Nature, 434(7031):387-391, 2005

[12]. A. Das and D. Kempe, “Algorithms for Subset Selection in Linear Regression”, Proc. Annual ACM Symposium on Theory of Computing, 45-54, 2008

[13]. H.V. Nguyen and L. Bai, “Cosine Similarity Metric Learning for Face Verification”, Computer Vision ACCV 2010, Springer, 709-720, 2011

[14]. V. Athitsos, J. Alon, and S. Sclaroff, “Efficient Nearest Neighbor Classification Using a Cascade of Approximate Similarity Measures”, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 486-493, 2005

[15]. Yali Li, Shengjin Wang, Qi Tian, Xiaoqing Ding, “A survey of recent advances in visual feature detection”, Neurocomputing, Pages 736-751, Volume 149, Part B, 3 February 2015

[16]. D. Navneet and B Triggs, “Histograms of Oriented Gradients for Human Detection”, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 886- 893, 2005

[17]. Viola, P., Michael Jones, J.: “Robust real-time face detection”. International Journal of Computer Vision 57(2), 137-154(2004)

[18]. Porikli, F. M.: “Integral histogram: A fast way to extract histograms in Cartesian spaces”, Proc. of the IEEE CVPR, vol. I, pp. 829836 (2005)

Appendix 1

```
function F = hogcalculator(img) %small image patch
cellpw = 64;    % window size
cellph = 64;    % window size
nblockw = 1;    % 1 patch
nblockh = 1;
nthet = 36;     % bin number 36 (HOG)
[M, N, K] = size(img);
delta = cellpw*nblockw * 0.5;
hx = [-1,0,1];
hy = -hx';
gradscalx = imfilter(double(img),hx);
gradscaly = imfilter(double(img),hy);
if K > 1
   maxgrad = sqrt(double(gradscalx.*gradscalx + gradscaly.*gradscaly));
   [gradscal, gidx] = max(maxgrad,[],3);
   gxtemp = zeros(M,N);
   gytemp = gxtemp;
   for kn = 1:K
      ttempx = gradscalx(:,:,kn);
      ttempy = gradscaly(:,:,kn);
      tmpindex = find(gidx==kn);
      gxtemp(tmpindex) = ttempx(tmpindex);
      gytemp(tmpindex) =ttempy(tmpindex);
   end
   gradscalx = gxtemp;
   gradscaly = gytemp;
else
   gradscal = sqrt(double(gradscalx.*gradscalx + gradscaly.*gradscaly));
end
% calculate gradient orientation matrix. plus small number for avoiding dividing zero.
gradscalxplus = gradscalx+ones(size(gradscalx))*0.0001;
gradorient = atan(gradscaly./gradscalxplus);
gradorient(gradorient<0) = gradorient(gradorient<0)+pi;
% generate gaussian spatial weight.
[gaussx, gaussy] = meshgrid(0:(cellpw*nblockw-1), 0:(cellph*nblockh-1));
weight = exp(-((gaussx-(cellpw*nblockw-1)/2).*(gaussx-(cellpw*nblockw-1)/2)...
   +(gaussy-(cellph*nblockh-1)/2).*(gaussy-(cellph*nblockh-1)/2))/(delta*delta));
```

```
% vote for histogram. there are two situations according to the interpolate
hist3dbig = zeros(nblockh+2, nblockw+2, nthet+2);
for bi = 1:(cellph*nblockh)
    for bj = 1:(cellpw*nblockw)
        iorbi = bi;
        jorbj = bj;
        gaussweight = weight(bi,bj);
        gs = gradscal(bi,bj);
        go = gradorient(bi,bj);
        binx1 = floor((jorbj-1+cellpw/2)/cellpw) + 1;
        biny1 = floor((iorbi-1+cellph/2)/cellph) + 1;
        binz1 = floor((go+(or*pi/nthet)/2)/(or*pi/nthet)) + 1;
        if gs < 1E-5
            continue;
        end
        binx2 = binx1 + 1;
        biny2 = biny1 + 1;
        binz2 = binz1 + 1;
        x1 = (binx1-1.5)*cellpw + 0.5;
        y1 = (biny1-1.5)*cellph + 0.5;
        z1 = (binz1-1.5)*(or*pi/nthet);
        % trilinear interpolation
        hist3dbig(biny1,binx1,binz1) = hist3dbig(biny1,binx1,binz1) + gs*gaussweight...
            * (1-(jorbj-x1)/cellpw)*(1-(iorbi-y1)/cellph)*(1-(go-z1)/(or*pi/nthet));
        hist3dbig(biny1,binx1,binz2) = hist3dbig(biny1,binx1,binz2) + gs*gaussweight...
            * (1-(jorbj-x1)/cellpw)*(1-(iorbi-y1)/cellph)*((go-z1)/(or*pi/nthet));
        hist3dbig(biny2,binx1,binz1) = hist3dbig(biny2,binx1,binz1) + gs*gaussweight...
            * (1-(jorbj-x1)/cellpw)*((iorbi-y1)/cellph)*(1-(go-z1)/(or*pi/nthet));
        hist3dbig(biny2,binx1,binz2) = hist3dbig(biny2,binx1,binz2) + gs*gaussweight...
            * (1-(jorbj-x1)/cellpw)*((iorbi-y1)/cellph)*((go-z1)/(or*pi/nthet));
        hist3dbig(biny1,binx2,binz1) = hist3dbig(biny1,binx2,binz1) + gs*gaussweight...
            * ((jorbj-x1)/cellpw)*(1-(iorbi-y1)/cellph)*(1-(go-z1)/(or*pi/nthet));
        hist3dbig(biny1,binx2,binz2) = hist3dbig(biny1,binx2,binz2) + gs*gaussweight...
            * ((jorbj-x1)/cellpw)*(1-(iorbi-y1)/cellph)*((go-z1)/(or*pi/nthet));
        hist3dbig(biny2,binx2,binz1) = hist3dbig(biny2,binx2,binz1) + gs*gaussweight...
            * ((jorbj-x1)/cellpw)*((iorbi-y1)/cellph)*(1-(go-z1)/(or*pi/nthet));
        hist3dbig(biny2,binx2,binz2) = hist3dbig(biny2,binx2,binz2) + gs*gaussweight...
            * ((jorbj-x1)/cellpw)*((iorbi-y1)/cellph)*((go-z1)/(or*pi/nthet));
    end
end
F(1:nthet) = hist3dbig(2, 2, 2:(nthet+1))
```

Appendix 2

```
function [Labelmag, Labelbin] = hogmagn_orien(img)

% check parameters's validity.
[M, N, K] = size(img);
% calculate gradient scale matrix.
hx = [-1,0,1];
hy = -hx';
gradscalx = imfilter(double(img),hx);
gradscaly = imfilter(double(img),hy);

if K > 1
   maxgrad = sqrt(double(gradscalx.*gradscalx + gradscaly.*gradscaly));
   [Labelmag, gidx] = max(maxgrad,[],3);
   gxtemp = zeros(M,N);
   gytemp = gxtemp;
   for kn = 1:K
      ttempx = gradscalx(:,:,kn);
      ttempy = gradscaly(:,:,kn);
      tmpindex = find(gidx==kn);
      gxtemp(tmpindex) = ttempx(tmpindex);
      gytemp(tmpindex) =ttempy(tmpindex);
   end
   gradscalx = gxtemp;
   gradscaly = gytemp;
else
   Labelmag = sqrt(double(gradscalx.*gradscalx + gradscaly.*gradscaly));
End

% calculate gradient orientation matrix.
% plus small number for avoiding dividing zero.
gradscalxplus = gradscalx+ones(size(gradscalx))*0.0001;
% unsigned situation: orientation region is 0 to pi.
Labelbin = atan(gradscaly./gradscalxplus);
Labelbin(Labelbin<0) = Labelbin(Labelbin<0)+pi;
Labelbin = floor(Labelbin.*(36/pi)) + 1;
save Labelmag;    %gradient matrix
save Labelbin;    %orientation matrix
```

Appendix 3

```
cellpw = 64;                        % window size
cellph = 64;
nthet = 36;
overlap = 0.5;                      % overlapping ratio 50%
orow = ceil((256 - cellpw)/(cellpw * (1 - overlap)) + 1);
ocolumn = orow;
imageN = 2080;
AallimageFV = cell(orow,ocolumn);    % local feature set
cord = cell(orow);                   % the coordinate of each fixation
for r = 1:orow
    for c = 1:ocolumn
        rstat = (r-1)*cellpw*(1-overlap)+1;
        cstat = (c-1)*cellph*(1-overlap)+1;
        cord{r,c} = [rstat, cstat];
    end
end
for r = 1:orow
    for c = 1:ocolumn
        rstat = cord{r,c}(1);
        cstat = cord{r,c}(2);
        for i = 1:imageN
            gradscal = mags(rstat:(rstat+cellpw-1),cstat:(cstat+cellph-1));
            gradorient = bins(rstat:(rstat+cellpw-1),cstat:(cstat+cellph-1));
            F = zeros(1, nthet);
            for b = 1:nthet
                bin = gradorient;
                mag = gradscal;
                bin(bin ~= b) = 0;
                bin(bin == b) = 1;
                mag = mag.*bin;
                F(b) = sum(sum(mag));
            end
            AallimageFV{r,c}(i,:) = F/sqrt(sum(F.*F)+0.001^2);
        end
    end
end
filename=['imageFV',num2str(overlap),'.mat'];     save(filename,'imageFV');
```

Appendix 4

```
load AallimageFV;
load Labelbin;
load Labelmag;
imageN = 2080;
cellpw = 64;
cellph = 64;
overlap = ;     % overlapping ratio
groups = 1;
groupf = 1;     % indicates which folder is processing
nthet = 36;     % we set 36 bins
num = 208;      % one folder contains 208 images
orow = (256 - cellpw)/(cellpw * (1 - overlap)) + 1;
ocolumn = orow;
para = ceil(1/(1-overlap) - 1);
leftiN = imageN;

image_mask = zeros(1,imageN);
block_mask = zeros(orow,ocolumn);
infor_gain_map = zeros(orow,ocolumn);
result = zeros(orow^2, num);
atten = zeros(orow^2, num);

cord = cell(orow, ocolumn);
for r = 1:orow
   for c = 1:ocolumn
      rstat = (r-1)*cellpw*(1-overlap)+1;
      cstat = (c-1)*cellph*(1-overlap)+1;
      cord{r,c} = [rstat, cstat];
   end
end

for groupN = groups:groupf
   group = 26*(groupN-1);
   Aimage_mask = zeros(1,imageN);
   for i = 1:8
      Aimage_mask(1, (260*(i-1)+1+group):(260*(i-1)+26+group)) = 1;
   end
```

```
for imgNum = 1:num
    dissim = zeros(1, imageN);
    classN = floor((imgNum-1)/26) + 1;
    tarimgN = (260-26)*(classN-1) + imgNum + group;

    for stepexp = 1:(orow*ocolumn)
        if sum(find(atten(1:stepexp, imgNum) == 256*256)) < 1
            dataimgFV = 0;
            tar_imgfv = 0;
            block_mask = zeros(orow,ocolumn);
            o_mask = zeros(256);
            image_mask = Aimage_mask;
            allimageFV = AallimageFV;
            leftiN = imageN - num;

            for step = 1:stepexp
                IMGMask = image_mask';
                IMGMask(IMGMask>0) = -1;
                IMGMask(IMGMask == 0) = 1;
                IMGMask(IMGMask == -1) = 0;
                imgmASK = repmat(IMGMask,1,nthet);

                for r = 1:orow
                    for c = 1:ocolumn
                        if (block_mask(r,c) == 0) &&
(sum(find(o_mask(cord{r,c}(1):(cord{r,c}(1)+cellpw-1),cord{r,c}(2):(cord{r,c}(2)+cellp
h-1))==0))~=0)
                            imgfv = allimageFV{r, c}(tarimgN,:);
                            allimageFV{r,c} = allimageFV{r,c}.*imgmASK;
                            allimageFV{r, c} = allimageFV{r,c}.*imgmASK;
                            dataimgFV(1:imageN, (1+(step-1)*nthet):(step*nthet)) =
allimageFV{r, c};
                            average_fv = sum(dataimgFV)'/leftiN;
                            average_fv = average_fv/norm(average_fv);
                            infor_gain_map(r,c) = sum(1 -
dataimgFV*average_fv/sqrt(step));
                            allimageFV{r, c}(tarimgN,:) = imgfv;
                        else
                            if (block_mask(r,c) == 0) &&
(sum(find(o_mask(cord{r,c}(1):(cord{r,c}(1)+cellpw-1),cord{r,c}(2):(cord{r,c}(2)+cellp
```

```
h-1))==0)) == 0)
                                    block_mask(r,c) = step;
                                end
                                infor_gain_map(r,c) = 0;
                            end
                        end
                    end

                    if sum(find(block_mask == 0)) ~= 0
                        max_infor = max(max(infor_gain_map));
                        [fix_row, fix_col] = find(infor_gain_map == max_infor);
                        fix_row = fix_row(1);
                        fix_col = fix_col(1);
                        block_mask(fix_row, fix_col) = step;
                        rstat = cord{fix_row, fix_col}(1);
                        cstat = cord{fix_row, fix_col}(2);
                        curatten =
size(find(o_mask(rstat:(rstat+cellpw-1),cstat:(cstat+cellph-1)) == 0),1);
                        mask = o_mask(rstat:(rstat+cellpw-1),cstat:(cstat+cellph-1));
                        mask(mask == 0) = step;
                        o_mask(rstat:(rstat+cellpw-1),cstat:(cstat+cellph-1)) = mask;

                        imgfv = allimageFV{fix_row, fix_col}(tarimgN,:);
                        if norm(imgfv) ~= 0
                            tar_imgfv((1+(step-1)*nthet):(step*nthet)) = imgfv;
                            imgfv = tar_imgfv'/norm(tar_imgfv);
                            imgmASK = repmat(imgmASK,1,step);
                            dataimgFV(1:imageN, (1+(step-1)*nthet):(step*nthet)) =
allimageFV{fix_row, fix_col};
                            dataimgFV = dataimgFV.*imgmASK;
                            tar_gain = dataimgFV*imgfv/sqrt(step);

                            dissim = tar_gain' + image_mask;
                            leftatten = size(find(o_mask == 0), 1);
                            if ((leftatten+curatten)/curatten) > (stepexp+1 - step)
                                k = (leftiN-30)/(stepexp-step+1);
                            else
                                k = (leftiN-30) * (curatten/(leftatten+curatten));
                            end
                            while k > 0
                                n = find(dissim == min(dissim), 2);
```

```
            m = size(n, 2);
            if m-k <= 0
                for i = 1:m
                    dissim(n(i)) = 1;
                    image_mask(n(i)) = step;
                end
            else
                for i = 1:k
                    dissim(n(i)) = 1;
                    image_mask(n(i)) = step;
                end
            end
            k = k - m;
        end
        leftiN = size(find(image_mask == 0), 2);

        % recalculate the feature vectors
        if  step < stepexp
            for r = (fix_row-para):(fix_row+para)
                if r > 0 && r <= orow
                    for c = (fix_col-para):(fix_col+para)
                        if c > 0 && c <= ocolumn
                            rstat = cord{r,c}(1);
                            cstat = cord{r,c}(2);
                            mask =
o_mask(rstat:(rstat+cellpw-1),cstat:(cstat+cellph-1));
                            if sum(find(mask == 0) ~= 0) && block_mask(r,c)
== 0
                                mask(mask>0) = -1;
                                mask(mask == 0) = 1;
                                mask(mask < 0) = 0;
                                for i = 1:2080
                                    if (image_mask(i) == 0 || i == tarimgN) &&
step < stepexp
                                        gradscal =
mask.*Labelmag{i}(rstat:(rstat+cellpw-1),cstat:(cstat+cellph-1));
                                        gradorient =
mask.*Labelbin{i}(rstat:(rstat+cellpw-1),cstat:(cstat+cellph-1));
                                        F = zeros(1, nthet);
                                        for n = 1:nthet
                                            bin = gradorient;
```

```
                                                            mag = gradscal;
                                                            bin(bin ~= n) = 0;
                                                            bin(bin == n) = 1;
                                                            mag = mag.*bin;
                                                            F(n) = sum(sum(mag));
                                                         end
                                                         allimageFV{r,c}(i,:) =
F/sqrt(sum(F.*F)+0.001^2);
                                                      end
                                                   end
                                                end
                                             end
                                          end
                                       end
                                    end
                                 end
                              end

                           end
                        end
                        maskcount = zeros(1,8);
                        for i = 1:8
                           maskcount(i) = size(find(image_mask((260*(i-1)+1):260*i) == 0), 2);
                        end
                        if sum(find(find(maskcount == max(maskcount)) == classN)) > 0
                           result(stepexp, imgNum) = 1;
                        end

                        atten(stepexp, imgNum) =  256*256 - leftatten;
                     end
                  end
               end
end
filename=['Ataratten',num2str(groups),'.mat'];
save(filename,'atten','result');
```