Non-parametric Approach for Practical Matching in 2D Images

Thesis by
Chao Zhang

In Partial Fulfillment of the Requirements for the
degree of
Doctor of Philosophy in Engineering

IWATE UNIVERSITY
Iwate, Japan

March 23, 2017
Defended February 9, 2017

Chao Zhang
ORCID: orcid.org/0000-0003-1961-6748

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Dr. AKASHI Takuya for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would like to thank the rest of my thesis committee: Dr. KONNO Kouichi, Dr. FUJIMOTO Tadahiro, for their insightful comments and encouragement, but also for the hard questions which inspire me to widen my research from various perspectives. Also, I am very grateful to Dr. CHIBA Norishige for his advices and gentility. I cannot forget all the students and staffs in the SMARTCV group and feel grateful for all kinds of help and assistance.

Last but not the least, I would like to thank my family: my parents and to my wife for supporting me spiritually throughout writing this thesis and my life in general.

ABSTRACT

Matching problems exist widely in many practical computer vision research tasks such as template matching, image matching, visual tracking, image registration, etc. Three main steps which form the matching procedure can be concluded as 1) feature selection, 2) similarity measure, and 3) search strategy design. Based on the basic intensity or color feature, many features have been designed recent years, which can be broadly categorized as local feature descriptor or global feature descriptor. With multiple alternatives available, selecting an appropriate feature for a specific matching problem becomes essential and a preprocessing (e.g. dimension reduction) can further improve the efficiency. On the other hand, based on the selected feature, similarity measure plays a role on quantifying the real-valued similarity/distance between two objects. With specific feature selected, an appropriate similarity measure method is supposed to be carefully selected from many parametric/non-parametric distance calculation methods. Besides, in case of the number of candidates is large (e.g. template localization, image retrieval), an efficient search strategy is needed instead of exclusive matching, because the cost of similarity measure grows proportionally with the increase of candidate number. In this dissertation, we introduce solutions of matching problems in multiple specific computer vision tasks.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

Matching problems exist widely in many practical computer vision research tasks such as template matching, image matching, visual tracking, image registration, etc. Three main steps which form the matching procedure can be concluded as 1) feature selection, 2) similarity measure, and 3) search strategy design. Based on the basic intensity or color feature, many features have been designed recent years, which can be broadly categorized as local feature descriptor or global feature descriptor. With multiple alternatives available, selecting an appropriate feature for a specific matching problem becomes essential and a preprocessing (e.g. dimension reduction) can further improve the efficiency. On the other hand, based on the selected feature, similarity measure plays a role on quantifying the real-valued similarity/distance between two objects. With specific feature selected, an appropriate similarity measure method is supposed to be carefully selected from many parametric/non-parametric distance calculation methods. Besides, in case of the number of candidates is large (e.g. template localization, image retrieval), an efficient search strategy is needed instead of exclusive matching, because the cost of similarity measure grows proportionally with the increase of candidate number. In this dissertation, we introduce solutions of matching problems in multiple specific computer vision tasks.

In Chapter 2 and 3, as an example of parametric matching, affine and projective model based template matching tasks are studied respectively. Both of the tasks require to use intensity feature only and the similarity measure method is limited to sum of absolute difference (SAD). In such cases, in order to search an approximate transformation over a very large searching space, we treat the searching procedure as an optimization procedure. Although homography can be estimated by combining key-point-based local features and random sample consensus (RANSAC), it can hardly be solved with feature-less images or high outlier rate images. Estimating the affine/projective transformation remains a difficult problem due to high-dimensionality and strong non-convexity. Our approach is to quantize the

parameters of projective transformation with binary finite field and search for an appropriate solution as the final result over the discrete sampling set. The benefit is that we can avoid searching among a huge amount of potential candidates. Furthermore, in order to approximate the global optimum more efficiently, we develop a level-wise adaptive sampling (LAS) method under genetic algorithm framework. With LAS, the individuals are uniformly selected from each fitness level and the elite solution finally converges to the global optimum.

In Chapter 4 and 5, for the tasks that parametric methods cannot be applied, non-parametric template matching methods are studied which do not assume any specific deformation models. Two different similarity measure methods are proposed in each chapter respectively. The first method is developed based on an assumption that the local rigidity, which is referred to as structural persistence between image patches, can help the algorithm to achieve better performance. A spatial relation test is proposed to weight the rigidity between two image patches. The second method called two-side agreement learning (TAL) is proposed which learns the implicit correlation between two sets of multidimensional data points. TAL learns from a matching exemplar to construct a symmetric tree-structured model. Using TAL can reduce the ambiguity in defining similarity which is hard to be objectively defined and lead to more convergent results.

In Chapter 6, non-parametric image matching problem with modified query image is studied. We use a compressed histograms of oriented gradients (HOG) feature descriptor to extract global visual similarity. For the nearest neighbor search problem, we propose random projection indexed KD-tree forests (rKDFs) to match image with local changes pair (ILP) efficiently instead of exhaustive linear search. rKDFs is built with large scale low-dimensional KD-trees. Each KD-tree is built in a random projection indexed subspace and contributes to the final result equally through a voting mechanism. In Chapter 7, non-parametric online visual tracking problem is studied. We propose a real-time tracking algorithm called coupled randomness tracking (CRT) which focuses on dealing with these two issues. One randomness represents random projection, and the other randomness represents online random forests (ORFs). In CRT, the gray-scale feature is compressed by a sparse measurement matrix, and ORFs are used to train the sample sequence online. During the training procedure, we introduce

a tree discarding strategy which helps the ORFs to adapt fast appearance changes caused by illumination, occlusion, etc. Our method can constantly adapt to the objective's latest appearance changes while keeping the prior appearance information.

Chapter 2

TEMPLATE MATCHING WITH AFFINE TRANSFORMATION

2.1 Summary

In this chapter, we address the problem of template matching under affine transformations with general images. Our approach is to search an approximate affine transformation over a binary Galois field. The benefit is that we can avoid matching with huge amount of potential transformations, because they are discretely sampled. However, a Galois field of affine transformation can still be impractical for exhaustive searching. To approach the optimum solution efficiently, we introduce a level-wise adaptive sampling (LAS) method under genetic algorithm framework. In LAS, individuals converge to the global optimum according to a level-wise selection and crossover while the population number is decreased by a population bounding scheme. In the experiment section, we analyse our method systematically and compare it against the state-of-the-art method on an evaluation data set. The results show that our method has a high accuracy performance with few matching tests compared against the state-of-the-art method.

2.2 Introduction

In this chapter, we consider the problem of template matching under arbitrary 2D affine transformations. Template matching is a classical computer vision problem which aims to find a global optimum area in the target image (*i.e.* source image) according to the hint provided by a rectangular template. In affine template matching, each candidate affine transformation corresponds to a candidate area in the target image. We only use the gray scale information of images as hint which is quantified by sum of absolute difference (SAD).

Recently, feature-based matching methods like SIFT and its variants are very efficient to estimate the 2D transformation matrix between template and target image. Only a few correctly matched key points are required for solving a system of linear equations. With matching results which contain

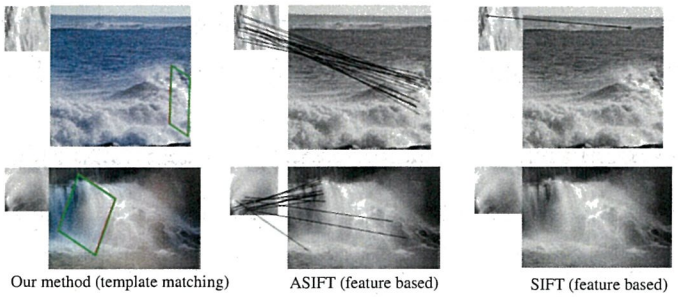


Figure 2.1: Our matching result (represented by green parallelogram) completely covers the ground truth area (represented red parallelogram) in both examples. Affine-SIFT (ASIFT) can well handle affine transformation in the case when template has strong features (upper), but mismatches in the case when template has weak features (lower). Common SIFT can not handle affine transformation well.

outliers, we can also use method like RANSAC Fischler and Bolles, 1981 to estimate the correct transformation matrix. Feature-based methods depend on the assumption that the key point matching results consist of inliers, there also exist images in which key points are hard to be detected like blur images, texture-less images, etc. Key points may also be mismatched heavily as a result of noise, illumination changes, etc. A common template matching method is usually considered to be effective against such special situations. Figure 2.1 shows two matching examples respectively when a template has strong features and weak features.

As we all know, template matching potentially requires a huge number of samples in order to ensure the global optimum solution can be obtained. Especially for affine template matching, the number of candidate transformations increases exponentially when more accurate solution is required to be obtained, because scaling, rotation and shear are taken into account additionally compared with common template matching. Matching with numerous candidate solutions is ineffective and not practical. In fact, it is possible to estimate only a small fraction of candidate solutions in order to solve the optimum solution if the following assumption is made: a template is smooth. Under this assumption, SAD will not change much around the ground truth area of a target image. This assumption provides chances for developing more efficient matching methods. At the same time, such methods can not guarantee the accuracy with highly textured template.

The rest of this chapter is structured as follows. In Section 2.3, we survey

template matching methods with transformations and the efforts that have been done on solving affine template matching problem. In Section 2.4, we introduce our method from two perspectives: 1) construction of Galois field. 2) level-wise adaptive sampling method over Galois field. In Section 2.5, we investigate the effects of tunable parameters and compare our method against the state-of-the-art method Korman et al., 2013. Finally, we conclude this chapter in Section 2.6.

2.3 Related Works

In this section, we mainly survey previous works on template matching considering geometric transformations. Despite the feature-based matching methods like SIFT Lowe, 2004, ASIFT Morel and Yu, 2009, direct methods also have been widely studied. A common direct template method only consider the translation in x-axis and y-axis, thus the degree of freedom (DF) is simply 2. However, many applications require methods to be robust with varied transformations.

Rotation and translation: Same with common template matching, target area in target image is still rectangular. The difference is, it is rotated and repositioned by translation. The DF in this situation is 3. Choi and Kim Choi and W.-Y. Kim, 2002 proposed a method combining the projection method and Zernike moments in two stages. Candidates with low cost feature extracted are selected at first stage, and rotation invariant matching is performed at second stage. Fredriksson et al.Fredriksson, Mäkinen, and Navarro, 2007 used string matching technique to deal with rotation.

Rotation, translation and scaling: In this situation, scaling is additionally attached to the matching problem, thus the DF grows to 5. The number of candidate areas becomes large and it is no longer practical for exhaustive searching. To accelerate matching procedure, kim et al.H. Y. Kim and Araújo, 2007 applied cascaded filters to exclude areas which have low probability to be selected as the final result. Akashi et al.Akashi et al., 2007 treated template matching as an optimization problem under genetic algorithm framework and applied their method into real-time eye detection by inheriting previous frame's matching result to the next. Genetic algorithm can evolutionarily select "promising" candidate areas to evaluate, thus can avoid exhaustive searching.

Affine transformation: Despite basic Euclidean transformations, shear and scaling are enhanced additionally. The DF then grows to 6. To the

best of our knowledge, few direct methods have been proposed under this situation compared with aforementioned two situations as a result of the broad search space. Korman et al., 2013 is probably the state-of-the-art work which matches template in a very sparse way under the smooth assumption. In this chapter, a discrete sampling net is constructed according to an accuracy parameter, after that, a branch-and-bound scheme is employed to search an approximate solution over the net. The basic idea of this chapter is to rule out a large portion of "unpromising" candidate transformations and focus on estimating the ones which are close to the ground truth. However, branch-and-bound scheme is still exhaustive to a certain extent, because the number of candidate transformations need to be estimated grows rapidly with the increase of expected accuracy. Insufficient samplings will lead to a totally different result. On the other hand, our method constructs a Galois field instead of a sampling net, and employs adaptive sampling to approach the ground truth from the perspective of optimization algorithm.

2.4 Methodology

Problem Description

Two grayscale images I_1 and I_2 are given as the input with each pixel's gray value normalized to [0, 1]. I_1 is defined as a template image with size of $n_1 \times n_1$. I_2 is defined as a target image with size of $n_2 \times n_2$. For clarity, we assume I_1 and I_2 are square images. An arbitrary pixel in I_1 and its mapped pixel in I_2 is denoted as p_1 and p_2 separately. We have

$$p_2 = T(p_1). \tag{2.1}$$

T is a 3×3 matrix which denotes affine transformation between p_1 and p_2. In the following formula, $\boldsymbol{k}$ includes operations such as rotation, scaling and shear. $\boldsymbol{t}$ includes translation operations:

$$T = \begin{bmatrix} \boldsymbol{k} & \boldsymbol{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}. \tag{2.2}$$

SAD is used to measure the similarity between I_1 and a candidate area in I_2. Normalized gray scale difference between each p_1 and according p_2 is summed, which can be written as:

$$S(I_1, I_2, T) = \frac{\sum_{p_1 \in I_1} |I_1(p_1) - I_2(p_2)|^m}{n_1^2}, m = \begin{cases} 0 & p_2 \notin I_2 \\ 1 & p_2 \in I_2 \end{cases}. \tag{2.3}$$

The purpose of our chapter is to infer an approximate affine transformation $\hat{T}$ from a given candidate set. In the best case, $\hat{T}$ equals to transformation $\overline{T}$. $\overline{T}$ is the closest transformation to ground truth T' among all the candidate transformations. An natural way to estimate $\hat{T}$ is to minimize SAD. Formally, our purpose can be denoted as:

$$\hat{T} = \underset{T \in \mathbb{F}_{2^{6n}}}{\arg\min} S(I_1, I_2, T). \tag{2.4}$$

The construction of candidate set will be introduced in the Section 2.4. However, from Equation 2.4, we can still not ensure that $\hat{T}$ is close enough to T', because SAD is related with not only transformation but also variation of template. Variation ν of template can be defined as the sum of maximal difference between each p_1 and its 8 neighbors $N_8(p_1)$. Formally,

$$\nu = \sum_{p_1 \in I_1} \max_{q \in N_8(p_1)} |I_1(p_1) - I_1(q)|. \tag{2.5}$$

Large variation means that an image is not smooth. In this case, two candidate transformations' SAD value will differ a lot even they are very close. Detail explanation will be discussed in the next section.

Galois Field of Affine Transformation

Matching with complete continuous affine transformation set which contains infinite candidates can be impractical. We build a discrete searching space according to binary Galois field to simplify this problem.

According to Hartley and Zisserman, 2003, a general affine transformation matrix can be decomposed into $T = TrR_2SR_1$, where R_1 represents matrix operation of 1st rotation, S is scale operation in x-axis and y-axis, R_2 is 2nd rotation, Tr is translation operation in x-axis and y-axis. By this decomposition, we will have 6 DFs given a certain affine transform. To construct a Galois field of affine, we summarize the range of each DF in Table 2.1.

Transformations over each decomposed DF can be modeled by a Galois field $\mathbb{F}_{2^n}$, n is a positive integer denoting the length of binary code and 2^n is the field's size. Elements in $\mathbb{F}_{2^n}$ are expressed as binary codes. For clarity, we assume n of each decomposed DF is the same. Each DF's range is then divided into 2^n discrete segments. $T \in \mathbb{F}_{2^{6n}}$ denotes a general affine transformation in 6 DFs. Acceptable margin of error can then be guaranteed on this

transformation	range	step amount	step size
rotation	$[0,\ 2\pi]$	2^n	$\frac{\pi}{2^{n-1}}$
translation	$[-n_2,\ n_2]$	2^n	$\frac{n_2}{2^{n-1}}$
scale	$[\frac{n1}{n2}\ ,\frac{n2}{n1}]$	2^n	$\frac{n_2^2-n_1^2}{n_1 n_2 2^n}$

Table 2.1: Value ranges of parameters for constructing a Galois Field of Affine Transformation.

Galois field. The maximum error of rotation is within $[-\pi/2^{n-1}, \pi/2^{n-1}]$, the maximum error of translation is within $[-n_2/2^{n-1}, n_2/2^{n-1}]$, the maximum error of scaling is within $[(-n_2^2 - n_1^2)/n_1 n_2 2^n, (n_2^2 - n_1^2)/n_1 n_2 2^n]$.

To quantify the error between two transformations T_1 and T_2, following formula is defined:

$$E(T_1, T_2) = |S(I_1, I_2, T_1) - S(I_1, I_2, T_2)|. \tag{2.6}$$

It has been proved in Korman et al., 2013 (Theorem 3.1) that the upper limit of $E(\overline{T}, T')$ is associated with three factors in discrete set of affine: step amount, variation of template, template size. For Galois filed of affine, we can rewrite:

$$E(\overline{T}, T') \leq O(\frac{\nu}{n \times n_1}) \tag{2.7}$$

With loose upper limitation, which may be caused by small n_1, small n, or large ν, there exists possibility that $E(\hat{T}, T') < E(\overline{T}, T')$. Note that $\overline{T}$ is the closest transformation to T' in the Galois field, not the transformation which can minimize $E(T, T')$. In such situation, it is impossible to estimate the right affine transformation by minimizing SAD and will not be taken into account in this chapter. In order to avoid such conditions, n_1 is limited in the experiments.

An approximate choice of n is needed in order to limit the maximum error to an acceptable range. However, size of Galois field grows exponentially with the increase of n. Typically, when $n = 8$, the total size of entire Galois field can be nearly $2.8e^{14}$. Considering a personal computer can not afford such a large amount of calculation, we will introduce our sampling method over $\mathbb{F}_{2^{6n}}$ in the next section.

Level-wise Adaptive Sampling (LAS)

In this section, we will introduce LAS which aims to achieve a satisfactory error rate instead of testing the complete $\mathbb{F}_{2^n}$. Our method is based on genetic algorithm (GA) Holland, 1975, which is biologically inspired. From the perspective of GA, our problem can be defined as a minimization problem of SAD. In crossover operation of GA, two coded individuals swap certain portions with each other. It is a good method to span search space around a sample point in multiple directions. However, in order to optimize $\hat{T}$ in such a broad search space, two major problems should be faced: 1) how to escape from local optimum. 2) how to control the optimization response time.

Preserving genetic variety: It has been argued in Hutter and Legg, 2006 that in order to prevent GA from falling into local optimum, genetic variety should be preserved somehow. Although mutation operation can surely increase the genetic variety randomly, it can also destroy individuals which are potentially to be close to $\hat{T}$. In a broad search space, the probability to create a "suitable" diversity is very low and mutation can contrarily slow down the speed of convergence. It is worth noting that in our problem, a large enough number of randomly initialized population keeps sufficient genetic variety for converging to $\hat{T}$. During the evolution, selection operation such as roulette wheel selection is likely to to select individuals which hold larger fitness for crossover operation. With the combination of selection and crossover, genetic variety decreases and the whole population converges to an optimum solution. However, if an individual happened to hold small SAD (*e.g.* a candidate area is flat) in the early stage of evolution, the whole population will easily fall into a local optimum especially when the search space is very broad. To preserve genetic variety, we select individuals from each SAD level uniformly. Each SAD level is a discrete interval which is occupied by a part of individuals. With maximum SAD in m th generation defined as S^m_{max}, minimum SAD in m th generation defined as S^m_{min} and the number of SAD level defined as σ, we can define i th SAD level as $[S^m_{min} + (i-1)(S^m_{max} - S^m_{min})/\sigma, S^m_{min} + i(S^m_{max} - S^m_{min})/\sigma]$. Each individual which is assigned to i th SAD level should have a fitness within this range. Individuals of next generation are then randomly selected from each SAD level. The number of individuals selected from each SAD level is the same. With the increase of σ, distribution of SAD in $m+1$ generation approximates to uniform distribution.

Fitness uniform selection scheme (FUSS) is proposed in Hutter and Legg, 2006, which selects a fitness value uniformly at first and then randomly select the nearest individual. The difference is, LAS can control the degree of uniform approximation by σ, which can directly affect the convergence speed. FUSS will take a longer time to converge, because the individuals with high fitness in FUSS make up only a small percentage of overall individuals.

Bounding population number: Evaluating a large number of population at initial generation is very important to avoid falling into local optimum. However, evaluating entire generations with same population number is time consuming and not practical. To accelerate the evolution procedure, we wish to rule out the candidate individuals which hold high SAD score. Instead of determining a fixed threshold, we learn a threshold at each generation which can rule out a certain fraction (λ percent) of individuals. Learning procedure is to adjust two constants α and β such that $S(I_1, I_2, T) < 0.1 \times \alpha + \beta$ holds for λ percent of the individuals. Figure 2.2 illustrates the matching frequency of each pixel in each generation. With the decrease of population number, matching frequency around each local optimum reduces, and finally the area with respect to global optimum transformation is localized. Involving the bounding scheme, the number of matching tests that LAS requires can be represented as:

$$\delta \sum_{i=0}^{m} \lambda^i, \delta\lambda^m > c. \tag{2.8}$$

δ is the population number of initial generation. c is a small constant which denotes the population number the last generation. The time complexity then can be ensured as long as the parameters are predetermined.

Approximation of SAD: Each matching test with respect to a single transformation has a time complexity of $O(n_1^2)$. To speed up each matching test, we wish to inspect only a small fraction of pixels instead of the entire pixels in template. We sample pixels at an equal interval on both width and height of template by a parameter ϵ to reduce the time complexity to $O(n_1^2/\epsilon^2)$. The Equation 2.3 can then be rewrote as following if the number of sampling pixels is enough.

$$S(I_1, I_2, T) \approx S(I_1', I_2, T), |I_1'| = n_1^2/\epsilon^2. \tag{2.9}$$

According to Chernoff bound, the number of sampling pixels should be $\log(1/\eta)n_1^2/\epsilon^2$ if we wish $|S(I_1, I_2, T) - S(I_1', I_2, T)| < \epsilon/n_1$ holds with prob-

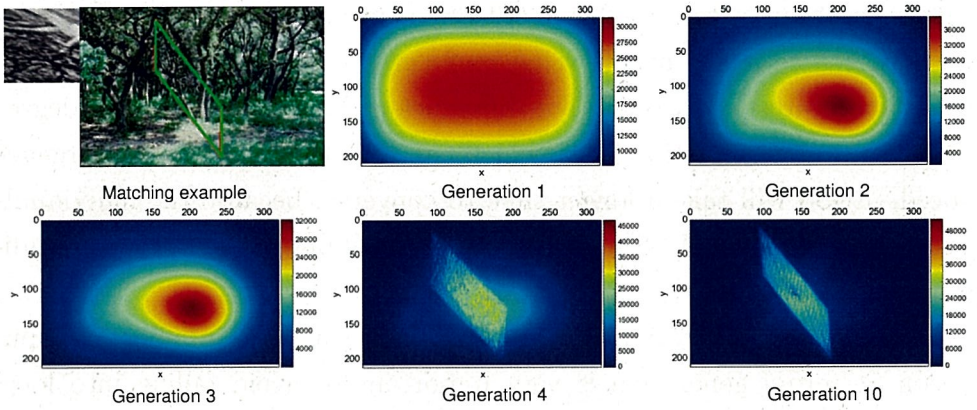


Figure 2.2: Heat map of matching frequency. This figure shows the frequency that each pixel has been used for calculating SAD. With the decrease of population number, the total matching frequency reduces while a more accurate candidate area can be localized.

ability $1 - \eta$. In our situation, $\eta = 1/e$. This also has been pointed out in Korman et al., 2013.

The entire procedure of LAS is described in Algorithm 2. All the transformations T are represented as binary Gray codes in Galois field. LAS runs in multiple generations, with each generation i generates a population $\mathbb{P}^i$. At first generation $\mathbb{P}^0$, individuals are sampled randomly from $\mathbb{F}_{2^n}$. Figure 2.3 illustrates the relation between SAD and the number of according individuals throughout the convergence process. With the generation number grows, the overall distribution translates from right to left as a result of selection and crossover. The amplitude decreases as a result of the population bounding scheme. Note that S_{min}^{2m+1} equals to S_{min}^{2m+2} and S_{max}^{2m+1} equals to S_{max}^{2m+2}, because the level-wise selection will not generate new solutions.

2.5 Experiments

To evaluate our algorithm, we use images from the famous SUN database Xiao et al., 2010, which has been used in evaluating many vision problems. We select 500 images as tests from category "waiting room" to "zoo". We randomly generate a ground truth affine transformation matrix for each test image, and make sure that the four corers of parallelogram generated by according matrix are all in the image. Pixels in the parallelogram are then warped to generate the square template. In our experiment, each

Algorithm 1 Level-wise Adaptive Sampling.

Require: Normalized template and target image, I_1, I_2.
Require: Population number δ of initial generation.
Require: Population bounding parameter λ.
Require: Population number c of last generation.
Ensure: Estimated transformation $\hat{T}$.

1: $\mathbb{P}^0 = \{T_0, ..., T_\delta\}$
2: $m = 0$
3: **while** $|\mathbb{P}^{2m}| > c$ **do**
4: $\quad \mathbb{P}^{2m+1} = \{T_i | T_i \in \mathbb{P}^{2m}, S(I_1, I_2, T_i) < \delta\lambda^m, S(I_1, I_2, T_i) \sim U(S^{2m+1}_{min}, S^{2m+1}_{max})\}$
5: $\quad \mathbb{P}^{2m+2} = crossover(\mathbb{P}^{2m+1})$
6: $\quad m = m + 1$
7: **end while**
8: **return** $\hat{T} \in \mathbb{P}^{2m+2}$ *s.t.* $S(I_1, I_2, \hat{T}) = S^{2m+2}_{min}$

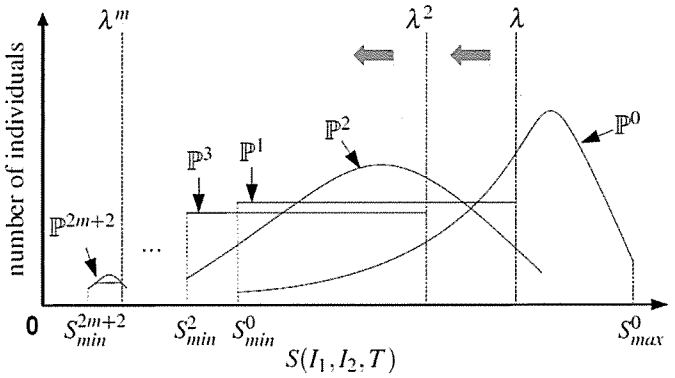


Figure 2.3: Illustration of SAD's distribution in each generation $\mathbb{P}$. Selection, bounding scheme and crossover on the individuals make the distribution move to left gradually, which is the procedure of estimating $\hat{T}$.

template has a size of 100 × 100 pixels.

Effect of parameters: We observe the change of SAD while changing the parameter δ and λ. Figure 2.4a shows that larger δ can improve the performance of SAD on the images which are not matched well using smaller δ. For the matching results which are close enough to the ground truth, it is hard to improve the performance by increasing δ. Figure 2.4b shows that small λ will only achieve rough results, because the algorithm converges too fast before a global optimum is localized. It is worth pointing out that even the ground truth transformations can have SAD larger than 0, because interpolation operations are involved during the creation of templates (warping). From Figure 2.4c, we can find out that our result

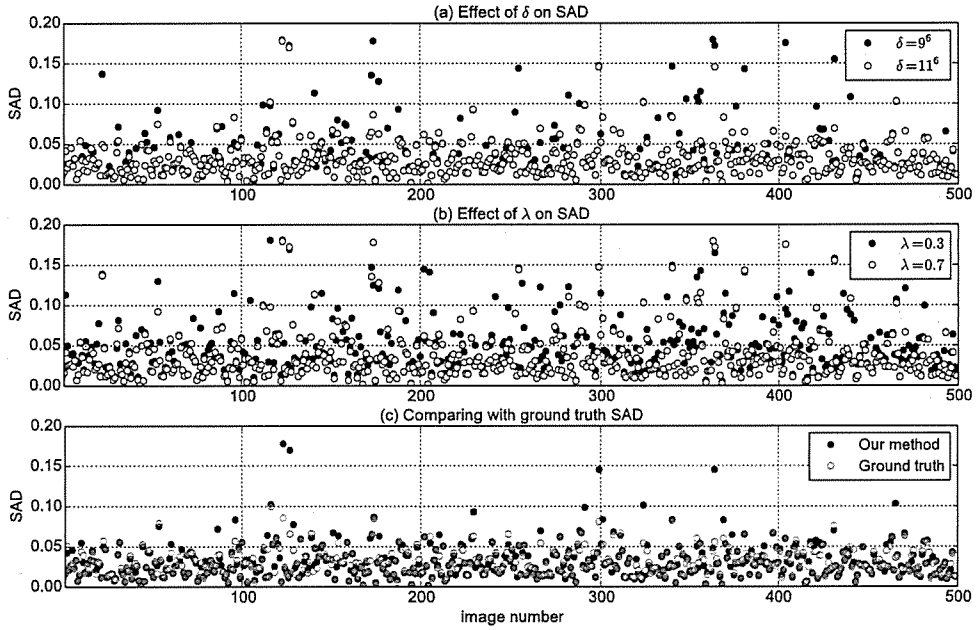


Figure 2.4: Parameter analysis on 500 images. (a) Effect of δ. Other parameters are: $\epsilon = 3, \lambda = 0.7$. (b) Effect of λ. Other parameters are: $\epsilon = 3, \delta = 9^6$. (c) Comparing tuned parameters with ground truth. Tuned parameters are: $\epsilon = 3, \delta = 11^6, \lambda = 0.7$.

can well fit the SAD of ground truth except mismatched tests.

Comparative results: We compare our algorithm with the sate-of-the-art method FAsT-Match Korman et al., 2013. We use overlap error to compare the accuracy which are defined as $1-(area(\hat{T}) \cap area(T'))/(area(\hat{T}) \cup area(T'))$ according to PASCAL measure (Everingham et al., 2010). We use number of matching tests to compare the efficiency which does not depend on type of programming languages and hard devices. In order to ensure the comparative results to be fair and accurate, the experiment is carried out under the following conditions: 1) No preprocessing like Gaussian blur. Although smoothing images will surely improve the accuracy, it will also bring complexities when analysing the results. 2) Set the approximation method of SAD as the same, number of sub-sampled pixels should be n_1^2/ϵ^2. 3) Set the number of matching tests as the same. It is difficult to control the number of matching tests of FAsT-Match, because it is dynamically determined. We only set its upper limit to avoid memory leak. 4) To keep the simplicity of algorithm, restarting an algorithm or other similar tricks for improving the accuracy are not allowed. From Figure 2.5a, we can see that with respect to different images, our method has a significant reduction on overlap error. From Figure 2.5b, we can see

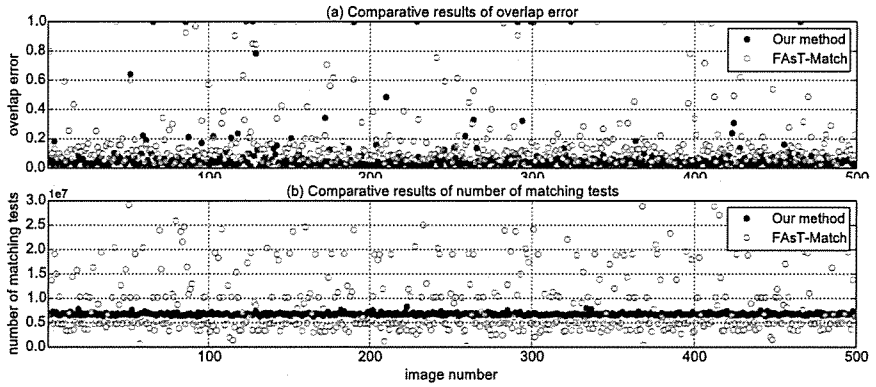


Figure 2.5: Comparative results with 500 images. Parameters are set as: $\delta = 11^6, \epsilon = 3, \lambda = 0.7$. (a) Overlap rate error on each test image. (b) Number of matching tests on each test image.

method	error ¡ 50%	error ¡ 10%	error ¡ 5%	average matching tests
FAsT-Match	92.2	48.4	11.8	8.6×10^6
Our method	97.4	91.0	68.0	6.7×10^6

Table 2.2: Accuracy of different overlap error criterion and average number of matching tests.

that our method is more stable in algorithm's complexity. By changing the criterion of overlap error, we report accuracy in Table 2.2.

We present examples of our matching results of Figure 2.5a in the Supplementary Material.

2.6 Conclusion

In this chapter, we presented a method to solve affine template matching problem in Galois field. For efficiency, we proposed level-wise adaptive sampling (LAS) method under genetic algorithm framework to estimate only a small fraction of candidate transformations. Experiments have shown that our algorithm is more accurate and faster than the state-of-the-art affine template matching method. The drawbacks of our algorithm can be concluded as: 1) The smooth assumption limits the application of our algorithm. For template with large variation, we have to increase δ. 2) Since GA bring about heuristics, there is no absolute assurance that our algorithm can find the global optimum by the limited matching tests. As the future work, we plan to extend our algorithm to projective template matching problem.

References

Akashi, Takuya et al. (2007). "Using genetic algorithm for eye detection and tracking in video sequence". In: *Journal of Systemics, Cybernetics and Informatics (JSCI)* 5.2, pp. 72–78.

Choi, Min-Seok and Whoi-Yul Kim (2002). "A novel two stage template matching method for rotation and illumination invariance". In: *Pattern Recognition (PR)* 35.1, pp. 119–129.

Everingham, Mark et al. (2010). "The pascal visual object classes (voc) challenge". In: *International journal of computer vision (IJCV)* 88.2, pp. 303–338.

Fischler, Martin A and Robert C Bolles (1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6, pp. 381–395.

Fredriksson, Kimmo, Veli Mäkinen, and Gonzalo Navarro (2007). "Rotation and lighting invariant template matching". In: *Information and Computation* 205.7, pp. 1096–1113.

Hartley, Richard and Andrew Zisserman (2003). *Multiple view geometry in computer vision*. Cambridge university press.

Holland, John H (1975). "Adaptation in natural and artificial systems. an introductory analysis with applications to biology, control and artificial intelligence". In: *Ann Arbor: University of Michigan Press* 1.

Hutter, Marcus and Shane Legg (2006). "Fitness uniform optimization". In: *IEEE Transactions on Evolutionary Computation (TEVC)* 10.5, pp. 568–589.

Kim, Hae Yong and Sidnei Alves de Araújo (2007). "Grayscale Template-matching Invariant to Rotation, Scale, Translation, Brightness and Contrast". In: *Pacific Rim Conference on Advances in Image and Video Technology (PSIVT)*. Springer-Verlag, pp. 100–113.

Korman, Simon et al. (2013). "FAsT-Match: Fast affine template matching". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 2331–2338.

Lowe, David G (2004). "Distinctive image features from scale-invariant keypoints". In: *International Journal of Computer Vision (IJCV)* 60.2, pp. 91–110.

Morel, Jean-Michel and Guoshen Yu (2009). "ASIFT: A new framework for fully affine invariant image comparison". In: *SIAM Journal on Imaging Sciences (SIIMS)* 2.2, pp. 438–469.

Xiao, Jianxiong et al. (2010). "SUN database: Large-scale scene recognition from abbey to zoo". In: *Computer vision and pattern recognition (CVPR)*. IEEE, pp. 3485–3492.

Chapter 3

TEMPLATE MATCHING WITH PROJECTIVE TRANSFORMATION

3.1 Summary

In this chapter, we address the problem of projective template matching which aims to estimate parameters of projective transformation. This work expands the application scope from affine transformation to projective transformation. Although homography can be estimated by combining key-point-based local features and RANSAC, it can hardly be solved with feature-less images or high outlier rate images. Estimating the projective transformation remains a difficult problem due to high-dimensionality and strong non-convexity. Our approach is to quantize the parameters of projective transformation with binary finite field and search for an appropriate solution as the final result over the discrete sampling set. The benefit is that we can avoid searching among a huge amount of potential candidates. Furthermore, in order to approximate the global optimum more efficiently, we develop a level-wise adaptive sampling (LAS) method under genetic algorithm framework. With LAS, the individuals are uniformly selected from each fitness level and the elite solution finally converges to the global optimum. In the experiment, we compare our method against the popular projective solution and systematically analyse our method. The result shows that our method can provide convincing performance and holds wider application scope.

3.2 Introduction

Parametric template matching has been studied for decades as a classical problem. Among different deformation and transformation models, projective transformation is one of the most common transformations that occurs between images. However, projectivities, which are the transformations within and between projective spaces, are hard to be estimated correctly due to high-dimensional parameters. In many real-world matching scenarios, there usually exists a projectivity between a template and a target image (i.e., source image) since the projectivity between real object plane and template is usually different from the projectivity between real

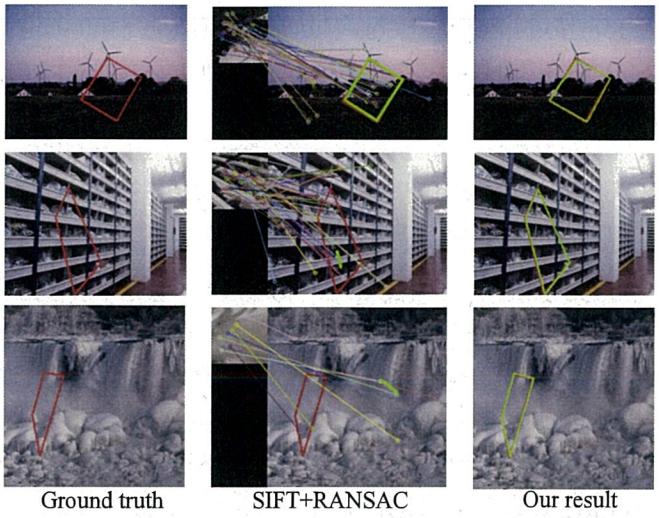


Figure 3.1: Matching examples. As we can see, SIFT+RANSAC can only handle the first example with slight projective transformation while our method matches all the ground truths well.

object plane and target image. The purpose of this work is to estimate the projectivity between two planar images: the template image and the target image. Specifically, in projective template matching, each candidate projectivity corresponds to a candidate polygon area in the target image. We aim to find a polygon which is most visually similar with the template image after eliminating the effect of transformation.

As a standard framework, local-feature-based methods such as SIFT and its variants are very efficient to estimate the 2D homography (projectivity) between the template and the target. Parameters of projectivity can be solved by a system of linear equations which can be written from a few inliers (i.e., correctly matched key points). We can also use method like RANSAC Fischler and Bolles, 1981 to eliminate the effects of outliers (i.e., incorrectly matched key points). However, there still exist some limitations in this framework: 1) For feature-less images, like medical images, key points are hard to be detected. Without inliers, projectivity cannot be solved. 2) Common local features (e.g., SIFT, ASIFT) are susceptible to projective transformation, so it is necessary to design a projective transformation invariant local feature. Affine-SIFT Morel and Yu, 2009 can handle matching with affine transformation but can hardly handle the projective transformation. 3) For images with heavy outliers, like noisy images, it is difficult to estimate the proper projectivity. In conclusion, the success of

feature-based methods depends on the assumption that the matching result of key point matching consists of inliers (at least four). Figure 1 shows three matching examples.

As we all know, the core drawback of template matching is that it potentially requires a huge number of candidate samples to evaluate in order to reach to the global optimum. In the case of projective template matching, it exponentially requires more computational cost at the time that more accurate parameters are required to be estimated. The reason is that eight degrees of freedom (DoFs) are required for defining a projective transformation. Due to this drawback, few existing works attempt to employ dense template matching directly with projective transformation. To make up for the drawback, how to search the candidate space effectively becomes an essential point in this paper. We quantize the eight DoFs of projective transformation with a finite set and then propose a meta-heuristic method to approach the global optimum effectively. The main contributions of this paper can be concluded as following:

- Overall, this paper proposes a solution to a long standing problem of projective template matching.
- We apply binary finite field to deal with very large DoF.
- We develop a new selection method called LAS to preserve the diversity under genetic algorithm framework, while keep the efficiency.

3.3 Related Work

The difficulty of template matching tasks increase as the dimension of DoF grows. Figure 3.2 shows some common transformation models with various dimensions of DoF. In this section, we mainly survey previous works involving these geometric transformations. Note that a simplest transformation model with two DoFs only involves translations of x-axis and y-axis.

Euclidean Template Matching

The dimension of DoF is three. Rotation and translation are considered in this transformation model. The result area in the target image is rectangular and kept rigid. Traditional approaches Brown, 1992 for solving this problem is to compute the correlation between each candidate and template which can be accelerated with fast Fourier transformation (FFT).

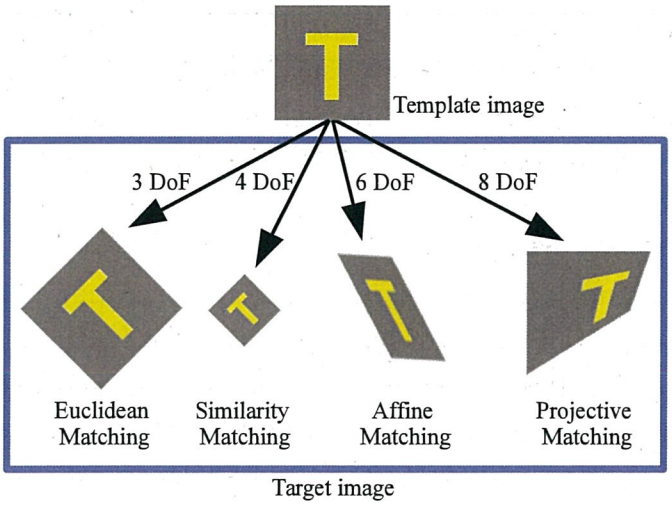


Figure 3.2: Template matching tasks with different transformation models. As the dimension of DoF grows, the matching difficulty also increases.

Low dimensional DoF allows exhaustive searching to some extent. Also, many efforts have been done to improve the performance of FFT. Generalized Fourier transform Nair, Rajagopal, and Wenzel, 2000 is one of the alternatives which offers a relatively robust and fast solution to the matching problem. On the other hand, rotation invariant features are considered as more feasible instead of exhaustive matching. Choi and Kim Choi and W.-Y. Kim, 2002 proposed a method which combines both the projection method and Zernike moments in two stages. At first stage, candidates with low cost feature extracted are selected. At second stage, rotation invariant matching is performed.

Similarity Template Matching

By involving the overall scaling variable, the dimension of DoF grows to four. This model is most widely applied in real-world applications. Exhaustive searching is no longer feasible due to the broad searching space. To improve the efficiency of matching process, Kim et al. H. Y. Kim and Araújo, 2007 proposed cascaded filters to exclude the candidates which hold low probability to be selected as the final result. Penate-Sanchez et al. Penate-Sanchez, Porzi, and Moreno-Noguer, 2015 treat the "probability" as "matchability", and apply dense convolutional neural network to learn and predict the matchability in advance. Hence, large amounts of unnecessary computations can be avoid. In Zhang and Akashi, 2015c,

the authors introduce a heuristic method which can sample the candidates adaptively by utilizing the property that the left-most bit of a binary-coded candidate affects the binary code most significantly and vice versa. Scaling, translation, and rotation are considered in the above works.

Affine Template Matching

Fewer works study on the affine matching since shearing is involved and the dimension of DoF grows to six. Affine invariant feature Morel and Yu, 2009 made a breakthrough on this problem. However, researches on direct matching are still needed since inliers with A-SIFT can not always be guaranteed, especially for feature-less images. FFT has also been extended for affine invariant matching Gundam and Charalampidis, 2015. In this work, template is first decomposed into non-overlapping concentric circular rings, and each ring's FT is calculated. Parameters of affine are then estimated under the assumption that rings may be rotated with respect to each other. S. Korman et al. Korman et al., 2013 proposed a method which matches the template in a very sparse way. A parameter-depended discrete sampling net is constructed and a branch-and-bound scheme is employed to search an approximate solution over the sampling net. C. Zhang and T. Akashi Zhang and Akashi, 2015a proposed a stochastic method to search the 2D affine parameters efficiently with SAD.

Projective Template Matching

With projection involved, the dimension of DoF grows to eight. Projective invariant feature has not been well developed yet and applying common local descriptors directly like SIFT will lead to a large amount of outliers. For direct methods (pixel-based methods), due to the high-dimensionality and high non-convexity, limited related literature can be found. Instead of global matching, F. Jurie et al. Jurie and Dhome, 2002 proposed a tracking-based matching approach which can greatly reduce the number of candidates. It can deal with projective transformation to a certain extent. However, the problem setting in this paper is more close to online tracking like Zhang, Yamagata, and Akashi, 2015 rather than template matching. To the best of our knowledge, our work first attempts to solve the global dense template matching problem under projection transformation model.

3.4 Methodology

Problem Description

Two grayscale images I_1 and I_2 are given as the input. Each pixel's gray value is normalized from [0, 255] to [0, 1]. Here, I_1 represents for $n_1 \times n_1$ template image and I_2 represents for $n_2 \times n_2$ target image, $n_1, n_2 \in \mathbb{N}^+$. For the convenience of denotation, here we assume that both I_1 and I_2 are square images. An arbitrary pixel $p \in I_1$ is mapped to I_2 via projectivity $\tau \in \mathbb{R}^{4\times4}$, which will be further defined in Section 3.4. We use p^τ to denote the mapped pixel in I_2.

For simplicity, sum of absolute difference (SAD) is used to measure the similarity between I_1 and a candidate area $I_c \in I_2$. With SAD defined, we can focus on studying the searching mechanism rather than feature extraction. Candidate area I_c corresponds to a candidate projectivity τ. Normalized SAD is utilized, which can be formally defined as:

$$S(I_1, I_2, \tau) = \frac{\sum_{p\in I_1} |I_1(p) - I_2(p^\tau)|^m}{n_1^2}, m = \begin{cases} 0 & p^\tau \notin I_2 \\ 1 & p^\tau \in I_2 \end{cases}. \tag{3.1}$$

The purpose of this paper is to estimate an approximate projectivity $\hat{\tau}$ which is a member of a sampling set. The sampling set can be constructed by finite set $\mathbb{F}$ which will be introduced in the following Section 3.4. In the case of the best result is achieved, $\hat{\tau} \in \mathbb{F}$ equals to $\overline{\tau} \in \mathbb{F}$. Projectivity $\overline{\tau}$ is the closest one to ground truth τ^* among all the candidate projectivities in $\mathbb{F}$. Because we use a discrete sampling set of transformation to approach the full continuous set of transformation, there is a strong possibility that the best transformation in the discrete sampling set does not equal to the best transformation in the full continuous set. The best transformation in the full continuous set is the ground truth τ^* and the best transformation in the discrete sampling set is $\overline{\tau}$, which is supposed to be the closest one to τ^*. With SAD defined, $\hat{\tau}$ can be indirectly estimated by minimizing the SAD:

$$\hat{\tau} = \arg\min_{\tau\in\mathbb{F}} S(I_1, I_2, \tau). \tag{3.2}$$

Geometric Model

In this section, we detailedly define the τ. 2D projectivity is widely utilized in multi-view geometry. The projectivity is defined as a non-singular 3×3 matrix. It at least needs four pairs of inliers to solve the eight parameters (projectivity is a homogeneous matrix and it only has eight DoFs

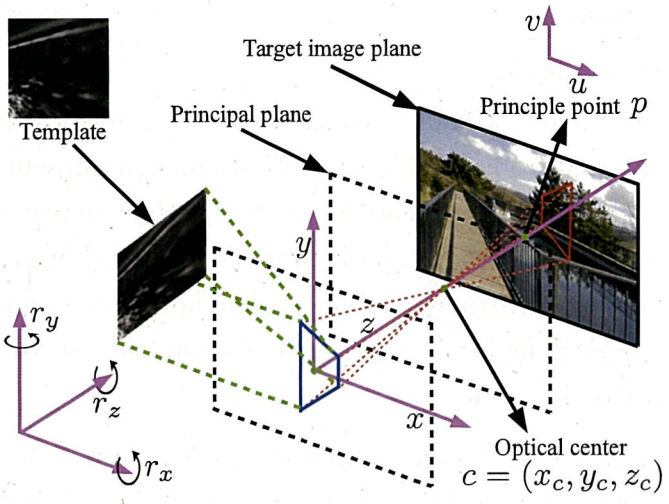


Figure 3.3: Pin-hole camera model. Template image is placed in world coordinate system, it is projected onto the target image plane by drawing the lines through the camera center C. By changing the appearance of template image in the world coordinate system, various polygon candidates can be observed in the target image plane. Tunable parameters of τ includes rotation r_x, r_y, r_z, translation x, y, scaling s_x, s_y and distance of principal plane z_c. Parameters z_x and z_y are set to zero.

even it has nine elements). Typically, parameters are estimated by finding the correspondences between template and target. However, in our case, it is hard to solve the projectivity directly since we optimize the τ with the feedback of SAD. Hence, each parameter must be assigned a meaning in our algorithm in order to tune the parameters within bounded ranges. Since projectivity is resulted by a chain of transformations, we decompose it into multiple transformation matrices instead of estimating the fused parameters directly. We model the τ under pin-hole camera geometry. As shown in Figure 3.3, varying the eight parameters can change the appearance of the template's outer contour observed by the pin-hole camera and

Table 3.1: List of each parameter's real number range, the step size and amount in sampling set.

Parameter	Range	Step amount	Step size
r_x	$[-\pi/2, \pi/2]$	2^n	$\pi/2^n$
r_y	$[-\pi/2, \pi/2]$	2^n	$\pi/2^n$
r_z	$[-\pi/2, \pi/2]$	2^n	$\pi/2^n$
x	$[-n_2/2, n_2/2]$	2^n	$n_2/2^n$
y	$[-n_2/2, n_2/2]$	2^n	$n_2/2^n$
s_x	$[n_1/n_2, n_2/n_1]$	2^n	$(n_2^2 - n_1^2)/n_1 n_2 2^n$
s_y	$[n_1/n_2, n_2/n_1]$	2^n	$(n_2^2 - n_1^2)/n_1 n_2 2^n$
z_c	$[0, +\infty]$	-	-

thus generate various candidate areas in the target image. Formally,

$$\begin{aligned}\tau = &\begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos r_x & \sin r_x & 0 \\ 0 & -\sin r_x & \cos r_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \\ &\begin{bmatrix} \cos r_y & 0 & -\sin r_y & 0 \\ 0 & 1 & 0 & 0 \\ \sin r_y & 0 & \cos r_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos r_z & \sin r_z & 0 & 0 \\ -\sin r_z & \cos r_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \\ &\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x & y & 0 & 1 \end{bmatrix} \times \begin{bmatrix} z_c & 0 & 0 & 0 \\ 0 & z_c & 0 & 0 \\ -x_c & -y_c & 0 & -1 \\ 0 & 0 & 0 & z_c \end{bmatrix}.\end{aligned} \tag{3.3}$$

Where r_x, r_y, r_z are the rotation angles with respect to each axis. Parameter x, y are translation parameters with respect to both target image plane and principle plane. Target image plane is parallel to the principle plane. Parameter s_x, s_y are scaling factors with respect to x-axis and y-axis. We set x_c and y_c to zero thus the target image plane is concentric with the principal plane. Parameter z_c is the z-axis's value of the optical center. During implementation, z_c is fixed as a positive integer since it can be positive infinity and unlimitable. A reasonable z_c is hard to be determined since larger z_c will lead to smaller candidate area in the target image plane and vice versa. In the other words, not only the scaling parameters s_x and

s_y but also the z_c can affect the size of a candidate polygon. As long as the z_c is fixed, the size of a candidate can be specifically tuned by s_x and s_y. However, if we choose a relatively large z_c, let us say $z_c = 1000$, the size of a candidate will be very small no matter how we tune the s_x and s_y within the given range $[n_1/n_2, n_2/n_1]$. Parameter z_c plays a role on limiting the extremity of the perspectivities. The decision of z_c depends on the size of an object you expect to observe from the target images in the specific applications. At the same time, matching insignificantly small regions is impractical in real-world applications. We suggest that the range of z_c should be determined empirically depend on specific applications. Each parameter's range is shown in Table 3.1.

With τ defined, we can calculate the p^τ by simply multiply the matrices:

$$p^\tau = p\tau, \text{where } p \in \mathbb{R}^{1\times 4}, \tau \in \mathbb{R}^{4\times 4}. \tag{3.4}$$

Note that p and p^τ are represented by homogeneous coordinate, specifically, $p = (p_x, p_y, 0, 1)$. When calculating the coordinate of p^τ, p^τ should also be converted to homogeneous form, that is, the value of fourth dimension in p^τ should be normalized to one.

Finite Field of Projective Transformation

Matching with complete continuous projective transformation set which contains infinite candidates can be impractical. To avoid this problem, we build a discrete set with binary finite set. We extend the smallest finite set $F_2 = \{0, 1\}$ to $\mathbb{F}_{2^n}$, where $n \in \mathbb{N}^+$ represents the length of each binary code and implicitly corresponds to the accuracy of sampling, $2^n = |\mathbb{F}_{2^n}|$. Each $\tau \in \mathbb{F}_{2^n}$ is coded by $7n$ bit binary code. Parameter over each decomposed DoF can be discretely sampled by each independent finite set. For denotation clarity, we assume that n in each finite set is the same. Table 3.1 shows the step size and amount when finite set is used to construct the sampling set. Each DoF's range is then divided into discrete segments.

To analyze whether minimizing SAD can help the algorithm reach to the best transformation in the discrete sampling set, we discuss what kind of factors will affect the error bound. With Equation 3.2, we can still not ensure the approximate solution $\hat{\tau}$ is close enough to $\overline{\tau}$ due to the drawback of SAD. An important factor which will affect the "approximate degree" is the variation υ Korman et al., 2013 of the template. Variation υ can

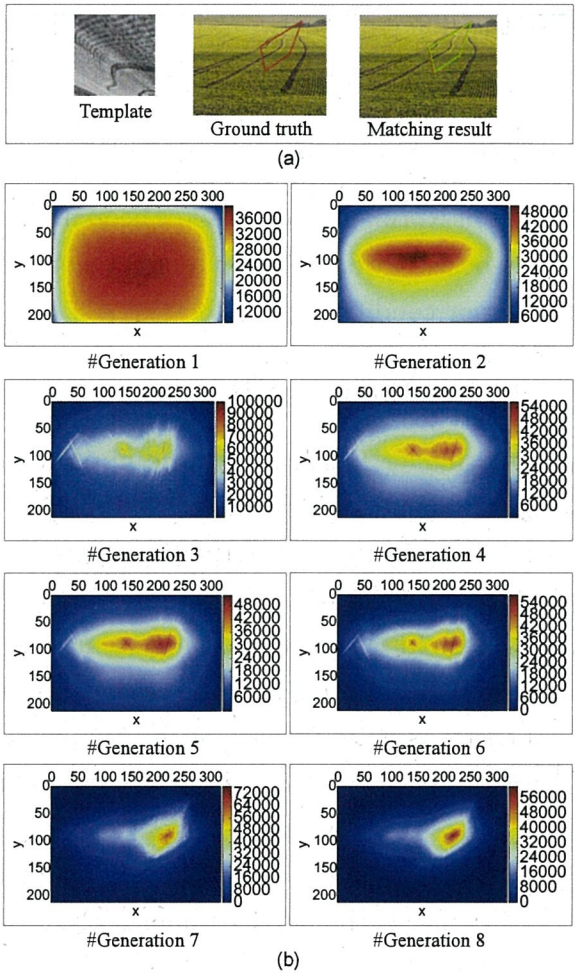


Figure 3.4: a) Example of projective matching; b) Heat map of matching frequency of each generation 1 to 8 with respect to example (a). This figure illustrates the frequency of each pixel that has been used to calculate the SAD during the matching procedure. With the increase of generation number, pixels close to the global optimum are more frequently to be matched while the total sum of matching frequency decrease, which is an important property in LAS.

be defined as the sum of maximum difference between each p and its eight neighbors $N_8(p)$. Formally,

$$\upsilon = \sum_{p \in I_1} \max_{q \in N_8(p)} |I_1(p) - I_1(q)|. \tag{3.5}$$

If we refer to simple cases especially when template is an edge image or other cross-domain imagesZhang and Akashi, 2015b, applying SAD directly will lead to a high υ, which means that slight translation near the ground truth area will yield large difference (i.e., the solution space is not smooth).

In order to qualify the difference between two candidate projectivities τ_1 and τ_2, we introduce the following projectivity error:

$$E(\tau_1, \tau_2) = |S(I_1, I_2, \tau_1) - S(I_1, I_2, \tau_2)|. \tag{3.6}$$

It has been addressed in Korman et al., 2013 (Theorem 3.1) that the upper limit of $E(\overline{\tau}, \tau^*)$ is associate with the size of template, variation of template, and step amount. In our condition,

$$E(\overline{\tau}, \tau^*) < O(\frac{\upsilon}{n_1 2^{7n}}). \tag{3.7}$$

Larger υ, smaller n_1, smaller n will loosen the upper bound of Equation 3.7. With loose upper bound, there exists case that $E(\hat{\tau}, \tau^*) < E(\overline{\tau}, \tau^*)$ while $\hat{\tau}$ is far away from $\overline{\tau}$ in the transformation space. In this case, minimizing the SAD only can never reach to the most approximate candidate in the sample set. To avoid this case, we limit n_1 in this paper since the variation υ is uncontrollable and n should be reasonably small considering the computational cost. As n_1 is the size of the template, we can fix it to avoid to be too small (e.g., $n_1 > 100$). Typically, in our implementation, we set $n = 8$. The size of the sampling set is $2^{7n} \approx 7.2 \times 10^{16}$. Obviously, it is hard for a personal computer to afford such a large-scale computation task. Instead of evaluating each sample exhaustively, we propose LAS to adaptively select the samples to evaluate, which will be introduced in the next section.

Level-wise Adaptive Sampling (LAS)

In this section, we introduce LAS which aims to reach the approximate solution instead of testing the complete $\mathbb{F}_{2^{7n}}$. To optimize τ based on the complete $\mathbb{F}_{2^{7n}}$, two main problems should be considered: 1) How to escape from local optimums; 2) How to guarantee the optimization response time. Because our methods is based on the genetic algorithm (GA) framework Holland, 1975, we inherit terms of GA such as population, generation, etc, which are briefly introduced as following:

- An individual is a candidate transformation to which the SAD function can be applied.
- A population is a set of individuals.
- Each successive population in an iteration is called a generation.

- Diversity refers to the average distance between individuals in a population.
- Crossover is genetic operator to recombine portions of individuals.

LAS reduces the population number as the generation number grows and the individuals are selected uniformly from each level of fitness value. At first, we refer to a simple example in order to give out the whole operational impression of our method, which is illustrated in Figure 3.4. This figure illustrates the heat map of each pixel's matching frequency through generation one to eight, we can observe two important properties of LAS from this figure: A) The total number of matching frequency decreases with the increase of generation number, which means that the computational cost has been adaptively reduced; B) The number of matching frequency with respect to pixels which are close to the global optimum increases, which means that our algorithm adaptively selects "hopeful" samples to evaluate rather than exhaustive searching.

For problem 1): preserving diversity. It has been argued in Hutter and Legg, 2006 that diversity should be preserved somehow if we want to prevent GA from falling into local optimums. Mutation operation can increase the diversity randomly and has been regarded as a typical process in GA. However, mutation can also randomly destroy individuals which are potentially closing to $\hat{\tau}$. In a broad search space, the probability of generating a "just right" diversity is very low and the mutation process may slow down the speed of convergence and be counterproductive. It is worth noting that in our problem, a large number of randomly initialized individuals is able to keep sufficient diversity for the algorithm converging to $\hat{\tau}$. Throughout the evolution process, classical selection method such as roulette wheel selection is more prone to select individuals which hold smaller SAD for crossover operation. With the combination of selection and crossover, diversity decreases and the whole population converges to an optimum solution. However, if an individual happened to hold small SAD (e.g., a flat candidate area) at the early stage of evolution, then the whole population will fall into a local optimum easily, especially when the sample space is very broad.

As a key to solve this problem, we select individuals from each SAD level uniformly. Each SAD level is a discrete interval which is occupied by a

part of individuals. With maximum SAD in m th generation denoted as S^m_{max}, minimum SAD in m th generation denoted as S^m_{min} and the number of SAD level denoted as nl, we can denote the range of i th SAD level as $[S^m_{min}+(i-1)(S^m_{max}-S^m_{min})/nl, S^m_{min}+i(S^m_{max}-S^m_{min})/nl]$. Each individual which is assigned to i th SAD level should hold a SAD value within this range. Individuals of next generation are then randomly selected from each SAD level. The number of individuals selected from each SAD level is the same. With the increase of nl, distribution of SAD in $m+1$ generation approximates to uniform distribution.

Fitness uniform selection scheme (FUSS) is proposed in Hutter and Legg, 2006, which selects a fitness value uniformly at first and then randomly select the nearest individual. The difference is, LAS can control the degree of uniform approximation by nl, which can directly affect the convergence speed. FUSS will take a longer time to converge, because the individuals with high fitness in FUSS make up only a small percentage of overall individuals.

For problem 2): limiting population size. Evaluating a large size of population at initial generation is very important to avoid to fall into local optimum. However, evaluating entire generations with same population size is time consuming and not practical. To accelerate the evolution procedure, we wish to rule out the candidate individuals which hold high SAD score. Instead of determining a fixed threshold, we learn a threshold at each generation which can rule out a certain fraction (λ percent) of individuals. Learning procedure is to adjust two constants α and β in different order of magnitude such that $S(I_1, I_2, \tau) < 0.1 \times \alpha + \beta$ holds for λ percent of the individuals. Specifically, parameters α and β are initialized to 0, thus the threshold is initialized to $0.1 \times 0 + 0 = 0$. Parameters α and β are increased by loops [0.1, 0.2, ..., 0.9] and [0.01, 0.02, ..., 0.1] respectively, and the value of object function $|\frac{\#\{individuals|SAD<treshold\}}{\#individuals} - \lambda|$ is observed. When the object function stops reducing, the best threshold is achieved. Involving the limiting scheme, the number of matching tests that LAS requires can be represented as:

$$\delta \sum_{i=0}^{m} \lambda^i, \delta\lambda^m > c. \tag{3.8}$$

Parameter δ is the population size of initial generation. Parameter c is a small constant which denotes the population size of the last generation.

The time complexity then can be ensured as long as the parameters are predetermined.

Approximation of SAD: Each matching test with respect to a candidate projectivity has a time complexity of $O(n_1^2)$. To speed up each matching test, we wish to inspect only a small fraction of pixels instead of the entire pixels in template. We sample pixels at an equal interval on both width and height of template by a parameter ϵ to reduce the time complexity to $O(n_1^2/\epsilon^2)$. The Equation 3.1 can then be rewrote as following if the number of sampling pixels is enough.

$$S(I_1, I_2, \tau) \approx S(I_1', I_2, \tau), |I_1'| = n_1^2/\epsilon^2. \tag{3.9}$$

According to Chernoff bound, the number of sampling pixels should be $\log(1/\eta)n_1^2/\epsilon^2$ if we wish $|S(I_1, I_2, \tau) - S(I_1', I_2, \tau)| < \epsilon/n_1$ holds with probability $1 - \eta$. In our situation, $\eta = 1/e$. This also has been pointed out in Korman et al., 2013.

The entire procedure of LAS is described in Algorithm 2. In row 4 of the algorithm, U means uniform and $S(I_1, I_2, \tau_i) \sim U(S_{min}, S_{max})$ indicates that the value of $S(I_1, I_2, \tau_i)$ is equally likely to be observed from S_{min} to S_{max}, which reflects that individuals are equally sampled from each SAD level. All the projectivities τ are represented as binary codes in finite set. LAS runs in multiple generations, with each generation i generates a population $\mathbb{P}^i$. At first generation $\mathbb{P}^0$, individuals are sampled randomly from $\mathbb{F}_{2^n}$. Figure 3.5 illustrates the relation between SAD and the number of according individuals throughout the convergence process. With the generation number grows, the overall distribution translates from right to left as a result of selection and crossover, which also means that SAD of the elite decreases. On the other hand, the amplitude decreases as a result of the population limiting scheme.

3.5 Experiment

Experiment Environment

We construct a benchmark inherited from the benchmark used in Zhang and Akashi, 2015a to evaluate our method. The original images are from the famous SUN dataset Xiao et al., 2010. Each test image in this benchmark corresponds to a randomly generated projective transformation with $z_c = 30$. To generate a random projectivity, at first, four points within a

Algorithm 2 Level-wise Adaptive Sampling.

Require: Template and target image, I_1, I_2.
Require: Population size δ of initial generation.
Require: Population limiting parameter λ.
Require: Population size c of last generation.
Ensure: Estimated transformation $\hat{\tau}$.

$\mathbb{P}^0 = \{\tau_0, ..., \tau_\delta\}$
$m = 0$
while $|\mathbb{P}^{2m}| > c$ **do**
    $\mathbb{P}^{2m+1} = \{\tau_i | \tau_i \in \mathbb{P}^{2m}, S(I_1, I_2, \tau_i) < 0.1 \times \alpha + \beta, S(I_1, I_2, \tau_i) \sim U(S_{min}^{2m+1}, S_{max}^{2m+1})\}$
    $\mathbb{P}^{2m+2} = crossover(\mathbb{P}^{2m+1})$
    $m = m + 1$
end while
return $\hat{\tau} \in \mathbb{P}^{2m+2}$ *s.t.* $S(I_1, I_2, \hat{\tau}) = S_{min}^{2m+2}$

test image are selected. Secondly, we warp the polygon constructed by four points into square template. We avoid to generate polygons which have too small area. Overall, there are 100 pairs of template and target images in this benchmark with various image size. All the ground truth bounding boxes are defined by the four points which are randomly selected at first.

We apply overlap rate to judge whether a matching result is successful by referring to the ground truth. Specifically, PASCAL criteria Everingham et al., 2010 is used to calculate the overlap rate:

$$\text{overlap rate} = \frac{\text{area}(BB_{re} \cap BB_{gr})}{\text{area}(BB_{re} \cup BB_{gr})}. \tag{3.10}$$

Where BB_{re} means polygon bounding box of result and BB_{gt} means polygon bounding box of ground truth. area$(\cdot)$ is a function to count number of pixels. Based on the overlap rate, we can determine whether a matching result is correct or wrong by setting a threshold. Specifically, we have

$$\text{answer} = \begin{cases} 1 & \text{if overlap rate} > \text{threshold} \\ 0 & \text{otherwise} \end{cases}. \tag{3.11}$$

Finally, success ratio $= \#\{\text{answer}|\text{answer} = 1\}/\#\text{test}$ as the accuracy criterion.

Comparison

To the best of our knowledge, SIFT and its invariants are the most stable, and widely-used feature descriptors in dealing with viewpoint changes during matching problems. However, they can only handle small viewpoint

changes which mean a very narrow range in the projective transformation space. Although comparing with SIFT is not fair since it is not a completely projective invariant feature, we take it as a baseline method. Figure 3.6(a) shows the success curves of our method and SIFT+RANSAC. We increase the threshold of overlap rate in x-axis while observing the change of success rate. As we can see, the success rate keeps larger than 90% until the threshold gets to 0.6. Even when the threshold is limited to 0.9, which is very strict and visually make little difference between result area and ground truth area, we can reach a success rate of 60%.

On the other hand, we compare the SAD value against ground truth case by case, which has been shown in Figure 3.6(b). Since we generate templates by warping the polygon area which is determined by four randomly generated points with interpolation, the $S(I_1, I_2, \tau^*)$ cannot exactly be zero. Except several images which are mismatched, most of the test images show close SAD value with the ground truths.

Effect of parameters

Several parameters are considered most likely to affect the success curve. They are δ, ς, ϵ, nl, λ and σ as shown in the caption of Figure 6.5 and previous content. To analyse how each parameter affect the success curve, we fix other parameters while observing one parameter. All the results are achieved under a same random seed. From Figure 6.5(a), we can see that at least 1.5×10^6 initial population size is needed in order to provide acceptable performance. With bad initialization, the algorithm is easy to fall into local optimums since the crossover operation can only generate new individuals covering a small range around each individual. In other words, the "gap" between individuals in the first generation are large when δ is small, many candidates in the "gap" are hard to be reached which may contain the global optimum. From Figure 6.5(b), we can see that crossover rate does not affect the success curve much and the best performance can be achieved when $\varsigma = 0.8$. From Figure 6.5(c), we can see that the best performance is achieved when ϵ is smallest. However, the accuracy does not reduce in proportion to the the decrease of ϵ, which means that taking more pixels' information into account when calculating SAD does not guarantee the improvement of accuracy. From Figure 6.5(d), we can see that the levels of SAD does not affect the accuracy much. From Figure 6.5(e), we can see that the accuracy improves in proportion to the the increase of λ.

High λ means that the decay rate of population size is small and larger amount of individuals will be evaluated. Hence, high λ results in better performance. Considering computational cost, we set $\lambda = 0.7$ in Figure 3.6. From Figure 6.5(f), we can see that like many matching problems, smooth parameter has great influence on accuracy. An appropriate choice is $\sigma = 3.0$.

Figure 3.10 shows some examples of succussed matching. We can observe that even feature-less templates or drastically warped can be correctly matched.

Noise-tolerance Experiment

In this section, we will check how our algorithm deals with Gaussian noise. It is important to see at which levels of noise the method can still work in order to test the robustness to real-world matching conditions. With Gaussian noise, each pixel in the image will be changed from its original value by a small amount. A histogram, a plot of the amount of distortion of a pixel value against the frequency with which it occurs, shows a Gaussian distribution of noise. The level of the noise is controlled by the expectation (set to 0 in the experiment) and the standard deviation (set to [0.01, 0.03, ..., 0.09] of Gaussian distribution. We apply the parameter setting for plotting Figure 3.6 to match and draw curves of each noise level in Figure 3.9. Figure 3.8 shows the illustration of different levels of noise and the corresponding matching results. From Figure 3.9, we can see that with the increase of noise, the matching accuracy decreases. However, even with the heaviest noise, our algorithm can still achieve a AUC of 0.67, which is better than the AUC of applying ASIFT without any noise (AUC=0.65) that has been shown in Figure 3.6.

Processing Time

In this section, we provide processing time of two kinds of parameter setting as a reference in Table 3.2. We test the processing time of our unoptimized code with a laptop equipped with Intel Core i7-4600M 2.90GHz CPU and 16 GB RAM. From Table 3.2 we can see that increasing approximation parameter ϵ can drastically reduce the processing time, while keep a satisfactory matching accuracy. Each template's size is 100×100 pixels and each target image's size is 320×240 pixels during the test. In real-world applications, we can further reduce the processing time by limiting the ex-

tremity of the projectivities. We can limit the range of parameter s_x, s_y and especially z_c to limit the extremity of the projectivities. For a specific application, we can empirically determine the ranges of these parameters to get rid of redundant candidates that will never appear in real world.

Table 3.2: Average processing time of tuned parameter settings.

AUC	processing time	δ	ς	ϵ	nl	λ	ς
0.88	206.4 sec/image	2.5×10^6	0.8	3	5	0.7	3.0
0.83	59.5 sec/image			8			

3.6 Conclusion

In this paper, we propose level-wise adaptive sampling (LAS) to optimize parameters of projective transformation in template matching problems. At first, We use binary finite set to construct a discrete sampling set. Then, instead of exhaustive searching, LAS selects the individuals in different fitness levels and thus can preserve the diversity better and help the algorithm to converge to the global optimum. The results on the benchmark show the efficiency of our method and many practical applications can be expanded based on this.

On the other hand, several drawbacks of this method can be concluded as: 1) Due to smoothness assumption, template with high variation value can cause failure in matching. Increasing initial population size can solve problem if reducing computational cost is not a priority. 2) Several minutes (1$\sim$3 minutes) are needed for producing a result with the best-performance-parameters and images in the benchmark. There still remains distance away from real-time applications. As the future work, we aim to accelerate the algorithm and expect real-world applications.

References

Brown, Lisa Gottesfeld (1992). "A survey of image registration techniques". In: *ACM computing surveys (CSUR)* 24.4, pp. 325–376.

Choi, Min-Seok and Whoi-Yul Kim (2002). "A novel two stage template matching method for rotation and illumination invariance". In: *Pattern Recognition (PR)* 35.1, pp. 119–129.

Everingham, Mark et al. (2010). "The pascal visual object classes (voc) challenge". In: *International journal of computer vision (IJCV)* 88.2, pp. 303–338.

Fischler, Martin A and Robert C Bolles (1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6, pp. 381–395.

Gundam, Madhuri and Dimitrios Charalampidis (2015). "Fourier transform-based method for pattern matching: affine invariance and beyond". In: *SPIE Defense+ Security*. International Society for Optics and Photonics, pp. 94770I–94770I.

Holland, John H (1975). "Adaptation in natural and artificial systems. an introductory analysis with applications to biology, control and artificial intelligence". In: *Ann Arbor: University of Michigan Press* 1.

Hutter, Marcus and Shane Legg (2006). "Fitness uniform optimization". In: *IEEE Transactions on Evolutionary Computation (TEVC)* 10.5, pp. 568–589.

Jurie, Frédéric and Michel Dhome (2002). "Real Time Robust Template Matching." In: *British Machine Vision Conference (BMVC)*, pp. 1–10.

Kim, Hae Yong and Sidnei Alves de Araújo (2007). "Grayscale Template-matching Invariant to Rotation, Scale, Translation, Brightness and Contrast". In: *Pacific Rim Conference on Advances in Image and Video Technology (PSIVT)*. Springer-Verlag, pp. 100–113.

Korman, Simon et al. (2013). "FAsT-Match: Fast affine template matching". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 2331–2338.

Morel, Jean-Michel and Guoshen Yu (2009). "ASIFT: A new framework for fully affine invariant image comparison". In: *SIAM Journal on Imaging Sciences (SIIMS)* 2.2, pp. 438–469.

Nair, Dinesh, Ram Rajagopal, and Lothar Wenzel (2000). "Pattern matching based on a generalized Fourier transform". In: *International Symposium on Optical Science and Technology*. International Society for Optics and Photonics, pp. 472–480.

Penate-Sanchez, Adrian, Lorenzo Porzi, and Francesc Moreno-Noguer (2015). "Matchability Prediction for Full-Search Template Matching Algorithms". In: *International Conference on 3D Vision (3DV)*. IEEE, pp. 353–361.

Xiao, Jianxiong et al. (2010). "SUN database: Large-scale scene recognition from abbey to zoo". In: *Computer vision and pattern recognition (CVPR)*. IEEE, pp. 3485–3492.

Zhang, Chao and Takuya Akashi (2015a). "Fast Affine Template Matching over Galois Field". In: *British Machine Vision Conference (BMVC)*.

– (2015b). "High-Speed and Local-Changes Invariant Image Matching". In: *IEICE Transactions on Information and Systems* 98.11, pp. 1958–1966.

– (2015c). "Simplifying Genetic Algorithm: A Bit Order Determined Sampling Method for Adaptive Template Matching". In: *Irish Machine Vision and Image Processing Conference (IMVIP)*.

Zhang, Chao, Yo Yamagata, and Takuya Akashi (2015). "Robust Visual Tracking via Coupled Randomness". In: *IEICE Transactions on Information and Systems* 98.5, pp. 1080–1088.

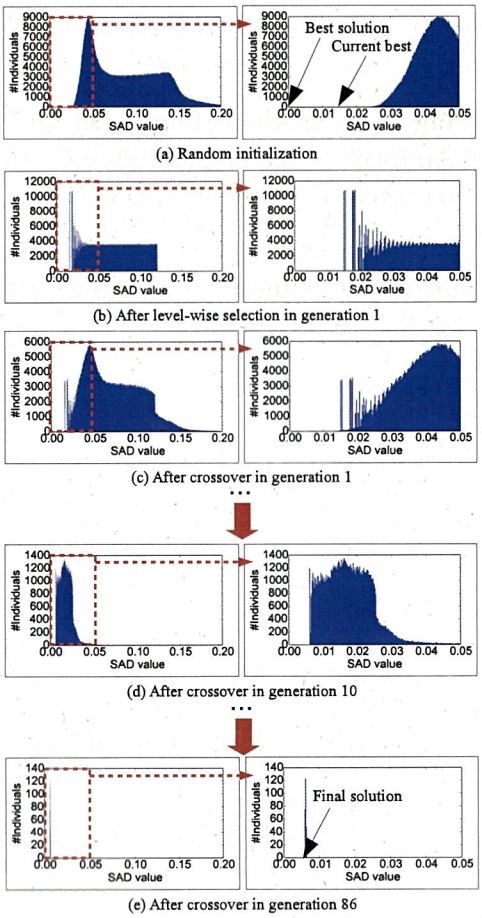
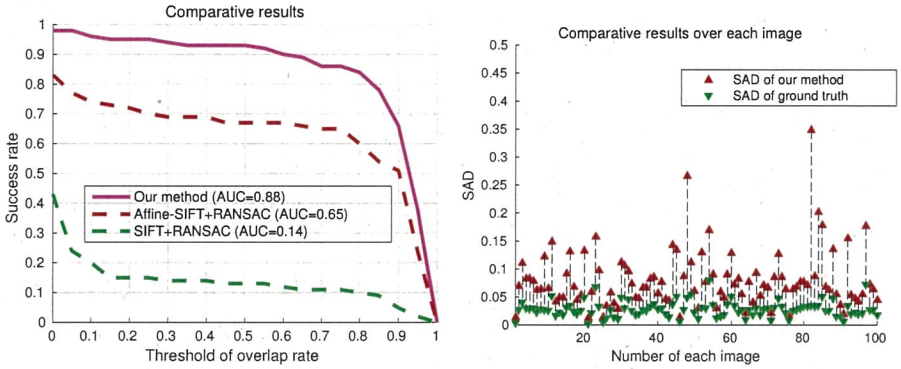


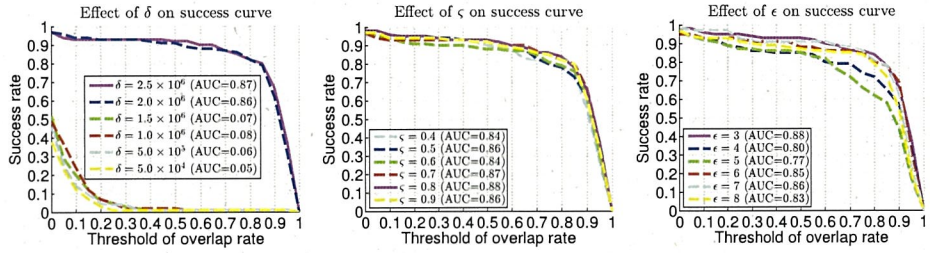
Figure 3.5: Illustration of evolution procedure. This figure shows the relation between SAD value (x-axis) and number of individuals (y-axis) throughout the evolution. a) 2.5×10^6 individuals (i.e., candidates) are initialized randomly in total. Better solutions are closer to the original point with lower SAD value. As a result of random process, we observe an elite individual with SAD value around 0.015. b) According to the initial limiting threshold λ, individuals which hold SAD value that is larger than λ are excluded in further stages. Furthermore, with the operation of LAS, samples distribute approximately uniformly with respect to SAD levels. c) To generate better possible individuals, crossover operation is performed. Although decrease of SAD score is not obvious in one generation, if we refer to (d), which is the distribution after 10 generations, we can find that the whole distribution has moved to left. e) As the result of LAS, after 86 generations, the number of individuals reduce to hundreds and the whole population converges to the final best solution.



(a) Fixed parameters: $\delta = 2.5 \times 10^6$, $\varsigma = 0.8$, $\epsilon = 3$, $nl = 5$, $\lambda = 0.7$, $\sigma = 3.0$

(b) Fixed parameters: $\delta = 2.5 \times 10^6$, $\varsigma = 0.8$, $\epsilon = 3$, $nl = 5$, $\lambda = 0.7$, $\sigma = 3.0$

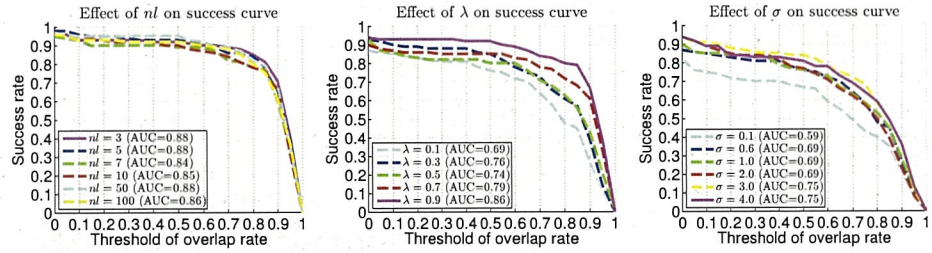
Figure 3.6: All the parameters which have been used to plot the figures are shown in the sub-captions. a) Comparative results on success rate between common solution SIFT+RANSAC, ASIFT+RANSAC and our method. SIFT+RANSAC obtains a poor performance since SIFT feature is not projective invariant. ASIFT+RANSAC achieves better performance than SIFT+RANSAC. Our method achieves promising results with detail parameters shown in the sub-caption. AUC stands for average accuracy of the success curve. b) Comparative results on SAD value between ground truth and our method. Ground truths also produce unavoidable SAD due to interpolation. The SAD value of our method is very close with the ground truth SAD.



(a) Fixed parameters: $\varsigma = 0.7$, $\epsilon = 3$, $nl = 5$, $\lambda = 0.7$, $\sigma = 3.0$

(b) Fixed parameters: $\delta = 2.5 \times 10^6$, $\epsilon = 3$, $nl = 5$, $\lambda = 0.7$, $\sigma = 3.0$

(c) Fixed parameters: $\delta = 2.5 \times 10^6$, $\varsigma = 0.8$, $nl = 5$, $\lambda = 0.7$, $\sigma = 3.0$



(d) Fixed parameters: $\delta = 2.5 \times 10^6$, $\varsigma = 0.8$, $\epsilon = 3$,0.8, $\lambda = 0.7$, $\sigma = 3.0$

(e) Fixed parameters: $\varsigma = 0.8$, $\delta = 2.5 \times 10^6$, $\epsilon = 5$,0.8, $\sigma = 3.0$, $nl = 5$

(f) Fixed parameters: $\varsigma = 0.8$, $\delta = 2.5 \times 10^6$, $\epsilon = 3$, $\lambda = 0.1$, $nl = 5$

Figure 3.7: Effect of parameters on success rate. We change one parameter while fixing other parameters which are shown in each sub-caption to see how a certain parameter affects the success curve. AUC stands for average accuracy of the success curve. a) Changing initial population size δ. b) Changing crossover rate ς. c) Changing approximation parameter ϵ. d) Changing number of SAD levels nl. e) Changing limiting parameter λ. f) Changing smooth parameter σ.

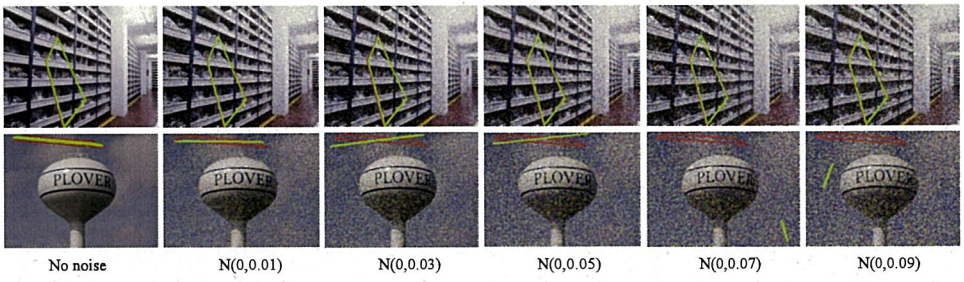


Figure 3.8: Matching results with different levels of Gaussian noise.

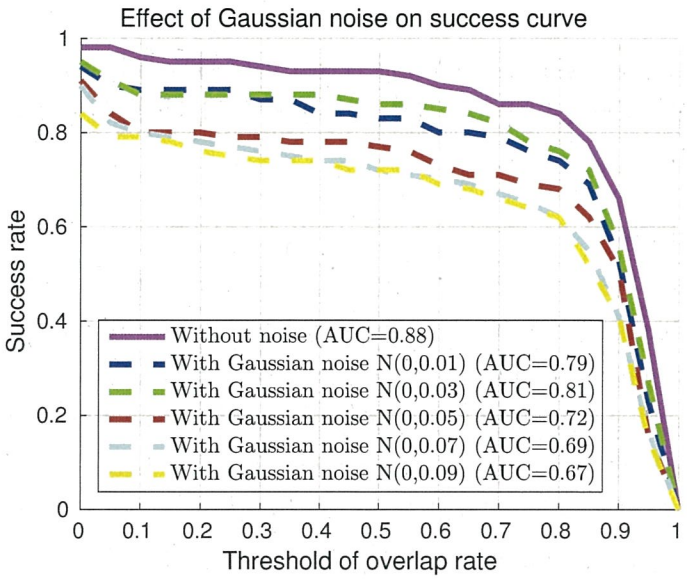


Figure 3.9: Noise-tolerance experiment with Gaussian noise. We control the level of noise by increasing the standard deviation of Gaussian from 0.01 to 0.09. Visual illustration of each noise level is shown in Figure 3.8

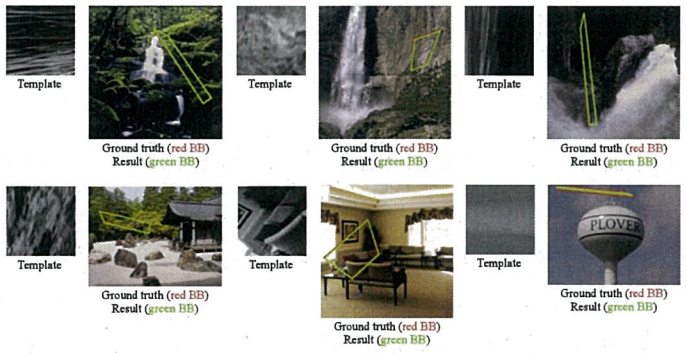


Figure 3.10: Examples of successful matching results. The green bounding box (BB), which represents our result, completely overlap the ground truth (shown in red BB) in these cases. Various kinds of transformations are generated in this benchmark, and many challenging cases with feature-less and narrow-long area have also been considered.

Chapter 4

ROBUST NON-PARAMETRIC TEMPLATE MATCHING WITH LOCAL RIGIDITY CONSTRAINTS

4.1 Summary

In this paper, we address the problem of non-parametric template matching which does not assume any specific deformation models. In real-world matching scenarios, deformation between a template and a matching result usually appears to be non-rigid and non-linear. We propose a novel approach called local rigidity constraints (LRC). LRC is built based on an assumption that the local rigidity, which is referred to as structural persistence between image patches, can help the algorithm to achieve better performance. A spatial relation test is proposed to weight the rigidity between two image patches. When estimating visual similarity under an unconstrained environment, high-level similarity (e.g. with complex geometry transformations) can then be estimated by investigating the number of LRC. In the searching step, exhaustive matching is possible because of the simplicity of the algorithm. Global maximum is given out as the final matching result. To evaluate our method, we carry out a comprehensive comparison on a publicly available benchmark and show that our method can outperform the state-of-the-art method.

4.2 Introduction

Template matching has been studied as a classical problem for a number of decades. Current techniques can match template with similar candidate windows in the target image while estimating translation, rotation, affine transformation and even some regular deformation. However, in real-world applications such as online tracking K. Zhang, L. Zhang, and Yang, 2012; C. Zhang, Yamagata, and Takuya Akashi, 2015, the foreground models in the templates usually deform complexly. Such deformation can hardly be modeled mathematically since the result in the target image is projected from the template after involving fusion of 3D transformations. Furthermore, external influences, such as occlusion, illumination change, and background clutter will increase the degree of non-linearity and the difficulty of template matching.

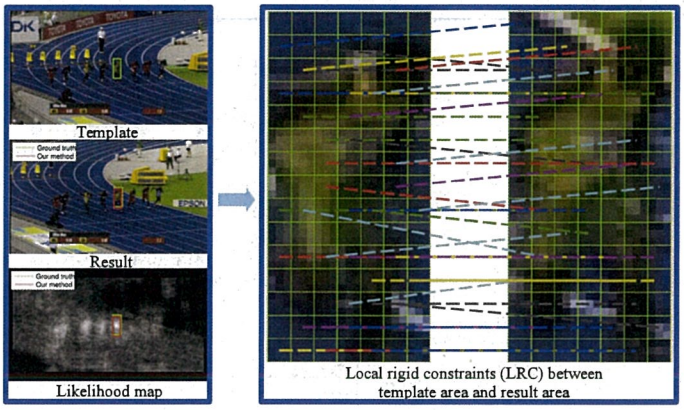


Figure 4.1: Matching example and its local rigidity constraints (LRC) which are generated by our program. Template area is predetermined within a reference image. Result area is represented by red rectangle, ground truth is presented by green rectangle in both target image and likelihood image. Dotted lines represent the corresponding LRC with 3×3 pixel rigid patches.

The drawbacks of existing parametric template matching frameworks can be mainly summarized as: 1) Dense matching at pixel-level is usually necessary in order to estimate more accurate parameters, which requires a lot of computing costs; 2) In the case of heavy occlusion, the occluded part may drastically affect the whole similarity score and lead to mismatch; 3) A large number of parameters may need to be estimated when complex transformations occur, which is very difficult due to high-dimensionality and strong non-convexity. These drawbacks limit the scope of applications and increase the dependence on environment.

As a common solution, histogram matching (HM) plays an important role in non-parametric template matching. HM can deal with the deformable matching problem by disregarding geometric relationship between pixels. However, we argue that disregarding geometry completely may increase the number of local optimums and thus increase the level of non-convexity. When partial occlusion occurs to the target object, the template is also easy to be mismatched to a local optimum Sato and Akashi, 2015.

In this paper, we address the importance of local rigidity constraints. As a circle can be approximated by many rigid straight lines, most of the objects can be approximated by rigid patches if the size of each rigid patch is small enough. Figure 4.1 shows an example of matching result with an

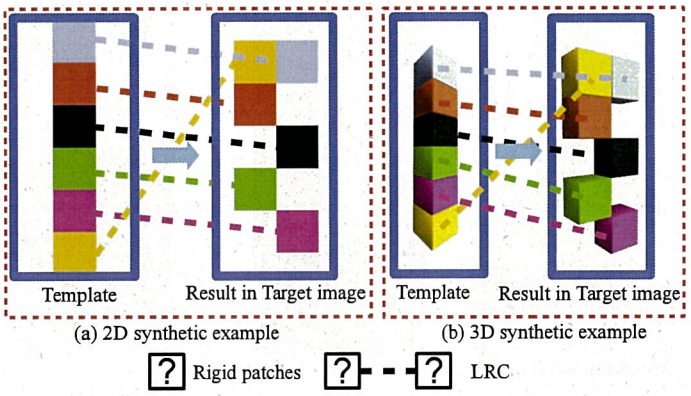


Figure 4.2: 2D and 3D rigidity with illustration examples. a) Each rectangle holds 2D local rigidity, since no deformation occur, rigidity can be preserved well in the target image. b) Each cube holds 3D local rigidity. Due to 3D transformation, each cube of result in the target image changes its appearance compared to the template. Hence, 3D rigidity is more difficult to be preserved in 2D images.

athlete. We specify a region of interest (ROI) as a template in the reference image which is taken at a running race. As the race progresses, the appearance of the athlete continuously changes. By decomposing the template and the result into 3×3 patches, we can find that although the global appearance has been changed (e.g. hands, head, legs and background), many patches still have corresponding relationships between the template and the result. Two patches and their corresponding relationship together form a constraint which implicitly reflects the inherent characteristics of the object. It is worth pointing out that there mainly exist two kinds of rigidity, which have been illustrated in Figure 4.2. Object holds 3D rigidity appears to be invariant in 3D space while changes appearance in 2D image due to projection. Hence, it is difficult to model the 3D rigidity and we only utilize the 2D rigidity in this paper.

By applying the rigidity, we extend a non-parametric template matching framework Dekel et al., 2015 named Best-Buddies (BB) similarity. The main idea of BB is that a pair of points plays an important role in estimating the similarity when each point is the nearest neighbor of the other. We apply BB to define the corresponding relationship between two rigid patches.

4.3 Related Work

In this paper, we mainly focus on the problem with single template and single result. Cross-domain matching C. Zhang and Takuya Akashi, 2015b is not considered in this paper (e.g. template is a sketch image while result is an RGB image). We survey previous works and contributions in this section. In section 2.1, we conclude the basic distance measurement methods of similarity. These methods are widely applied in the applications of template matching, which will be summarized in Section 2.2 and 2.3 respectively in terms of using geometry model or not using.

Visual Similarity Estimation

As the most important component in template matching, popular direct methods such as sum of absolute differences (SAD), sum of squared difference (SSD), normalized cross-correlation (NCC), and zero-means normalized cross-correlation (ZNCC) have been widely applied Lucas, Kanade, et al., 1981; C. Zhang and Takuya Akashi, 2015a; Di Stefano, Mattoccia, and Tombari, 2005. Due to the computational efficiency, direct methods are suitable for dense matching. On the other hand, feature-based methods, such as SIFT Lowe, 2004, ASIFT Morel and Yu, 2009 are very effective especially dealing with rotation, scaling, and simple transformation. However, feature-based methods are usually high-dimensional (e.g. SIFT is typically 128-dimensional), it is inefficient for dense matching and is usually employed with key points. It has also been discussed in Tuytelaars and Mikolajczyk, 2008 that the most damaging effect on the matching results of keypoint-based local features are the non-planarities or non-rigid deformation, which abound in our testing images. To cover the advantages of both direct method and feature-based method, Dekel et al. Dekel et al., 2015 found an intrinsic relationship between two groups of points, thus can reduce the outlier rate without increasing the number of feature dimension.

Parametric Template Matching

Lucas, et al. Lucas, Kanade, et al., 1981 proposed parametric optical flow to estimate inliers between a template and a target. Further developed by feature-based methods, Lucas and Kanade's framework has become an essential approach in many matching problems. Combined with RANSAC Fischler and Bolles, 1981, parameters of transformation can be estimated. However, a sufficient number of inliers (i.e., distinct features) are necessary

in order to estimate the parameters. On the other hand, although affine or projective transformation can be calculated by solving a system of linear equations, parameters of non-rigid transformations are difficult to be calculated. C. Zhang and T. Akashi C. Zhang and Takuya Akashi, 2015a proposed a stochastic method to search the 2D affine parameters efficiently with the fitness function of SAD. It is also difficult to be applied in real images since the real-world transformations are more complex. D. J. Tan, et al. Tan et al., 2014 modelled the 2D deformation with cubic B-Splines. Larger number of control points are required if more complex deformation want to be matched. Overall, to the best of our knowledge, most of the parametric methods can hardly be applied to "wild" images which may contain incalculable and unpredictable deformation.

Non-parametric Template Matching

It is more efficient and feasible to design deformation-invariant features instead of estimating the specific parameters when matching with the real-world deformation. Despite histogram matching Swain and Ballard, 1991; Ullah and Kaneko, 2004; Comaniciu, Ramesh, and Meer, 2000, D. P. Huttenlocher et al. Huttenlocher, Klanderman, Rucklidge, et al., 1993 designed a distance function which measures the level of mismatch between two point sets. The distance is calculated when point of set A is the farthest from any point of set B and vice versa. This idea is quite similar with Dekel et al., 2015. The difference is, instead of calculating the farthest distance, Dekel et al., 2015 counts the number of matches which satisfy the condition: point of set A is the nearest from any point of set B and vice versa. Y. Rubner et al. Rubner, Tomasi, and Guibas, 2000 introduced a function named Earth Mover's Distance (EMD) to measure the minimum cost that must be paid from one point set to another. EMD allows partial matches, which means it is robust with occlusion and clutter. D. Simakov et al. Simakov et al., 2008 proposed bidirectional similarity (BDS). BDS considers two point sets are similar if all patches of set A are contained in set B and vice versa.

It is worth pointing out that the term "local rigidity constraint" has also been used by Loeckx et al., 2004 and related image registration papers. However, the definition is quite different with our method since in Loeckx et al., 2004, local rigidity constraint is treated as a penalty term of the cost function, which is based on Jacobian matrix. In our paper, LRC is treated as a one-to-one map limitation (i.e. restriction) between a rigid patch of

template and a rigid patch of candidate.

4.4 Methodology

Two color images are given as the input with each pixel and each channel normalized to [0,1]. I_T is defined as $T_w \times T_h$ pixel template image extracted from a reference image and I_S is defined as a $S_w \times S_h$ pixel target image (i.e. source image). Each rigid patch is defined as a $s \times s$ pixel square patch. $T_w, T_h, S_w, S_h, s \in \mathbb{N}^+$. A candidate I_C in target image I_S is an ROI defined by a search window. Following the traditional sliding window search method, we have candidates arranged in order from top left to bottom right in the target image. To clarify the meaning of reference, target, and template image, we define them as following:

Reference image: Base image from which a template is cropped.

Template image: A region cropped from the reference image manually, and holds semantic meaning (usually an object) for matching with similar objects in the target image.

Target image: Also known as source image, is an image in which the object described by template exists, but may change in appearance due to internal and external influences.

Problem The problem of this paper can be defined as:

$$\arg\max_{I_C \in I_S} \mathrm{LRCS}(I_T, I_C), \tag{4.1}$$

where $\mathrm{LRCS}(I_T, I_C)$ is a function to estimate the LRC similarity between a template and a candidate, which will be introduced in the following section.

Feature of Local Rigid Patch

Feature vector of a rigid patch is denoted by:

$$\mathbf{f} = \mathbf{p}^{(L,C,G)}, \mathbf{f} \in \mathbb{R}^{6s^2+2}, \tag{4.2}$$

where L, C, G are feature spaces. $\mathbf{p}^L \in \mathbb{R}^2$ denotes a patch's center location in image I. Specifically, p_1^L represents x-axis value and p_2^L represents y-axis value. $\mathbf{p}^C \in \mathbb{R}^{s \times s \times 3}$ denotes a patch's color feature (e.g. RGB). $\mathbf{p}^G$ represents a patch's spatial structure. The dimensionality of $\mathbf{p}^G$ depends on the presentation of spatial structure. In this paper, $\mathbf{p}^G \in \mathbb{R}^{s \times s \times 3}$.

Definition 1 The operator to investigate a neighbor pixel's feature value is denoted by $[\cdot]_{x,y}$. x, y is the relative coordinate to the location of cor-

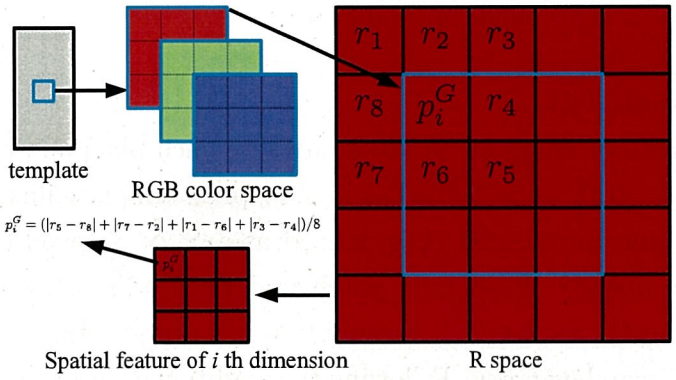


Figure 4.3: A simple case of calculating spatial feature in R channel where $s = 3$, $C = RGB$, $\omega(\cdot) = 1$. $\mathbf{p}^G$ is calculated by only investigating the relationship between 8 nearest neighbors.

responding pixel. Specifically, when $x = 1$, $y = 1$, and p_i^C is the feature value obtained at $(2, 2)$ in the image coordinate, then $[p_i^C]_{x,y}$ equals to the feature value of the same dimension i which is located at $(3, 3)$.

With Definition 1 in mind, we represent a patch's spatial feature by investigating the relationship between each pixel in the patch and its neighbor pixels.

$$\begin{aligned} p_i^G := & |[p_i^C]_{\omega(T_w),\omega(T_h)} - [p_i^C]_{-\omega(T_w),0}|/8 \\ & + |[p_i^C]_{-\omega(T_w),\omega(T_h)} - [p_i^C]_{0,-\omega(T_h)}|/8 \\ & + |[p_i^C]_{-\omega(T_w),-\omega(T_h)} - [p_i^C]_{0,\omega(T_h)}|/8 \\ & + |[p_i^C]_{\omega(T_w),-\omega(T_h)} - [p_i^C]_{\omega(T_w),0}|/8, \end{aligned} \tag{4.3}$$

where ω is a linear function to dynamically determine the neighbor pixels to investigate according to the template's size. Instead of fixing the position of neighbor pixels to investigate (e.g. local binary pattern), we change the position dynamically based on a simple fact: the size of rigid parts depend on the size of object in the template.

The design of $\mathbf{p}^G$ is important since we can confirm how the complex deformation affect the patches by checking the spatial structure feature in both template and candidate images. To gain more insight into the feature design, we refer to a simple case when $s = 3$, $C = RGB$, $\omega(\cdot) = 1$. Figure 4.3 illustrates this case.

Definition 2 The operator to calculate the feature distance between two rigid patches is denoted by $\|\cdot, \cdot\|_*$, which is defined as:

$$\begin{aligned}\|\mathbf{f_1}, \mathbf{f_2}\|_* : \mathbb{R}^{6s^2+2}, \mathbb{R}^{6s^2+2} \rightarrow \mathbb{R}^+ : \mathbf{f_1}, \mathbf{f_2} \mapsto w_l\|\mathbf{p}^L - \mathbf{q}^L\|_2^2 \\ +w_c\|\mathbf{p}^C - \mathbf{q}^C\|_2^2 + w_g\|\mathbf{p}^G - \mathbf{q}^G\|_2^2,\end{aligned} \tag{4.4}$$

where $\mathbf{f_1}$ and $\mathbf{f_2}$ are the feature vectors of two rigid patches. w_l, w_c, $w_g \in \mathbb{R}$ are the weights of each feature space. These weights balance the feature space to describe better appearance model of a template. In the experiment of parameter analysis, we will comprehensively study how the w_l, w_c and w_g affect the performance. (each of them is varied from 0.5 to 3.0).

Local Rigidity Constraint Similarity

With the distance between two feature vectors defined, we can estimate the similarity between two rigid patches. Following traditional method such as SAD, we may estimate the similarity between template and candidate by using the sum of patches' distance. However, it has been proved to be inefficient when deformation occurred in the target image and the corresponding relationships between pixel pairs no longer exist. Instead of using the feature distance to estimate the distance directly, we extend the method in Dekel et al., 2015.

Definition 3 The operator to judge wether constraint exists between a rigid patch of template image I_T and a rigid patch of candidate image I_C is denoted by $< \cdot, \cdot >_{I_T, I_C}$, which is defined as:

$$\begin{aligned}< \mathbf{f_1}, \mathbf{f_2} >_{I_T, I_C} : \mathbb{R}^{6s^2+2}, \mathbb{R}^{6s^2+2} \rightarrow \{0, 1\} : \mathbf{f_1}, \mathbf{f_2} \mapsto \\ \begin{cases} 1, & \mathrm{NN}(\mathbf{f_1}, I_C) = \mathbf{f_2} \wedge \mathrm{NN}(\mathbf{f_2}, I_T) = \mathbf{f_1} \\ 0, & \text{otherwise} \end{cases}\end{aligned} \tag{4.5}$$

Where $\mathrm{NN}(\mathbf{f_1}, I_C) = \arg\min_{\mathbf{f_i} \in I_C} \|\mathbf{f_1}, \mathbf{f_i}\|_*$, and $\mathrm{NN}(\mathbf{f_2}, I_C) = \arg\min_{\mathbf{f_i} \in I_T} \|\mathbf{f_2}, \mathbf{f_i}\|_*$. Both $\mathbf{f_1}$ and $\mathbf{f_2}$ are feature vector of single patch which is extracted from I_T and I_C respectively. Similar operator is also defined in Dekel et al., 2015. This operator is similar with binary quantization, which converts a pair of feature distance in real number into a countable number. This operator can also be seen as a compression procedure. By summing up 0/1 number, the degree of similarity can be compressed from high-dimensional feature space.

Definition 4 The LRC similarity between a template and a candidate can

Algorithm 3 Template matching with LRC.

Require: Template image extracted from reference image: I_T
Require: Target image: I_S
Require: Size of rigid patch: s
1: **for** i from 1 to T_w **do**
2: **for** j from 1 to T_h **do**
3: **if** $i - s/2 \geqq 0 \wedge j - s/2 \geqq 0 \wedge i + s/2 \leqq S_w \wedge j + s/2 \leqq S_h$ **then**
4: Preprocessing with Gaussian smoothing
5: Calculate LRCS(I_T, I_C), the center of I_C locates at (i,j)
6: **end if**
7: **end for**
8: **end for**
9: Return $\arg\max_{I_C \in I_S} \text{LRCS}(I_T, I_C)$

then be defined as:

$$\text{LRCS}(I_T, I_C) : \mathbb{R}^{T_w \times T_h}, \mathbb{R}^{T_w \times T_h} \to \mathbb{R}^+ : I_T, I_C \mapsto \frac{1}{\min(\|\mathbf{f^T}\|_1, \|\mathbf{f^C}\|_1)} \sum_{i,j} < \mathbf{f_i^T}, \mathbf{f_j^C} >_{I_T, I_C}, \tag{4.6}$$

where $\mathbf{f_i^T}$ is the feature vector of i^{th} rigid patch extracted from I_T, $\mathbf{f_j^C}$ is the feature vector of j^{th} rigid patch extracted from I_C, $\mathbf{f^T} = \{\mathbf{f_1^T}, \mathbf{f_2^T}, ...\}$, $\mathbf{f^C} = \{\mathbf{f_1^C}, \mathbf{f_2^C}, ...\}$.

Overall, Equation 4.5 and 4.6 specify the genera expressions in Equation 1 and 2 of Dekel et al., 2015. Our contribution is to add a spatial relation test defined in Equation 4.3 to feature extraction, which helps to match rigid patches. The whole procedure of our algorithm is concluded in Algorithm 4.

Discussion

In mathematics and physic, the definition of "rigidity" can also be referred to as "stiffness", which means the property of a solid body to resist deformation. In our paper, the "rigidity" has the similar meaning, which means the property of an image patch to resist geometry deformation. Furthermore, as each image patch is locally existed with respect to an image, we name it as "local rigidity". Let us show a specific situation to visually illustrates the difference between BBS and LRC in Figure 4.4. As we can see from Figure 4.4, by involving such a certain pattern of spatial relation test, LRC tends to match patches that are structurally persistent. From this example, it is hard to judge directly whether the matching result of

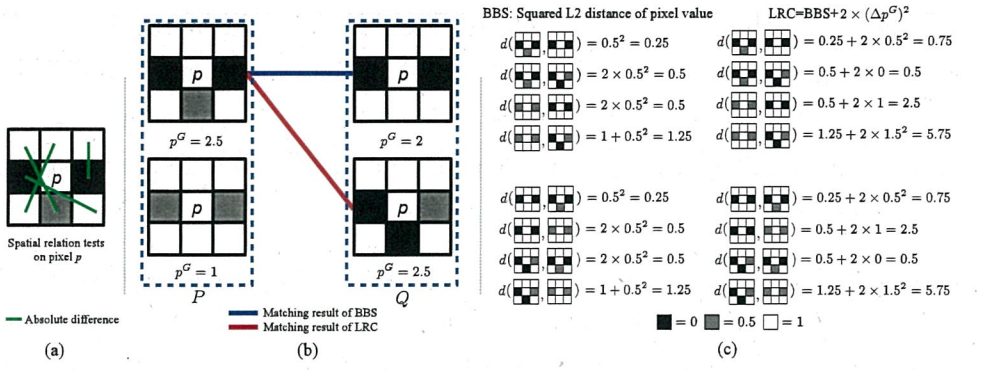


Figure 4.4: A specific matching example to compare the detail between BBS and LRC. Overall, LRC tends to match patches that are structurally persistent, which is referred to as "rigidity" in our paper. On the other hand, BBS tends to be influenced by partial consistency of color feature. a) Each green line represents a spatial relation test, which is manually designed. b) Two sets of image patches P and Q are used for matching. For clarity, each image patch is set with only 1 channel and each pixel's value is normalized to [0,1]. At this example, as each image patch is dependent from an image, only the center pixel's p^G is calculated to calculate the spatial relationship of an image patch. Outside this example, the average value of 3×3 pixels' p^G is calculated because the outer pixels exist. c) After adding a weighted p^G, the matching result is changed.

LRC is better than BBS. However, the result of LRC is more in line with our assumption: the matched image patches which are structurally persistent (rigid) play more important role on similarity estimation, and both the symmetric and real-data experiments show that this assumption can help improving the performance.

Analysis

In order to understand the efficiency, we first show a simple 2D case which is illustrated in Figure 4.5. To increase the matching difficulty, two different background models (in red points) are generated, and each of them is mixed with the foreground model (in blue points). We match (a) and (b) and compare the results generated without LRC (c, d) and with LRC (e, f). By comparing with result (c) and (e), we can see that the number of matched foreground points is roughly doubled while the number of matched background points only increased by 12. The proportion of matched foreground points increases from 57% to 69%. By comparing result (d) and (f), we can also find that the number of matched foreground points is roughly

doubled while the number of matched background points only increased by 7. The proportion of matched foreground points increases from 65% to 75%. This example shows that considering LRC can further improve the matching rate of foreground and separate the background well comparing with Dekel et al., 2015.

To prove LRC as a better method, we have to prove two assumptions: 1) The expectation of a pair of rigid patches to be matched is highest when two patches are from the same foreground (same distribution). Conversely, the expectation drops sharply when two foreground models leave each other. 2) If rigidity exist, considering the neighbor patches can enhance the phenomenon described in (1). We prove these two assumptions under one-dimensional case. First we generate a point set P under normal distribution $N(0, 0.1)$, $\|P\| = 100$, and then extend P to $\overline{P} = \{P,\ P-d,\ P+d\}, d \in \mathbb{R}, \|\overline{P}\| = 300$. Similarly, we generate $\overline{Q}$ under $N(\mu, \sigma)$ and extend it to $\overline{Q} = \{Q,\ Q-d,\ Q+d\}, \|\overline{Q}\| = 300$. Note that the points in $\overline{P}$ and $\overline{Q}$ no longer obey simple Gaussian distribution since $P-d$, $P+d$, $Q-d$, $Q+d$ are involved. The expectation of two points (p, q), $p \in \overline{P}, q \in \overline{Q}$ to be matched can be defined as E_1. The case in which $\overline{P}$ and $\overline{Q}$ are simple Gaussian distributions has been proved in Dekel et al., 2015:

$$\begin{aligned} E_1 := \iint_{-\infty}^{+\infty} \Big(& f_{\overline{P}}(p) F_{\overline{Q}}(|x-p| \le |p-q|)^{\|\overline{Q}\|-1} \times \\ & f_{\overline{Q}}(q) F_{\overline{P}}(|x-q| \le |p-q|)^{\|\overline{P}\|-1} \Big) \mathrm{d}p\mathrm{d}q, \end{aligned} \tag{4.7}$$

where $F_Q(\cdot)$ and $F_P(\cdot)$ are the probability functions. $F_Q(\cdot)$ describes the probability that a $x \in \mathbb{R}$ with a given distribution over $\overline{Q}$ will be found to have a value which satisfies the condition within the parentheses. Function $F_P(\cdot)$ holds the same definition. Function $f_P(p)$ represents probability density function which equals to $1/\|\overline{P}\|$, $f_Q(q)$ equals to $1/\|\overline{Q}\|$. On the other hand, we define the expectation considering local rigidity as E_2:

$$\begin{aligned} E_2 := \iint_{-\infty}^{+\infty} \Big(& f_{\overline{P}}(p) F_{\overline{Q}}(\|\mathbf{x}-\mathbf{p}\|_2 \le \|\mathbf{p}-\mathbf{q}\|_2)^{\|\overline{Q}\|-1} \times \\ & f_{\overline{Q}}(q) F_{\overline{P}}(\|\mathbf{x}-\mathbf{q}\|_2 \le \|\mathbf{p}-\mathbf{q}\|_2)^{\|\overline{P}\|-1} \Big) \mathrm{d}p\mathrm{d}q, \end{aligned} \tag{4.8}$$

where $\mathbf{x} = \{x^-, x^+\}$, $\mathbf{p} = \{p^-, p^+\}$, $\mathbf{q} = \{q^-, q^+\}$. Variable $p^- \in \overline{P}$ and $p^+ \in \overline{P}$ are the the closest left point and the closest right point to

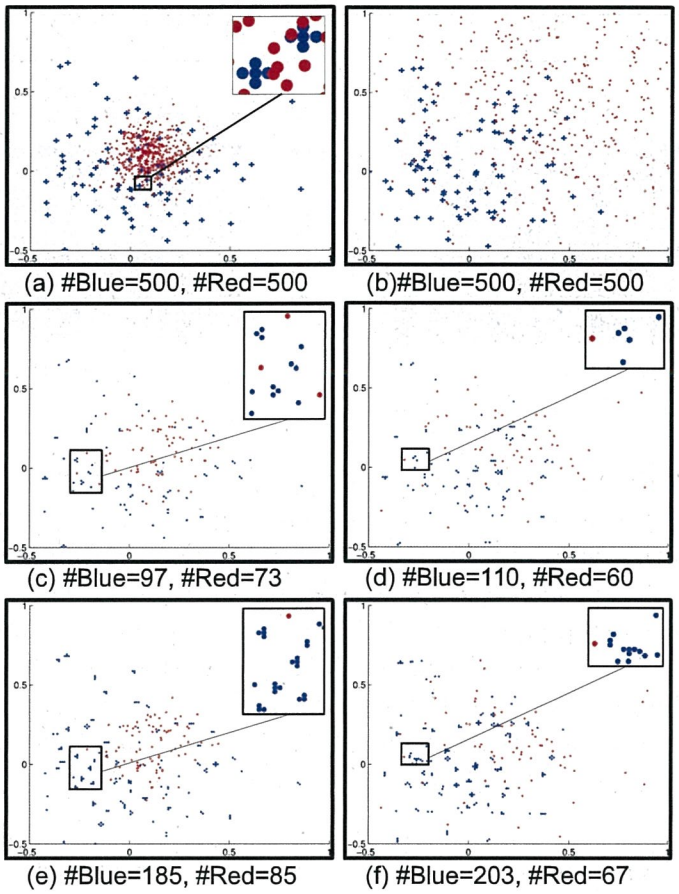


Figure 4.5: 2D case with synthetic data generated by Gaussian distribution. **a)** We first generate 100 blue points from a normal distribution $N(\mu_1, \sigma_1)$. To give each blue point rigidity, we generate four points around each blue point vertically and horizontally. Totally, the combination of 500 blue points is treated as foreground model. As shown in the enlarged part of (a), the combination of blue points shows like a cross. We then generate 500 red points as background from a different distribution $N(\mu_2, \sigma_2)$. **b)** Similarly, we first generate 100 blue points from $N(\mu_1, \sigma_1)$ and then extend them to 500. Background is drawn from $N(\mu_3, \sigma_3)$. **c)** The matching result of (a) by Dekel et al., 2015 without considering the LRC. **d)** The matching result of (b) by Dekel et al., 2015 without considering the LRC. **e)** The matching result of (a) considering LRC. **f)** The matching result of (b) considering LRC.

$p \in \overline{P}$ respectively. The meaning of this denotation also applies to $\mathbf{x}$ and $\mathbf{q}$. From Figure 4.6, we can observe two properties, 1) higher expectation can be observed when parameters μ, σ are closer to $(0, 0.1)$; 2) in (b), the expectation drops faster than (a) when (μ, σ) become larger than $(0, 0.1)$.

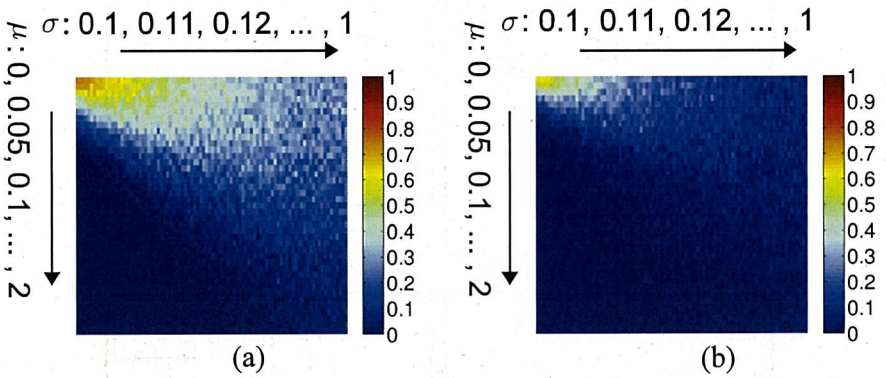


Figure 4.6: Expectation of a pair of points (p, q) $(p \in \overline{P}, q \in \overline{Q})$ to be matched. $P \in \overline{P}$ is a distribution with parameter $(\mu = 0,\ \sigma = 0.1)$, $Q \in \overline{Q}$ is a distribution with dynamic parameters μ and σ. Parameter μ changes from 0 to 2, with each step equals to 0.05. Parameter σ changes from 0.1 to 1, with each step equals to 0.01. Each combination of μ and σ plots a pixel in both heat map (a) and (b). Left top point shows the expectation when distribution $Q \in \overline{Q}$ is the same with $P \in \overline{P}$. (a) is the result of BB Dekel et al., 2015. (b) is the result of LRC. As we can observe, comparing with (a), the expectation drops faster when μ and σ increase.

These two figures show that our method is more sensitive with the difference of distribution and thus results in better performance. These two properties we observed can well prove the two assumptions we made.

4.5 Experiment

Experiment Environment

We use the benchmark used in Dekel et al., 2015 to evaluate our method. This benchmark is inherited from online tracking benchmark Wu, Lim, and Yang, 2013. Hence, it is very challenging for global template matching task. Many real-world difficulties have been considered in this benchmark such as occlusion, illumination change, background clutter, deformation, etc. There are 106 pairs of template and target images in this benchmark with various image size. All the ground truth bounding boxes are annotated manually with a semantic foreground defined.

We use the overlap rate to judge whether a matching result is successful by referring to the ground truth. Specifically, PASCAL criteria Everingham

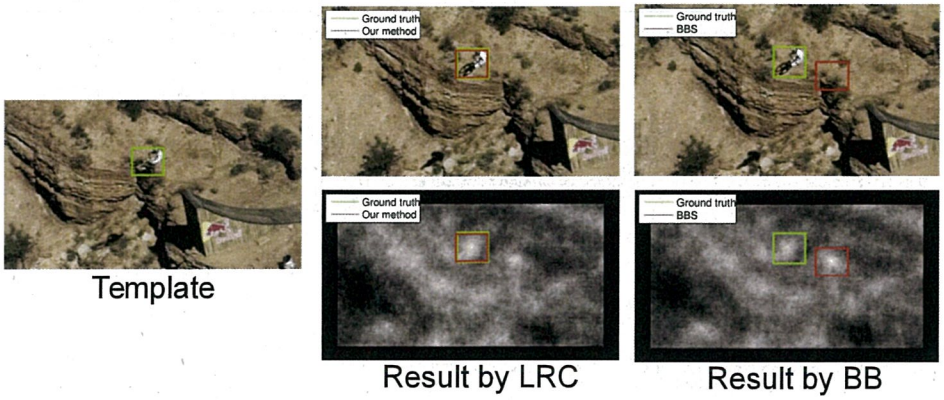


Figure 4.7: Example of comparison. A bike passes trough a series of scenes, the template is selected from previous frames and the target image is selected from later frames. Likelihood mainly focuses on two ROIs: an ROI which has the same background with template and an ROI which has the same foreground with the template. It is difficult to answer which one is correctly matched since a foreground is usually semantically defined by users. In this example, bike is defined as the foreground and our method correctly matched.

et al., 2010 is used to calculate the overlap rate:

$$\text{overlap rate} = \frac{\text{area}(BB_{re} \cap BB_{gr})}{\text{area}(BB_{re} \cup BB_{gr})}. \tag{4.9}$$

Where BB_{re} means bounding box of result and BB_{gt} means bounding box of ground truth. area$(\cdot)$ is a function to count number of pixels. Based on the overlap rate, we can achieve the answer about whether a matching result is correct or wrong by setting a threshold. Specifically, we have

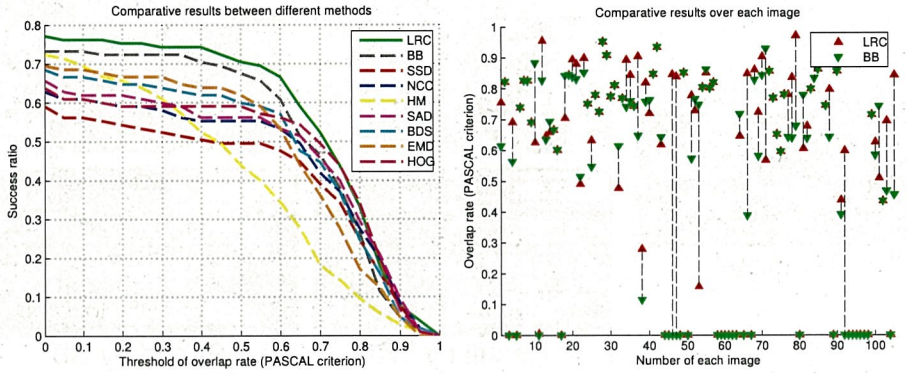
$$\text{answer} = \begin{cases} 1 & \text{if overlap rate} > \text{threshold} \\ 0 & \text{otherwise} \end{cases}. \tag{4.10}$$

Finally, success ratio $= \#\{\text{answer}|\text{answer} = 1\}/\#\text{test}$ as the accuracy criterion.

All the experiments have been done on a PC equipped with Intel Core-i7 2.9GHz and 16 GB RAM.

Comparison

We compare our method with both classical methods and state-of-the-art methods. Classical methods such as SAD, SSD, HM and NCC have been comprehensively studied in Ouyang et al., 2012. Among recent methods,



(a) (b)

Figure 4.8: Comparative result. a) Success ratio curves with threshold of overlap rate be changed from 0 to 1. b) Case-by-case comparison with BB. LRC improves the overlap rate on many test cases in the benchmark.

BB, BDS are patch-to-patch similarity measurements, which are closest to our method. HOG is a dense feature combined with SSD during the comparison. Figure 5.9(a) illustrates the comparison result of accuracy at a glance. For clarity, we dynamically change the threshold and each threshold corresponds to a success ratio value. Each curve represents one method's result and it is worth noting that BB only partially improves the accuracy against previous methods when the threshold is smaller than 0.63. When the threshold exceeds 0.63, other methods such as HOG, SAD can even outperform BB. This is because dense feature matching methods can adjust the location of final matching result better when less deformation occur. On the other hand, LRC can not only improves the success ratio in case of threshold < 0.63, but also maintain the same level of accuracy with dense feature matching method when the overlap rate becomes higher. Table 6.1 shows the average success ratio over all the matching tests in the benchmark. BB improves the accuracy by 5% and LRC improves the accuracy against BB by 4%.

Effect of Parameters

In this section, we systematically report the results for studying how each parameter affects the performance of our matching method. Six parameters $w_l, w_c, w_g, \gamma, C, s$ are studied which have been mentioned in Section 3. The results are concluded in Figure 6.5. From (a) to (f) we can see that all the six parameters affect the final result in a certain extent. The best

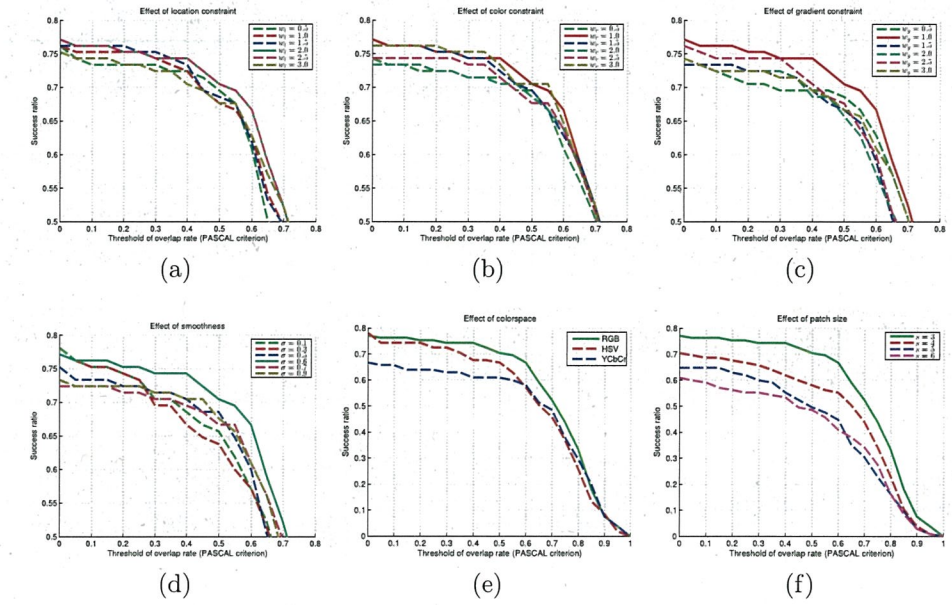


Figure 4.9: Effect of parameters on success ratio. (a) Varying the parameter of location feature's weight w_l from 0.5 to 3.0. (b) Varying the parameter of color feature's weight w_c from 0.5 to 3.0. (c) Varying the parameter of spatial feature's weight w_g from 0.5 to 3.0. (d) Varying the parameter σ which affects the degree of smoothness from 0.1 to 0.9. (e) Comparing the results over three different color spaces. (f) Varying the parameter of patch size s from 3 to 6.

performance is achieved when $w_l = 2$, $w_c = 1$, $w_g = 1$, $\gamma = 0.6$, $C = RGB$, $s = 3$. All the solid curves show the parameters we have used in the comparative experiment. Unexpectedly, increasing the patch size will cause a sharp decrease on accuracy, that means our method needs to pay a certain amount of computational cost to keep the accuracy. In our implement, about 2 seconds are needed for matching a 480×270 pixel target image with 19×45 pixel template. Processing time is directly proportional to the template size and target size. In addition to the patch size, smooth level also affect the performance a lot. Smoothness assumption is a very important precondition for template matching. An edge image (which is not smooth) without preprocessing is not suitable for template matching since a little displacement will change the matching score drastically. Over-smoothed images will also lose important feature information and lead to failed matching. Figure 4.10 shows some examples of matching results with tuned parameters. Our approach succeeded in many different conditions

Figure 4.10: Examples of matching results.

Table 4.1: Comparative result of average success ratio.

Method	average success ratio (%)
LRC (our method)	**59**
BB Dekel et al., 2015	55
BDS Simakov et al., 2008	50
SAD	49
EMD Rubner, Tomasi, and Guibas, 2000	49
HOG Dalal and Triggs, 2005	49
NCC Lewis, 1995	47
SSD	43
HM	41

such as: drastic appearance change, illumination change, small size, etc.

4.6 Conclusion

In conclusion, this chapter presents a template matching method which has no need to define a specific deformation model. Local rigidity constraint (LRC) has been proposed, which is defined as a pair of matched patches. Counting number of LRC is treated as the visual similarity between a template and a candidate. All the one-dimensional, two-dimensional synthetic experiments and real matching test show the efficiency of considering the local rigidity (if any) can improve the matching accuracy. However, several

drawbacks have limited the application of this method. 1) Since only translation has been considered, scaling and rotation cannot be sensed during the matching. 2) The matching accuracy of non-rigid objects, such as fluid, can hardly be improved.

As the future work, we intend to enhance this method for more intense environment changes in order to solve the problems which have been reflected in most of the failure tests. Finally, we hope to improve the matching accuracy and expect further real-world applications.

References

Comaniciu, Dorin, Visvanathan Ramesh, and Peter Meer (2000). "Real-time tracking of non-rigid objects using mean shift". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 142–149.

Dalal, Navneet and Bill Triggs (2005). "Histograms of oriented gradients for human detection". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 886–893.

Dekel, Tali et al. (2015). "Best-Buddies Similarity for Robust Template Matching". In: *Computer Vision and Pattern Recognition (CVPR)*.

Di Stefano, Luigi, Stefano Mattoccia, and Federico Tombari (2005). "ZNCC-based template matching using bounded partial correlation". In: *Pattern Recognition Letters (PRL)* 26.14, pp. 2129–2134.

Everingham, Mark et al. (2010). "The pascal visual object classes (voc) challenge". In: *International journal of computer vision (IJCV)* 88.2, pp. 303–338.

Fischler, Martin A and Robert C Bolles (1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6, pp. 381–395.

Huttenlocher, Daniel P, Gregory Klanderman, William J Rucklidge, et al. (1993). "Comparing images using the Hausdorff distance". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 15.9, pp. 850–863.

Lewis, JP (1995). "Fast normalized cross-correlation". In: *Vision interface*, pp. 120–123.

Loeckx, Dirk et al. (2004). "Nonrigid image registration using free-form deformations with a local rigidity constraint". In: *Medical Image Computing and Computer Assisted Intervention (MICCAI)*. Springer, pp. 639–646.

Lowe, David G (2004). "Distinctive image features from scale-invariant keypoints". In: *International Journal of Computer Vision (IJCV)* 60.2, pp. 91–110.

Lucas, Bruce D, Takeo Kanade, et al. (1981). "An iterative image registration technique with an application to stereo vision". In: 81, pp. 674–679.

Morel, Jean-Michel and Guoshen Yu (2009). "ASIFT: A new framework for fully affine invariant image comparison". In: *SIAM Journal on Imaging Sciences (SIIMS)* 2.2, pp. 438–469.

Ouyang, Wanli et al. (2012). "Performance evaluation of full search equivalent pattern matching algorithms". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 34.1, pp. 127–143.

Rubner, Yossi, Carlo Tomasi, and Leonidas J Guibas (2000). "The earth mover's distance as a metric for image retrieval". In: *International Journal of Computer Vision (IJCV)* 40.2, pp. 99–121.

Sato, J and T Akashi (2015). "Investigation of Face Tracking Accuracy by Obscuration Filters for Privacy Protection". In: *Irish Machine Vision & Image Processing Conference (IMVIP)*, pp. 121–124.

Simakov, Denis et al. (2008). "Summarizing visual data using bidirectional similarity". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1–8.

Swain, Michael J and Dana H Ballard (1991). "Color indexing". In: *International Journal of Computer Vision (IJCV)* 7.1, pp. 11–32.

Tan, David Joseph et al. (2014). "Deformable Template Tracking in 1ms". In: *British Machine Vision Conference (BMVC)*.

Tuytelaars, Tinne and Krystian Mikolajczyk (2008). "Local invariant feature detectors: a survey". In: *Foundations and Trends® in Computer Graphics and Vision* 3.3, pp. 177–280.

Ullah, Farhan and Shun'ichi Kaneko (2004). "Using orientation codes for rotation-invariant template matching". In: *Pattern Recognition (PR)* 37.2, pp. 201–209.

Wu, Yi, Jongwoo Lim, and Ming-Hsuan Yang (2013). "Online object tracking: A benchmark". In: *Computer vision and pattern recognition (CVPR)*. IEEE, pp. 2411–2418.

Zhang, Chao and Takuya Akashi (2015a). "Fast Affine Template Matching over Galois Field". In: *British Machine Vision Conference (BMVC)*.

– (2015b). "High-Speed and Local-Changes Invariant Image Matching". In: *IEICE Transactions on Information and Systems* 98.11, pp. 1958–1966.

Zhang, Chao, Yo Yamagata, and Takuya Akashi (2015). "Robust Visual Tracking via Coupled Randomness". In: *IEICE Transactions on Information and Systems* 98.5, pp. 1080–1088.

Zhang, Kaihua, Lei Zhang, and Ming-Hsuan Yang (2012). "Real-time compressive tracking". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 864–877.

Chapter 5

TWO-SIDE AGREEMENT LEARNING FOR NON-PARAMETRIC TEMPLATE MATCHING

5.1 Summary

We address the problem of measuring matching similarity in terms of template matching. A novel method called two-side agreement learning (TAL) is proposed which learns the implicit correlation between two sets of multi-dimensional data points. TAL learns from a matching exemplar to construct a symmetric tree-structured model. Two points from source set and target set agree to form a two-side agreement (TA) pair if each point falls into the same leaf cluster of the model. In the training stage, unsupervised weak hyper-planes of each node are learned at first. After then, tree selection based on a cost function yields final model. In the test stage, points are propagated down to leaf nodes and TA pairs are observed to quantify the similarity. Using TAL can reduce the ambiguity in defining similarity which is hard to be objectively defined and lead to more convergent results. Experiments show the effectiveness against the state-of-the-art methods qualitatively and quantitatively.

5.2 Introduction

Relationship between similarity estimation and template matching: Matching similarity estimation is one of the fundamental key problems to many computer vision tasks. Generally, given two input point sets P and Q, a numerical output is required in order to quantify the similarity between P and Q. Each point in the point sets belongs to n-dimensional feature space which depends on specific applications. Template matching, which is a classical problem and has been studied for a number of decades, is a typical application that largely depends on the performance of visual similarity estimation. Template matching can also be expressed in $P - Q$ matching form because any image can be divided into patches and each patch can be treated as a n-dimensional point. In real-world matching scenarios, there usually exist deformation between a reference image and a target image, this requires an algorithm to be able to estimate the visual similarity under unconstrained environment and does not depend on any

ideal deformation models (e.g. affine transformation, projective transformation). In addition, external influences, such as occlusion, illumination change and background clutter will increase the degree of non-linearity and the difficulty of template matching.

Non-parametric way for dealing with appearance-variant matching task: To deal with mentioned problems, instead of deformation-model-based approaches, many works try to improve matching performance in non-parametric way. Among them, the relationship between image patches have been proved as an important property. For a long time, there exists an argument that whether similarity should be treated as a symmetric or asymmetric relation Tversky, 1977. As an example of asymmetric methods, Hausdorff distance Huttenlocher, Klanderman, Rucklidge, et al., 1993 takes the largest distance of all the distances from a point in one set to the closest point in the other set as the output. Formally, $\max(\mathrm{dmax}(P,Q),\mathrm{dmax}(Q,P))$, where function $\mathrm{dmax}(P,Q)$ calculates the largest directed distance which starts from P. As an example of symmetric methods, Best-Buddies similarity (BBS) Dekel et al., 2015 finds that a pair of points plays an important role in matching if each point is the nearest neighbor of the other. Formally, $\sum_P \sum_Q bb(p,q)$, where function $bb(p,q)$ equals to 1 if point p and q are each other the nearest neighbor, and otherwise equals to 0. Either symmetric way or asymmetric way reveals an important principle that when estimating the similarity between P and Q, quantifying point-wise relationship usually yields better performance than taking a distance measurement after extracting feature of whole P and Q separately.

Problems in conventional methods: However, most of the conventional methods burden with manual designed matching mechanisms. The drawback is obvious because for all kinds of foreground models, there is only one predetermined method can be used to estimate the similarity, which is difficult to be redesigned when confronted with failure.

Our contributions: We propose a data-driven method called two-side agreement learning (TAL) based on the assumption that to each specific combination of foreground and background in the template, there underlies an appropriate matching mechanism. With a given matching exemplar, we extract a single positive sample and a large number of negative samples for learning. The contributions can be concluded as follows:

- We design a symmetrically tree-structured model which contains two randomized clustering tree (RCT) as shown in Figure 5.1.
- We propose a new unsupervised quality measurement method for node splitting of RCT, which is formulated in Equation 5.11.
- A cost function is proposed for model selection, which is formulated in Equation 5.13.
- Based on the above model, a data-driven distance estimation method is proposed, which is formulated in Equation 5.2.
- The effectiveness of the proposed method is proved both with synthetic data and real images.

5.3 Related Work

In this section, we mainly review the related works from two points of view. In Section 2.1, similarity estimation methods with respect to template matching are reviewed including parametric and non-parametric ways. In Section 2.2, similarity learning techniques are reviewed which may easy to be confused with the proposed method, and the difference is also described.

Similarity Estimation for Template Matching

Paramatric matching: Classical methods, based on such as SAD, SSD, NCC, and ZNCC have been widely applied Lucas, Kanade, et al., 1981; Zhang and Akashi, 2015; Di Stefano, Mattoccia, and Tombari, 2005. Lucas, et al. Lucas, Kanade, et al., 1981 proposed parametric optical flow to estimate inliers between a template and a target. Further developed by feature-based methods, Lucas and Kanade's framework has become an essential approach in many matching problems. C. Zhang and T. Akashi Zhang and Akashi, 2015 proposed a stochastic method to search the 2D affine parameters efficiently with a fitness function of SAD. D. J. Tan, et al. Tan et al., 2014 modelled 2D deformation with the cubic B-Splines. Larger number of control points are required if more complex deformation want to be matched. Most of the parametric methods can hardly be applied to "wild" images which may contain incalculable and unpredictable deformation.

Non-parametric matching: As a common solution, histogram matching (HM) Swain and Ballard, 1991; Ullah and Kaneko, 2004; Comanici-

u, Ramesh, and Meer, 2000 plays an important role in non-parametric template matching. HM can deal with deformable matching problem by disregarding the geometric relationship between pixels. In addition to mentioned methods Huttenlocher, Klanderman, Rucklidge, et al., 1993; Dekel et al., 2015, Y. Rubner et al. Rubner, Tomasi, and Guibas, 2000 introduced a function named Earth Mover's Distance (EMD) to measure the minimum cost that must be paid from one point set to another. EMD allows partial matches, which means it is robust with occlusion and clutter. D. Simakov et al. Simakov et al., 2008 proposed bidirectional similarity (BDS). BDS considers two point sets are similar if all points of set P are contained in set Q and vice versa.

Similarity Learning

Instead of using predefined metrics such as Bhattacharyya coefficient, Kullback-Leibler divergence, a majority of similarity learning methods focus on learning metrics based on Mahalanobis distance or bilinear similarity Davis et al., 2007; Chechik et al., 2010. Differently with metric learning methods, our method attempts to learn a measurement over each data point instead of each feature channel. For d-dimensional points, metric learning requires to estimate $O(d^2)$ parameters which become much harder in high-dimensional situation without considering dimension reduction methods. In the case of template matching, a typical 3×3 image patch with RGB feature yields 27-dimensional feature vector and thus 729 parameters are required to be estimated. On the other hand, the number of matching exemplars for training are usually limited in template matching. Similarly in the case of ranking metric learning, supervised predefined order is needed for training and it is hard to learn from only a single positive sample and a large number of negative samples. Besides metric learning, Shrivastava et al. Shrivastava et al., 2011 use linear SVM to learn data-driven "uniqueness" from a single positive sample and a very large negative set of samples. The uniqueness is than used to quantify the similarity. However, it is hard to be applied on template matching since it only works based on the precondition that each pair of image patches is roughly spatially consistent.

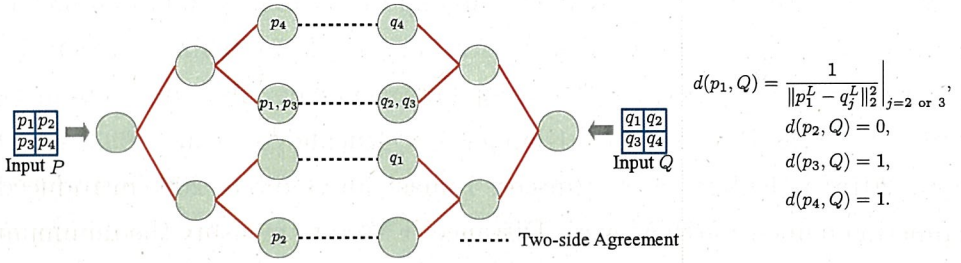


Figure 5.1: Example of symmetrically structured randomized clustering tree (RCT). Two RCTs are exactly the same and connected by the leaf nodes. Point sets P and Q are input from left root node and right root node respectively. Points are assigned to different leaf nodes and two-side agreements are constructed between each pair of leaf nodes in alignment.

5.4 Methodology

Problem Setting

If we treat a patch of an image as a multi-dimensional point, and a template as a point set which includes multiple points, then the template matching problem can be converted to general $P - Q$ form: $P = \{p_i\}_{i=1}^{M}$ and $Q = \{q_j\}_{j=1}^{M}$, where $p_i, q_j \in \mathbb{R}^d$ and M is the number of patches a template can be divided. The number of feature dimension d is proportional to the patch size. For a $pz \times pz$ image patch with RGB feature, $d = 3 \times pz \times pz$. For clarity, we only use color feature in this paper to concentrate on the analysis of matching mechanisms. Under exhaustive slide-window-detection framework, the goal of our approach can be formulated as:

$$\hat{Q} = \underset{Q \in I_{target}}{\arg\max} TAD(P, Q). \tag{5.1}$$

Where P is the template extracted from a reference image and Q is the candidate area within each search window of the target image. Distance function TAD will be introduced in the next section.

Two-side Agreement Distance

The basic form of TAD can be presented as:

$$TAD(P, Q) = \frac{1}{|P|} \sum_{i=1}^{|P|} d(p_i, Q). \tag{5.2}$$

Where function d is an asymmetric distance measurement function based on two-side agreements, which can be specifically defined as:

$$d(p_i, Q) = \begin{cases} 0, \forall q_j \in Q,\ p_i^C \neq q_j^C \\ 1,\ \exists q_j \in Q\ s.t.\ (\|p_i^L - q_j^L\|_2^2 = 0) \wedge (p_i^C = q_j^C) \\ \frac{1}{\|p_i^L - q_j^L\|_2^2},\ (\forall q_j \in Q,\ \|p_i^L - q_j^L\|_2^2 \neq 0) \\ \qquad\qquad \wedge (\exists q_j \in Q\ s.t.\ p_i^C = q_j^C) \end{cases} . \tag{5.3}$$

Where L denotes the spatial location of each image patch with respect to the image coordinate, C denotes which leaf node an image patch reaches. p_i^C equals to q_j^C when the two nodes that p_i and q_j reach can be connected by a two-side agreement. q_j is randomly selected from the candidates which meet the conditions when $d(p_i, Q) = 1/\|p_i^L - q_j^L\|_2^2$ because either of them has the possibility to be matched with p_i. In the right part of Figure 5.1, some examples of calculating $d(p_i, Q)$ have been shown. The combination of leaf nodes can also be seen as a number of compact clusters. The remained problem turns out to be how to assign each image patch s into each leaf node such that TAD of positive sample and negative sample can be distinguished at the most. We wish to learn a function with input s and an output of reached leaf cluster:

$$f(s) : \mathbb{R}^d \rightarrow \mathbb{N}^+ : s \mapsto s^C, \tag{5.4}$$

and extend f to a symmetric model:

$$ff(P, Q) : \mathbb{R}^{d \times |P|}, \mathbb{R}^{d \times |Q|} \rightarrow \mathbb{R} : P, Q \mapsto TAD(P, Q). \tag{5.5}$$

Two-side Agreement Learning

As mentioned in Section 5.2, we take both the advantages of tree designs of density forest Criminisi, Shotton, and Konukoglu, 2011 and random forest Breiman, 2001 to model function f with randomized clustering tree (RCT). Positive sample and negative samples are sampled from the foreground and background models of the training image respectively. Similar with ranking metric learning, the template is known to be more similar with the positive sample than the negative samples. The tree construction process is based on the binary random tree Breiman, 2001. The difference is, we redesign the information gain (Equation 5.11) which can use unlabelled samples to determine a hyperplane (Equation 5.12) for node splitting.

Node splitting: RCT is a full binary tree with each internal node initialized with a set of N random tests $\varphi = \{\langle \phi_i, \theta_i \rangle\}_{i=1}^{N}$. Each random test can be treated as a candidate hyper-plane that can divide the data of the node into two parts. Specifically,

$$s \in S^m, m = \begin{cases} L, & s\phi_i \geq \theta_i \\ R, & s\phi_i < \theta_i \end{cases}. \tag{5.6}$$

Where S^m denotes a sample set of a certain node. S^L and S^R represent the sample sets that belong to a left child node and a right child node respectively. A best splitting hyper-plane $\langle \hat{\phi}_i, \hat{\theta}_i \rangle$ is selected according to a quality measurement. For labelled samples, entropy or Gini index are usual choices for the quality measurement. However, in our case, although P, Q are labelled as positive or negative, image patches p_i, q_j are not labelled. This requires us to define an unsupervised quality measurement to select a suitable hyper-plane.

Unsupervised quality measurement: The density forest provides an unsupervised quality measurement method under the assumption that the data in each node distributes with Gaussian distribution Criminisi, Shotton, and Konukoglu, 2011. The general information gain with respect to a node's sample set S and a random test is defined as:

$$I(S, \langle \phi_i, \theta_i \rangle) = H(S) - \sum_{m \in L,R} \frac{|S^m|}{|S|} H(S^m). \tag{5.7}$$

As the extension of information entropy, general differential entropy is defined as:

$$H(S) = -\int_S g(x) \log g(x) dx. \tag{5.8}$$

Where $g(x)$ is a general distribution function. With the assumption that the data in the sample set S obey a multi-variate Gaussian distribution, the above equation can then be rewritten by replacing $g(x)$ with Gaussian:

$$\begin{aligned} H(S) &= \frac{1}{2} \log((2\pi e)^d \det(\Sigma(S))) \\ &= \frac{1}{2} \log \det(\Sigma(S)) + c. \end{aligned} \tag{5.9}$$

Where c is a constant number equals to $\frac{1}{2} \log(2\pi e)^d$. The differential of multi-variate Gaussian entropy H is then defined by the determinant of the

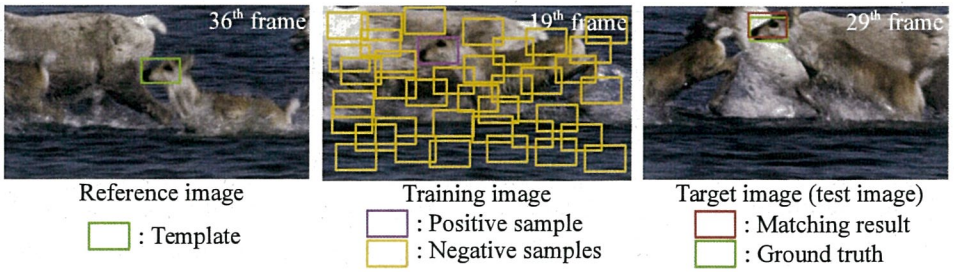


Figure 5.2: Example of sampling from training image. Training image (i.e. matching exemplar) is given alone and different with reference and target images. A single positive sample and a large number of negative samples can be extracted from the training image.

covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$, which can be seen as the volume of the hyper-ellipsoid that bounds the uncertainty of the data distribution. Substituting Equation 5.9 into Equation 5.7, we can get:

$$I(S, \langle \phi_i, \theta_i \rangle) = \frac{1}{2} \log\det(\Sigma(S)) - \frac{1}{2} \sum_{m\in L,R} \frac{|S^m|}{|S|} \log\det(\Sigma(S^m)). \tag{5.10}$$

Since the value of first term $\frac{1}{2}\log\det(\Sigma(S))$ is fixed as long as the sample set S of a node is fixed, it is not needed when maximizing I for selecting a best random test with respect to a given node. Note that $\Sigma(S)$ is the covariance matrix calculated from observation (real data), and the real data can distribute in a more complex way. It has been pointed in Pei, Kim, and Zha, 2013 that such measurement has the problem of rank-deficiency, and suggests to use the trace of covariance matrix instead of determinant. Although it has been argued in Sim and Roy, 2005 that trace is not suitable for covariance based metric due to the lack of invariance to scales and sensitiveness to the parameters, it is not the problem in our case because the RGB color feature is naturally well scaled by itself. In addition, we add two penalty terms to avoid splitting off degenerate clusters and ensure a full binary tree can be built. Based on the above discussion, Equation 5.10 can be rewritten as:

$$\begin{aligned} I'(S, \langle \phi_i, \theta_i \rangle) = &-\frac{1}{2} \sum_{m\in L,R} \frac{|S^m|}{|S|} \log \mathrm{tr}(\Sigma(S^m)) \\ &+\lambda_1 \max\left(\frac{|S^L|}{|S^R|}, \frac{|S^R|}{|S^L|}\right) \\ &+\lambda_2 \frac{\|\mathrm{centroid}(S^L) - \mathrm{centroid}(S^R)\|_\infty}{\sum_{m\in\{L,R\}} \max_{s\in S^m} \|s - \mathrm{centroid}(S^m)\|_\infty}. \end{aligned} \tag{5.11}$$

Where function centroid$(\cdot)$ returns the centroid of input sample set, λ_1 and λ_2 are constant numbers. The first weighted penalty term avoids an internal node to generate hyper-planes that split off extremely unbalanced child nodes. This is important for RCT since every node other than the leaf nodes should has two child nodes. The second penalty term is similar with Pei, Kim, and Zha, 2013, it increases as the centroid of two child nodes get apart and samples in each child node distribute closely with the centroid. With unsupervised quality measurement defined, we can select a best hyper-plane by maximizing the I':

$$\langle \hat{\phi}_i, \hat{\theta}_i \rangle = \underset{\langle \phi_i, \theta_i \rangle \in \varphi}{\arg\max} I'(S, \phi_i, \theta_i). \tag{5.12}$$

Sampling: Besides reference image and target image, we additionally use a matching exemplar (training image) to provide positive sample set I^{pos} (where $|I^{pos}| = 1$ in our template matching application) and negative sample set I^{neg}, $|I^{neg}| = \mathcal{N}$, and $\mathcal{N} \gg 1$. Each image sample $Q \in I^{pos} \cup I^{neg}$ has M image patches. As shown in Figure 5.2, positive sample is defined as the ground truth manually annotated. Negative samples are selected randomly from the entire training image and do not overlap with the positive sample.

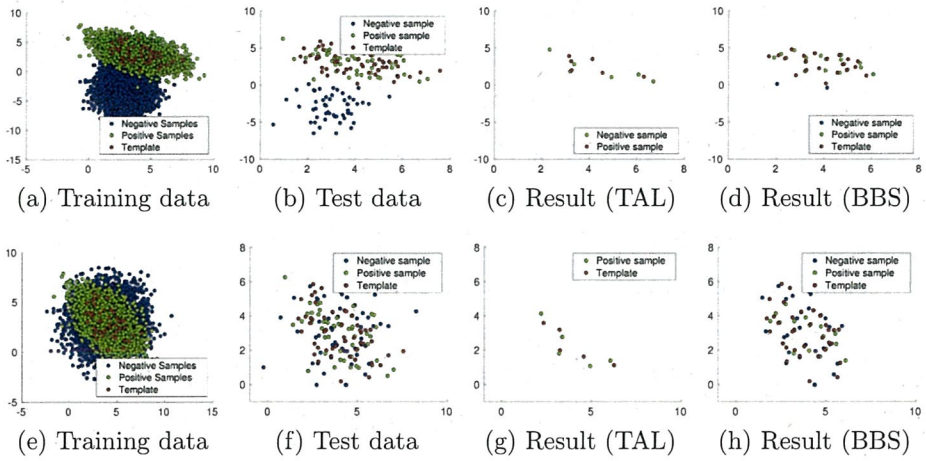
Cost function for tree selection: Slide-window search leads to dense matching which requires efficient similarity estimation. Considering the computational cost, we select a best symmetric RCT model from candidate model set $\{ff^t\}_{t=1}^T$ instead of performing weighted combination of each independent model's result. Given positive sample set S^{pos} and negative sample set S^{neg}, the cost function is defined and the tree model with minimum cost function value is selected for matching:

$$\hat{ff} = \underset{ff^t \in \{ff^t\}_{t=1}^T}{\arg\min} \frac{|I^{pos}| \sum_{Q \in I^{neg}} TAD(P,Q)}{|I^{neg}| \sum_{Q \in I^{pos}} TAD(P,Q)} + \lambda \sum_{Q \in I^{neg}} TAD(P,Q)^2. \tag{5.13}$$

Where a L2 regularization term weighted by λ is added to avoid the situation that $\sum_{Q \in I^{neg}} TAD(P,Q)$ and $\sum_{Q \in I^{pos}} TAD$
(P,Q) are both too small.

Analysis

The central point of this section is to generate synthetic numerical samples from two different mathematical distributions (Gaussian distributions) and



(a) Training data (b) Test data (c) Result (TAL) (d) Result (BBS)

(e) Training data (f) Test data (g) Result (TAL) (h) Result (BBS)

Figure 5.3: Visualization of matching results under two different 2D background models. a) Positive samples are drawn from distribution A and negative samples are drawn from distribution B. Template is also drawn from A and kept fixed during matching. b) One positive sample and one negative sample are drawn from A and B respectively for test. c) Matching result of TAL. Result points (p_i, q_j) are drawn as long as $d(p_i, Q) = 1$. d) Matching result of BBS. Result points are drawn as long as $bb(p_i, q_j) = 1$. e) Template and positive samples are the same with (a). Instead of B, distribution C is used to draw negative samples. (f∼h) Similar with (b∼d).

Table 5.1: Number of correctly distinguished pairs of test. A pair of test is correctly distinguished if positive sample has higher score than negative sample.

Number of test pairs / Method	100	200	300	400	500	600	700	800	900	1000
BBS (background B)	100	200	300	400	500	600	700	800	900	1000
TAL (background B)	100	200	300	400	500	600	700	800	900	1000
BBS (background C)	61	125	199	247	349	410	474	544	618	692
TAL (background C)	**76**	**147**	**226**	**300**	**375**	**455**	**533**	**615**	**694**	**772**

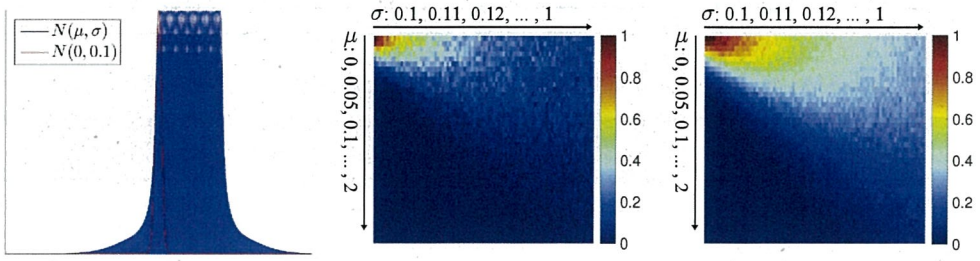
check whether two samples drawn from the same distribution has higher similarity score then the two samples which are drawn from two different distributions. We first visualize the matching results in 2D case (d=2) and then calculate the matching expectation of TAL with 1D case ($d = 1$). The state-of-the-art alternative Dekel et al., 2015 is compared in both cases.

2D case: Points of positive samples (foreground) and negative samples (background) are drawn from two different multivariate Gaussian distribution $N(\mu, \mathbf{\Sigma})$ respectively. Each sample and template consists of 50 2D data points. We prepare 3 kinds of distributions for generating da-

ta: for distribution A, $\mu = (4, 3)$, $\Sigma = (2, -1; -1, 2)$; for distribution B, $\mu = (3, -3)$, $\Sigma = (1, 0; 0, 4)$; for distribution C, $\mu = (4, 3)$, $\Sigma = (3, 0; 0, 3)$. Distribution B and C are both used to draw background points, the difference is, since C has the same μ with A, the background points are much more mixed with the points of foreground generated by A comparing with B. Figure 5.3 visualizes the matching results. As we can see, the matching results of TAL (Figure 5.3c and 5.3g) does not contain any points of negative sample, this indicates that negative samples can be sensitively distinguished from positive samples by TAL even the background is mixed with the foreground (Figure 5.3f). On the other hand, although BBS can less rule out the points of positive sample (Figure 5.3d), it confuses with mixed background (Figure 5.3h). Furthermore, we randomly generate more pairs of test data to validate two methods' distinguish ability between positive and negative sample, the statistical result is shown in Table 5.1. As we can see, in the case of generating background with distribution B, both TAL and BBS can distinguish between positive and negative sample well. When it comes to the case of distribution C, TAL has better distinguish performance.

1D case: To prove TAL as a better method, we have to show that 1) the expectation of two data points to be matched is highest when two points are sampled from the same distribution. Conversely, the expectation drops when two points are sampled from two different distributions; 2) Expectation of TAL drops more sharply when two distributions leave each other. At first, we generate a point set P under normal distribution $N(0, 0.1)$, $|P| = 1000$. Similarly, we generate Q under $N(\mu, \sigma)$, where μ ranges from 0 to 2 at intervals of 0.05, σ ranges from 0.1 to 1 at intervals of 0.01, and $|Q| = 1000$. We calculate $BBS(P, Q)/1000$ to approximate $E(BBS(p_i, q_j))$, $TAL(P, Q)/1000$ to approximate $E(TAL(p_i, q_j))$. As we can observe from Figure 5.4, 1) in both (b) and (c), higher expectation can be observed when parameters (μ, σ) are closer with (0, 0.1); 2) In (b), the expectation drops faster than (c) when (μ, σ) increase. These two observations show that TAL is more sensitive with the difference between distributions and thus results in better performance.

Influence of the number of negative samples: In general, the number of negative samples affects our algorithm from two perspectives: 1) With more negative samples, the clustering ability of RCT tree increases



(a) Distributions (b) $E(TAL(p_i, q_j))$ (c) $E(BBS(p_i, q_j))$

Figure 5.4: Expectation of a pair of points (p_i, q_j) to be matched. P is sampled from $N(0, 0.1)$ and Q is sampled from $N(\mu, \sigma)$. Parameters μ and σ are dynamically increased to plot each pixel in the heat maps (b) and (c). a) Probability density function of each distribution. b) Heat map of expectation generated by TAL. c) Heat map of expectation generated by BBS Dekel et al., 2015.

as information gain can be more reasonably calculated from more sufficient data for node splitting. 2) With more negative samples, the cost function can be calculated from more samples thus contributes to selecting a better model. However, increasing the number of negative samples can not always improve the performance of algorithm. With shallow depth of RCT tree, the number of internal hyperplanes is not enough to divide the data into "pure" clusters and the number of leaf nodes is also not enough to hold all kinds of clusters. In this condition, increasing the number of negative samples will conversely reduce the performance. We visually show how the number of negative samples affect the whole performance of our algorithm in 1D Gaussian case, which is shown in Figure 5.5. Under ideal conditions, highest expectation should be observed at top left, where $\mu = 0, \sigma = 0.1$. As we can observe from Figure 5.5, when the number of negative samples is 1, the high values of expectation do not gather on the top left area. When the number of negative samples increases to 100, the high values gather most closely on the top left area. However, further increasing the number of negative samples can not make the high values gather more closely due to the limitation of tree depth. This observation well supports our explanation on the influence of the number of negative samples.

Implementation and Complexity

In this section, we analyze the complexity of our algorithm, which can theoretically reflect the processing speed and memory cost. Instead of constant

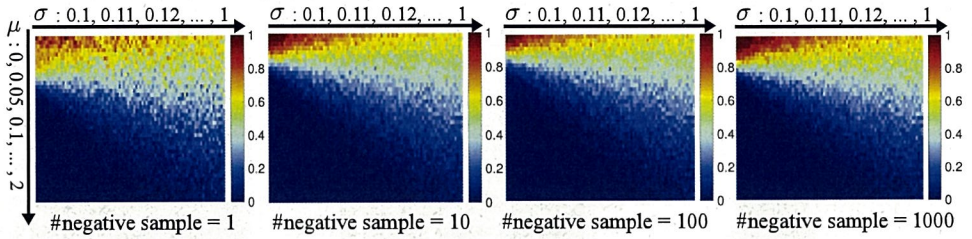


Figure 5.5: Influence of the number of negative samples. The number of negative samples varies from 1 to 1000. The number of positive samples is set to 1, the tree depth is set to 3 and the dimensionality of each sample is set to 100. Heat maps of the matching expectation are shown. Each heat map is generated by models with different number of negative samples.

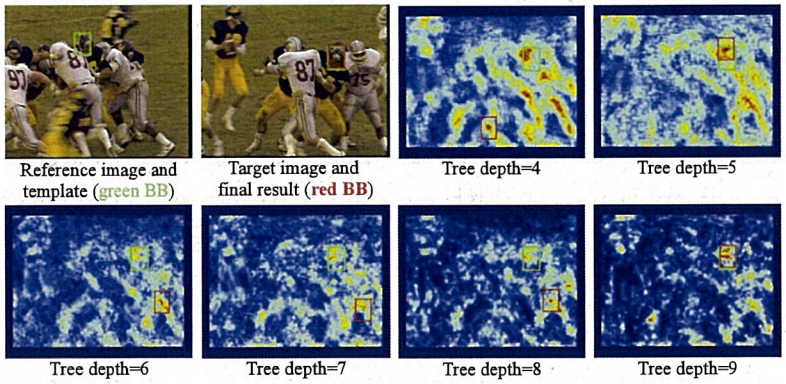


Figure 5.6: Influence of the tree depth. With the increase of tree depth, the red and yellow regions gradually shrink and the blue region expands. This observation indicates that deeper RCT has higher distinguish ability between positive sample and negative sample.

number, TAL dynamically determines patch size pz according to the size of template (ranges from 2 to 5 pixel in the experiment). The influence of pz and template size can be concluded as: in case of small template, large pz will lead to insufficient patches for training a reliable model. In case of large template, small pz will make each patch feature-less and burden with high computational cost in training stage. The stride of sliding-window-detection is set to the size of single image patch. Besides, calculating TAD over each detection window independently will result in redundant computation since detection windows always share image patches with each other. To improve the efficiency, we construct a buffer vector C where $C_j = q^C$ to assign each non-overlapped image patch of the target image to the according cluster in advance. The size of P and Q depend on the size of

template, the patch size pz, and the stride parameter of detection window. Assuming that a target image can be divided into $k \times k$ non-overlapped image patches, and a template can be divided into $k' \times k'$ image patches, by using buffering vector, the complexity reduces from $O(k'^2(k-k')^2)$ to $O(k^2)$.

Each splitting operation with hyper-plane has complexity of $O(d)$, we randomly select 5 channels from the patch feature instead of full d-dimensional feature to construct each hyper-plane. With depth of tree given by D, the main complexity for matching a target image is $O(Ddk^2)$. The D is set to 9 during the comparative experiment. On the other hand, BBS is a symmetric matching method and it needs to compute full d-dimensional Euclidean distance between each patch in the template and each patch in the target image, thus it has complexity of $O(dk'^2k^2)$ using buffer matrix, which is larger than TAL.

5.5 Experiments

Qualitative Evaluation

Influence of hyper-parameters: Main hyper-parameters of TAL include tree depth, number of trees, number of random tests, and number of feature channels for splitting. In Figure 5.6, we take an example of matching and plot the according likelihood map to analyse the influence of tree depth. We can see that when the depth is small, multiple local optimums can be observed. With increase of the tree depth, the number of negative local optimums decrease while the positive optimum remains. As a conclusion, we state that deeper structured tree can better distinguish between positive and negative sample. Based on the matching example shown in Figure 5.6, Figure 5.7 shows the change of cost function value with respect to the tree depth and the number of candidate trees respectively. As we can see, both increasing the tree depth and number of candidate trees can reduce the cost function value and thus contribute to selecting a better model. Considering the computational cost, we set tree depth as 9 and the number of candidate trees as 100 in the experiment.

Robustness with multi-view geometry: In real-world applications of non-parametric template matching (e.g. 3D reconstruction, product inspection), the key characteristic expected by users is the ability to handle matching tasks that not limited to ideal geometry models. We use the

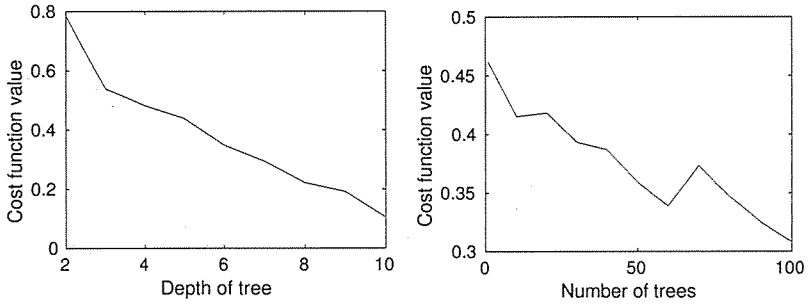


Figure 5.7: Curves to show the relationship between hyper-parameters (tree depth and number of trees) and cost function value based on the test example shown in Figure 5.6. The cost function is defined in Equation 13. Lower cost function value means that a selected model can distinguish between positive and negative samples better.

famous multi-view sequence Graffiti [1] to evaluate this characteristic and plot the results of TAL, BBS and SSD in Figure 5.8. As we can see, both TAL and BBS can deal with template matching under multi-view environment. However, in the case of the last target image, which exist large deformation caused by drastic view point change, BBS fails in matching while our method can still keep successful. As a baseline method, classical SSD cannot deal with multi-view situation well.

Quantitative Evaluation

We use the benchmark built by Dekel et al., 2015 to evaluate our method quantitatively. This benchmark is inherited from online tracking benchmark Wu, Lim, and Yang, 2013. In this benchmark, various difficulties in real scenes have been taken into account (e.g. illumination variation, occlusion, deformation, background clutter), and it is more challenging for global template matching task than ROI based online tracking task. There are 105 pairs of template and target image in this benchmark in various sizes. Each pair consists frame t and $t+20$ of a sequence as template and target image respectively, and t is randomly selected. Additional training frame for TAL is selected from $[t+15, t+19] \cup [t+21, t+25]$ randomly. Only one kind of random seed is used throughout the experiment. The ground truth bounding boxes are annotated manually with a semantic foreground defined.

We use the overlap rate to judge whether a matching result is successful by

[1] http://www.robots.ox.ac.uk/ vgg/data/data-aff.html

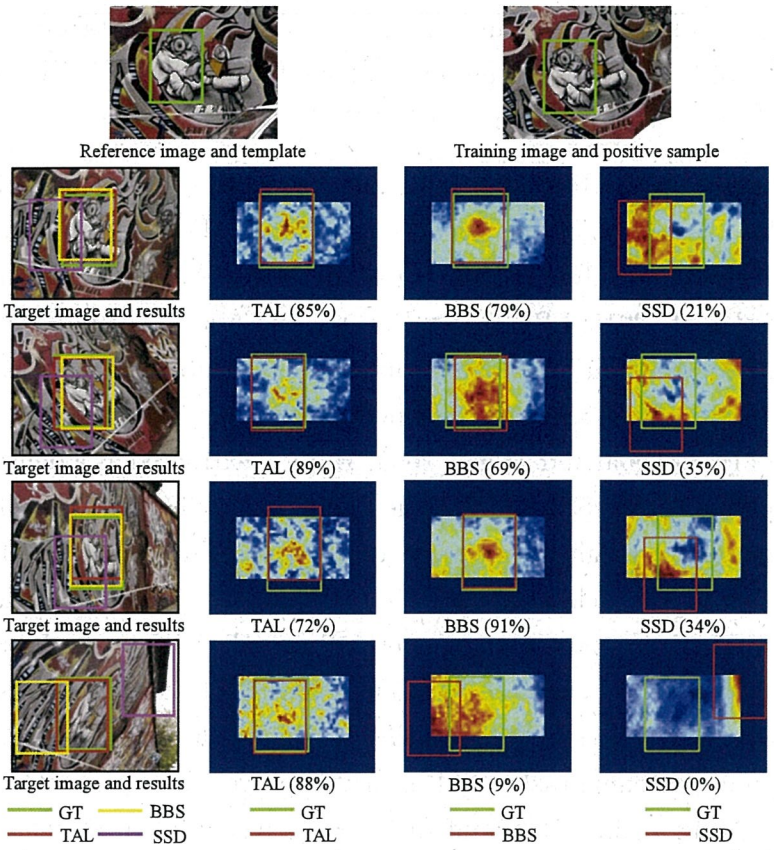


Figure 5.8: Qualitative comparison on Graffiti sequence. Between any two frames of the sequence, the camera's view point is changed. An object (comic person) is predetermined on this sequence. Each frame is manually annotated, and the center of the ground truth (GT) is kept as the object's center. Numbers within parentheses represent overlap rate.

referring to the ground truth. Specifically, the criteria used by PASCAL challenge Everingham et al., 2010 is applied to calculate the overlap rate:

$$\text{overlap rate} = \frac{\text{area}(BB_{re} \cap BB_{gr})}{\text{area}(BB_{re} \cup BB_{gr})}. \tag{5.14}$$

Where BB_{re} represents bounding box of result and BB_{gt} represents bounding box of ground truth. area$(\cdot)$ is a function to count number of pixels within the input area. Based on the overlap rate, we can obtain the answer about whether a matching result is correct or wrong by setting a threshold. Specifically,

$$\text{answer} = \begin{cases} 1 & \text{if overlap rate} > \text{threshold} \\ 0 & \text{otherwise} \end{cases}. \tag{5.15}$$

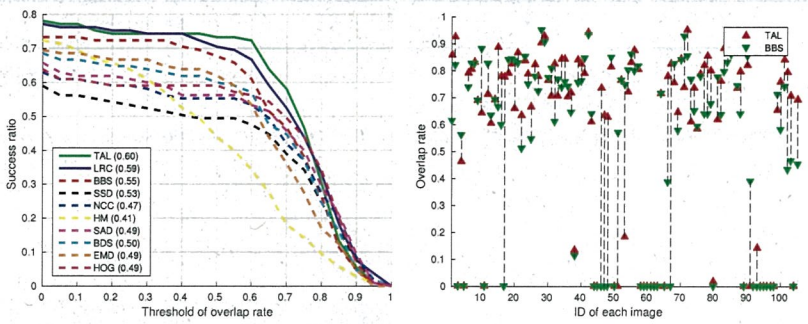
Finally, success ratio = #{answer|answer = 1}/#tests is taken as the accuracy criterion. All the experiments have been done on a PC equipped with Intel Core-i7 2.9GHz and 16 GB RAM.

We compare our method with both classical methods and state-of-the-art methods. Classical methods such as SAD, SSD, HM and NCC have been comprehensively studied in Ouyang et al., 2012. Among recent methods, BBSDekel et al., 2015, LRCZhang, Haitian, and Akashi, 2016, BDSSimakov et al., 2008 are patch based similarity measurements, which are closest to our method. Other methods include EMDRubner, Tomasi, and Guibas, 2000 and HOG Dalal and Triggs, 2005. HOG is extracted as a dense feature and combined with SSD during the comparison. Figure 5.9a illustrates the comparative result of accuracy at a glance. The threshold of overlap rate is dynamically changed and each threshold corresponds to a success ratio. Each curve represents a method's result. We can observe from Figure 5.9a that the curve of TAL outperforms the other methods overall. Especially, when threshold equals to 0.5, which is a widely-used criteria in detection or matching tasks, TAL nearly improves the accuracy by 6% and 3% comparing against BBS and LRC respectively. When threshold equals to 0.6, TAL nearly improves the accuracy by 9% and 4.5% comparing against BBS and LRC respectively. Also, we plot the overlap rate of TAL and BBS case by case in Figure 5.9b. We can see that in most of the cases, TAL can improve the performance comparing against BBS. Figure 5.10 shows some matching results on the benchmark. The likelihood maps generated by TAL converge on the ground truth more than ones generated by BBS.

5.6 Conclusion

In this chapter, we introduced a new method called Two-side agreement learning (TAL) to improve the accuracy of non-parametric template matching with a single matching exemplar for training. We compare our method against several widely used methods on public benchmark and show the effectiveness. TAL can work well under real-world scenes and make user easier to define the "similarity" since a matching exemplar is allowed to be provided.

Our method can fail when extreme changes occur between template and target image as well as other methods do. For example, drastic scaling



(a) (b)

Figure 5.9: Comparative result. Average success ratio of each method is shown after the according legend. Average success ratio is the mean of non-zero sample points on each curve. a) Success ratio curves with threshold of overlap rate is changed from 0 to 1. b) Case-by-case comparison with BB. TAL improves the overlap rate on many test cases in the benchmark.

change, illumination change, occlusion, etc. Part of the reasons are that we only use RGB color feature rather than many state-of-the-art features such as SIFT, HOG, etc. After TAL has been proved as effective, integrating such features can further improve the accuracy and contribute to many computer vision tasks which may benefit from object localization.

References

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Chechik, Gal et al. (2010). "Large scale online learning of image similarity through ranking". In: *The Journal of Machine Learning Research (JMLR)* 11, pp. 1109–1135.

Comaniciu, Dorin, Visvanathan Ramesh, and Peter Meer (2000). "Real-time tracking of non-rigid objects using mean shift". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 142–149.

Criminisi, A, J Shotton, and E Konukoglu (2011). "Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning". In: *Microsoft Research Cambridge, Tech. Rep. MSRTR-2011-114* 5.6, p. 12.

Dalal, Navneet and Bill Triggs (2005). "Histograms of oriented gradients for human detection". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 886–893.

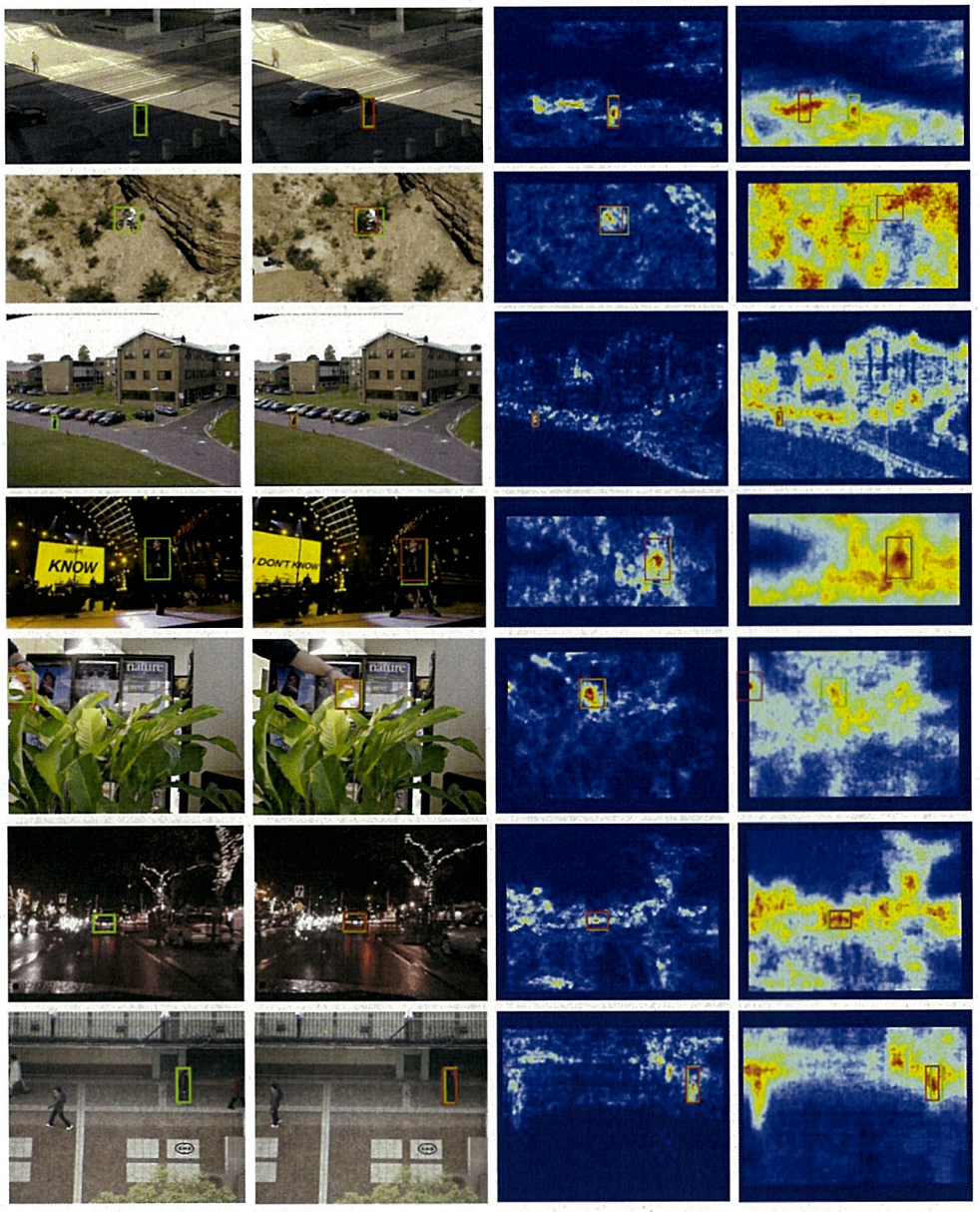


Figure 5.10: Examples of matching results on the benchmark. Likelihood maps of BBS and TAL are shown.

Davis, Jason V et al. (2007). "Information-theoretic metric learning". In: *International Conference on Machine Learning (ICML)*. ACM, pp. 209–216.

Dekel, Tali et al. (2015). "Best-Buddies Similarity for Robust Template Matching". In: *Computer Vision and Pattern Recognition (CVPR)*.

Di Stefano, Luigi, Stefano Mattoccia, and Federico Tombari (2005). "ZNCC-based template matching using bounded partial correlation". In: *Pattern Recognition Letters (PRL)* 26.14, pp. 2129–2134.

Everingham, Mark et al. (2010). "The pascal visual object classes (voc) challenge". In: *International journal of computer vision (IJCV)* 88.2, pp. 303–338.

Huttenlocher, Daniel P, Gregory Klanderman, William J Rucklidge, et al. (1993). "Comparing images using the Hausdorff distance". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 15.9, pp. 850–863.

Lucas, Bruce D, Takeo Kanade, et al. (1981). "An iterative image registration technique with an application to stereo vision". In: 81, pp. 674–679.

Ouyang, Wanli et al. (2012). "Performance evaluation of full search equivalent pattern matching algorithms". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 34.1, pp. 127–143.

Pei, Yuru, Tae-Kyun Kim, and Hongbin Zha (2013). "Unsupervised random forest manifold alignment for lipreading". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 129–136.

Rubner, Yossi, Carlo Tomasi, and Leonidas J Guibas (2000). "The earth mover's distance as a metric for image retrieval". In: *International Journal of Computer Vision (IJCV)* 40.2, pp. 99–121.

Shrivastava, Abhinav et al. (2011). "Data-driven visual similarity for cross-domain image matching". In: *ACM Transactions on Graphics (TOG)*. Vol. 30. 6. ACM, p. 154.

Sim, Robert and Nicholas Roy (2005). "Global a-optimal robot exploration in slam". In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 661–666.

Simakov, Denis et al. (2008). "Summarizing visual data using bidirectional similarity". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1–8.

Swain, Michael J and Dana H Ballard (1991). "Color indexing". In: *International Journal of Computer Vision (IJCV)* 7.1, pp. 11–32.

Tan, David Joseph et al. (2014). "Deformable Template Tracking in 1ms". In: *British Machine Vision Conference (BMVC)*.

Tversky, Amos (1977). "Features of similarity". In: *Psychological review* 84.4, p. 327.

Ullah, Farhan and Shun'ichi Kaneko (2004). "Using orientation codes for rotation-invariant template matching". In: *Pattern Recognition (PR)* 37.2, pp. 201–209.

Wu, Yi, Jongwoo Lim, and Ming-Hsuan Yang (2013). "Online object tracking: A benchmark". In: *Computer vision and pattern recognition (CVPR)*. IEEE, pp. 2411–2418.

Zhang, Chao and Takuya Akashi (2015). "Fast Affine Template Matching over Galois Field". In: *British Machine Vision Conference (BMVC)*.

Zhang, Chao, Sun Haitian, and Takuya Akashi (2016). "Robust Non-parametric Template Matching with Local Rigidity Constraints". In: *IEICE Transactions on Information and Systems* 99.9, in print.

Chapter 6

HIGH-SPEED AND LOCAL-CHANGES INVARIANT IMAGE MATCHING

6.1 Summary

In recent years, many variants of key point based image descriptors have been designed for the image matching, and they have achieved remarkable performances. However, to some images, local features appear to be inapplicable. Since theses images usually have many local changes around key points compared with a normal image, we define this special image category as the image with local changes (IL). An IL pair (ILP) refers to an image pair which contains a normal image and its IL. ILP usually loses local visual similarities between two images while still holding global visual similarity. When an IL is given as a query image, the purpose of this work is to match the corresponding ILP in a large scale image set. As a solution, we use a compressed HOG feature descriptor to extract global visual similarity. For the nearest neighbor search problem, we propose random projection indexed KD-tree forests (rKDFs) to match ILP efficiently instead of exhaustive linear search. rKDFs is built with large scale low-dimensional KD-trees. Each KD-tree is built in a random projection indexed subspace and contributes to the final result equally through a voting mechanism. We evaluated our method by a benchmark which contains 35,000 candidate images and 5,000 query images. The results show that our method is efficient for solving local-changes invariant image matching problems.

6.2 Introduction

During the last decade, image matching and retrieval technology have been widely studied. At the same time, with the development of internet and image editing technology, images are showing more and more diversity in our daily life. The explosion of image data requires image descriptors to be not only lighter but also more discriminative for matching and retrieval tasks. SIFT David G. Lowe, 2004 feature and other key point based image features well solved this problem and are now becoming one of the most popular research branches. However, key point based frameworks have become less effective against images with local changes (IL). IL can not be

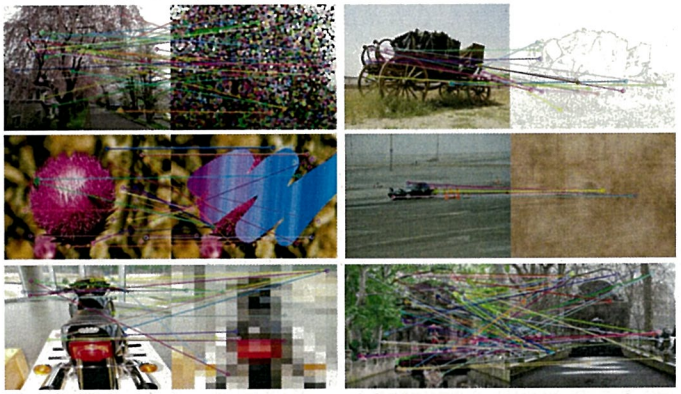


Figure 6.1: Examples of ILP. In this figure, we use SURF feature to match the key points detected in the image pairs. Because of the local changes occurred in the right column images, the key points can not be matched well. Our task is to grasp the visual similarity in ILP when such local changes exist.

defined by a single image since "changes" exist with respect to a normal image. IL is an image in an IL pair (ILP). An ILP includes two images: a normal image and its IL.

Main reasons for why local features are less effective in matching ILP can be concluded as : 1) In some IL, detection of corner points is difficult. 2) An ILP may contains many local changes (*e.g.* an ILP contains two photos which are taken at same place but in different seasons). 3) Multiple similar regions may exist in an IL. An ILP can be image with image, image with sketch, image with noise image, image with painting, image with synthetic image, image with blur image, image with edge image, etc. Figure 6.1 shows some examples of ILP. As we can see, various local changes can be considered such as changes of illumination, color, edge, shape, texture. The SURF Bay, Tuytelaars, and Van Gool, 2006 descriptor failed in matching with each key point, which will lead to a failure of matching the whole image.

Image matching is a basic research topic for various applications of computer vision, such as near-duplicate image detection (NDID), content-based image retrieval (CBIR), texture classification. The task is to search a large database of candidate images with a query image and then finds the result which matches to the query image. In our problem, there is one query as the input and one image as the output. The input and output are exactly

the same despite the local changes. Most of the image matching problems are studied from two points of view: 1) image feature presentation 2) nearest neighbor (NN) search technique.

Low-level features, such as histogram based gradient features and key point based features, usually have trade-off problem between the number of dimensions and the discriminative ability. In other words, features composed of more dimensions usually have stronger ability to present an image's visual similarity. Although higher-level features learned by sparse coding/deep learning can surely present image's visual similarity well with less dimensions, the coding process is time consuming. In our research, the original high-dimensional HOG feature is projected onto a low-dimensional subspace while trying to keep discriminative ability based on Achlioptas, 2003. According to the compressive sensing theory, a small number of randomly generated linear measurements can preserve most of the salient information. The projection processing does not cost much time when the projection matrix is very sparse.

On the other hand, most of the current data structures for effective NN search can only index data points in low-dimensional feature space. These data structures become less efficient with the growing of dimension number due to the curse of dimensionality. The difficult point is that it is hard to solve the exact NN problem efficiently in high-dimensional feature space while the accuracy is low when solving the exact NN problem in low-dimension feature space. In this chapter, our solution is to find numerous approximate nearest neighbors (ANN) in thousands of low-dimension feature spaces and then vote for the best ANN as the final output. By doing this, we can accelerate the matching procedure while achieve satisfactory matching accuracy. To the best of our knowledge, there is little study on image matching problem with various types of local changes. Our method builds a huge number of subspaces to reduce noise's effect brought by local changes, and finally grasps the global similarity between query and candidate images.

6.3 Related Work

Feature Descriptors for Visual Similarity Evaluation

Among recent state-of-art works, local feature descriptors for quantifying images' visual similarity have been proved to be very effective. SIFT David

G. Lowe, 2004 and its variants are representative. Furthermore, D. C. Hauagge *et al.* Hauagge and Snavely, 2012 proposed a local feature descriptor which based on detecting and representing local symmetries for matching pairs of photos taken at urban scenes. K. Grauman *et al.* Grauman and Darrell, 2005 proposed a technique that compares images by matching their distributions of local invariant features.

On the other hand, global feature descriptors are also used for evaluating the visual similarity. A. Oliva *et al.* Oliva and Torralba, 2006 noted the global image features play an important role on scene perception. S. Lazebnik *et al.* Lazebnik, Schmid, and Ponce, 2006 noted that a global feature representation can be surprisingly effective for identifying the overall scene. P. Li *et al.* Li et al., 2012 proposed a method to enrich the discriminative ability of local feature with global information. They noted that the current local descriptors will fail to match when an image has multiple similar regions. C. Zhang *et al* Zhang and Akashi, 2015 proposed a compressed HOG descriptor for IL image matching. They used random projection to compress the high-dimensional HOG feature into low-dimensional feature. However, in matching procedure, only a simple brute-force method with L1 distance measure is applied.

Image Matching and NN Search

NN search problem for image matching has been widely studied. For exact image matching, brute-force is an efficient method especially the number of feature dimension is large. A. Torralba *et al.* Torralba, Fergus, and Weiss, 2008 applied brute-force search to match images which are converted into binary code from GIST descriptor. When the number of feature dimension is small, many data structures can be applied for image matching such as Kd-tree, R-tree, Ball tree, SR-tree. C. Silpa *et al.* Silpa-Anan and Hartley, 2008 introduced an optimized Kd-tree Friedman, Bentley, and Finkel, 1977 algorithm which is used to match SIFT descriptors. On the other hand, instead of finding the nearest image to the query image, approximate image matching aims to find images which are within a certain distance threshold to query image, such as image retrieval. ANN search can deal with the high-dimensional feature effectively by reducing the dependency on dimensionality. Y. Ke *et al.* Ke, Sukthankar, and Huston, 2004 employed local sensitive hashing (LSH) to index the local descriptors for near-duplicate detection. LSH also applied the random projection for searching ANNs

over high-dimensional data. P. Wu *et al.* Wu et al., 2011 used multiple randomly projected kd-trees to search ANN. Each kd-tree search the ANN in a random projected low-dimensional space and rank the results by distance at last.

Image Matching with IL

To the best of our knowledge, there are few chapters for studying all types of IL. Sketches and paintings are most studied problems belong to ILP matching. A. Shrivastava *et al.* Shrivastava et al., 2011 defined IL as cross-domain images, the authors mainly considered the matching task for sketches, paintings and photos taken in different seasons which are all included in the definition of IL. They learned the weights for each HOG feature's dimension with single positive query image and a very large set of negative images by SVM. The training process is very time consuming and hard to be finished within query time. Other similar chapters include Eitz et al., 2011 for matching sketches with photographs, Russell et al., 2011 for matching paintings with photographs, Chong, Gortler, and Zickler, 2008 for matching images under different illumination conditions. Furthermore, Zhang and Akashi, 2015 used a compressed HOG descriptor and brute-force NN search to match the ILP. In this chapter, the dimension number of original HOG descriptor is reduced from 6384 to 500 in order to reduce the burden of matching time. However, after projection with a single random sparse matrix, the original feature lost original information naturally. The balance between matching accuracy and matching time is still not be solved well in this chapter.

Our work is mainly based on work Zhang and Akashi, 2015; Wu et al., 2011. We use the feature descriptor proposed in Zhang and Akashi, 2015 and enhance the ANN method proposed in Wu et al., 2011 for high-speed exact ILP matching.

6.4 Methodology

Problem Setting

We have a set P_c of n pre-processed candidate image feature vectors $\{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n\}$, where $\mathbf{p}_n \in \mathbb{R}^m$. m is the number of compressed feature vector's dimension. Given an arbitrary query feature vector $\mathbf{q}_n \in \mathbb{R}^m$ from query set P_q, return $\mathbf{p}_n$ which is closest to $\mathbf{q}_n$ under the distance measurement function. In the following sections, we will first introduce how to generate

set P_c and P_q from candidate image set I_c and query IL set I_q, and then we will introduce how to find $\mathbf{p}_n$ by using random projection indexed KD-tree forests (rKDFs).

Feature Compression

HOG feature Dalal and Triggs, 2005 counts occurrences of gradient orientations in cells/blocks/windows and merge them into one feature vector $\mathbf{p}' \in \mathbb{R}^n$, n is the dimension number of original HOG feature. In this chapter, we treat the whole image as a single window, and construct grid-like structure for extracting feature with units called block and cell. We define R as a $m \times n$ random measurement matrix, and $r_{ij} = R(i,j)$. $R(i,j)$ denotes the entry in row i, column j of matrix R. Each r_{ij} is independent with others and decided by the following probability distribution,

$$r_{ij} = \sqrt{s} \times \begin{cases} +1 & \text{with probability } \frac{1}{2s} \\ 0 & \text{with probability } 1 - \frac{1}{s}. \\ -1 & \text{with probability } \frac{1}{2s} \end{cases} \tag{6.1}$$

Achlioptas *et al.* Achlioptas, 2003 state that when the $s = 1$ or 3, R satisfies the Johnson-Lindenstrauss lemma. Such kind of matrices can achieve favorable compression performance. The method of Wu et al., 2011 also uses $s = 3$ to generate the random measurement matrix. When $s = 3$, only $1/3$ data need to be processed. However, when the size of candidate image set I_c is very large, the procedure of pre-processing becomes time consuming. For each query image, although compression operation only needs to be performed once, we hope to avoid large amount of numerical calculation in order to reduce query time as much as possible. Fortunately, this random sparse matrix has been proved to be effective even $s \gg 3$ Zhang, Yamagata, and Akashi, 2015. In this chapter, we set $s > n/2$. Therefore, only $2/n$ data need to be processed at most. Parameter s is determined by rule of thumb. For example, in both Zhang, Yamagata, and Akashi, 2015 and Zhang and Akashi, 2015, s is set as $n/4$ for efficient compression procedure. In addition, since no floating-point arithmetic is needed expect a square root operation, the compression process needs little computational cost. Also, this random measurement matrix only needs to be generated once during the pre-processing procedure.

The process of compression can be seen as a projection from the high-

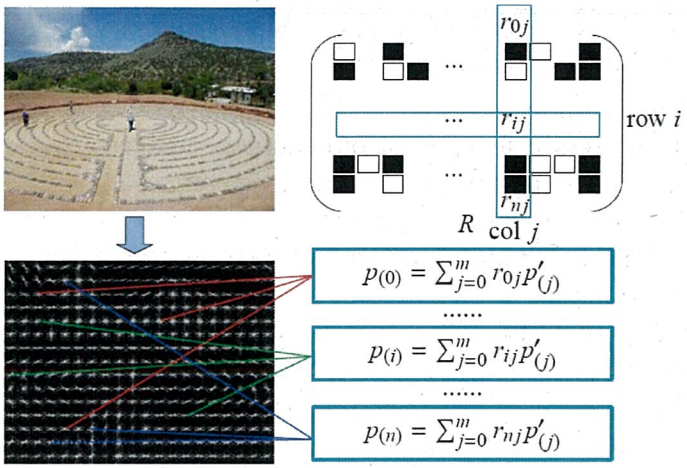


Figure 6.2: Compressing HOG feature from m-dimension to n-dimension. The dimensions of $\mathbf{p}'$ are randomly selected for generating $\mathbf{p}$.

dimensional space to low-dimensional space. We define $\mathbf{p}'$ as high-dimensional HOG feature ($\mathbf{p}' \in \mathbb{R}^n$), $\mathbf{p}$ as low-dimensional compressed feature ($\mathbf{p} \in \mathbb{R}^m$). For the sparse random projection, $n \gg m$. The compression procedure can be presented as,

$$\mathbf{p}^{(m\times 1)} = R^{(m\times n)}\mathbf{p}'^{(n\times 1)}. \tag{6.2}$$

This quick and simple matrix multiplication complies our requirements for computing speed. However, with larger s, the loss of feature's information will be unavoidable. We apply multiple random matrices to remedy this problem. We use $\mathbf{p}_{ij}$ to denote the feature vector of image I_i which is compressed by random matrix R_j. As a result, each image will be presented by γ compressed feature vectors in total instead of a single vector. Although dimension number of each $\mathbf{p}_{ij}$ is much smaller than the descriptor proposed in Zhang and Akashi, 2015, the combination of all the $\mathbf{p}_{ij}$ can hold more information. Furthermore, lower-dimensional $\mathbf{p}_{ij}$ is much easier to be processed by KD-tree.

The theoretical foundation of why such a simple matrix can do data compression well is proved in Baraniuk et al., 2007. R.Baraniuk *et al.* give a simple proof that R satisfies the restricted isometry property. At the same time R satisfies the Johnson-Lindenstrauss lemma, thus it has high probability to reconstruct $\mathbf{p}'$ from $\mathbf{p}$ with minimum error. Figure 6.2 shows the feature compression procedure. R is created with Equation 1, black squares represent positive entries, and the white squares represent negative

Algorithm 4 Feature extraction and compression.

Require: Candidate image set : I_c
Require: Query image set : I_q
Require: Compressed feature set of candidate images: P_c
Require: Compressed feature set of query images : P_q
Require: Gaussian blur kernel size : k
Require: Number of random matrices : γ
Require: Parameter : s

```
for i from 1 to γ do
    Generate projection matrix R_i with Equation 1 and s
end for
for each image I_i in I_c ∪ I_q do
    I_i = gaussianBlur(I_i, k)
    p' = extractHOG(I_i)
    for j from 1 to γ do
        p_ij = compress(p', R_j)
        l_2 = normalize(p_ij)
        if I_i ∈ I_c then
            Push p_ij to P_c
        else
            Push p_ij to P_q
        end if
    end for
end for
Return P_c and P_q
```

entries. In order to calculate i th dimension's value $\mathbf{p}_{(i)}$, dimensions of $\mathbf{p}'$ are randomly selected and combined according to R.

The whole compression procedure can also be considered as a procedure to improve the original HOG's feature level. The problem of HOG feature is that it is not clear which dimension of $\mathbf{p}'$ performs a more important role in further application, which dimension of $\mathbf{p}'$ is useless. After compression, each dimension of $\mathbf{p}$ is calculated from multiple dimensions of $\mathbf{p}'$, thus more information is included in $\mathbf{p}$'s single dimension than $\mathbf{p}'$. As we all know that with higher level features, less dimensions are needed to hold the same discriminative ability. From this point of view, we can also understand why the random projection works for feature dimension reduction with less loss of discriminative ability.

The preprocessing algorithm can be concluded with Algorithm 4.

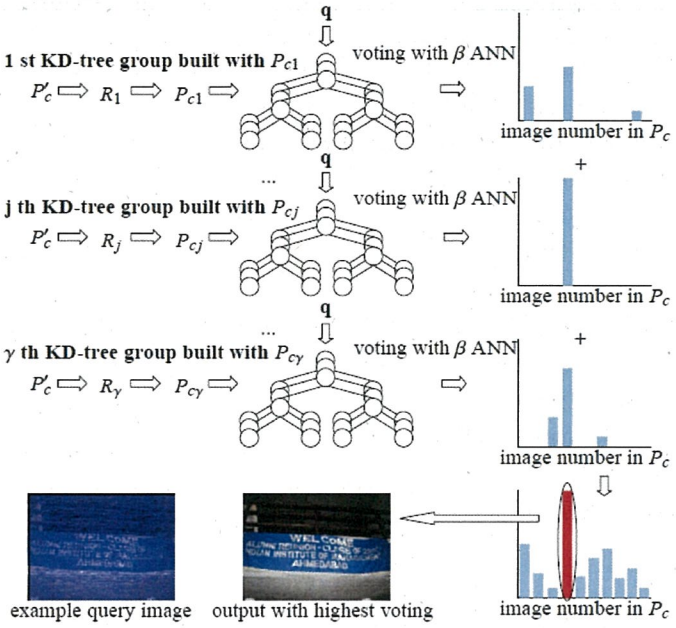


Figure 6.3: The main process of our modified image retrieval algorithm

Random Projection Indexed KD-tree Forests

KD-tree Robinson, 1981 is a widely used tree structure for searching ANN in multi-dimensional data space. It is a binary tree with each node has a hyper-plane (typically one dimension) to divide the data space into two subspaces. The feature vectors which are left to the hyper-plane will be assigned to left child node, and the feature vectors which are right to the hyper-plane will be assigned to right child node. As one of the ANN algorithms, KD-tree works effectively when dealing with low-dimensional data. However, KD-tree works poorly especially the number of feature vector's dimension is large since it will degrade to linear search Gionis, Indyk, Motwani, et al., 1999. In our matching problem, KD-tree seems to be inapplicable because the dimension number of HOG feature is large. By using the compression method mentioned above, dimension number of HOG feature can be reduced. In our condition, we set compressed feature vector's dimension extremely small to build KD-tree in an effective way (*e.g.*, $n = 6384, m = 10$). Such KD-tree is very light both in memory and search time. However, much information on the original feature vectors will be lost naturally and ANN of query ICL becomes hard to search by single KD-tree. To solve this problem, our idea is to build a large scale KD-tree forests (*e.g.*, $\alpha = 8,000$) with each tree indexed by a random matrix. Each

Algorithm 5 Build rKDFs.

Require: Compressed feature set of candidate images: P_c
Require: Number of random matrices : γ
Require: Number of trees in one group : δ
Require: Maximum depth of one tree: λ
Ensure: $|P_c| > 0$

```
for j from 1 to γ do
    for i from 1 to δ do
        Initialize KD-tree KT with root node n and data P_cj
        while depth(n) < λ do
            splitNode(n)
            n = findLeaf(KT)
        end while
        Push KT into tree group KG_j
    end for
end for
Return KD-tree groups KG
```

tree in rKDFs is built with different input compressed data sets which are generated by different random matrices. "indexed" in rKDFs means that we use one random matrix to discriminate a certain tree from others. ANN results returned by a single KD-tree in rKDFs are very inaccurate but better than random guesses, because the compressed feature space's dimension is too small to reflect the original feature space's data distribution. We vote with ANNs provided by each tree by a histogram and at last select the ANN which is most voted as the final output. Figure 6.3 illustrates the whole processing. P_{cj} denotes a compressed feature set which is compressed via random matrix R_j.

We now introduce how to build rKDFs. P_c includes γ feature subspaces which are returned by Algorithm 4. We build δ trees in one subspace in parallel. δ trees in one subspace form a tree group. For each tree, data in the according subspace is partitioned recursively from the root node to leaf nodes. In initialization process, dimensions with large variance are selected as candidate dimensions to partition the data (*e.g.*, five dimensions). At each node, we first randomly select a dimension for splitting and then calculate the median value of this dimension. After that, all the feature vectors in the node will be split into two child nodes according to the median value. The split operation will stop until the depth of the tree reaches to the depth threshold λ. In order to store all the trees, we need

space complexity about $O(\alpha m \times |P_c|)$. The building algorithm is concluded in Algorithm 5.

We now introduce how to search with rKDFs. To find the best ANN of a given feature vector $\mathbf{p} \in P_q$, we need to search with $\gamma \times \delta$ trees. After preprocessing, $\mathbf{p}$ has already been projected into γ subspaces. In each subspace, we search ANNs of $\mathbf{p}$ with a tree group which is returned by Algorithm 5. However, these ANNs are very inaccurate since each is outputted by a single tree. In order to boost the accuracy, ANNs searched by a tree group are ranked by distance and output best β ANNs for voting the final NN. The voting mechanism is established under this assumption: exact NN of a query image has higher probability to appear in the ANNs of each sub feature space. We need time complexity about $O(\alpha n \times \log |P_c|)$ to search with one query. The searching algorithm is concluded in Algorithm 6.

The differences between our matching method and Wu et al., 2011 can be concluded as following. 1) We introduced randomized kd-tree forests Vedaldi and Fulkerson, 2010 to divide tress into groups according to different subspaces. 2) Method of Wu et al., 2011 uses only about 20 trees to search ANNs, in our condition, number of trees is 8,000 and more. 3) Method in Wu et al., 2011 ranked all the ANNs by distance and treat the top rank which is closest to the query in subspace as the final NN. This method will become less effective when the dimension number of original feature vector is very large like the HOG feature. Because the distance measurement in subspace can not well reflect the distance in the original feature space. Our method vote with all the ANNs to determine the final NN which appeared most frequently as an ANN. 4) Our method compresses feature dimension from thousands to 10 while method of Wu et al., 2011 compresses feature dimension from hundreds to 10. Random projection is a random method which does not depend on any training data, thus building large number of KD-trees in a same subspace is risky and unwarranted. Our method disperses the risk to each subspace and achieve better performance overall.

Algorithm 6 Search with rKDFs.

Require: Compressed feature set of query images : P_q
Require: Compressed feature set of candidate images : P_c
Require: Number of random matrices : γ
Require: KD-tree groups KG
Require: Number of ANNs outputted by a single KD-tree group: β
Ensure: $|P_q| > 0$ and $|P_c| > 0$

```
for i from 1 to |P_q| do
    initialize histogram with |P_c| bins
    for j from 1 to γ do
        ANN = search(KG_j, p_ij, β)
        vote(histogram, ANN)
    end for
    Return arg max_{1≤o≤|P_c|} bin_o
end for
```

6.5 Experiment

Experiment Environment

We use the benchmark used in Zhang and Akashi, 2015 to evaluate our method. It is a challenging benchmark which contains 5,000 query images and 35,000 candidate images. 5,000 query images are modified with local changes based on normal images which are randomly selected from 35,000 candidate images. Width of images is between 454 pixels to 1272 pixels, the of images is between 482 pixels to 1024 pixels. Many types of IL are included in the query set, and the local changes can be mainly concluded into three categories: changes of color-texture information, changes of edge-gradient information, and changes with special filters. Color-texture information can be changed by the adding of text and scribbling, illumination changes, image binarization, etc. Edge-gradient information can be changed by image rotation, local deformation, text adding, scribbling, etc. Special image filters will largely change image's local feature and keep global similarity like crayon drawing, oil paint, pencil drawing, pixel explosion, stained glass, etc. This benchmark is an one-to-one matching task benchmark, a query image and its ground truth are exactly the same despite the local changes. To the best of our knowledge, there are few similar benchmarks for one-to-one image matching task involving various types of local changes.

We did all the experiments with a PC equipped with Intel Core-i5 2.5GHz

Table 6.1: Parameter setting for each experiment. Parameter value with "–" is variable during the according experiment. Results are summarized in Figure 6.5 (shown in last page).

Sub figure No.	α	β	s	d	k	n	m
(a)	–	10	6000	35000	31	6384	10
(b)	400	–	6000	35000	31	6384	10
(c)	400	10	–	35000	31	6384	10
(d)	8000	10	6000	–	31	6384	10
(e)	8000	10	2000	35000	–	2700	10
(f)	400	10	6000	35000	31	6384	–

CPU and 6 GB RAM.

Effect of parameters

In this section, we systematically report the experimental results for studying how each parameter affects the performance of our matching method. In this chapter, some parameters are fixed to reduce complexity of experiment. We set the number of trees $\delta = 4$ in each tree group, image size as 320×240, gradient angle's range as $[0°, 180°]$, σ_x of Gaussian blur as 8 and σ_y of Gaussian blur as 6. Figure 6.5 (shown in last page) summarizes the effects of various parameters. Experimental conditions for each sub figure are given out in Table 6.1. Two evaluation criteria are observed during experiments: error rate and matching time per query image in milliseconds. Error rate is defined as follows,

$$\text{error rate} = 1.0 - \frac{\sum_{p \in P_q} \text{match}(p, P_c)}{|P_q|}, \tag{6.3}$$

match function returns 1 if a query image can be approximately matched according to ground truth, returns 0 if not.

As Figure 6.5(a) shows, increasing the number of trees α of rKDFs improves the performance significantly. Error rate stops decreasing from a certain value of α. The matching time per query image increases linearly as α increases. As Figure 6.5(b) shows, when the number of ANNs outputted by each tree group is increased, the voting process appears to be more accurate. As Figure 6.5(c) shows, with the increase of s, error rate declines in a stepwise fashion. Larger s leads the algorithm to generate a more sparse random matrix to compress the original HOG feature vector.

Table 6.2: Example of parameter settings for extracting HOG feature. n is the dimension number of HOG feature. To calculate the error rate, we set $\alpha = 8000$, $\beta = 10$, $d = 35000$, $k = 31$, $m = 10$.

n	error rate	block	block stride	cell	bin	s
1836	0.149	(64,64)	(16,16)	(64,64)	9	1000
2396	0.099	(32,32)	(16,16)	(32,32)	9	2000
2700	**0.066**	**(16,16)**	**(16,16)**	**(16,16)**	**9**	**2000**
5508	0.110	(64,64)	(16,22)	(32,32)	9	5000
6384	0.076	(32,32)	(16,16)	(16,16)	6	6000
9576	0.071	(32,32)	(16,16)	(16,16)	9	6000
12768	0.080	(32,32)	(16,16)	(16,16)	12	6000
29376	0.087	(64,64)	(16,16)	(16,16)	9	6000
35964	0.092	(32,32)	(8,8)	(16,16)	9	6000

In our method, larger s shows to be a more appropriate choice. As one of the possible reasons, excessive compression may cause bad influence on calculating the visual similarity of IL conversely. As Figure 6.5(d) shows, with the increase of candidate images, error rate maintains. This can explain that our method is robust in change of search space. We can also find that the processing time increase linearly with the increase of d, this is very important for practical applications. From Figure 6.5(e) and (f), we can find that there exists minimum error rate while increasing k and m step by step. We can tune both the parameters by a validation set. Furthermore, the dimension number of original HOG feature vector n also plays an important role. Some parameter tuning examples are shown in Table 6.2. We found out that parameter setting with 9 bins performs better than 6 bins, the best accuracy is achieved when block size, cell size, block stride are all set to (16,16). After tuning on a validation data set, we report the lowest error rate in the next section.

Comparison

We compare our method with others from two aspects for the different needs of practical applications, 1) Considering accuracy as priority, 2) considering matching time as priority. The compared methods are listed below:

– **N-BoF-SIFT**: In experiment, we combine BoF with SIFT David G. Lowe, 2004. The visual vocabulary is built by randomly sampled images from the candidate dataset and each packaged feature is finally normalized.

We extract 128 dimension SIFT descriptors for all the detected key points and then use K-means clustering method, which is usually used in many BoF implementations, to cluster visual words. Initial centroid positions of K-means are chosen according to Arthur and Vassilvitskii, 2007. For the assignment task, we use fast approximate nearest neighbor (FLANN) Muja and David G Lowe, 2009 to assign the novel features to the closest terms in the vocabulary. After normalization, we use the packaged feature to match the data set by L1 distance.

– **RP-HOG**: The method in Zhang and Akashi, 2015 compresses the HOG feature with random projection and then match the NN by brute-force with L1 distance measurement.

– **N-HOG**: Original HOG features Dalal and Triggs, 2005 are extracted from each image and then normalized by L2 norm. We use brute-force method to match the NN with L1 distance measurement.

– **GIST**: Gist feature Oliva and Torralba, 2006 is a global image feature which convolves a gradient filter to encode the amount and strength of edges. After Gist is extracted, we use brute-force method to match the NN with L1 distance measurement.

– **Fisher Vector**: GMM is used to construct a visual word dictionary at first. We extract SIFT feature as the local feature of each image. Fisher Vector is encoded by the SIFT feature and the prior obtained GMM, and finally normalized by L2 norm.

– **VLAD**: VLAD can be seen as the simplification of Fisher Vector. In experiment, K-Means is used instead of GMM for visual word generating, and KD-tree is used for vector quantization. It is also normalized by L2 norm at last.

Table 6.3 and 6.4 show the comparison results. From Table 6.3 we can see that SIFT appears to be very ineffective even the visual words are set to 10,000. By this point we can prove the effectiveness of our dataset for evaluating the KF images. Furthermore, our method is about 54% faster than N-HOG at a same accuracy level. We successfully converted a high-dimensional feature matching problem into a low-dimensional matching problem and improved the accuracy. From Table 6.4 we can see that our method is about 74% faster than RP-HOG at a same accuracy level and outperforms other methods both in accuracy and time. Although Fisher Vector and VLAD can perform very well in standard image retrieval tasks,

Figure 6.4: Examples of comparison with method Zhang and Akashi, 2015

Table 6.3: Comparison results considering accuracy as priority. Dimension number of descriptors are shown after according method's name.

method	error rate	matching time per query image (ms)
our-method-accurate	**0.061**	134.3
BOW-SIFT-10000 Csurka et al., 2004	0.426	541.2
RP-HOG-3000 Zhang and Akashi, 2015	0.135	128.5
N-HOG-6384 Dalal and Triggs, 2005	0.067	294.6
GIST-6400 Oliva and Torralba, 2006	0.211	295.3
Fisher-Vector-7680 Perronnin and Dance, 2007	0.637	352.1
VLAD-6400 Delhumeau et al., 2013	0.66	294.9

it can not perform well in our problem. The main reason can be concluded as: both of the methods are developed based on the local features such as SIFT, the matching error brought by local features can be further expanded during the transformation of features. As a conclusion, our method can match ILP in high-speed with large scale candidate database, at the same time, accuracy is satisfactory. Dense sampling methods such as SIFT Flow Liu, Yuen, and Torralba, 2011 is recently showing the effectiveness. However, to compute a 128-dimensional SIFT feature for each pixel is very time consuming and impractical. With the increase of data base's size, both the time and space complexity grow dramatically.

Figure 6.4 intuitively shows some matching examples comparing to Zhang and Akashi, 2015. A synthesized image example and a distorted image example are shown to be mismatched by Zhang and Akashi, 2015 while our method can still match correctly.

6.6 Conclusion

This chapter presented a problem which aims to match a special category of images called IL. The proposed method applied a compressed HOG descriptor for extraction and introduced rKDFs for high-speed NN search.

Table 6.4: Comparison results considering matching time as priority. Dimension number of descriptors are shown after according method's name.

method	error rate	matching time per query image (ms)
our-method-fast	0.159	**5.5**
BOW-SIFT-500 Csurka et al., 2004	0.584	92.6
RP-HOG-500 Zhang and Akashi, 2015	0.151	21.3
N-HOG-1836 Dalal and Triggs, 2005	0.187	88.6
GIST-512 Oliva and Torralba, 2006	0.362	23.7
Fisher-Vector-1280 Perronnin and Dance, 2007	0.630	57.4
VLAD-1280 Delhumeau et al., 2013	0.560	57.9

There still exist some limitations in practical applications. The main limitation is that compressed HOG descriptor is not rotation invariant, thus IL presented in different rotation angles will fail in matching. Furthermore, our method will fail in matching when multiple candidate images in a candidate data set appear to be visually similar (*e.g.*, successive frames in video). Our method will also fail in matching when both images showing a same basic geometric shape (*e.g.*, an image of sun and an image of ball).

In the future, we plan to develop rotation invariant descriptor which can also grasp global visual similarity in order to solve the problems mentioned above. Furthermore, since each tree can perform matching independently, the matching process can be further accelerated by GPU.

References

Achlioptas, Dimitris (2003). "Database-friendly random projections: Johnson-Lindenstrauss with binary coins". In: *Journal of computer and System Sciences (JCSS)* 66.4, pp. 671–687.

Arthur, David and Sergei Vassilvitskii (2007). "k-means++: The advantages of careful seeding". In: *ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, pp. 1027–1035.

Baraniuk, Richard et al. (2007). "A simple proof of the restricted isometry property for random matrices". In: *Constructive Approximation* 2008.

Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool (2006). "Surf: Speeded up robust features". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 404–417.

Chong, Hamilton Y, Steven J Gortler, and Todd Zickler (2008). "A perception-based color space for illumination-invariant image processing". In: *ACM Transactions on Graphics (TOG)*. Vol. 27. 3. ACM, p. 61.

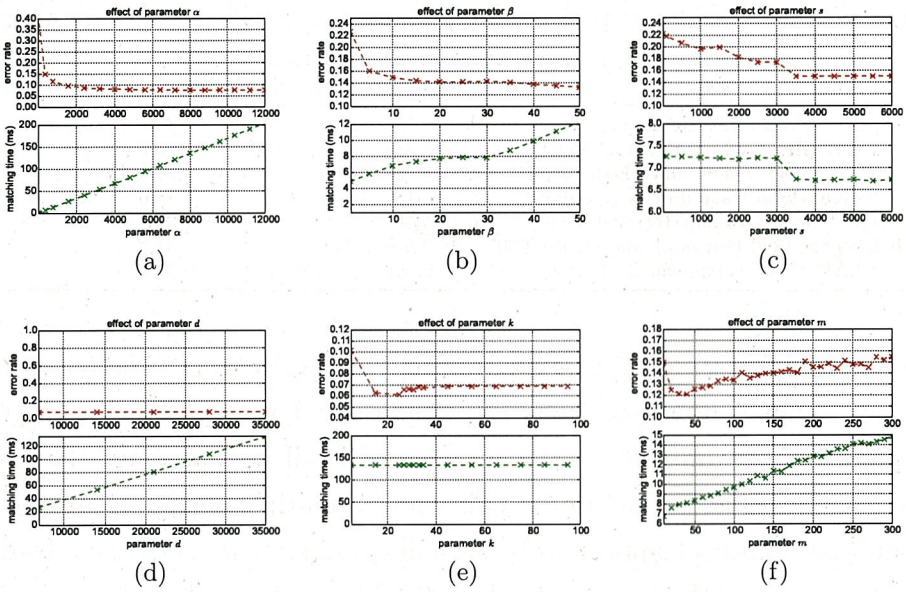


Figure 6.5: Effect of parameters on error rate and processing time per query image. (a) Increasing the number of trees: α. (b) Increasing the number of ANNs of each tree group: β. (c) Increasing the number of compression factor: s. (d) Increasing the size of candidate dataset: d. (e) Increasing the size of blur kernel: k. (f) Increasing the dimension number after compression: m.

Csurka, Gabriella et al. (2004). "Visual categorization with bags of keypoints". In: *European Conference on Computer Vision (ECCV)*. Vol. 1, p. 22.

Dalal, Navneet and Bill Triggs (2005). "Histograms of Oriented Gradients for Human Detection". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Ed. by Cordelia Schmid, Stefano Soatto, and Carlo Tomasi. Vol. 2, pp. 886–893. URL: `http://lear.inrialpes.fr/pubs/2005/DT05`.

Delhumeau, Jonathan et al. (2013). "Revisiting the VLAD image representation". In: *ACM international conference on Multimedia (ACMMM)*. ACM, pp. 653–656.

Eitz, Mathias et al. (2011). "Sketch-based image retrieval: Benchmark and bag-of-features descriptors". In: *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 17.11, pp. 1624–1636.

Friedman, Jerome H, Jon Louis Bentley, and Raphael Ari Finkel (1977). "An algorithm for finding best matches in logarithmic expected time". In: *ACM Transactions On Mathematical Software (TOMS)* 3.3, pp. 209–226.

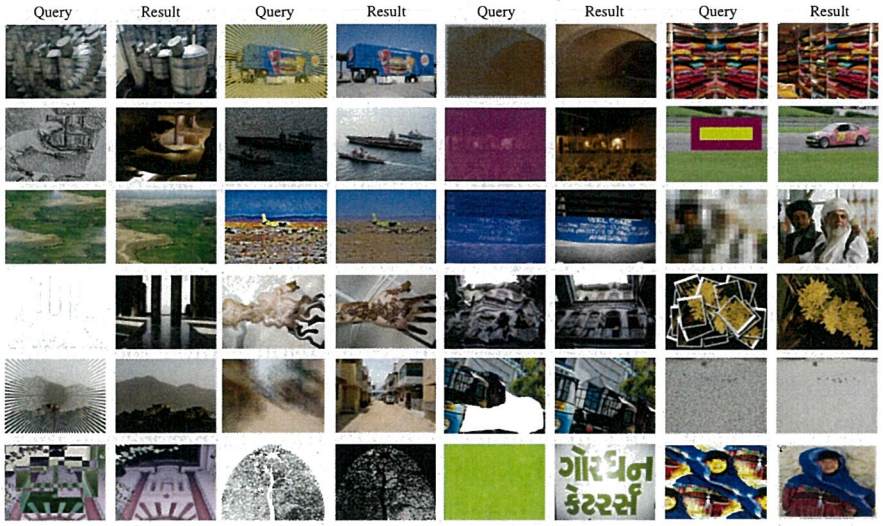


Figure 6.6: Examples of successful ILP matching.

Gionis, Aristides, Piotr Indyk, Rajeev Motwani, et al. (1999). "Similarity search in high dimensions via hashing". In: *International Conference on Very Large Databases (VLDB)*. Vol. 99, pp. 518–529.

Grauman, Kristen and Trevor Darrell (2005). "Efficient image matching with distributions of local invariant features". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2. IEEE, pp. 627–634.

Hauagge, Daniel Cabrini and Noah Snavely (2012). "Image matching using local symmetry features". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 206–213.

Ke, Yan, Rahul Sukthankar, and Larry Huston (2004). "Efficient near-duplicate detection and sub-image retrieval". In: *ACM Multimedia (ACM-MM)*. Vol. 4. 1, p. 5.

Lazebnik, Svetlana, Cordelia Schmid, and Jean Ponce (2006). "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2. IEEE, pp. 2169–2178.

Li, Peng et al. (2012). "Image local invariant features matching using global information". In: *International Conference on Information Science and Technology (ICIST)*. IEEE, pp. 627–633.

Liu, Ce, Jenny Yuen, and Antonio Torralba (2011). "Sift flow: Dense correspondence across scenes and its applications". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 33.5, pp. 978–994.

Lowe, David G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision (IJCV)* 60, pp. 91–110.

Muja, Marius and David G Lowe (2009). "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration." In: *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, pp. 331–340.

Oliva, Aude and Antonio Torralba (2006). "Building the gist of a scene: The role of global image features in recognition". In: *Progress in brain research* 155, pp. 23–36.

Perronnin, Florent and Christopher Dance (2007). "Fisher kernels on visual vocabularies for image categorization". In: *Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1–8.

Robinson, John T (1981). "The KDB-tree: a search structure for large multidimensional dynamic indexes". In: *ACM SIGMOD Conference on Management of Data (SIGMOD)*. ACM, pp. 10–18.

Russell, Bryan C et al. (2011). "Automatic alignment of paintings and photographs depicting a 3d scene". In: *International Conference on Computer Vision (ICCV)*. IEEE, pp. 545–552.

Shrivastava, Abhinav et al. (2011). "Data-driven visual similarity for cross-domain image matching". In: *ACM Transactions on Graphics (TOG)* 30.6, 154:1–154:10. ISSN: 0730-0301. DOI: 10.1145/2070781.2024188. URL: http://doi.acm.org/10.1145/2070781.2024188.

Silpa-Anan, Chanop and Richard Hartley (2008). "Optimised KD-trees for fast image descriptor matching". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1–8.

Torralba, Antonio, Robert Fergus, and Yair Weiss (2008). "Small codes and large image databases for recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1–8.

Vedaldi, Andrea and Brian Fulkerson (2010). "VLFeat: An open and portable library of computer vision algorithms". In: *ACM Multimedia (ACMMM)*. ACM, pp. 1469–1472.

Wu, Pengcheng et al. (2011). "Randomly projected KD-trees with distance metric learning for image retrieval". In: *Advances in Multimedia Modeling*. Springer, pp. 371–382.

Zhang, Chao and Takuya Akashi (2015). "Compressive Image Retrieval with Modified Images". In: *Asian Control Conference (ASCC)*. IEEE, pp. 814–819.

Zhang, Chao, Yo Yamagata, and Takuya Akashi (2015). "Robust Visual Tracking via Coupled Randomness". In: *IEICE Transactions on Information and Systems* 98.5, pp. 1080–1088.

Chapter 7

ROBUST VISUAL TRACKING VIA COUPLED RANDOMNESS

7.1 Summary

Tracking algorithms for arbitrary objects are widely researched in the field of computer vision. At the beginning, an initialized bounding box is given as the input. After that, the algorithms are required to track the objective in the later frames on-the-fly. Tracking-by-detection is one of the main research branches of online tracking. However, there still exist two issues in order to improve the performance. 1) The limited processing time requires the model to extract low-dimensional and discriminative features from the training samples. 2) The model is required to be able to balance both the prior and new objectives' appearance information in order to maintain the relocation ability and avoid the drifting problem. In this chapter, we propose a real-time tracking algorithm called coupled randomness tracking (CRT) which focuses on dealing with these two issues. One randomness represents random projection, and the other randomness represents online random forests (ORFs). In CRT, the gray-scale feature is compressed by a sparse measurement matrix, and ORFs are used to train the sample sequence online. During the training procedure, we introduce a tree discarding strategy which helps the ORFs to adapt fast appearance changes caused by illumination, occlusion, *etc.* Our method can constantly adapt to the objective's latest appearance changes while keeping the prior appearance information. The experimental results show that our algorithm performs robustly with many publicly available benchmark videos and outperforms several state-of-the-art algorithms. Additionally, our algorithm can be easily utilized into a parallel program.

7.2 Introduction

Visual tracking without depth information has become an important research area of computer vision. A typical real-world application is video surveillance Akashi et al., 2007. We have to deal with many problems when tracking one objective with a single camera, such as illumination, occlusion, scale variation, deformation, motion blur, in-plane rotation, out-

of-plane rotation, *etc.* Many of the current tracking methods depend on the training data collected in advance. By comparison with such methods, a tracking task for an arbitrary objective without prior knowledge is more difficult, because the appearance of the objective will be changed due to various conditions during the tracking process. There is a question which may seem contradictory, of whether new information should be incorporated for prediction purposes while the prior information should be saved for relocation. In the past decades, many tracking algorithms have been proposed with better and better performance. In many comprehensive surveys Yilmaz, Javed, and Shah, 2006; Li et al., 2013; Wu, Lim, and Yang, 2013, various object tracking methods have been investigated. We will introduce the state-of-the-art surrounding feature dimension reduction and online learning based on the last decade's chapters.

Feature Dimension Reduction

Before an appearance model is built, feature extraction is usually the first step. No matter whether the feature is global or local, it should be low-dimensional in order to reduce the entire processing time. In recent years, sparse presentation and compressive sensing theories have attracted a lot of theoretical and applied research interest. As one of the various techniques, principal component analysis (PCA) and its variations are widely applied in online tracking. The method in Ross et al., 2008 proposes an online algorithm that incrementally learns and adapts a low dimensional eigenspace representation to reflect the appearance changes of the objective. The method in Kwon and Lee, 2010 proposes a tracking model which can be decomposed into several basic observation models. Each decomposed model can be seen as a feature template that is constructed by sparse principal component analysis (SPCA). All the observation models are combined to cover a specific appearance of the objective. The method in Kwon and Lee, 2011 builds a high-level tracker selecting framework which focuses on the novel point that the trackers should be adapted or constructed depending on the current situation. Among different models, SPCA is used to build the appearance model.

It is also possible to use sparse presentation (SP) to code feature with low dimension. For instance, Jia, Lu, and Yang, 2012 develops a structural local sparse appearance model. Unlike the traditional SP based trackers which only consider the holistic presentation, this chapter addresses the

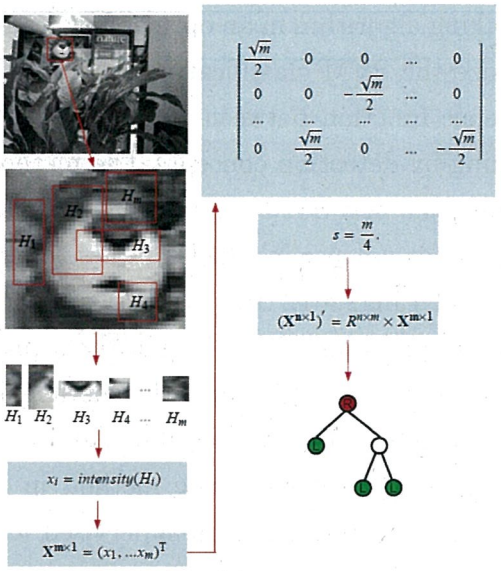


Figure 7.1: Compressing rectangular gray-scale feature from m-dimension to n-dimension.

importance of partial and spatial information. The method in Bao et al., 2012 improves the performance of L1 tracker by adding a ℓ_2 norm regularization on the coefficients associated with the templates. Most of the L1 trackers and their variations model the target appearance by a sparse linear combination of templates. The method in T. Zhang et al., 2012 models the particles as linear combinations of dictionary templates under the particle filter framework. Since each template is updated dynamically, the combination can adapt the latest target's appearance. Since the information of the high-dimensional feature can be preserved based on the compressive sensing theory, compressive sensing can also be used for feature reduction. The method in K. Zhang, L. Zhang, and Yang, 2012 uses a very sparse measurement matrix to compress high-dimensional Haar-like features to a low-dimensional domain.

Online Learning

Online learning has recently become more and more popular due to the successful application of machine learning algorithms in the field of object detection. A tracking-by-detection concept is proposed and many online learning methods are derived from their off-line versions. Most of the online learning methods are based on the support vector machine (SVM) or boosting. For SVM group, the method in Avidan, 2004 integrates the SVM

classifier into a tracking algorithm in an optical flow framework. This chapter tries to maximize the SVM classification score, instead of minimizing an intensity difference function between successive frames. This idea exemplifies the tracking-by-detection concept. The method in Hare, Saffari, and Torr, 2011 develops a new SVM algorithm called kernelized structured output SVM, which does not use labelled samples to update the classifier. The method in Bai and Tang, 2012 treats the tracking problem as a ranking problem which uses the ranking SVM to rank the samples extracted from the next frame. For the boosting group, the method in H. Grabner, M. Grabner, and Bischof, 2006 proposes an online AdaBoost feature selection algorithm for the tracking problem. The method in H. Grabner, Leistner, and Bischof, 2008 introduces a semi-supervised learning scheme into the online boosting classifier. By doing this, update errors caused by each learning sample are limited. The method in Babenko, Yang, and Belongie, 2009 proposes a multiple instance learning method instead of traditional sampling methods. However, relatively few researchers pay attention to solving online tracking problems under the original Random Forests (RFs) framework Breiman, 2001; Saffari et al., 2009.

It is that RFs have an overfitting problem, especially when the data to be trained has large noise information or is structured in high-dimensions. However, it is also worth pointing out that RFs have the advantage of fast convergence. On the other hand, they can be easily implemented and can handle parallel processing with GPGPU naturally, since every tree is independent from the others. These are very potential features for real-time tracking, since the current machine learning research is more and more inseparable from the development of GPU. In order to overcome the shortcomings of the RFs, our method runs online random forests (ORFs) with only 50-dimensional training data and a shallow decision tree structure. By doing this, we can limit the disadvantages of RFs and achieve favorable tracking results. We apply similar ideas from the work K. Zhang, L. Zhang, and Yang, 2012 for feature dimension reduction and introduce a tree discarding strategy into the ORFs framework Saffari et al., 2009. We find the ORFs perform well with compressed features and the whole model becomes more robust by periodically discarding trees.

7.3 Feature Compression

Gray-scale feature and random projection

In this section, we will introduce the basic definitions of the gray-scale feature and random projection. The rectangular gray-scale feature can be defined as the sum of each pixel's gray value inside a rectangle. The rectangle can be any size at any position inside a bounding box (bb) area. The gray-scale feature is a little different from Haar-like feature since it does not need to calculate the difference between multiple rectangles in the feature extraction step. It can be simply calculated by using integral image Viola and Jones, 2001. We define a rectangle as H_i. For each H_i, we extract the gray-scale feature, which is denoted by x_i, where $0 \leq i \leq m$. m is the number of rectangles extracted in one bb. A feature vector $\mathbf{X}$ is defined by combing every element x_i. The left part of Figure 7.1 shows the feature extraction procedure intuitively. Let $R^{m \times n}$ be a very sparse measurement matrix generated by equation 7.1.

$$r_{ij} = \sqrt{s} \times \begin{cases} +1 & \text{with probability } \frac{1}{2s} \\ 0 & \text{with probability } 1 - \frac{1}{s}. \\ -1 & \text{with probability } \frac{1}{2s} \end{cases} \tag{7.1}$$

In this probability distribution, $r_{ij} = R(i,j)$. $R(i,j)$ denotes the entry in row i, column j of matrix R. r_{ij} are all independent from each other. Generating this random matrix is totally independent from the data, with only one parameter s having to be tuned considering the balance between the feature's discriminative ability and computational cost. Not limited to 1 or 3 in real applications, s can be a larger number. It has been proved that the compression can be efficient even when $s = m/4$ in work K. Zhang, L. Zhang, and Yang, 2012. In this chapter, we set $s = m/4$. We found that only $4/m$ of the data needs to be processed during the projection procedure with such a sparse measurement matrix. Since no floating-point arithmetic is needed in addition to a square root operation, the compression process needs little computational cost. Also, this distribution only needs to be calculated once at the first frame and kept fixed until end. To be robust with scale variation in the tracking problem, H_i should be selected in multi-scales. The width of a rectangle W_{H_i} should be in the range $[1, W_{bb}]$ and the height of a rectangle H_{H_i} should be in the range $[1, H_{bb}]$. W_{bb} with H_{bb}

being the width and height of bb. W_{H_i} and H_{H_i} are all integers. Therefore, we can generate $W_{bb} \times H_{bb}$ types of rectangles in a certain bb. Then for every pixel in the bb area ($W_{bb} \times H_{bb}$ pixels in sum), we extract all types of rectangles for feature extraction. The final feature vector $\mathbf{X}$'s dimension is $(W_{bb} \times H_{bb})^2$.

Feature compression

In this section, we will introduce how to compress the gray-scale feature in a given bb. The compression procedure is a projection procedure which can be expressed as Equation 7.2.

$$\mathbf{X}^{'} = R\mathbf{X}. \tag{7.2}$$

$(\cdot)^{'}$ indicates the compression operation. For example, $\mathbf{X}^{'}$ indicates the compressed feature vector. The right part of Figure 7.1 shows the feature compression procedure intuitively. Although R is created with quite large randomness, it is able to preserve the original information stably during tracking. There is a theoretical basis Achlioptas, 2003 which states when the m is suitably high, the distances between the points in a vector space can be preserved with high probability. Our setting satisfies this theoretical basis since m is between 10^6 and 10^{10}. On the other hand, during the projection procedure, the weighted sum or difference between x_i is calculated due to $\sqrt{s}$ and $-\sqrt{s}$ in the distribution.

The projection can also be considered as a procedure to improve the level of the gray-scale feature. The compressed feature vector is very similar to the N-rectangle Haar-like feature. However, they are obviously different because the compressed feature is calculated based on a huge number of rectangles. It is well known that with higher level features, less dimensions are needed to hold the same discriminative ability. From this view point, it is clear why the random projection works for feature dimension reduction with less loss of discriminative ability.

7.4 Tracking by detection

Problem setting

In this section, we will define some symbols to illustrate the tracking problem after we have explained how to calculate $\mathbf{X}^{'}$ in section 7.3. The main purpose of our algorithm is to estimate the objective's position (represented

by bb) continuously which has been specified in the first frame I_0. Sample $\mathbf{X^P}$ is defined by the gray-scale feature of an image patch extracted from a bb area with certain position P. There are three types of samples: positive samples $\mathbf{X_p^P} \in \chi_p$, negative samples $\mathbf{X_n^P} \in \chi_n$, candidate samples $\mathbf{X_c^P} \in \chi_c$. χ_p, χ_n and χ_c are sets composed by according samples. The corresponding compressed feature can be expressed as $(\mathbf{X_p^P})' \in \chi_p'$, $(\mathbf{X_n^P})' \in \chi_n'$, $(\mathbf{X_c^P})' \in \chi_c'$. Then in each frame $I_i, i > 0$, our classifier's purpose can be expressed as $\hat{C} = sign(h(\mathbf{X}))$, where $h : \chi_c' \to \mathbb{R}$. $C \in \{1, -1\}$, in which 1 means positive label, and -1 means negative label. In our algorithm, $h(\mathbf{X})$ is the function to solve the average probability density of positive labels. $1 - h(\mathbf{X})$ is the average probability density of negative labels. When training the classifier with χ_P' and χ_N', every sample's label C has been determined. And lastly, the main purpose of our algorithm can be expressed as Equation 7.3. $\mathbf{X_{I(i)}}$ denotes the tracking result of frame I_i. $\mathbf{X_{I(i)}}$ is selected from χ_c'.

$$\mathbf{X_{I(i)}} = \underset{\mathbf{x} \in \chi_c'}{\arg\max}\, h(\mathbf{X}). \tag{7.3}$$

Preparing samples

In this section, we will introduce how we sample χ_p and χ_n from the previous frame I_{i-1} and sample χ_c from the current frame I_i. With the use of Euclidean distance, we can extract samples with the following equation 7.4.

$$\begin{aligned} \chi_c &= \{\mathbf{X_c} |\ \|P(\mathbf{X_c}) - P(\mathbf{X_{I(i)}})\| < \gamma\}, \\ \chi_p &= \{\mathbf{X_p} |\ \|P(\mathbf{X_p}) - P(\mathbf{X_{I(i-1)}})\| < \alpha\}, \\ \chi_n &= \{\mathbf{X_n} |\ \delta < \|P(\mathbf{X_n}) - P(\mathbf{X_{I(i-1)}})\| < \beta, \alpha < \delta < \beta\}. \end{aligned} \tag{7.4}$$

Distance threshold γ, α, δ and β are all positive real numbers which indirectly determine the number of samples. $P(\mathbf{X})$ is the function to return the 2D position of certain uncompressed sample or compressed sample in the image. This is quite a direct way to extract samples, since we assume that between two continuous frames, only a small amount of displacement of the objective can be observed. Figure 7.2 illustrates how we do sampling during tracking by drawing bb of each sample. These bbs are drawn with the parameters which are used for experimental evaluation. This allows us to visually discover the learning and detection scope of CRT.

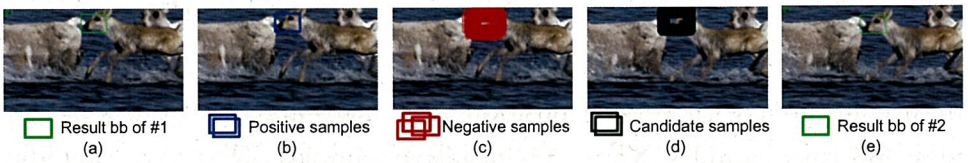


Figure 7.2: Sampling operation between two successive frames. a) Result bb of frame #1 is determined by the previous prediction. b) About 45 positive samples are selected from frame #1 for online training. c) About 80 negative samples are selected from frame #1 for online training. d) About 1900 candidate samples are selected from frame #2 for prediction. e) A candidate sample with highest classification score is determined as frame #2's tracking result.

Online random forests

Sample arrival

We denote ORFs with $o = \{t_1, ..., t_T\}$, whereas T is an integer that indicates the number of trees in the entire forest. In our experiment, T is set to 100. In the non-parallel program, we consider the sample arrives sequentially at ORFs. A sample will be the input of the classifier only if the previous sample has been learned or predicted. Every sample will be trained K times from its arrival, and K is determined by a poisson distribution Poisson(θ_0) referring to work Oza, 2005. In our experiment, θ_0 is set to 1. For those samples which $K = 0$, will not be used for training. The samples which are not included during the training procedure are called out-of-bag (OOB) samples. Considering computational consumption, we do not use these OOB samples to compute the out-of-bag-error (OOBE). The reason is low that a OOBE sometimes does not mean that the objective is being tracked well. Especially after background or obstacles are wrongly learned as the positive feature for a certain time, the OOBE cannot reflect the tracking result's quality.

Training

For the random binary decision trees in ORFs, training is the procedure of splitting each node from top to bottom. We can simply use the attributes of the feature to split the nodes. However, it is time consuming to measure every attribute's quality by entropy or Gini coefficient. Instead of that, we use test functions to split each node Saffari et al., 2009. A random test is defined as a pair $(test(\mathbf{X}'), \sigma)$. σ is a real number threshold, and

it determines whether the according sample should be split into left child node or right child node. When $test(\mathbf{X}') > \sigma$, $\mathbf{X}'$ will fall into the right child node, otherwise it will fall into the left node. Each σ is randomly selected from a numerical range which is determined in advance. This specified range's lower limit is the sum of the minimum value on each feature dimension, and its upper limit is the sum of the max value on each feature dimension. Both limits should be specified in a rational range. If we have a large number of training samples and feature dimensions, the absolute value of the limit will be a relatively larger number. Test function $test(\mathbf{X}') = \mathbf{X}'\mathbf{M}^{\mathrm{T}}$, where each dimension's value of $\mathbf{M}$ is a real number randomly generated between 0 and 1.

For every node in the random trees, we generate a certain number of random tests. We denote a random test set included by a node as $S = \{(test_1(\mathbf{X}'), \sigma_1),$
$..., (test_N(\mathbf{X}'), \sigma_N)...\}$. It is a trade-off relationship between the number of random tests and accuracy performance. For every random test, we use a normalized information gain (IG) to evaluate its quality. The one with the highest normalized IG is selected to split the current node. Node n contains a set of samples. The calculation of IG for random test s in node n can be denoted in Equation 7.5. Split Entropy (SpE), Prior Entropy (PrE), Posterior Entropy (PoE) are calculated respectively in Equation 7.6. PR is the number of positive samples which are assigned to the right child node from the current node. NR is the number of negative samples which are assigned to the right child node from the current node. PL is the number of positive samples which are assigned to the left child node from the current node. NL is the number of negative samples which are assigned to the left child node from the current node. SU is the total number of samples in the current node. Before being split, n must meet other two additional conditions which are proposed as non-recursive strategy Saffari et al., 2009. 1) The number of samples in node n must be larger than θ_1. 2) The value of information gain for the split must be larger than θ_2. 3) The depth of the node n must be smaller than θ_3. After being split, the left child node and right child node will keep the parent node's samples, thus it can be used to calculate the probability density for classification.

$$IG(n, s) = \frac{PrE - PoE}{PrE \times SpE}. \tag{7.5}$$

$$
\begin{aligned}
PR &= |\{\mathbf{X}^{'}|\mathbf{X}^{'} \in \chi_{p}^{'}, \mathbf{X}^{'} \in n, test_{N}(\mathbf{X}^{'}) > \sigma_{N}\}|, \\
NR &= |\{\mathbf{X}^{'}|\mathbf{X}^{'} \in \chi_{n}^{'}, \mathbf{X}^{'} \in n, test_{N}(\mathbf{X}^{'}) > \sigma_{N}\}|, \\
PL &= |\{\mathbf{X}^{'}|\mathbf{X}^{'} \in \chi_{p}^{'}, \mathbf{X}^{'} \in n, test_{N}(\mathbf{X}^{'}) < \sigma_{N}\}|, \\
NL &= |\{\mathbf{X}^{'}|\mathbf{X}^{'} \in \chi_{n}^{'}, \mathbf{X}^{'} \in n, test_{N}(\mathbf{X}^{'}) < \sigma_{N}\}|, \\
SU = |\{\mathbf{X}^{'}|\mathbf{X}^{'} \in n\}|, p_1 &= \frac{(PR+NR)}{SU}, p_2 = \frac{(PL+NL)}{SU}, \\
p_3 = \frac{(PR+PL)}{SU}, p_4 &= \frac{(NR+NL)}{SU}, p_5 = \frac{PR}{(PR+NR)}, \\
p_6 = \frac{NR}{(PR+NR)}, p_7 &= \frac{PL}{(PL+NL)}, p_8 = \frac{PL}{(PL+NL)}, \\
SpE &= -\ln p_1 - \ln p_2, \\
PrE &= -\ln p_3 - \ln p_4, \\
PoE &= (-\ln p_5 - \ln p_6) \times p_1 + (-\ln p_7 - \ln p_8) \times p_2.
\end{aligned}
\tag{7.6}
$$

Discarding

Discarding trees is a very necessary step for tracking. Without discarding, the entire ORFs cannot track the objective adaptively throughout the time. A common method Leistner et al., 2009 is to discard a certain random tree by measuring its OOBE. Specifically, a random tree with a higher OOBE has a higher probability of being discarded. This strategy can surely deal with slow illumination changes or occlusion changes. However, it can hardly deal with drastic illumination changes or occlusion changes. Taking Figure 7.3 as an example, aa the singer is suddenly exposed to light, the OOBE instantaneously becomes large. If we use the strategy mentioned above, most of the trees will be discarded in order to incorporate the new feature caused by shining light. This strategy can temporarily hold the changes, however it will lose the original information of the objective. This will cause the loss of relocation ability.

Our discarding strategy is to discard and retrain half of the trees periodically while the other half of the trees continue to be updated with the initial appearance information of the objective. Improving the performance of Random Forests by discarding trees is a widely used technology. For example, Robnik-Šikonja, 2004 discards trees with negative margin, which is decided by a voting mechanism. The difference is, our strategy discards trees with "time", while Robnik-Šikonja, 2004 discards trees with "voting

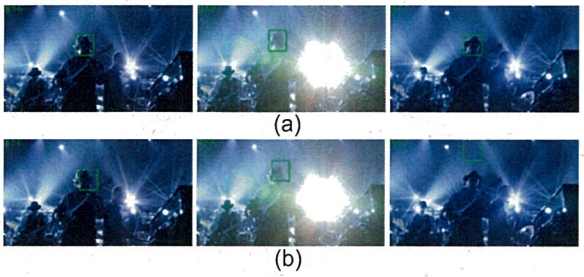


Figure 7.3: Comparison of tracking results on Shaking sequence. (a) ORFs with our discarding strategy. (b) ORFs without discarding strategy.

mechanism". Since we discard the trees according to frequency, we will introduce how the frame rate affects our method. At first, the discarding parameter will not be affected with the frame rate, because this frequency parameter is determined in advance. Secondly, the training procedure will not be affected with the sequence's frame rate, because we train the entire ORFs every frame. Then, the effect of the discarding operation can be affected by the sequence's frame rate. With a lower frame rate sequence, our discarding method can improve the performance of the ORFs accordingly, and with a higher frame rate sequence, although we cannot improve the performance of the ORFs, the performance will not be reduced. Lastly, the tracking performance will be affected by the frame rate in the same way as most of the online tracking algorithms. In the experiment, we discard the first half of the trees every two frames. By doing this, our classifier can deal with intense illumination changes and occlusions while keeping the objective's original information. If we discard the trees every five frames or ten frames, when sudden environment changes occur in the low frame rate sequence, the tracker will lose the target objective more easily. In Figure 7.3, (a) shows the tracking results without this discarding strategy and (b) shows the tracking results with this discarding strategy. In (b), the classifier prevent the bb drifting from the objective when the light dims again.

Predicting

Figure 7.4 illustrates the prediction procedure with a certain bb's compressed feature $\mathbf{X}_\mathbf{c}^{'}$. The arrows in $1st$ and Tth random tree illustrate how a $\mathbf{X}_\mathbf{c}^{'}$ falls from the root node to a certain leaf node. Each arrow is determined by the selected random test on each node. At the leaf node which $\mathbf{X}_\mathbf{c}^{'}$

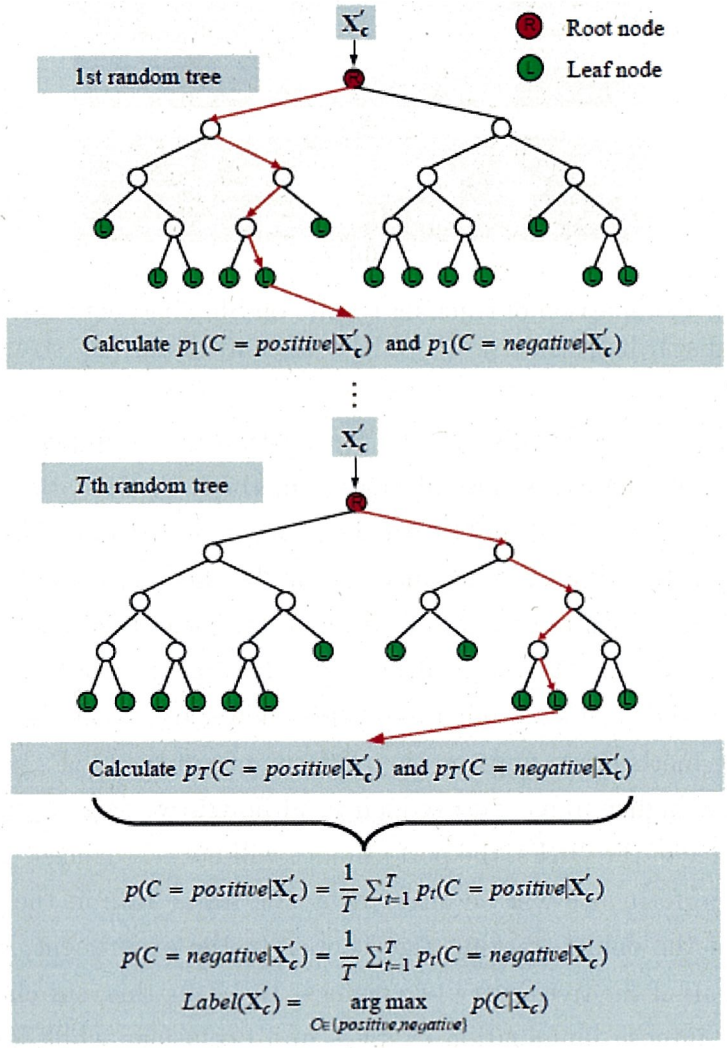


Figure 7.4: Prediction with online random forests.

falls, the probability density of both the positive label and negative label will be calculated. The final average probability density can be calculated by averaging the statistical results of all the trees.

The entire algorithm of CRT is depicted in Algorithm 7.

7.5 Experiments

Evaluation settings

In recent years, many sequences for tracking evaluation have been publicly available. However at the same time, one sequence may have several versions of annotation data which are edited by different people. In order to ensure comparative experiments to be fair and accurate, we selected 20 se-

quences with their original annotation data. Animal, Shaking from Kwon and Lee, 2010, Box from Santner et al., 2010, Kitesurf, Biker, Bolt, Skiing from K. Zhang, L. Zhang, and Yang, 2012, others from Babenko, Yang, and Belongie, 2009, etc. These 20 sequences can reflect the 9 attributes defined in Wu, Lim, and Yang, 2013. We compared our results with the best experimental results reported in K. Zhang, L. Zhang, and Yang, 2012 to avoid tuning other algorithm parameters. The trackers to be compared include the compressive tracker (CT) K. Zhang, L. Zhang, and Yang, 2012, the fragment tracker (Frag) Adam, Rivlin, and Shimshoni, 2006, the on-line AdaBoost tracker (OAB) H. Grabner, M. Grabner, and Bischof, 2006, the semi-supervised tracker (SemiB) H. Grabner, Leistner, and Bischof, 2008, the MILTrack algorithm (MIL) Babenko, Yang, and Belongie, 2009, the ℓ_1-tracker (ℓ_1-T) Mei and Ling, 2011, the TLD tracker (TLD) Kalal, Matas, and Mikolajczyk, 2010, and the Struck algorithm (Struck) Hare, Saffari, and Torr, 2011. Since most of the tracking algorithms run with randomness, the experimental results' accuracy fluctuates within a certain range of accuracy. In order to objectively evaluate the performance of the algorithm, we repeated the experiment 10 times and calculated the average value for each result.

We use the overlap rate as the criteria to judge whether a tracking result is successful or not. The overlap rate is calculated with a tracking result bb (BB_{tr}) and a ground truth BB(BB_{gt}). Specifically, we employ the PASCAL Everingham et al., 2010 measure, which states that the overlap rate between successful BB_{tr} and BB_{gt} should exceed 50%. This widely used criteria is shown in Equation 7.7. Based on this criterion, the success rate is calculated with the total number of bbs and the number of succeeded bbs.

$$\frac{area(BB_{tr}) \cap area(BB_{gt})}{area(BB_{tr}) \cup area(BB_{gt})} > 0.5. \tag{7.7}$$

We also evaluate the center location error (CLE) with ground truth data. First we calculate the sum of the distance between each BB_{tr}'s center and BB_{gt}'s center. Then we compute the average distance based on each sequence's frame number. Note that the CLE will be largely influenced by drifting. If the tracker completely loses the objective at a certain time during the tracking, the CLE will become very large. Although most of the trackers have the ability to relocate, when the detecting bb drifts away from the objective, it is very hard for online tracers to relocate (global

Table 7.1: Sequences used in the experiment.

Sequence	Resolution	Initial BB	#Frames
Animal	704 × 400	300, 5, 100, 70	71
Box	640 × 480	476, 143, 86, 112	1161
Coupon book	320 × 240	142, 62, 62, 98	327
Cliff bar	320 × 240	138, 120, 38, 59	329
David indoor	320 × 240	122, 58, 75, 97	462
Girl	320 × 240	128, 46, 104, 127	502
Occluded face 2	320 × 240	112, 50, 92, 116	812
Sylvester	320 × 240	121, 58, 51, 50	1345
Shaking	624 × 352	225, 135, 60, 70	365
Soccer	640 × 360	302, 135, 66, 80	392
Twinings	320 × 240	126, 165, 73, 53	472
Tiger 1	320 × 240	116, 44, 38, 42	354
Tiger 2	320 × 240	16, 30, 34, 39	365
Panda	312 × 233	58, 100, 27, 22	1014
Jumping	352 × 288	147, 110, 33, 32	313
Kitesurf	480 × 270	201, 40, 30, 32	84
Biker	640 × 360	254, 92, 33, 42	180
Bolt	480 × 270	265, 70, 42, 74	293
Walking	768 × 576	692, 439, 24, 79	412
Skiing	640 × 360	454, 190, 48, 47	67

exploration is needed). Therefore, there is less value in evaluating the CLE when a tracker cannot continuously keep tracking the objective. We do not show the CLE of the TLD tracker during the sequences in which the TLD tracker can easily lose the objective completely.

Experimental results

Combined with low-dimensional feature (compressed by random projection) and tree structure classifier (ORFs), the running efficiency of the CRT is trustworthy. In fact, the CRT runs at an average 18 FPS on an Intel Core-i7 3.4 GHz CPU with a 8 GB RAM. It's slower than the CT but outperforms the other compared trackers in processing time. Table 7.2 shows the SR estimated with 9 trackers. The top 3 results are shown in bold font. The rank of each result is shown in the parentheses. Our tracker CRT achieved the most 1st-ranks among 13 sequences. The CRT got the highest average rank among 9 trackers. Especially with sequences Tiger1 and Tiger2, the CRT outperformed other trackers by 15% to 80% and 30% to 80%. The robustness of the CRT is highlighted in the Tiger1

Figure 7.5: Examples of results comparing with CT on several sequences.

and Tiger2 sequences from many aspects such as fast motion, occlusion, rotation, illumination change, *etc.* In the Shaking sequence, the CRT adapts the drastic illumination change better than other trackers. In the Animal sequence, our tracker can catch up with the fast motion of a deer. Other appearance changes can also be well maintained such as the Girl (in-plane and out-of-plane rotation), the Cliff bar and the Coupon book (background clutters), *etc.* However, in the soccer and box sequences, we obtained an SR below 50%. In both of the video sequences, the CRT easily fails in tracking when heavy prolonged occlusions occur. For heavy prolonged occlusions, our classifier continues learning from the wrong appearance information and changes the probability density in each leaf node. When the objective appears again, the CRT tends to track the obstructions. Table 7.3 shows the CLE achieved by 9 trackers on 13 sequences. Although the CRT did not achieve the best Average CLE Rank, it outperforms many trackers in many sequences with CLE. It is worth pointing out that the CLE of the Soccer and Box sequences is relatively high because the CRE easily fail in these two sequences due to heavy prolonged occlusion and the result bb usually drifts away from the objective. Figure 7.5 shows some examples on different sequences while comparing with CT for clarity. From the results of our experiments, we can see that for practical use, our method tends to perform well on the sequences without prolonged occlusion and out of view frames. For the other sequences, our tracking algorithm can perform satisfactorily.

7.6 Conclusion

In this chapter, we applied random projection to reduce the gray-scale feature's dimension and realize real-time tracking under the online random forests framework. The feature level is upgraded after compression with the dimension reduced. Random projection fits the random forests well by reducing the risk of overfitting. We also discovered that discarding trees periodically effectively solved the problem of balancing the new appearance information of the objective for adaptive tracking and the original appearance information of the objective for relocation. The results of the experiments show that our method performed robustly with many benchmark sequences and outperformed many state-of-the-art trackers. On the other hand, it also showed that in some sequences, the proposed method tends to perform poorly due to prolonged occlusion, as mentioned in Section 4.2. Future plans include trying to overcome this problem by introducing an obstacle detection mechanism. If our tracker can refuse to learn the feature of the obstacles, we believe performance can be further improved.

Table 7.2: Evaluation1: Success Rate(SR). The rank of each result is shown in the parentheses. The top 3 results are shown in bold font.

Sequence	CRT	CT	Struck	MILT	TLD	OAB	ℓ_1-T	SemiB	Frag
Animal	**99(1)**	**76(3)**	**97(2)**	73(5)	75(4)	15(7)	5(8)	47(6)	3(9)
Box	49(4)	**89(2)**	**92(1)**	**65(3)**	**92(1)**	13(7)	5(8)	38(5)	16(6)
Coupon book	**100(1)**	**100(1)**	**99(2)**	**99(2)**	16(6)	**98(3)**	**100(1)**	37(4)	27(5)
Cliff bar	**91(1)**	**89(2)**	**70(3)**	65(5)	67(4)	23(7)	38(6)	65(5)	22(8)
David indoor	**76(3)**	**89(2)**	**98(1)**	68(4)	**98(1)**	31(7)	41(6)	46(5)	8(8)
Girl	**100(1)**	78(4)	**99(2)**	50(8)	57(7)	71(5)	**90(3)**	50(8)	68(6)
Occluded face2	**100(1)**	**100(1)**	78(4)	**99(2)**	46(7)	47(6)	**84(3)**	40(8)	52(5)
Sylvester	78(4)	75(5)	**87(2)**	**80(3)**	**94(1)**	70(4)	46(6)	68(5)	34(7)
Shaking	**97(1)**	**92(2)**	1(9)	**85(3)**	16(7)	40(4)	10(8)	31(5)	28(6)
Soccer	**18(3)**	**78(1)**	14(5)	17(4)	10(7)	8(9)	13(6)	9(8)	**27(2)**
Twinings	**95(2)**	**89(3)**	**98(1)**	72(5)	46(7)	**98(1)**	83(4)	23(8)	69(6)
Tiger 1	**93(1)**	**78(2)**	**73(3)**	39(5)	65(4)	24(7)	13(9)	28(6)	19(8)
Tiger 2	**92(1)**	**60(2)**	22(6)	**45(3)**	41(4)	37(5)	12(9)	17(7)	13(8)
Panda	**93(1)**	**81(2)**	13(8)	**75(3)**	29(7)	69(4)	56(6)	67(5)	7(9)
Jumping	**100(1)**	**100(1)**	18(6)	**99(2)**	**99(2)**	**86(3)**	9(7)	84(4)	36(5)
Kitesurf	**94(1)**	**68(3)**	40(6)	**90(2)**	65(5)	31(7)	31(7)	67(4)	10(8)
Biker	**88(1)**	**75(2)**	35(5)	21(8)	42(4)	42(4)	31(6)	**62(3)**	26(7)
Bolt	**60(3)**	**79(2)**	10(6)	**83(1)**	1(8)	1(8)	2(7)	16(5)	39(4)
Walking	**100(1)**	**89(3)**	**100(1)**	32(6)	55(5)	**86(3)**	**98(2)**	81(4)	32(6)
Skiing	**100(1)**	**70(3)**	**80(2)**	42(6)	59(5)	69(4)	10(7)	69(4)	7(8)
Ave. SR Rank	**1.65**	**2.30**	**3.75**	4.00	4.80	5.25	5.95	5.45	6.55

References

Achlioptas, Dimitris (2003). "Database-friendly random projections: Johnson-Lindenstrauss with binary coins". In: *Journal of computer and System Sciences (JCSS)* 66.4, pp. 671–687.

Table 7.3: Evaluation2: Center Location Error(CLE). The rank of each result is shown in the parentheses. The top 3 results are shown in bold font.

Sequence	CRT	CT	Struck	MILT	TLD	OAB	ℓ_1-T	SemiB	Frag
Animal	**14(1)**	**17(2)**	**17(2)**	32(4)	–(–)	62(5)	155(7)	**25(3)**	99(6)
Box	152(5)	**14(2)**	**11(1)**	**14(2)**	–(–)	**74(3)**	196(7)	119(4)	160(6)
Coupon book	17(5)	**4(1)**	10(4)	**6(2)**	–(–)	**9(3)**	**6(2)**	74(7)	63(6)
Cliff bar	**12(2)**	**7(1)**	**20(3)**	**7(1)**	–(–)	33(4)	35(6)	56(7)	34(5)
David indoor	26(5)	**16(3)**	**9(1)**	19(4)	**12(2)**	57(8)	42(7)	37(6)	73(9)
Girl	**20(3)**	21(4)	**10(1)**	25(6)	–(–)	23(5)	**13(2)**	50(8)	26(7)
Occluded face 2	**16(2)**	**10(1)**	25(4)	**16(2)**	–(–)	36(5)	**19(3)**	39(6)	58(7)
Sylvester	11(4)	**9(2)**	**9(2)**	**10(3)**	**7(1)**	12(5)	42(7)	14(6)	47(8)
Shaking	**10(2)**	**9(1)**	166(7)	**12(3)**	–(–)	22(4)	192(8)	133(5)	134(6)
Soccer	119(6)	**16(1)**	95(4)	**64(3)**	–(–)	96(5)	189(8)	135(7)	**54(2)**
Twinings	**12(3)**	**9(2)**	**7(1)**	14(4)	15(5)	**7(1)**	**10(3)**	70(6)	15(5)
Tiger 1	**9(1)**	**10(2)**	**12(3)**	27(4)	–(–)	42(6)	48(7)	39(5)	39(5)
Tiger 2	**11(1)**	**13(2)**	22(4)	**18(3)**	–(–)	22(4)	57(7)	29(5)	37(6)
Panda	**9(2)**	**6(1)**	67(5)	**9(2)**	20(4)	**10(3)**	**10(3)**	**10(3)**	69(6)
Jumping	**10(3)**	**6(1)**	42(6)	**10(3)**	**8(2)**	11(4)	99(7)	11(4)	29(5)
Kitesurf	**8(2)**	**9(3)**	30(5)	**5(1)**	–(–)	33(6)	51(7)	10(4)	55(8)
Biker	14(4)	**11(1)**	14(4)	38(5)	–(–)	**12(2)**	74(7)	**13(3)**	72(6)
Bolt	73(4)	**8(1)**	157(6)	**8(1)**	–(–)	227(7)	**58(3)**	101(5)	**43(2)**
Walking	**5(3)**	**5(3)**	**2(1)**	8(6)	6(4)	**5(3)**	**4(2)**	7(5)	59(7)
Skiing	**10(1)**	**10(1)**	166(6)	15(4)	–(–)	**12(3)**	189(7)	**11(2)**	134(5)
Ave. CLE Rank	**2.95**	**1.75**	3.50	**3.15**	–(–)	4.30	5.50	5.05	5.85

Adam, Amit, Ehud Rivlin, and Ilan Shimshoni (2006). "Robust fragments-based tracking using the integral histogram". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society Conference on.* Vol. 1. IEEE, pp. 798–805.

Akashi, Takuya et al. (2007). "Using genetic algorithm for eye detection and tracking in video sequence". In: *Journal of Systemics, Cybernetics and Informatics (JSCI)* 5.2, pp. 72–78.

Avidan, Shai (2004). "Support vector tracking". In: *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on* 26.8, pp. 1064–1072.

Babenko, Boris, Ming-Hsuan Yang, and Serge Belongie (2009). "Visual tracking with online multiple instance learning". In: *Computer Vision and Pattern Recognition (CVPR). IEEE Conference on.* IEEE, pp. 983–990.

Bai, Yancheng and Ming Tang (2012). "Robust tracking via weakly supervised ranking svm". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on.* IEEE, pp. 1854–1861.

Bao, Chenglong et al. (2012). "Real time robust l1 tracker using accelerated proximal gradient approach". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on.* IEEE, pp. 1830–1837.

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Everingham, Mark et al. (2010). "The pascal visual object classes (voc) challenge". In: *International journal of computer vision (IJCV)* 88.2, pp. 303–338.

Grabner, Helmut, Michael Grabner, and Horst Bischof (2006). "Real-Time Tracking via On-line Boosting." In: *British Machine Vision Conference (BMVC)*. Vol. 1. 5, p. 6.

Grabner, Helmut, Christian Leistner, and Horst Bischof (2008). "Semi-supervised on-line boosting for robust tracking". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 234–247.

Hare, Sam, Amir Saffari, and Philip HS Torr (2011). "Struck: Structured output tracking with kernels". In: *International Conference on Computer Vision (ICCV), IEEE Conference on*. IEEE, pp. 263–270.

Jia, Xu, Huchuan Lu, and Ming-Hsuan Yang (2012). "Visual tracking via adaptive structural local sparse appearance model". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*. IEEE, pp. 1822–1829.

Kalal, Zdenek, Jiri Matas, and Krystian Mikolajczyk (2010). "Pn learning: Bootstrapping binary classifiers by structural constraints". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*. IEEE, pp. 49–56.

Kwon, Junseok and Kyoung Mu Lee (2010). "Visual tracking decomposition". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*. IEEE, pp. 1269–1276.

– (2011). "Tracking by sampling trackers". In: *International Conference on Computer Vision (ICCV), IEEE Conference on*. IEEE, pp. 1195–1202.

Leistner, Christian et al. (2009). "Semi-supervised random forests". In: *Computer Vision, IEEE Conference on*. IEEE, pp. 506–513.

Li, Xi et al. (2013). "A survey of appearance models in visual object tracking". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 4.4, p. 58.

Mei, Xue and Haibin Ling (2011). "Robust visual tracking and vehicle classification via sparse representation". In: *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on* 33.11, pp. 2259–2272.

Oza, Nikunj C (2005). "Online bagging and boosting". In: *International Conference on Systems, Man, and Cybernetics (SMC)*. Vol. 3. IEEE, pp. 2340–2345.

Robnik-Šikonja, Marko (2004). "Improving random forests". In: *Machine Learning: ECML 2004*. Springer, pp. 359–370.

Ross, David A et al. (2008). "Incremental learning for robust visual tracking". In: *International Journal of Computer Vision (IJCV)* 77.1-3, pp. 125–141.

Saffari, Amir et al. (2009). "On-line random forests". In: *International Conference on Computer Vision (ICCV) Workshops, IEEE Conference on*. IEEE, pp. 1393–1400.

Santner, Jakob et al. (2010). "Prost: Parallel robust online simple tracking". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*. IEEE, pp. 723–730.

Viola, Paul and Michael Jones (2001). "Rapid object detection using a boosted cascade of simple features". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society Conference on*. Vol. 1. IEEE, pp. I–511.

Wu, Yi, Jongwoo Lim, and Ming-Hsuan Yang (2013). "Online object tracking: A benchmark". In: *Computer vision and pattern recognition (CVPR)*. IEEE, pp. 2411–2418.

Yilmaz, Alper, Omar Javed, and Mubarak Shah (2006). "Object tracking: A survey". In: *Acm computing surveys (CSUR)* 38.4, p. 13.

Zhang, Kaihua, Lei Zhang, and Ming-Hsuan Yang (2012). "Real-time Compressive Tracking". In: *European Conference on Computer Vision (ECCV)*. ECCV'12. Florence, Italy: Springer-Verlag, pp. 864–877. ISBN: 978-3-642-33711-6.

Zhang, Tianzhu et al. (2012). "Robust visual tracking via multi-task sparse learning". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*. IEEE, pp. 2042–2049.

Algorithm 7 Coupled Randomness Tracking.

Require: Image Sequence I
Require: Number of trees in ORFs T
Require: Parameter number of Poisson distribution θ_0
Require: Minimum number of samples for split θ_1
Require: Minimum value of information gain for split θ_2
Require: Maximum depth of every random tree θ_3

frame number $i \leftarrow 0$
Draw initial bb
$i \leftarrow i + 1$
while I_i exists and $i > 0$ **do**
  if $i\%2$ *is* 0 **then**
    for m from 1 to $T/2$ **do**
      $discardTree(t_m)$
    end for
  end if
  Do Sampling by Equation 7.4 with I_{i-1} and I_i
  Do Feature compression by Equation 7.2
  for j th decision tree from 1 to T **do**
    for every $\mathbf{X}^{'}$ in $\chi_p^{'}$ and $\chi_n^{'}$ **do**
      $K \leftarrow Poisson(\theta_0)$
      if $K > 0$ **then**
        for k from 1 to K **do**
          $n = findLeaf(\mathbf{X}^{'})$
          $updateNode(n, (\mathbf{X}^{'}, C))$
          if $|n| > \theta_1$ and $\exists s \in S : IG(n, s) > \theta_2$ and $depth(n) < \theta_3$ **then**
            $s_n = \arg\max_{s \in S} IG(n, s)$
            split(n,s_n)
          end if
        end for
      end if
    end for
  end for
  for every $\mathbf{X}^{'}$ in $\chi_c^{'}$ **do**
    Calculate $p(C = -1|\mathbf{X}^{'})$ and $p(C = 1|\mathbf{X}^{'})$
  end for
  $\mathbf{X_{I(i)}} = \arg\max_{\mathbf{X}' \in \chi_c^{'}} p(C = 1|\mathbf{X}^{'})$
  draw bb with $P(\mathbf{X_{I(i)}})$
  $i \leftarrow i + 1$
end while

Chapter 8

CONCLUSION

In this dissertation, many matching tasks are studied following the procedures: 1) feature selection, 2) similarity measure, and 3) search strategy design. These tasks can be mainly categorized as: Parametric (geometric model based) tasks and non-parametric (non-geometric model based) tasks. Parametric methods are very limited in application because a predetermined geometric is needed in order to do the matching. One the other hand, non-parametric methods can be more widely applied and robust against the appearance change. In Chapter 2 and 3, as an example of parametric matching, affine and projective model based template matching tasks are studied respectively. In Chapter 4 and 5, for the tasks that parametric methods cannot be applied, non-parametric template matching methods are studied which do not assume any specific deformation models. In Chapter 6, non-parametric image matching problem with modified query image is studied. In Chapter 7, non-parametric online visual tracking problem is studied.

As the future work, we plan to apply the research results to real-world scenes and contribute to the industry. Although many template matching applications have been developed last decades, we believe that our methods, which are more robust with the object's geometric changes and object's appearance changes, can help improve the existing industry systems toward a more practical level.