HIGH ACCURACY AND LANGUAGE INDEPENDENT DOCUMENT RETRIEVAL WITH A FAST INVARIANT TRANSFORM

Qiong Liu[1], Hironori_Yano[2], Don Kimber[1], Chunyuan Liao[1], Lynn Wilcox[1]

[1]FX Palo Alto Laboratory, 3400 Hillview Ave. Bldg. 4, Palo Alto, CA 94304, USA (s)
[2]Internet Service Department, Fujifilm Corporation, Tokyo, Japan

ABSTRACT

This paper presents a tool and a novel Fast Invariant Transform (FIT) algorithm for language independent e-documents access. The tool enables a person to access an e-document through an informal camera capture of a document hardcopy. It can save people from remembering/exploring numerous directories and file names, or even going through many pages/paragraphs in one document. It can also facilitate people's manipulation of a document or people's interactions through documents. Additionally, the algorithm is useful for binding multimedia data to language independent paper documents. Our document recognition algorithm is inspired by the widely known SIFT descriptor [4] but can be computed much more efficiently for both descriptor construction and search. It also uses much less storage space than the SIFT approach. By testing our algorithm with randomly scaled and rotated document pages, we can achieve a 99.73% page recognition rate on the 2188-page ICME06 proceedings and 99.9% page recognition rate on a 504-page Japanese math book [2].

Index Terms— Document Retrieval, Image Descriptor, SIFT, SURF, Paper Document, Cell Phone Interface.

1. INTRODUCTION

People frequently want to find the original or related e-files of paper documents. With current technology, we have to explore numerous directories and file names to find these documents. If a document has many pages, we also need to go through many pages and paragraphs to find a specific patch location in the document. To overcome this problem, more and more people prefer using proper keywords to find files or specific pages. However, selecting proper and distinctive keywords is not always an easy task. Even if people can select one or several keywords correctly, a simple text input may frequently lead to a large number of text matches that a person cannot easily handle.

In this paper, we present a tool for accessing an e-document (e.g. pdf file) by capturing an image of a document hardcopy. Figure 1 shows two usage scenarios where a user accesses an e-document by capturing a document hardcopy or an image of a 3D object (cookie jar in the example). Beyond the usage for regular desktop systems, this interface is also good for cell phones whose keyboards are too tiny for comfortable typing. Additionally, in a tele-collaboration scenario, when regular collaboration cameras cannot capture a paper hardcopy clearly, the system can automatically retrieve the original document for display or manipulation based on some blurred images.

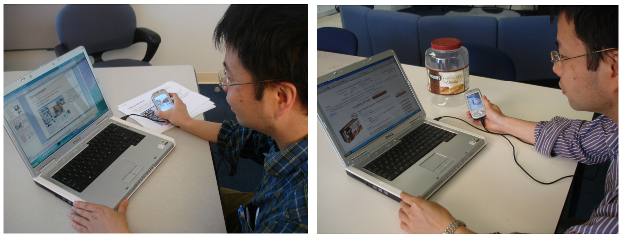

Figure 1. Some usage examples of the proposed system.
Left: get an original file by capturing a hardcopy.
Right: get more information of a 3D object (e.g. cookie jar).

To realize the proposed system, we must address several challenges. First, we need to find robust features to represent a document hardcopy or captured object under various camera setups and capture conditions. Second, the feature search in this algorithm should be fast enough to handle a large number of files. Third, it is better that the algorithm can be language independent so that the system can support the document access functions in different languages. With these challenges in mind, we designed a novel image descriptor that can overcome the language problem. This new descriptor can achieve high accuracy on document recognition. Additionally, it uses much less memory and can be constructed and searched much faster than the well known SIFT feature.

In the following sections, we first talk about early work related to this research. In section 3, we give an overview of our system. In section 4, we describe the image descriptor construction process. In section 5, we report preliminary evaluations of our algorithm.

2. RELATED WORK

There are various ways to find an e-document based on a hardcopy. A typical approach for this is to feed keywords to a text based search engine. Many efficient text-based search algorithms have been developed in various commercial products. By using text, these algorithms can also use language knowledge to assist the search. These services are useful when people can find distinctive keywords from a document hardcopy. When a user cannot find a sufficient number of distinctive keywords or the hardcopy is in a language unknown to a system, it will be hard to use this approach.

Some systems, such as [1], use OCR packages to convert scanned document hardcopies to text and use text for e-document retrieval. These systems are only useful for document hardcopies that have enough distinctive text in a certain language. They also require very high resolution cameras to capture small characters clearly for OCR packages. Paper [3] tries to overcome the OCR package limitation by using word bounding box relations and achieves 60~80% recognition rate. However, it is still limited to documents that have clear word bounding boxes (i.e. western languages). When a document does not have clear bounding boxes (e.g. Chinese and Japanese documents), the approach in [3] cannot work either. Those requirements limit the deployments of those systems.

In [4], Lowe proposed a robust low level image feature set, called SIFT, for general object recognition. This approach works well with a small object dataset. It was also tried on a 50 page document set [12]. However, because the SIFT feature has a very high dimension (128 dimensions) and its feature construction involves Gaussian weighting of gradients over a large area, the memory requirement, feature construction time and searching time cost of using this algorithm will become prohibitive on a large dataset. For example, a 612 by 792 image in our dataset uses more than 1.6MB for saving its features.

In [8], Bay et al. proposed a fast 64 dimension descriptor called SURF. Unlike SIFT, SURF uses integral images to save keypoint finding and orientation assignment time. On the other hand, SURF still needs to access all sampling points in a 20s (s is the scale of a keypoint) by 20s region (minimum 400 points) and it still needs to use a Gaussian window to weight all computed wavelet coefficients. Similarly to SIFT, SURF also needs to verify all points in a sampling region with a keypoint-orientation dependent transform and use all verified points for feature extractions.

Through the project reported in this paper, we explore a method for constructing descriptors with pre-computed pyramid data. We also consider skipping the Gaussian weighting process in the descriptor construction procedure. Moreover, we want to investigate descriptor construction with less image values for computation time saving.

3. SYSTEM OVERVIEW

To make the system work for both camera-equipped cell phones and cameras directly connected to PCs, we separate the software into three modules: mobile-client module, service-proxy module, and file-manipulation module.

The mobile-client module is a client application that is used to collect input images for the system. It can be deployed on a cell phone or a PC connected to a camera to respond to user's image capture requests. The service-proxy module is a web service that analyzes the captured image, extracts low level image features, and searches the original/appearance-similar e-files based on matched features. The file manipulation module is a service application module that has the authority to access those original e-files and related applications. It normally resides on a user's document access machine.

4. ALGORITHM DESIGN

A system for finding correspondences between a camera-captured image and an image converted from original e-document can be separated into three modules: key point detector, descriptor constructer, and correspondence locator. In these three modules, the descriptor's construction complexity and dimensionality have direct and significant impact to the performance of the whole system. The goal of our algorithm is a local image descriptor that has a comparable distinctiveness with state-of-the-art descriptors and significantly reduced computational complexity and dimensionality. Our algorithm, named FIT for Fast Invariant Transform, is inspired by the well known SIFT descriptor and can be constructed and searched much faster than SIFT. In this section, we will briefly describe the SIFT descriptor and explain the difference between our descriptor and the SIFT descriptor.

4.1. The SIFT Descriptor Construction

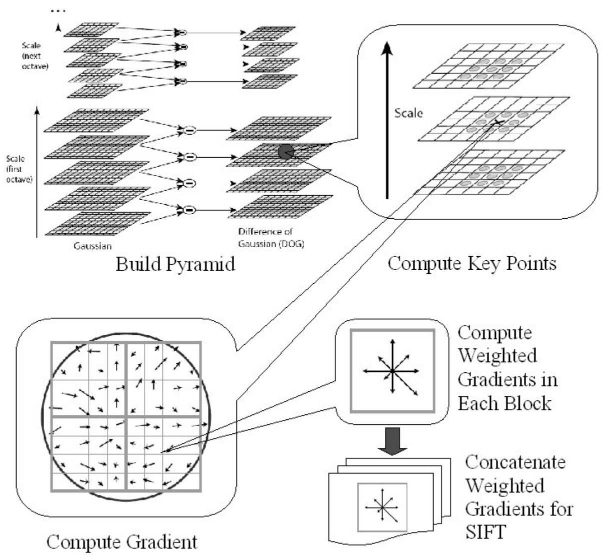
...
Scale (next octave)
Scale (first octave)
Gaussian
Difference of Gaussian (DOG)
Build Pyramid
Scale
Compute Key Points
Compute Gradient
Compute Weighted Gradients in Each Block
Concatenate Weighted Gradients for SIFT

Figure 2. The construction of a SIFT descriptor

The SIFT feature computation can be summarized by the following steps:

1. Gradually Gaussian-blur the input-image to construct a Gaussian-pyramid.
2. Construct the Difference of Gaussian (DOG) pyramid by computing the difference of any two consecutive Gaussian-blurred images in the Gaussian pyramid.
3. Find local maximums and local minimums in the DOG space and use the locations and scales of these maximums and minimums as key-point locations in the DOG space.
4. Compute gradients around each key-point (at least a 16 by 16 region) at the key-point scale and assign an orientation to each key-point based on nearby gradients.
5. Compute 8-direction Gaussian weighted gradients' histograms in 16 sub-blocks (minimum size is 4 by 4).
6. Concatenate the 16 histograms from 16 sub-blocks to form a 128 dimensional vector as a feature descriptor.

Figure 2 illustrates the SIFT feature construction process.

4.2. FIT Descriptor Construction

In the SIFT feature construction steps (steps 4-6), we found interesting issues for further study. The first issue that interests us is the signal sampling rate. More specifically, is the 2D sampling rate of 16 by 16 or higher at a key point level a must for SIFT's distinctiveness? Since SIFT is proven to be distinctive by many object recognition tasks, we guess the aliasing problem can be ignored at this sampling rate. Then the following question is: is the sampling rate too high? If the sampling rate is much higher than the Nyquist sampling rate, we have to pay much more computation cycles than we need for the feature computation.

The second interesting issue for us is the Gaussian weighting and histogram accumulating process. Since the non-linear operation (set negative gradient values to zero) does not increase the information in a descriptor for its distinctiveness, we can ignore that part in our analysis. If we ignore the non-linear process, the Gaussian windowing process will be equivalent to a Gaussian filtering process. On the other hand, filtering key point level data with a Gaussian window is equivalent to operating on data at a larger scale in the spatial-scale space. Fortunately, the Gaussian pyramid can provide us larger scale data of the signal. In other words, if we can use larger scale data properly, we should be able to bypass the expensive Gaussian weighting process.

The third issue that interests us is the histogram computation. The local histograms make the feature robust to feature point localization errors. On the other hand, the feature point localization errors may not be a big problem if we can operate on data at a larger scale with lower frequency signal.

Finally, the SIFT algorithm only uses one Gaussian window for its feature construction, which may limit the scale information usage by a feature. With all these considerations in mind, we design the following procedure for the FIT feature construction:

1-4. Gaussian-pyramid construction, Difference of Gaussian (DOG) pyramid construction and keypoint/orientation search. Currently, we use the same approach as SIFT.

5. Identify descriptor sampling points based on each key point location in the Gaussian pyramid space. We use 5 scale-dependent 3D vectors from a key point to corresponding sampling points to identify sampling points for the key point.
6. Compute 8 scale-dependant gradients at each sampling point. If a gradient is less than 0, the gradient will be set to 0.
7. Concatenate gradients from all sampling points of a key point to form a 40 dimensional vector as a feature descriptor.

Figure 3 illustrates the FIT feature construction process. Based on the comparison between SIFT and FIT computation procedures, we can see that FIT and SIFT have similar steps for DOG pyramid building, key-point localization and orientation assignment. We keep these steps intact for proper comparisons between our FIT approach and the SIFT approach. They may be further improved in our later explorations. Unlike SIFT which accumulates histograms of Gaussian weighted gradients at a key point level, the FIT descriptor directly computes its features at multiple scales higher than the key point scale. This approach can greatly reduce the number of image-pixel-operations involved in feature extraction. Moreover, FIT uses the pre-computed pyramid to save computational cost on the expensive Gaussian weighting process.

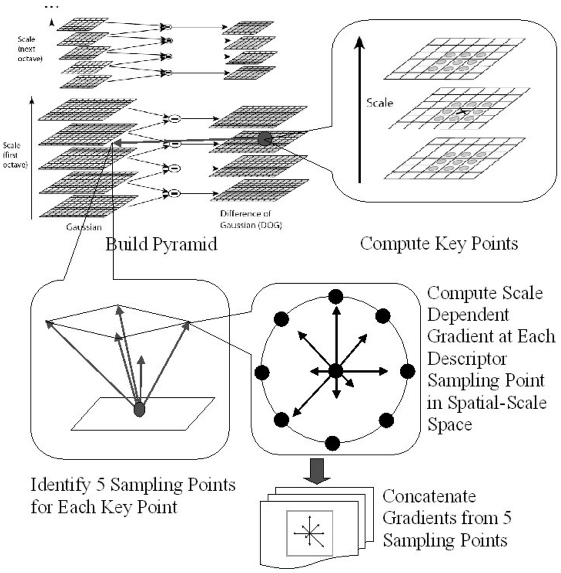
...
Scale (next octave)
Scale (first octave)
Gaussian
Difference of Gaussian (DOG)
Build Pyramid
Scale
Compute Key Points
Compute Scale Dependent Gradient at Each Descriptor Sampling Point in Spatial-Scale Space
Identify 5 Sampling Points for Each Key Point
Concatenate Gradients from 5 Sampling Points

Figure 3. The construction of a FIT descriptor

For describing descriptor details in the spatial-scale space, we need to define a local sub-coordinate system originated at a key point. In the sub-coordinate system, the key point has coordinates (0,0,0), and the *u* direction will align with the key point orientation in spatial domain. By rotating the *u* axis 90 degrees in a counter clockwise direction in the spatial domain centred at the origin, we can get the *v* direction in the spatial domain. The *w* axis corresponding to scale change is perpendicular to the spatial domain and pointing to the scale-increase direction.

The descriptor information is collected at 5 sampling points. We define these sampling points with 3D vectors $\vec{O}_i \quad (i = 0,1,2,3,4)$ from the sub-coordinate origin to sampling point locations and these vectors can be described with the following equation using one variable *d* in spatial domain and one variable *sd* in scale domain:

$$\begin{aligned}
\vec{O}_0 &= [0 \quad 0 \quad 0] \\
\vec{O}_1 &= [d \quad 0 \quad sd] \\
\vec{O}_2 &= [0 \quad d \quad sd] \\
\vec{O}_3 &= [-d \quad 0 \quad sd] \\
\vec{O}_4 &= [0 \quad -d \quad sd]
\end{aligned} \tag{1}$$

Around each sampling point $\vec{O}_i$, we define 8 points $\vec{O}_{ij}$ (j=0, …, 7) on a circle with radius r_i according to the following equation:

$$\vec{O}_{0j} = \vec{O}_0 + \left[r_0 \cdot \cos\left(\frac{2 \cdot \pi \cdot j}{8}\right) \quad r_0 \cdot \sin\left(\frac{2 \cdot \pi \cdot j}{8}\right) \quad 0 \right]$$
$$for \quad i = 0 \tag{2}$$
$$\vec{O}_{ij} = \vec{O}_i + \left[r_i \cdot \cos\left(\frac{2 \cdot \pi \cdot j}{8}\right) \quad r_i \cdot \sin\left(\frac{2 \cdot \pi \cdot j}{8}\right) \quad sd \right]$$
$$for \quad i = (1,2,3,4)$$

If we denote *I([x,y,s])* as a value in a 3D spatial-scale space, where *(x,y)* corresponds to a location in the spatial domain (image domain), *s* corresponds to a Gaussian filter scale in the scale domain, *I* corresponds to the image intensity level at that location, and I_{ij} corresponds to the image intensity difference between sampling point $\vec{O}_i$ point $\vec{O}_{ij}$, we can compute a vector $\vec{V}_i$ for the sampling point *i* with the following equation:

$$I_{ij} = \max\left(I(\vec{O}_i) - I(\vec{O}_{ij}), \quad 0\right)$$
$$V_{ij} = I_{ij} \Big/ \sqrt{\sum_{j=1}^{8} I_{ij}^2} \tag{3}$$
$$\vec{V}_i = [V_{i0}, V_{i1}, V_{i2}, V_{i3}, V_{i4}, V_{i5}, V_{i6}, V_{i7}]$$

By concatenating vectors collected at 5 sampling points, we can get the descriptor vector $\vec{V}$ for a key point by the following equation:

$$\vec{V} = \left[\vec{V}_1, \vec{V}_2, \vec{V}_3, \vec{V}_4, \vec{V}_5\right] \tag{4}$$

In equations 1-4, parameters $d, \quad sd, \quad and \quad r_i$ all depend on the key point scale of a sub-coordinate system. Assume the key point scale is *s*. By setting three constants $dr, \quad sdr, \quad and \quad rr$, we can represent $d, \quad sd, \quad and \quad r$ in terms of *s* using the following equation:

$$d = dr \cdot s$$
$$sd = sdr \cdot s$$
$$r_0 = rr \cdot s \tag{5}$$
$$r_i = r_0 \cdot (1 + sdr) \quad for \quad i = 1,2,3,4$$

Even though the computation of FIT is much simpler, all information preserving operations in SIFT [4] and SURF [8] computations have corresponding operations in the FIT procedure. More specifically, we use operations on existing larger scale signals to substitute the expensive Gaussian weighting process and compute gradients on properly down-sampled signals. From this aspect, we believe that a FIT descriptor can work as efficiently as SIFT and SURF on capturing distinctive information. On the other hand, since FIT gives us more freedom to move sampling points corresponding to a key point in a 3D spatial-scale space, it gives us a good chance to find a more optimal descriptor than the SIFT and SURF descriptors. Additionally, since FIT accesses many less points in the spatial-scale space than SIFT and SURF for feature extraction, the FIT feature composition is expected to be fast. Different from PCA-SIFT [7], which builds a more efficient key point representation with more descriptor construction time, FIT searches for a more efficient key point representation with less descriptor construction time.

5. ALGORITHM EVALUATION

We gave the FIT algorithm a preliminary test on the 2188-page ICME06 proceedings and a 504-page Japanese text book [2]. In the 504-page Japanese document, we removed 8 blank pages. To load more page representations in computer memory, we set the training image size to 306 by 396 pixels, and grey level at each pixel to 8 bits for FIT feature extraction. For our test, we randomly scaled (0.18~2) and rotated (0º~360º) the image of each page to generate 12 test images for each page, and fed features of these test images to an ANN (Approximate Nearest Neighbor) algorithm [5] for fast correspondence search in training features. By using the FIT features, we achieved 99.73% page recognition rate for the ICME06 proceedings and 99.9% page recognition rate for the Japanese text book.

Because the SIFT feature for the 2188-page proceeding is too large for our current computer memory, we compared the FIT feature with our fine tuned SIFT feature on 1000 pages from the ICME06 proceedings. We got 99.9% page recognition rate for FIT and 99.93% recognition rate for SIFT. Even though the SIFT feature has a little higher recognition rate, FIT only uses less than 1/3 of SIFT's storage space. Moreover, with the same ANN settings and an Intel 2.4G CPU, the average search time with FIT descriptor was only 24 ms while the average search time with SIFT descriptor was 220 ms. According to the principles described in [6], this speedup can be even bigger for a larger dataset because of the lower dimensionality of FIT.

Because we did not fully optimize our FIT implementation yet, it is still too early to report a fair speed comparison between the constructions of these two descriptors. Since FIT construction removes many expensive and redundant operations in the SIFT construction, we are sure that the FIT construction is faster than the SIFT construction.

7. REFERENCES

[1] ABBYY, ABBYY FineReader OCR, http link: http://finereader.abbyy.com/?param=137503, June, 2008.

[2] Hal Tasaki, Math, URL: http://www.gakushuin.ac.jp/~881791/mathbook/MB080315.pdf.

[3] B. Erol, E. Antunez, and J. Hull, HOTPAPER: multimedia interaction with paper using mobile phones, Proceeding of ACM MM '08, October 2008.

[4] D.G. Lowe, “Distinctive image features from scale-invariant keypoints”, *International Journal on Computer Vision*, vol. 60, pp. 91-110, 2004.

[5] D.M. Mount, “ANN Programming Manual”, http link: http://www.cs.umd.edu/~mount/ANN/Files/1.1.1/ANNmanual_1.1.1.pdf

[6] S. Arya, D. M. Mount. “Approximate nearest neighbor queries in fixed dimensions”, In Proc. 4th ACM-SIAM Sympos. Discrete Algorithms, pages 271-280, 1993.

[7] Y. Ke1, R. Sukthankar, PCA-SIFT: A More Distinctive Representation for Local Image Descriptors, In Proc. of CVPR 2004.

[8] Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359, 2008.