Word Retrieval in Historical Document using Character-Primitives

Partha Pratim Roy
Laboratoire d'Informatique
Université François Rabelais
Tours, France
partha.roy@univ-tours.fr

Jean-Yves Ramel
Laboratoire d'Informatique
Université François Rabelais
Tours, France
ramel@univ-tours.fr

Nicolas Ragot
Laboratoire d'Informatique
Université François Rabelais
Tours, France
nicolas.ragot@univ-tours.fr

Abstract

Word searching and indexing in historical document collections is a challenging problem because, characters in these documents are often touching or broken due to degradation/ageing effects. For efficient searching in such historical documents, this paper presents a novel approach towards word spotting using string matching of character primitives. We describe the text string as a sequence of primitives which consists of a single character or a part of a character. Primitive segmentation is performed analyzing text background information that is obtained by water reservoir technique. Next, the primitives are clustered using template matching and a codebook of representative primitives is built. Using this primitive codebook, the text information in the document images are encoded and stored. For a query word, we segment it into primitives and encode the word by a string of representative primitives from codebook. Finally, an approximate string matching is applied to find similar words. The matching similarity is used to rank the retrieved words. The proposed method is tested on historical books of French alphabets and we have obtained encouraging results from the experiment.

1. Introduction

With the growing popularity of internet and web-technology, Digital Libraries and archives are becoming interested in mass-digitization and transcription of the collection of historical documents. The purpose is to make the historical documents available to the wider users for accessing the valuable information. The searching/retrieval of content information depends on the transcription of these documents. However, automatic text transcription, performed by the available commercial OCR systems are not satisfactory. OCR provides low performance of transcription in such images due to degradation occurred by ageing, strains, repetitive use, etc.

Word-spotting [5, 9] provides an alternative approach for indexing and retrieval. Word spotting treats each word as a whole entity and thus avoids the difficulty of character segmentation and recognition. It produces a ranked list of word images according to similarity of the query word image. The matching for word spotting is done at image level through word shape coding, generally generated by a set of features at different zones of the word image.

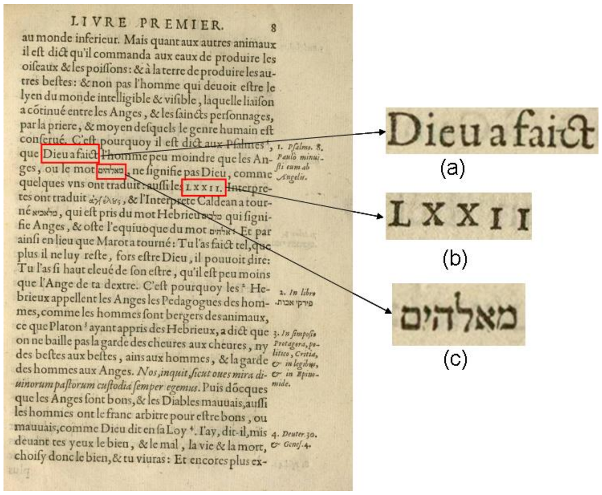
Many existing approaches on word spotting rely on accurate segmentation of words or connected components [6]. Rath and Manmatha [9] used vertical projection profile, upper and lower boundary projection profile and applied dynamic time warping (DTW) distance for matching similar words. To take care the word segmentation problem, Leydier et al. [3] have used differential features and a cohesive elastic distance. Gatos and Pratikakis [1] proposed a segmentation-free word spotting approach for historical printed documents using salient region detection by template matching at an initial stage. Often, word shape coding is used to encode the words in normal printed documents. Lu et al. [4] proposed a technique to retrieve document images by a set of topological codes based on shape features including character ascenders/descenders, holes, water reservoirs information etc.

However, holistic word spotting approaches may not perform well if the words are not segmented correctly due to degradation. Sometimes, a word can be over-segmented due to error. Another difficulty arises from the severely touching or broken characters. Incorrect segmentation of touching characters is still one of the main causes for connected component based approaches. We show an example of document image in Fig.1 to illustrate these problems. Sometimes, the inter-character spacing is not uniform which leads difficulties to word segmentation. Also, it can be noticed that the document contains some words in other languages than English. Thus, the word spotting method needs to be script independent and robust to take care of word segmentation problem.

In this paper, we propose a robust indexing scheme to

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.142

IEEE computer society



LIVRE PREMIER. 8

au monde inferieur. Mais quant aux autres animaux il eſt dict qu'il commanda aux eaux de produire les oiſeaux & les poiſſons : & à la terre de produire les autres beſtes : & non pas l'homme qui deuoit eſtre le lyen du monde intelligible & viſible, laquelle liaiſon a cõtinué entre les Anges, & les ſaincts perſonnages, par la priere, & moyen deſquels le genre humain eſt conſerué. C'eſt pourquoy il eſt dict aux Pſalmes [1], que Dieu a faict l'homme peu moindre que les Anges, ou le mot מאלהים, ne ſignifie pas Dieu, comme quelques vns ont traduit : auſſi les LXXII. Interpretes ont traduit ἀγγέλους, & l'Interprete Caldean a tourné מלאכיא, qui eſt pris du mot Hebrieu מלאך qui ſignifie Anges, & oſte l'equiuoque du mot אלהים. Et par ainſi en lieu que Marot a tourné : Tu l'as faict tel, que plus il ne luy reſte, fors eſtre Dieu, il pouuoit dire : Tu l'as ſi haut eleué de ſon eſtre, qu'il eſt peu moins que l'Ange de ta dextre. C'eſt pourquoy les [2] Hebrieux appellent les Anges les Pedagogues des hommes, comme les hommes ſont bergers des animaux, ce que Platon [3] ayant appris des Hebrieux, a dict que on ne baille pas la garde des cheures aux cheures, ny des beſtes aux beſtes, ains aux hommes, & la garde des hommes aux Anges. *Nos, inquit, ſicut oues mira diuinorum paſtorum cuſtodia ſemper egemus.* Puis dõcques que les Anges ſont bons, & les Diables mauuais, auſſi les hommes ont le franc arbitre pour eſtre bons, ou mauuais, comme Dieu dit en ſa Loy [4]. I'ay, dit-il, mis deuant tes yeux le bien, & le mal, la vie & la mort, choiſy donc le bien, & tu viuras : Et encores plus ex-

[1] *Pſalmo. 8. Paulò minuiſti eum ab Angelis.*

[2] *In libro* פירקי אבות.

[3] *In ſimpoſio Protagora, politico, Critia, & in legibus, & in Epimenide.*

[4] *Deuter. 30. & Geneſ. 4.*

Figure 1. A document from historical collection shows (a) The inter-word spacing is not uniform for word segmentation. (b) Difficult to group characters in a word due of large inter-character spacing. (c) Presence of text words in different scripts.

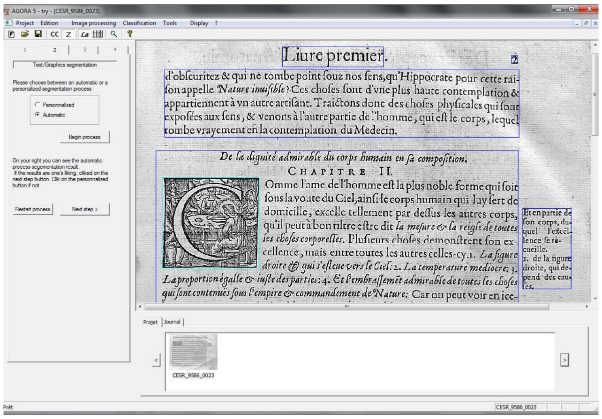
search query word in printed document images. We describe the text string using basic features called primitive. A primitive consists of a single character or a part of a character. Primitive segmentation is performed using background information of text. To handle the background information, water reservoir concept has been used. The primitives are next clustered using template matching and a codebook of primitives is built. Finally, the text information in the book are encoded using primitive codebook. A query word is also represented by a string of primitives. Next, an approximate string matching is applied to each encoded strings in the documents for retrieval of similar words. The matching distance is used to rank the retrieved words.

The proposed methodology is generic and can be applied to different scripts as the method uses dynamic codebook vocabulary for text encoding. The main contributions of this paper are the use of primitive-components instead of whole word and connected components and encoding the text using primitive-components. The advantage of this approach is its robustness to touching and broken characters. Our approach does not need proper segmentation of words. Thus, the system is flexible enough to find location of query words in complex layout.

The rest of the paper is organized as follows. In Section 2, we explain the tool used for pre-processing and text segmentation. In Section 3, we detail the proposed indexing approach. The word retrieval is discussed in Section 4. The experimental results of the approach are presented in Section 5. Finally conclusion is given in Section 6.

2. Preprocessing and Layout Analysis

Historical printed documents are not easy to analyze due to complex layout and degradation effect. In our system, we have used AGORA[8] page layout analysis tool for layout analysis and word segmentation because of its superior performance in historical document analysis. AGORA performs a hybrid content analysis approach combining top-down and bottom-up methods. Next, the system produces a list of homogeneous regions or blocks from the layout using foreground and background information and sends back to the user. This initial representation is used in interactive scenarios analysis. The user builds scenarios according to his needs (location of the dropcaps, notes at margins, etc.) allowing labelling, merging or removing blocks contained in the intermediate representation. Next, these scenarios are applied to the rest of the images in batch processing. Finally, AGORA provides segmentation and labelling of each block (text, picture, dropcaps, etc.) in the document (See Fig.2) and performs segmentation of lines and words.



Text/Graphics segmentation

Please choose between an automatic or a personalized segmentation process

○ Personnalized
● Automatic

Begin process

On your right you can see the automatic process segementation result.
If the results are one's liking, cliked on the next step button. Clik on the personnalized button if not.

Restart process
Next step >

Liure premier. 2

d'obſcuritez & qui ne tombe point ſouz nos ſens, qu'Hippocrate pour cette raiſon appelle *Nature inuiſible* ? Ces choſes ſont d'vne plus haute contemplation & appartiennent à vn autre artiſant. Traictons donc des choſes phyſicales qui ſont expoſées aux ſens, & venons à l'autre partie de l'homme, qui eſt le corps, lequel tombe vrayement en la contemplation du Medecin.

De la dignité admirable du corps humain en ſa compoſition.

CHAPITRE II.

COmme l'ame de l'homme eſt la plus noble forme qui ſoit ſous la voute du Ciel, ainſi le corps humain qui luy ſert de domicille, excelle tellement par deſſus les autres corps, qu'il peut à bon tiltre eſtre dit *la meſure & la reigle de toutes les choſes corporelles.* Pluſieurs choſes demonſtrent ſon excellence, mais entre toutes les autres celles-cy. 1. *La figure droite & qui s'eſleue vers le Ciel:* 2. *La temperature mediocre:* 3. *La proportion égalle & iuſte des parties:* 4. *Et l'embraſſemẽt admirable de toutes les choſes qui ſont contenuës ſous l'empire & commandement de Nature:* Car on peut voir en ice-

Et en partie de ſon corps, duquel l'excellence ſe recueille.
1. de la figure droite, qui depend des cauſes.

Figure 2. Different blocks (marked by rectangle) produced by AGORA Layout analysis.

3 Text Indexing

After pre-processing by AGORA, the text block is segmented into words according to inter-word spacing analysis. The words having larger inter-word space selected by user in scenario analysis are segmented. Due to the non-uniform spacing in historical documents, word segmentation error can not be avoided in the pre-processing stages. Our approach of word spotting can take care this segmentation error even in presence of noise and touching problems. Details of the text indexing method are discussed below.

3.1 Extraction of Primitives from Words

Our word retrieval system is based on the primitives extracted from text characters. To obtain the character primitives, we first apply a connected component analysis to the word image and extract individual components. Each connected components are next segmented into primitive characters according to its background information.

To get the background portion we apply the water reservoir concept. Water reservoir is a metaphor to illustrate the cavity region of a component [7]. The water reservoir principle is as follows. If water is poured from a side of a component, the cavity regions of the background portion of the component where water will be stored are considered as reservoirs of the component. By top (bottom) reservoirs of a component we mean the reservoirs obtained when water is poured from the top (bottom) of the component. Since, touching of neighbor characters affects mostly the upward or downward reservoir rather leftward or rightward reservoirs, we have used only the information of top and bottom reservoirs. The reservoirs having height H ($H > \lambda * S_w$) are considered for segmentation. S_w is the stroke-width of the word and it is computed by analysing statistical mode of object pixels' run lengths in horizontal and vertical directions. λ is chosen according to the experiment dataset. Next, the selected top (bottom) reservoirs are segmented at lowest (highest) reservoir point and component is split into basic primitive shapes.

As our method segments primitives from connected components of the document image, the text characters (e.g. 'i', '$\grave{a}$', '$\acute{e}$', etc.) having multiple components will also be segmented into different primitives. These components are grouped together by checking their overlapping position horizontally. Two components are grouped together, if one is completely overlapped by the other component or they are overlapped partially by overlapping ratio of T_r. T_r is set to 0.8 according to experiment data. Overlapped components are grouped into a single primitive. Finally, with a given set of training pages of a book, we extract all the primitive p_i forming the set $P^n = \{p_1, p_2, \ldots p_n\}$. See Fig.3, where a word is segmented into its primitive characters.

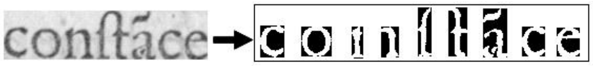


Figure 3. A word image and its corresponding primitives.

3.2 Primitive Codebook

After collecting all the character primitives, we find out the representative primitives of them and build a codebook. The idea is, different character objects can be represented by a small number of shared primitives. The representatives are learnt through an unsupervised clustering algorithm of all primitives involved in training. These representative are defined as centers of clusters $\mu^m = \{\mu_1, \mu_2, \ldots \mu_m\}$, m $\leq$ n. The similarity (S) between two primitives (A and B) is measured by the template matching using cross correlation equation which is given by:

$$S(A,B) = \frac{\sum_{y=0}^{h-1}\sum_{x=0}^{w-1}\tilde{A}(x,y)\tilde{B}(x,y)}{\sqrt{\sum_{y=0}^{h-1}\sum_{x=0}^{w-1}\tilde{A}(x,y)^2\sum_{y=0}^{h-1}\sum_{x=0}^{w-1}\tilde{B}(x,y)^2}}$$

where, $\tilde{A}(x,y) = A(x,y) - \bar{A}$ and $\tilde{B}(x,y) = B(x,y) - \bar{B}$. $\bar{A}$, $\bar{B}$ are the mean of pixel values. h and w are the normalized height and width of image size. The clustering method is performed in an incremental fashion. The codebook is expanded by comparing each new primitive to the existing codebook primitives. When there is a new sample, we find the similarity of each of the existing cluster centres. If the similarity results are less than a pre-defined threshold T_D, we add this primitive as a new cluster centre. Next, a codebook of primitives is built using these cluster centres. The size of codebook depends on the shape variation of primitive components in training set. The process discussed above is detailed in algorithm 1.

Algorithm 1 Creation of Primitive Codebook

```
//create the list of codebook $\mu^m$ from primitives $P_n$
$\mu^m \Leftarrow \{p_1\}$
for all primitives $p_i$ of $p_2, \ldots, p_n$ do
    for all primitives of codebook $\mu_j$ of $\mu^m$ do
        $R_j = S(p_i, \mu_j)$
    end for
    $R_{min} = min(R_1, R_2, \ldots R_m)$
    if $R_{min} < T_D$ then
        $\mu^m \Leftarrow \mu^m \cup p_i$
    end if
end for
```

3.3 Indexing by Primitive Labelling

After the codebook is created from the segmented primitives of the training pages, the words from all pages of that book are indexed using the codebook. To do so, the codebook primitives are first indexed by unique labels $L^m = \{L_1, L_2, \ldots, L_m\}$. Let L_i be the label of codebook primitive μ_i. Next, for each word of the book, the segmented primitives are tagged using the label of most similar codebook primitive.

Let, a word W being segmented into primitive sequence $h_1h_2 \ldots h_l$, where l is the total number of primitives obtained in W. Each primitive h_i is matched by similarity measure (S) to all the codebook primitives μ^m and h_i is labelled by the same of nearest codebook primitive. Finally, W is translated into a sequence of labels $L_{h1}L_{h2} \ldots L_{hl}$, where L_{hi} is the label of h_i. We store the label of the primitive along with its position in the string for indexing purpose. Thus, the 2 dimensional word image has been converted to a 1 dimensional string of codebook labels of L^m.

Due to noise, a primitive may not be represented by a same codebook primitive always. To take care of this problem, during index generation we store more than one nearest codebook primitives for each primitive. To do so, the codebook primitives are ranked in descending order based on the similarity measure and top ‘c’ models are chosen. In our experiment, we have used 3 nearest codebook primitive models. In Fig.4, we have shown the different codebook primitives for each primitive from a touching character. The similarity distances are also marked for each models.

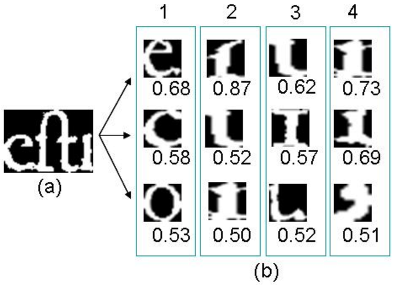


Figure 4. A touching part of a word is segmented into 4 primitives. Each of the primitives are mapped to its 3 nearest primitives in codebook.

4 Retrieval of Query Word

For searching a query word Q from the collection of document images, Q is first segmented into primitives as explained in Section 3.1. Next, we find out most similar codebook models of μ^m for each primitive segments. Q is then encoded into a sequence of labels $L_{q1}L_{q2}\dots L_{qt}$, where $L_{qi} \in L^m$ and t is the number of primitives in Q. To handle noise, ‘c’ nearest codebook models are also stored for each query primitives.

Now, the objective is to find words that have similar sequence of labels. Thus, the matching between query word Q and a target word W is formulated as matching of 2 sequences of primitives. As, primitives are labelled by codebook primitives L^m, the problem is to match primitive label strings of query and target word.

Approximate string matching algorithm [2] has been used in our system for text searching. This method has frequently been used in the literature to refer to a class of pattern matching techniques, by which k errors are allowed between a pattern string S and a text string T. The length of the strings S and T may be different. The algorithm finds all substrings of the text T that have at most k errors (character that are not same) with the pattern S. When $k = 0$ (no mismatches) it is simple string matching algorithm.

In our process, the approximate string matching algorithm is adapted to handle ‘c’ choices of each primitive character. Let, s_i and t_j be the labels of $S[i]$ and $T[j]$, where $s_i = \{L_1^i, L_2^i, \dots, L_c^i\}$ and $t_j = \{L_1^j, L_2^j, \dots, L_c^j\}$. The matching function of s_i and t_j are performed based on following formula.

$$match(s_i, t_j) = \begin{cases} 1 & \texttt{if } |s_i \cap t_j| \geq 1 \\ 0 & \texttt{otherwise} \end{cases}$$

5. Experimental setup and results

The “Centre d’Etude Superieur de la Renaissance” (CESR) of Tours has created a Humanistic Virtual Library with bitmap versions of several books. They have a collection of precious historical books currently numbering around 3000 copies dating from the middle of the XIV century to the beginning of the XVII century and some of them are already scanned or photographed. The languages used, Latin or French, bring an additional factor of variability to the books. An example of scanned page of these books is shown in Fig.1.

The proposed system is tested on a set of documents images taken from this collection. The dataset contains 45 pages of a historical book written mostly in French language. The pages are analyzed by AGORA for preprocessing and different word blocks are segmented. There were total 8675 word blocks, some of them are not segmented correctly due to non-uniform character spacing. We have built the primitive codebook from 24 pages. We found total 57324 primitives from these training pages and we have obtained 183 representative primitives using our clustering algorithm. During primitive matching, the size normalization is done by 20×20. Next, the words of 45 images have been tested for indexing.

An experiment is performed first for retrieval of words from the database according to query word images. A ranking based on string matching distance is done to evaluate the retrieved word. When two or more words are found at same distance, they are discriminated by the accumulated similarity measures of primitives. In Fig.5, we show output of 5 query word images. The query words are shown in 1st row. The results show the robustness of our approach in different kinds of problem. It is reasonable, how noise or missing characters in segmented words affect the ranking process.

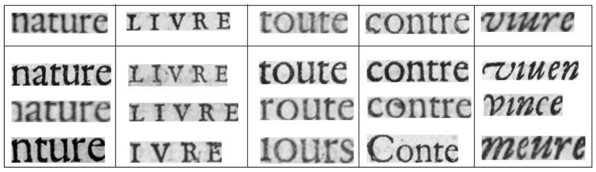
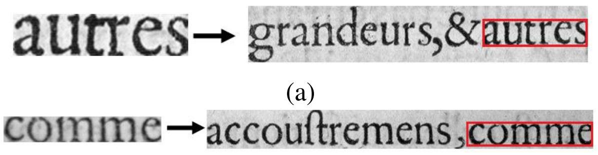


Figure 5. Some query words and their retrieval results arranged column-wise according to the rank.

To check whether a query word is detected correctly or not by our system, we draw a bounding box of the extracted primitive characters in the word block. By viewing the results on the computer's display we check the word detection results. To get the idea of our word detection results, some of the results are shown in Fig.6.



(a)

(b)

Figure 6. Two query images and their location detection in word blocks.

To evaluate the quantitative performance of the system with a query word against a collection of segmented words, we use common ratio of precision (P) and recall (R) for evaluation of word retrieval. Each retrieved word is considered as relevant or not depending on the ground truth of the data. The precision measures the quality of the retrieval system in terms of the ability of the system to include only relevant items in the result. Whereas recall measures the effectiveness of the system in retrieving the relevant items. The result is performed on 20 query word images. In Table 1, we show the performance of the system. We have compared our method using connected component (CC) based approach. In this CC approach, the isolated connected components are used similar to primitives and string matching is done with the codebook of connected component. We have obtained 326 representatives in the codebook using similar clustering technique. From Table 1, we noticed that our primitive based approach outperforms the CC-based approach. One of the reason is that, our method handles touching characters better in such degraded documents. We have also tested the DTW based word spotting approach [9] in these degraded documents. When the words are segmented correctly, the performances are similar. But, when the words are not segmented properly, our method works better.

Table 1. Word Retrieval Results

Approach	Precision	Recall
CC based	70.39%	74.58%
Primitive based	79.46%	81.21%

6. Conclusion

We have presented a robust word spotting system for historical documents. In stead of connected components, we consider character primitives in our approach. The algorithm uses background information to segment characters into primitives. The primitives are then used to encode the word in a string of labels. This algorithm is efficient to locate the words in noisy environment.

The proposed system does not require proper word segmentation as the word matching is mapped to string matching problem. As the codebook vocabulary is based on primitives, efficient indexing is possible than that of connected components or whole word based approach. The searching process is very fast as the matching is performed in string level. The methodology has been made generic to be applied independently of languages. As, it is based on the primitives and the primitives are learnt from training data. The efficiency can be improved by using more precise codebook analysis and context information. This system can also be extended for OCR or transcription purpose.

7. Acknowledgements

This work has been supported by the AAP program of Université François Rabelais, Tours, France (2010-2011).

References

[1] B. Gatos and I. Pratikakis. Segmentation-free word spotting in historical printed documents. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 271–275, 2009.

[2] P. A. V. Hall and G. R. Dowling. Approximate string matching. *ACM Computing Surveys*, 12:381–402, December 1980.

[3] Y. Leydier, A. Oujia, F. LeBourgeois, and H. Emptoz. Towards an omnilingual word retrieval system for ancient manuscripts. *Pattern Recognition*, 42:2089–2105, 2009.

[4] S. Lu, L. Linlin, and C. L. Tan. Document image retrieval through word shape coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1913–1918, 2008.

[5] S. Marinai, E. Marino, and G. Soda. Indexing and retrieval of words in old documents. In *Proceedings of International Conference on Document Analysis and Recognition*, page 223, Washington, DC, USA, 2003.

[6] R. F. Moghaddam and M. Cheriet. Application of multi-level classifiers and clustering for automatic word spotting in historical document images. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 511–515, 2009.

[7] U. Pal, A. Belaid, and C. Choisy. Touching numeral segmentation using water reservoir concept. *Pattern Recognition Letters*, pages 261–272, 2003.

[8] J.-Y. Ramel, S. Leriche, M. L. Demonet, and S. Busson. User-driven page layout analysis of historical printed books. *International Journal on Document Analysis and Recognition*, 9(2-4):243–261, 2007.

[9] T. M. Rath and R. Manmatha. Word image matching using dynamic time warping. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, volume 2, pages 521–527, 2003.