Okapi-Chamfer Matching For Articulate Object Recognition

Hanning Zhou
FX Palo Alto Lab, Inc.
3400 Hillview Ave.
Palo Alto, CA 94304
hanning@fxpal.com

Thomas Huang
University of Illinois
405 North Mathews Ave.
Urbana, IL 61801
huang@ifp.uiuc.edu

Abstract

Recent years have witnessed the rise of many effective text information retrieval systems. By treating local visual features as terms, training images as documents and input images as queries, we formulate the problem of object recognition into that of text retrieval. Our formulation opens up the opportunity to integrate some powerful text retrieval tools with computer vision techniques. In this paper, we propose to improve the efficiency of articulated object recognition by an Okapi-Chamfer matching algorithm. The algorithm is based on the inverted index technique.

The inverted index is a widely used way to effectively organize a collection of text documents. With the inverted index, only documents that contain query terms are accessed and used for matching. To enable inverted indexing in an image database, we build a lexicon of local visual features by clustering the features extracted from the training images. Given a query image, we extract visual features and quantize them based on the lexicon, and then look up the inverted index to identify the subset of training images with non-zero matching score. To evaluate the matching scores in the subset, we combined the modified Okapi weighting formula with the Chamfer distance. The performance of the Okapi-Chamfer matching algorithm is evaluated on a hand posture recognition system. We test the system with both synthesized and real world images. Quantitative results demonstrate the accuracy and efficiency our system.

1. Introduction

Object recognition is an important task for many computer vision applications such as surveillance, human computer interaction and robotics. There are two types of approaches for object recognition. One is 3D-model-based, the other is appearance-based. Appearance-based object recognition typically involves matching an input image with an images database of the object from several characteristic views. According to the matching criterion, the appearance-based approaches can be categorized into two kinds. The first is based on matching salient local features [22, 28, 14, 7, 12, 20]. This kind of approaches detects salient points and uses local invariant feature descriptors to represent the object class. They have been very successful in recognizing rigid objects under partial occlusion and slight pose variation. They usually require the object to have consistent and distinct texture. The second kind of appearance-based approaches rely on global-shape descriptors, such as moment invariants, Fourier descriptor or shape context [4, 27, 2]. They can handle object of little texture, but usually require clean segmentation.

The object classes we are interested in are postures of articulated objects (e.g., the human hand and body). Their deformations have many degrees of freedom. It requires a large image database to cover all the characteristic shapes under different views. Matching a query image with all training images in the database is time-consuming unless one uses a PC cluster [8]. To improve the efficiency, [1] proposes to embed the manifold of hand shapes into a lower-dimensional Euclidean space. [23] proposes to use Parameter Sensitive Hashing. [26] proposes tree-based filtering. Most of these approaches are based on global-shape matching, because there is very little salient texture on the hand, and the traditional local feature based approaches are not directly applicable.

This paper takes a different approach to improve the efficiency in recognizing postures of articulated objects. We formulate the problem of posture recognition as that of text retrieval. A training image is treated as a document; a test image is treated as a query. This formulation enables us to accelerate the database matching with the inverse index. Based on the inverted index, we can identify the subset of training images that share at least one quantized visual feature with the query image at constant computational cost. This technique significantly improves the efficiency of image retrieval, because we only have to match with training images in the subset.

Another effective tool in text retrieval systems is the Okapi weighting formula. It matches a query with a document based on a weighted sum of the terms that appears in both the document and the query. However the Okapi weighting formula is not sufficient for matching images, because it disregards the location of a term in the document. In the case of image retrieval, the image positions of local features are very important. Therefore, we propose to combine the Okapi weighting formula with the Chamfer distance. We assign to each local feature a spacial tag recording its relative image position, and use this tag to calculate the Chamfer distance.

From computer vision point of view, the Okapi-Chamfer matching algorithm integrates local features into a global shape matching algorithm. The local features are extracted from the binary patches along the boundary between foreground and background. The geometry of the boundary is modeled by the spacial tags.

2. Related Works

In the content-based image retrieval community, the idea of indexing low-level features has been proposed. For instance, [15] indexes the image database with a code book for color features; [21] uses the ratio between the two dimensions of an image as the index. However, to our best knowledge, the only image retrieval system that takes advantage of the inverted index technique is [25], but their lexicon is built locally from one individual video sequence, and used in quite a different context.

In terms of combining local feature with global matching, the proposed Okapi-Chamfer algorithm is related to the idea of connecting local invariant features with a deformable geometrical model [5, 16], but our method do not require manually specify the geometric model. Therefore it can accommodate a wide range of deformations of articulated objects.

In terms of quantizing local feature descriptors, our work bares some similarity with [25]. The difference is that [25] uses the original SIFT descriptor [13] to characterize textured regions, while our method simplifies SIFT and use a group of local features to describe the silhouette of lowtexture objects.

The problem of articulated posture estimation (of hand or human body) has been investigated in the context of tracking. One category of tracking algorithms [10, 17] search the configuration space of a 3D kinematic model, and estimate the posture with an analysis-bysynthesis approach. The second category [24, 2], which this paper belongs to, uses the segmented hand region as a query image to retrieve similar images in database of labelled training images.

3. Extracting and Quantizing Visual Features

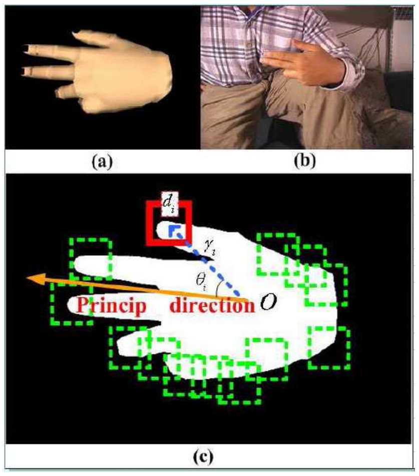

Figure 1. (a) is an example of training image. (b) is an example of testing image. (c) illustrates the feature extraction process.

For a query image of the hand (Figure 3(b)), the basic steps for feature extraction are:

- Segment the hand region based on the skin-color histogram.
- Find discriminative patches along the boundary between the hand region and the background.
- Extract local orientation histogram feature from each patch.
- Quantized the local feature based on a pre-trained lexicon.

For a training image generated with computer graphics (Figure 3(a)), we can get the hand region directly. The next two steps are the same as those for a query image. The lexicon is trained by clustering the local features extracted from all the training images. The details of each step are given in the following subsections.

3.1. Skin Color Histogram based Segmentation

For a query image, we convert it into HSI (hue, saturation and intensity) color space. The HSI image H is mapped to a likelihood ratio image L. The value of pixel (u, v) in L is defined as:

$$L(u,v) = \frac{p(H(u,v)|skin)}{p(H(u,v)|nonskin)} \quad (1)$$

where $H(u, v)$ is the hue and saturation value of pixel (u, v). We train the likelihood $p(H(u, v)|skin)$ and $p(H(u, v)|nonskin))$ by collecting color histogram from manually segmented images of the hand and the background regions. We segment the hand region by thresholding L [9]. In contrast to traditional matching algorithms based on global measurements (e.g., Chamfer distance between two sets of contour points), our matching algorithm collectively matches local features. Therefore it is more robust against mis-segmentation. We can use a low threshold (so that more pixels are segmented into the hand region) which fits the general scenarios.

3.2. Local Orientation Histogram Feature Descriptor

Figure 3(c) illustrates the visual feature extraction. The local features are generated from a binary image of the segmented hand. We do not use salient point detector such as Harris corner detector or difference of Gaussian detector, because there is no reliable texture in a query image. On the binary image, 24×24 subwindows (denoted by $d_i, i = 1 \ldots n$) are selected if at least 20% but no more than 80% of the pixels in the subwindow belongs to the hand region. The thresholds are set to eliminate subwindows that are almost entirely within the hand region or in the background.

We characterize the shape in the subwindow with a local orientation histogram feature descriptor as described in Section 6.1 of [13]. The histogram covers a angle from 0 degree to 360 degrees, with 8 bins. The difference is that we do not subdivide the subwindow into smaller blocks, because cascading features from the subdivided blocks in the same subwindow will make the feature descriptor orientation dependent. The final descriptor for each subwindow is an 8 dimensional vector f_i.

To ensure the descriptor is invariant with respect to the rigid motion of the hand (including in-plane rotation, scaling and 2D translation), we take the following steps (the notations are illustrated in Figure 3):

- Find the centroid O of the hand region.
- Measure the radius γ_i between O and the center of each subwindow d_i.
- Normalize γ_i by dividing it with the medium of all γ_i (i = 1 . . . n).
- Measure the angle θ_i.
- Take the mode among all θ_i (i = 1 . . . n), and define it as the principle angle $\theta_{principle}$.
- calculate relative angle $\theta_i^{rel} = \theta_i - \theta_{principle}$. Since we are mapping from Manifold S_1 to R, we have to handle the problem of sudden jump between 0 degree and 360 degrees[1].

Each 8 dimensional feature vector is assigned a tag of its relative spacial information, denoted by $s_i = (\gamma_i, \theta_i)$. At this stage, an image d can be represented by a list $\{f_i, s_i\}_{i=1 \ldots n}$.

3.3 Building the Lexicon and Quantizing the Image

Given a collection of training images, we can generate a large number[2] of feature vectors denoted by f_i $(i = 1 \ldots D)$. To build a lexicon for the feature vectors, we used the EM algorithm to find $|V|$ clusters, where $|V|$ is the pre-specified lexicon size. We run EM clustering with increasing $|V|$, until the average quantization error is smaller than a threshold. The center c_j $(j = 1 \ldots |V|)$ of each cluster represents a unique term in the lexicon. Figure 3.3 shows some examples of the patches in a cluster.

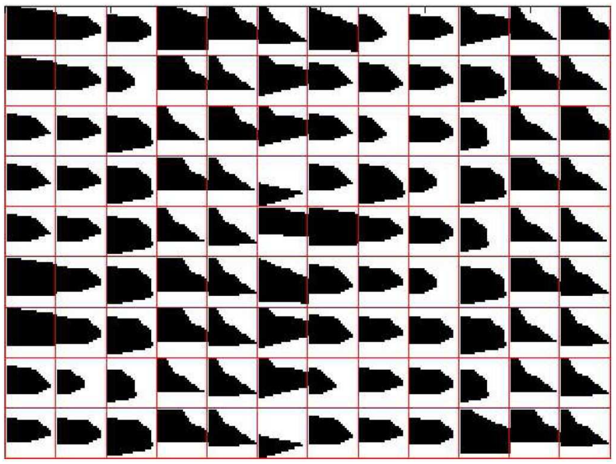

Figure 2. Examples of patches in a particular cluster.

Majority of the patches in this cluster corresponds to the angular shape between the proximal phalanges of two adjacent fingers. Given the lexicon, a feature vector f_i can be quantized into a term

$$t_i = \text{argmin}_{j=1 \ldots |V|} \mathbf{Dist}(f_i, c_j) \tag{2}$$

where **Dist** is Euclidean.

[1]We apply the following mapping to make sure θ_i^{rel} is always positive: $\theta_i^{rel} = \theta_i^{rel} + 360$, if $\theta_i^{rel} < 0$. When measuring distance between two angles θ_i^{rel} and θ_j^{rel} inside $Dist_{Chamfer}$ of Equation (5), we first ensure $\theta_i^{rel} < \theta_j^{rel}$ and then define $dist(\theta_i^{rel}, \theta_j^{rel}) = min\{(\theta_j^{rel} - \theta_i^{rel}), (\theta_i^{rel} - \theta_j^{rel} + 360)\}$

[2]In our case, D = 11647966.

In building the lexicon, we only use the feature descriptor part f_i (but not the spacial tag s_i), so multiple occurrences of the same term with different spacial tags are not distinguished.

With the lexicon, a training image of n local features can be represented by a list $\{t_i, s_i\}_{i=1...n}$. This is similar to a document consists of n terms. The difference is that in most text retrieval systems, the spacial information of one occurrence of one term is not recorded[3], while in our image retrieval system, each term t_i has a 2D spacial tag s_i. The spacial tags are used for evaluating Chamfer distance in Equation (5).

A test image is also quantized with Equation (2).

4. Matching with the Okapi-Chamfer Weighting Formula

Having defined the lexicon in Section 3.3, we can formulate the problem of object recognition as that of text retrieval. In the following subsections, a training image is called a document; an image database is called a collection of documents; a test image is called a query. Both a document and a query are a list of terms. The following subsections give the details on the Okapi- Chamfer matching algorithm.

4.1. Vector Space Model

Assuming the size of the lexicon is $|V|$, a document (or query) can be represented by a vector $\vec{x} = [x_1, \ldots x_|V|]$, where x_i is an importance weight of a term w_i in the document (or query). The vector space of $\vec{x}$ is denoted by Π. Typically $x_i = 0$, if w_i does not appear in the document (or query). Therefore the vector $\vec{x}$ is very sparse.

The similarity score between the query $\vec{q}$ and document $\vec{d}$ can be measured by an inner product $\mathbf{Sim}(\vec{q}; \vec{d}) = \sum_{i=1}^{|V|} q_i d_i$, which can be evaluated on co-occurring terms in both the document and the query, as Equation (3) shows.

$$\mathbf{Sim}(\vec{q}; \vec{d}) = \sum_{w \in \vec{q} \cap \vec{d}} W(w) \tag{3}$$

The function W is a weighting function that is typically defined on the basis of two popular heuristics: term frequency (TF) and inverse document frequency (IDF).

TF $= c(w; \vec{d})$ counts the number of occurrences of a term w in document $\vec{d}$. The intuition is that w is more important if it occurs more frequently in a document/query. IDF $= log\frac{N+1}{df(w)}$, where $df(w)$ is the number of documents that include term w, and N is the total number of documents in the collection. The intuition is w is more discriminative if it only appears in a few documents.

[3]In other words, multiple occurrences of the same term is not distinguished among each other. Only the term frequency within the document matters.

The vector space model is so general that many similarity measurements originally deducted from probabilistic model [18] or language model [11] can be implemented as Equation (3) with different weighting function W. In some sense, one can think of W as a kernel function that transform the vector space Π into a higher dimensional feature space. Therefore the linear function in Equation (3) can approximate various matching functions.

4.2. The Modified Okapi Weighting

This paper uses the modified Okapi weighting, because it satisfies a set of prescribed constraints and performances among the best in extensive text information retrieval experiments [6].

In the original Okapi formula, a negative IDF occurs when the a query term has very high document frequency (e.g., a verbose query[4]). When the IDF part of the original Okapi formula is negative, it will give misleading importance weight, because a higher TF will only push the weight further to the negative side, and this defeats the purpose of the TF heuristic.

In the case of object recognition, the query image is verbose. From an arbitrary input query image, many non-disseminative local features will be extracted and included in the query. Therefore we need to adopt the modification suggested in [6]. The modified version changes the IDF part so that it is always positive, as in the following equation.

$$\begin{aligned} Sim(\vec{q}; \vec{d}) &= \textstyle\sum_{w \in \vec{q} \cap \vec{d}} (log\frac{N+1}{df(w)} \times \frac{(k_3+1) \times c(w,\vec{q})}{k_3 + c(w,\vec{q})} \\ &\times \frac{(k_1+1) \times c(w,\vec{d})}{k_1(1-b+b\frac{|\vec{d}|}{avdl}) + c(w,\vec{d})}) \end{aligned} \tag{4}$$

where $|\vec{d}|$ denotes the length of document $\vec{d}$ and $avdl$ is the average length of documents in the collection. k_1, $k2$ and b are the same parameters as defined in the original Okapi weighting formula [19].

4.3 Okapi-Chamfer Matching

The original Okapi weighting formula considers the content of a document invariant under permutation of terms. However, the image position of the local features will affect the label of an image. Therefore we need to model the spacial distribution of the local features and integrate it to the weighting formula.

To record the spacial information, we use the spacial tag s_i, which records the relative spacial location that term t_i occurs in the document, as defined in Section 3.2.

[4]versus a key-word query, which only include discriminative terms.

To match the spacial tags in the query with those in the document, we use the Chamfer distance, which has proved to be an effective way of matching spacial distribution of silhouette points [1]. It is computationally expensive to compute the Chamfer distance between two point-sets. Fortunately, in our case, we are computing the Chamfer distance between two sets of local features. The number of elements in each set is much smaller.

Combining the modified Okapi with the Chamfer distance, we define the similarity measure as follows.

$$Sim(\vec{q};\vec{d}) = \sum_{w\in\vec{q}\cap\vec{d}}(log\frac{N+1}{df(w)} \times \frac{(k_3+1)\times c(w,\vec{q})}{k_3+c(w,\vec{q})} \times \frac{(k_1+1)\times c(w,\vec{d})}{k_1(1-b+b\frac{|\vec{d}|}{avdl})+c(w,\vec{d})}) + \mu \times \textbf{Dist}_{Chamfer}(S_{\vec{q},w}, S_{\vec{d},w}) \tag{5}$$

where μ is the coefficient that decides how much effect the spacial distribution has upon the matching score. $\textbf{Dist}_{Chamfer}$ denotes the Chamfer distance between two sets of spacial tags. They are defined as follows.

$$\begin{aligned} S_{\vec{q},w} &= \{s_i \in \vec{q} | t_i = w\} \\ S_{\vec{d},w} &= \{s_i \in \vec{d} | t_i = w\} \end{aligned} \tag{6}$$

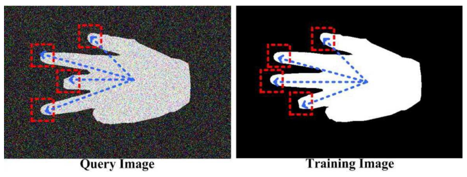

Figure 3. An illustration of Chamfer distance between matched terms in a query image and a training image. The red boxes indicates the locations where the matched term w appears in the two images.

The pair-wise distance between individual spacial tags is Euclidean. Figure 4.3 illustrates the Chamfer distance between a query image and a training image according to a particular term w. In Figure 4.3 the matched term w happens to represent the arch pattern of the finger tips. In practice, w can be any matched shape pattern.

4.4 Fast Matching with the Inverted Index

Formulating the image retrieval problem as that of text retrieval based on vector space model, we are ready to use a powerful tool in the area of text retrieval: the inverted index.

An inverted index includes two components: a lexicon of distinct terms and for each term, a list of documents that contain the term. Consider the following two documents:

$D1$: This is an interesting paper.

$D2$: That is a boring paper.

The inverted index for these two documents would be:

$$\begin{aligned} This &\rightarrow \{D1\} \\ is &\rightarrow \{D1, D2\} \\ a &\rightarrow \{D1\} \\ interesting &\rightarrow \{D1\} \\ paper &\rightarrow \{D1, D2\} \\ That &\rightarrow \{D2\} \\ an &\rightarrow \{D2\} \\ boring &\rightarrow \{D2\} \end{aligned}$$

With the inverted index, only documents that contain at least one query term are accessed and matched with the query, in order to retrieve the matching results.

In a collection of N documents, the computational cost of matching with inverted index is $O(B|V_q|)$, where B is the average number of training documents in which a query term appears. $|V_q|$ is the vocabulary size of a query. In comparison, the cost of matching with every documents in the collection is $O(N|V_q|)$.

To take advantage of the inverted index, B should be as small as possible. One typical way to reduce B is to eliminate the stop words from the lexicon. Stop words are terms that appears in many documents. According to IDF heuristic, stop words are not very discriminative and will give low weight in the Okapi formula. For details, please refer to [3].

By organizing the database of training images with the inverted index and eliminating stop words from the lexicon of visual features, we accelerate the speed of processing one query by a factor of 82.2 on average.

5 Experimental Results

To evaluate the performance of the Okapi-Chamfer matching algorithm, we test it on a database of 16384 images of hand at different postures. We use a 3D kinematic mesh model to generate 1024 hand shapes. Each shape is rendered from 16 different view angles. We follow a similar procedure as [8] in choosing the 1024 hand shapes and the 16 view angles.

Assuming all three flexing joint angles in one finger are linearly related and the abduction angles are fixed, we parameterize the finger configuration with $\vec{\theta} = [\theta_{thumb}\ \theta_{index}\ \theta_{middle}\ \theta_{ring}\ \theta_{pinky}]$. Each component of $\vec{\theta}$ determines the three flexing joint angles of one finger. We take four different values for each of the five components, which are 8, 26, 48 and 75 degrees.

The 16 view angles are the combinations of 4 discrete rotations along X axis and 4 discrete rotations along Y axis. The rotation angles are 0, 20, 40 and 60. We do not need to generate multiple images of the same hand shape under different in-plane rotations (i.e. rotation along Z axis), because the local visual feature is invariant to in-plane rotations.

During testing, we used two kinds of query images. One kind of query images are generated with a 3D kinematic hand model from several view points. The rotation angles along X axis and Y axis range from 0 to 60 degrees. The angle along Z axis range from 0 to 360 degrees. We also added Gaussian noise the synthesized query images to imitate the segmentation error due to sensor noise. We tested only 500 synthesized query images and their finger configurations are randomly sampled.

The second kind is real-world images. The real hand geometry is slightly different from the 3D computer graphics model used to generate the image database. The finger con- figurations are manually labeled.

Figure 5 shows some samples of retrieval results of synthesized query images without in-plane rotation. The left most column is the query image and the other columns

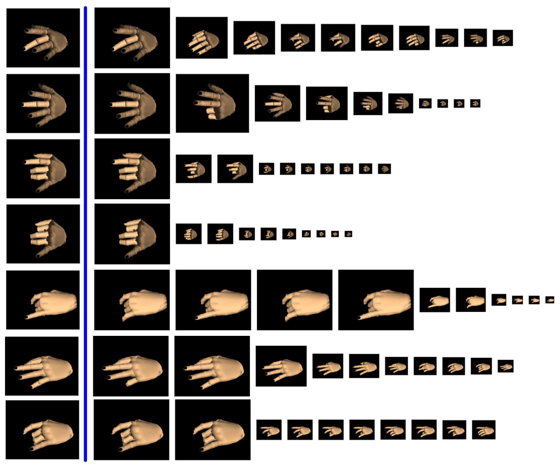

Figure 4. Some samples of retrieval results of synthesized query images without in-plane rotation. The left most column is the query image and the other columns are the retrieval results. The size of the retrieved images are proportional to their matching scores.

Figure 5 shows some samples of retrieval results of synthesized query images with in-plane rotations. This demonstrates that our local feature is invariant to in-plane rotations.

Figure 5 shows some samples of retrieval results of real-world query images. This demonstrates that our matching algorithm is robust against clutter background and variation in the geometry of the hand.

The goal of our recognition system is to recover the finger configurations (i.e. the joint angles) from an input im-

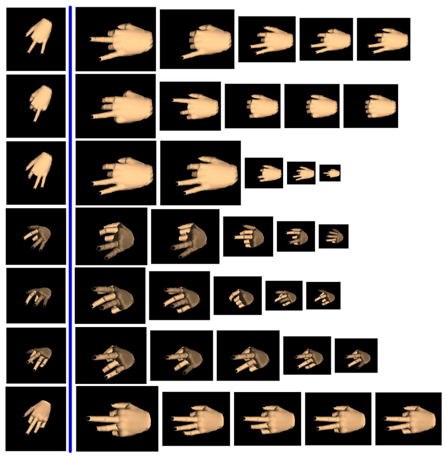

Figure 5. Some samples of retrieval results of synthesized query images with in-plane rotation. are the retrieval results ordered according to their matching score. The size of the retrieved images are proportional to their matching scores.

age. We take the label of rank one retrieval result as the estimation result for a query image. In the quantitative evaluations, instead of showing a precision-recall curve, we measure the root mean square error between the estimated parameter $\hat{\vec{\theta}}$ and the ground truth parameter $\vec{\theta}$ which is defined as $\sqrt{E\{\|\hat{\vec{\theta}} - \vec{\theta}\|^2\}}$, where $\|\cdot\|$ is ℓ^2-norm.

In Figure 5, the vertical axis is the root mean square error (RMSE). The horizontal axis is the variance of Gaussian noises that are added to the synthesized input image. The figure demonstrates that using the modified Okapi or the Chamfer distance alone will give larger errors. With a proper coefficient μ the Okapi-Chamfer matching algorithm (the blue curve) gives much smaller errors. When μ is set too large, using the Okapi-Chamfer matching (the green curve) will be almost equivalent to using the Chamfer distance alone (the red curve). The recognition error when using the modified Okapi alone is not very sensitive to image noise. This is because the modified Okapi disregards the position of the local features. The image noise only affects the TF part of the weighting formula, but will not affect the IDF part of the weighting, as long as at least one of the previously detected local feature survives the noise.

The four curves are different only in terms of matching algorithm. The same inverted index is used for all four curves.

The speed of our posture recognition system is around 3 seconds/query on a Pentium III 1GHz PC in a database of 16384 training images. Although the current Matlab im-

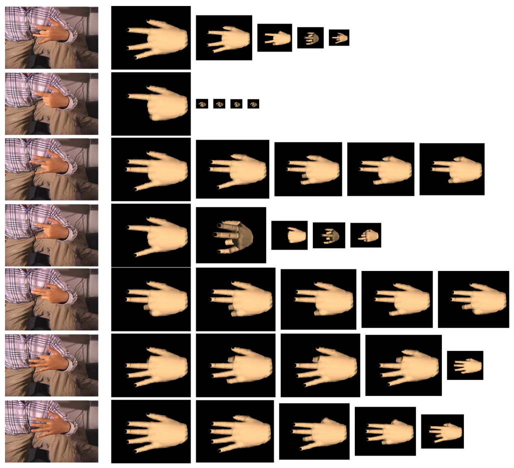

Figure 6. Some samples of retrieval results of real-world query images.

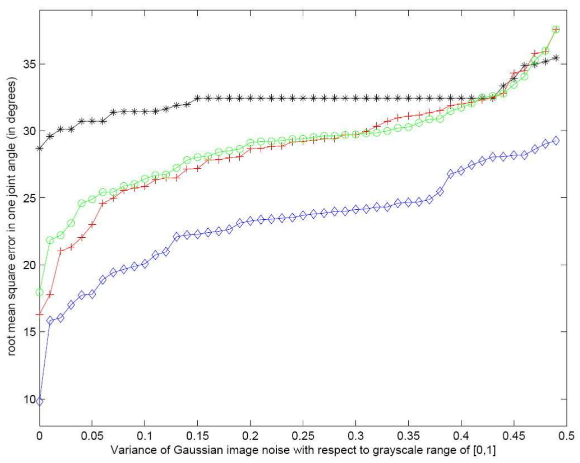

Figure 7. RMSE of parameter θ. The black curve is using modified Okapi only (i.e. $\mu = 0$ in Equation (4)). The red curve is using Chamfer distance only. The green curve is using Okapi-Chamfer matching with $\mu = 15$. The blue curve is using Okapi-Chamfer matching with $\mu = 5$.

plementation is meant to validate the framework and is not optimized for speed, its speed is comparable to the state of the art. [8] reported 3.7 seconds/query and [1] reported 2.3 seconds/query. Both are in a comparable setup as that of our experiments. After optimizing the implementation and porting the code to C/C++, we can reach the goal of realtime posture recognition.

6. Conclusions

The main contributions of our work are:

- Formulated the problem of object recognition as that of text retrieval.
- Introduced the inverted index technique to organize an image database.
- Proposed and implemented a matching algorithm that combines the modified-Okapi weighting formula with the Chamfer distance.

The current framework of posture recognition is briefly summarized into the following steps:

- Extract and quantize local features from a query image.
- For each local feature in the test image, find a subset of training images that contains at least one local feature in the query.
- Within the subset, match the Chamfer distance between the two sets of spacial tags that are associated with a mutual local feature.
- Combine the Chamfer distance with the modified Okapi formula to get the weight of current local feature.
- Sum up the weights of all local features as the final matching score.

There are many possible improvements that we are still investigating. For example, in Step 3, we could obtain more accurate matching score by incorporating the original visual feature into the Chamfer distance. This can reduce the effect of quantization error in building the lexicon. On the other hand, we chose linear combination of Okapi weighting and Chamfer distance, because it is the most simple form and involves only one coefficient. However other ways of combining the two measurements might further improve the accuracy. More specifically, we can put Chamfer distance inside the Okapi weight formula by discounting the TF part according to the Chamfer distance.

Since labeling real-world images is time consuming and error prone, our tests on real-world query images are not very extensive. We will evaluate our method on large annotated databases such as the American Sign Language database [1].

7 Acknowledgments

The authors would like to thank Dr. Chengxiang Zhai for proofreading the manuscript and inspiring discussions. We also thank Dennis Lin for his help in rendering the 3D mesh model of the hand. We thank the anonymous reviewers for their valuable suggestions. This work is supported in part by NSF Grant IIS 01-38965 and NSF Alliance Program.

References

[1] V. Athitsos, J. Alon, G. Kollios, and S. Sclaroff. Boostmap: A method for efficient approximate similarity rankings. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume II, pages 268–275, Washington DC, July 2004.

[2] V. Athitsos and S. Sclaroff. Estimating 3D hand pose from a cluttered image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Madison, Wisconsin, June 2003.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, London, 1st edition, 1999.

[4] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. Technical Report UCB//CSD00-1128, UC Berkeley, January 2001.

[5] M. C. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Santa Barbara, CA, June 23-25 1998.

[6] H. Fang, T. Tao, and C. Zhai. A formal study of information retrieval heuristics. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 40–54, South Yorkshire, UK, July 2004. Springer-Verlag, New York, Inc.

[7] V. Ferrari, T. Tuytelaars, and L. V. Gool. Simultaneous object recognition and segmentation by image exploration. In *Proc. European Conference on Computer Vision*, volume I, pages 40–54, Prague, Czech Republic, May 11-14 2004.

[8] A. Imai, N. Shimada, and Y. Shirai. 3D hand posture recognition by training contour variantion. In *Proc. IEEE International Conference on Automatic Face and Gesture Recognition*, pages 895 – 900, Korea, May 17-19 2004.

[9] M. Jones and J. Rehg. Statistical color models with application to skin detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume I, pages 274–280, Fort Collins, CO, June 23-25 1999.

[10] J. J. Kuch and T. S. Huang. Vision-based hand modeling and tracking for virtual teleconferencing and telecollaboration. In *Proc. of IEEE International Conf. on Computer Vision*, pages 666–671, Cambridge, MA, June 1995.

[11] J. Lafferty and C. Zhai. Probabilistic relevance models based on document and query generation. In W. B. Croft and J. Lafferty, editors, *Language Modeling and Information Retrieval*. Kluwer Academic, Amsterdam, 2003.

[12] S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *Proc. British Machine Vision Conference*, volume 2, pages 959–968, September 2004.

[13] D. Lowe. Object recognition from local scale-invariant features. In *Proc. IEEE International Conference on Computer Vision*, pages 1150–1157, Corfu, Greece, September 1999.

[14] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[15] A. Mojsilovic, J. Kovacevic, J. Hu, and R. Safranek. Matching and retrieval based on the vocabulary and grammar of color patterns. *IEEE Transactions on Image Processing*, 9:38–54, 2000.

[16] D. Ramanan and D. A. Forsyth. Using temporal coherence to build models of animals. In *Proc. IEEE International Conference on Computer Vision*, pages 338–345, Nice, France, Oct. 2003.

[17] J. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *Proc. of IEEE International Conf. Computer Vision*, pages 612–617, 1995.

[18] S. Robertson, C. V. Rijsbergen, and M. Porter. Probabilistic models of indexing and searching. In R. N. Oddy, editor, *Information Retrieval Research*. Butterworth-Heinemann, Newton, MA, 1981.

[19] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 232–241. Springer-Verlag, New York, Inc., 1994.

[20] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. Segmenting, modeling, and matching video clips containing multiple moving objects. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 914–921, Washington DC, July 2004.

[21] R. Schettini, I. Gagliardi, and G. Ciocca. Quick look system. http://www.ivl.disco.unimib.it/Activities/Quicklook.html, Oct. 2004.

[22] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:530–535, May 1997.

[23] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter sensitive hashing. In *Proc. IEEE International Conference on Computer Vision*, volume 2, pages 750 – 757, Nice, France, October 2003.

[24] N. Shimada, K. Kimura, and Y. Shirai. Real-time 3-D hand posture estimation based on 2-D appearance retrieval using monocular camera. In *Proc. of Intl. Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Realtime Systems*, pages 23–30, Vancouver, Canada, July 2001.

[25] J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume I, pages 488–495, Washington DC, 2004.

[26] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla. Filtering using a tree-based estimator. In *Proc. IEEE International Conference on Computer Vision*, volume 2, pages 1063 – 1070, Nice, France, October 2003.

[27] A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla. Shape context and Chamfer matching in cluttered scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume I, pages 127–133, Madison, WI, June 2003.

[28] T. Tuytelaars and L. J. V. Gool. Content-based image retrieval based on local affinely invariant regions. In *Proceedings of the Third International Conference on Visual Information and Information Systems*, pages 493–500, Washington DC, 1999. Springer-Verlag.