Robust Image Retrieval using Multiview Scalable Vocabulary Trees

David Chen[a], Sam S. Tsai[a], Vijay Chandrasekhar[a],
Gabriel Takacs[a], Jatinder Singh[b], Bernd Girod[a] *

[a]Stanford University, Stanford, California, USA
[b]Deutsche Telekom Laboratories, Los Altos, California, USA

ABSTRACT

Content-based image retrieval using a Scalable Vocabulary Tree (SVT) built from local scale-invariant features is an effective method of fast search through a database. An SVT built from fronto-parallel database images, however, is ineffective at classifying query images that suffer from perspective distortion. In this paper, we propose an efficient server-side extension of the single-view SVT to a set of multiview SVTs that may be simultaneously employed for image classification. Our solution results in significantly better retrieval performance when perspective distortion is present. We also develop an analysis of how perspective increases the distance between matching query-database feature descriptors.

Keywords: image retrieval, feature matching, scalable vocabulary tree, perspective distortion

1. INTRODUCTION

One successful approach to content-based image retrieval in recent years relies on robust local features. The Scale-Invariant Feature Transform (SIFT) extracts features from an image that can survive moderate geometric and illumination distortions.[1] An alternative to SIFT, called Speeded-Up Robust Features (SURF), offers similar image retrieval performance.[2] To avoid an expensive linear search through the database, local features are typically organized into a Scalable Vocabulary Tree (SVT).[3] An SVT enables a fast preliminary search for a small subset of the closest matching database images. Then, pairwise comparison of the query image with database images in this subset is computationally practical.

Unfortunately, SVT-based retrieval performs poorly if objects of interest are imaged under severe perspective. First, many keypoints identified by a local feature detector in a fronto-parallel view cannot be repeatably identified in a perspective view. Second, circular regions around keypoints in a fronto-parallel view deform into elliptical regions in a perspective view, causing the feature descriptors to change significantly.

Many previous solutions focus on *perspective rectification* on the query side. Douxchamps and Macq search for projective parameters that minimize sum-of-absolute differences between two stereo images.[4] Their exhaustive search is not designed for real-time image retrieval. Koser and Koch segment out planar patches in the query image, re-project the patches, and extract SIFT features.[5] Their approach requires accurate depth estimation, which is a nontrivial task. If object surfaces in the image can be approximated by planar patches, the affine-invariant feature detector proposed by Mikolajczyk and Schmid can perform well in spite of large viewpoint changes, albeit at much higher computational cost than the SIFT or SURF detector.[6]

The complexity of perspective rectification is relatively high for the query side, especially for our application, where the same mobile camera that captures the query photo might also perform feature extraction. Thus, a better solution is to perform *perspective matching* on the server, as effectively demonstrated by Lepetit et al.[7,8] Database images in fronto-parallel view are projectively transformed to mimic different types of perspective expected in the query images.

Lepetit's perspective matching system uses randomized decision trees and keypoint localization which is not scale-invariant. In this paper, we propose perspective matching using SVTs, which are more commonly used than randomized decision trees. The SVTs are built from SIFT or SURF features, which are scale-invariant.

[*] Please send correspondence to David Chen (dmchen@stanford.edu) and Sam S. Tsai (sstsai@stanford.edu).

Visual Communications and Image Processing 2009, edited by Majid Rabbani, Robert L. Stevenson, Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 7257, 72570V · © 2009 SPIE-IS&T · CCC code: 0277-786X/08/$18 · doi: 10.1117/12.805606

Several canonical views are chosen: one represents the original fronto-parallel view, while the others represent different perspective views. A set of *multiview SVTs* is constructed, where a different SVT is built for each canonical view. This multiview SVT database structure is well suited for a multi-processor server. In Sec. 2, the effect of perspective distortion on local features is discussed, providing an analysis and explanation not found in prior work on why a single-view tree fails. Multiview SVTs are presented in Sec. 3 to remedy the shortcomings of a single-view SVT. Experimental results presented in Sec. 4 show significant gains in matching accuracy are obtained with multiview SVTs.

2. EFFECT OF PERSPECTIVE DISTORTION ON IMAGE RETRIEVAL

The nodes of an SVT are the centroids determined by hierarchical k-means clustering of database feature descriptors. Suppose an SVT has N nodes. During the training phase, all feature descriptors extracted from the i^{th} database image are classified through the SVT, using a greedy nearest-neighbor search.[9] After classification, it is known how many times $n_i(j)$ the j^{th} node in the SVT is visited by the i^{th} database feature set, resulting in a vector of node visit counts $\mathbf{d}_i = [n_i(1),\ n_i(2),\ \ldots\ ,\ n_i(N)]$. Similarly, when a query feature set is received on the server, its descriptors are classified through the SVT, generating a vector $\mathbf{q} = [n_q(1),\ n_q(2),\ \ldots\ ,\ n_q(N)]$. The dissimilarity between the i^{th} database feature set and the query feature set is given by

$$D_i\left(\mathbf{d}_i, \mathbf{q}\right) = \left\| \frac{\mathbf{W}\mathbf{d}_i}{\|\mathbf{W}\mathbf{d}_i\|} - \frac{\mathbf{W}\mathbf{q}}{\|\mathbf{W}\mathbf{q}\|} \right\| , \tag{1}$$

where $\mathbf{W} = \operatorname{diag}\left(w_1, \ldots, w_N\right)$ is a matrix used to assign different entropy-related weights to different nodes and the norm could be L_1 or L_2.[3] The database feature sets with the smallest D_i values are further considered for a more elaborate comparison.

Using the SVT, information about a feature set is condensed to node visit counts. A perspectively degraded query feature set and the matching fronto-parallel database feature set will have different numbers of features and different descriptors, as discussed in Sec. 2.1 and 2.2. Consequently, their node visit counts can be very dissimilar, preventing the correctly matching database feature set from being included in the set of minimum-D_i candidates.

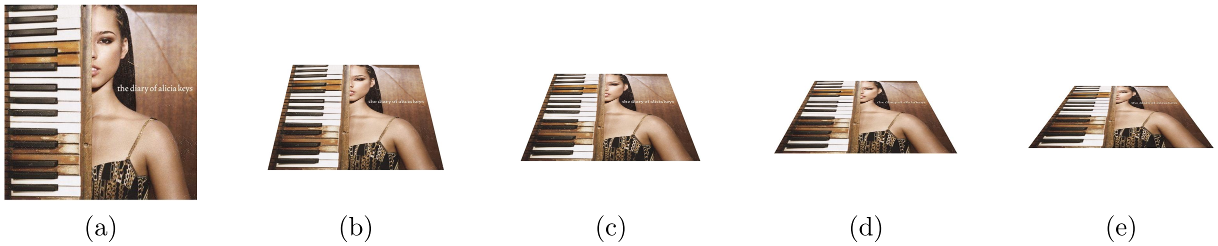
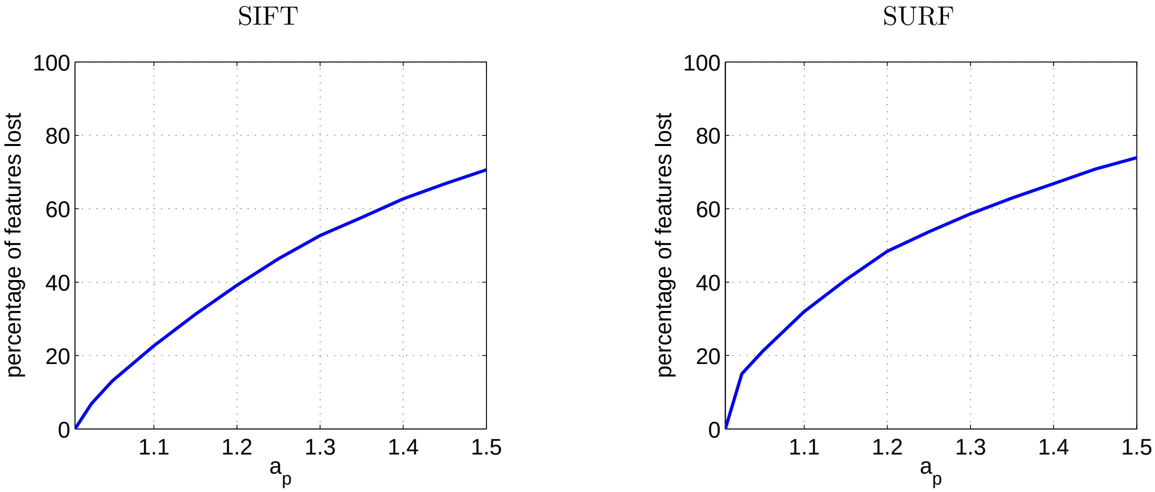


Figure 1. (a) Fronto-parallel CD cover image of resolution 500×500 pixels ($a_p = 1.0$). CD cover images obtained from (a) by applying increasingly more perspective foreshortening (b) $a_p = 1.2$, (c) $a_p = 1.3$, (d) $a_p = 1.4$, and (e) $a_p = 1.5$.

2.1 Change in Number of Query Features

Perspective foreshortening eliminates many keypoints that are detected in a fronto-parallel query image. A circular blob in the fronto-parallel view deforms into an elongated ellipse after perspective foreshortening, and the ellipse may no longer qualify as a feature. Fig. 1(a) shows an undistorted, fronto-parallel CD cover image. By applying increasingly more severe perspective foreshortening vertically, the CD cover image becomes progressively more distorted as shown in Fig. 1(b)-(e). Suppose the fronto-parallel CD cover has vertices with normalized coordinates $\{(0,0),(1,0),(1,1),(0,1)\}$. The projective transformation for parameter a_p maps these vertices to the coordinates $\{(0,0),(1,0),(a_p,1/a_p),(1-a_p,1/a_p)\}$.

Fig. 2 plots the percentage of SIFT and SURF features lost, relative to an undistorted CD cover image, as a function of the severity of perspective foreshortening. The statistics are averaged across 50 different CD cover images of resolution 500×500 pixels. As perspective distortion worsens, the percentage of features lost becomes



(a) (b)

Figure 2. Average percentage of features lost for (a) SIFT and (b) SURF, for 50 CD cover images.

significant. Suppose an SVT is built from database images in fronto-parallel view. Fig. 2 demonstrates that the query and database images can disagree significantly in the number of features for a CD cover. Thus, the scoring metric in (1), which requires that the number of features for an object of interest is similar in the query and database images, will become inaccurate.

2.2 Change in Query Feature Descriptors

Even if a keypoint is repeatably detected in the perspective view, the feature descriptor can still be severely distorted. After perspective foreshortening, different gradient responses will be measured in the image patch around a keypoint. Both SIFT and SURF calculate descriptors based on local gradients, so the deformation directly translates into distortion in the feature descriptor.

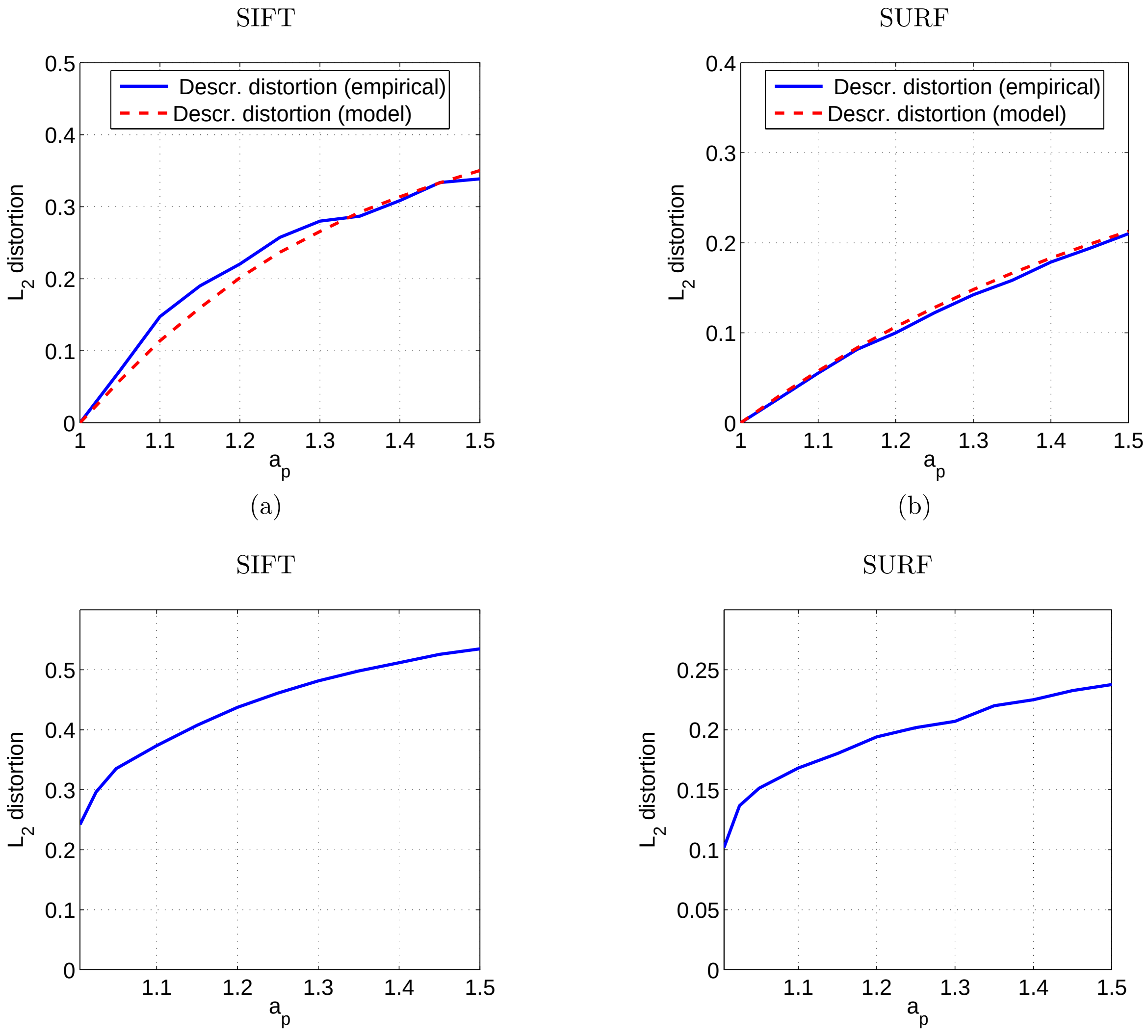
We analyze in Appendix A how the L_2-norm distortion in the descriptor increases with the severity of perspective. A Gaussian blob is chosen as our example of an image patch. Then, we can accurately model the change in the gradient field of the Gaussian blob as perspective increases. Distortions in the SIFT and SURF descriptors are natural consequences of the gradient field's distortion.

Fig. 3 shows the L_2-norm distortion in the SIFT and SURF descriptors of the Gaussian blob as perspective increases. The *empirical* curves are found by calculating SIFT or SURF descriptors on synthetic images containing only Gaussian blobs, while the *model* curves are found by computing gradient vector fields as derived in Appendix A. Distortion curves for real image patches will deviate from these curves because Gaussian blobs are too simple to account for complex patterns in real image patches. Nevertheless, our model clearly identifies distortion of the gradient field as the key reason for increased distortion in feature descriptors.

Also confirmed experimentally in Fig. 3, the average L_2-norm distortion in SIFT or SURF descriptors for real image patches increases with the severity of perspective. The distortion is considerable for even moderate amounts of perspective and rises above half of the inter-centroid distances at different levels of the SVT. Many of these distorted descriptors will be incorrectly classified in the fronto-parallel SVT, and the resulting node visit counts will be increasingly inaccurate.

3. MULTIVIEW SCALABLE VOCABULARY TREES

To accurately classify perspectively distorted query images, we propose the creation of multiview SVTs. Our test material is a large collection of CD covers. Each CD cover has four sides and a front face, so five multiview SVTs is a reasonable design choice, where each SVT corresponds to one canonical view. The original fronto-parallel database images belong to the *front* view, as in Fig. 1(a). Then, to generate images for a different view, a projective transformation is applied to the fronto-parallel images, as depicted in Fig. 4. For example, in the



(a) (b) (c) (d)

Figure 3. Average L_2 distortion in feature descriptors of synthetic images containing only Gaussian blobs, for (a) SIFT and (b) SURF. Models are derived from Appendix A. Average L_2 distortion in feature descriptors of 50 CD cover images, for (c) SIFT and (d) SURF.

bottom view, perspective foreshortening occurs in a bottom-to-top direction. The *front* view database images are projectively transformed into *bottom* view images, as depicted in Fig. 4. By varying the direction of perspective foreshortening, the *top*, *left*, and *right* views can also be generated. Local features are extracted from the images in each view and contribute to the construction of that view's SVT.

On a multi-processor server, several SVTs can simultaneously serve an incoming query. The top database candidates to emerge from each SVT are further subjected to pairwise comparison using a ratio test and geometric consistency checking (GCC).[1] Across all views, the database image associated with the maximum number of post-ratio-test, post-GCC feature matches is selected to be the best match. Our solution is attractive because it utilizes the ample computational capabilities of a server to handle the problem of perspective distortion.

Multiview SVTs offer substantial improvements over a single SVT. Unlike a fronto-parallel database image, a correctly pre-distorted database image can closely match a perspectively degraded query image in the number of features and in the characteristics of feature descriptors. Multiview SVTs preemptively prepare for perspective distortion and mimic the degradations. The statistical mismatches between a query feature set and the matching database feature set described in Sec. 2 can be greatly reduced with this pre-distortion approach.

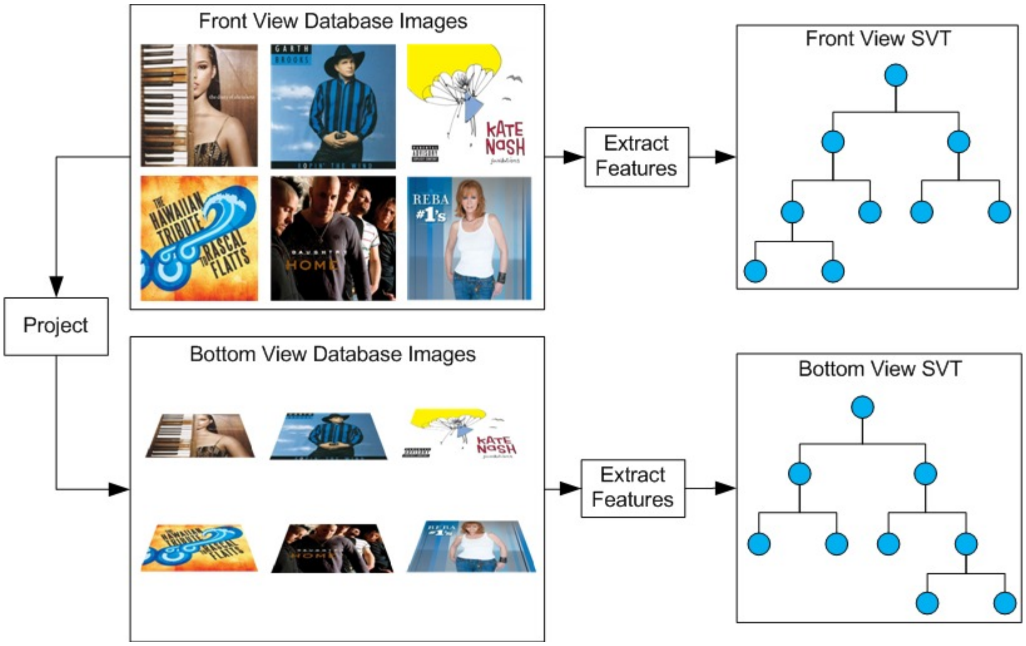


Figure 4. Construction of the *front* and *bottom* SVTs in a multiview set.

4. EXPERIMENTAL RESULTS

The performance of multiview SVTs is evaluated using a database of 10,000 CD cover images, each of resolution 500×500 pixels. Fifty of these covers are randomly chosen and photographed from different views, as shown in Fig. 5. Each SVT in the multiview set has a depth of 6 and a branch factor of 10, meaning the tree contains 1 million leaf nodes.

For each SVT, the 25 database images attaining the smallest values of D_i in (1) are checked for geometric consistency against the query image. Among all SVTs, the database image with the largest number of geometrically consistent feature matches is reported. If no database image contains at least 10 geometrically consistent feature matches, a *No Match* result is reported. The matching rate is the fraction of query images correctly identified. A *No Match* result is considered an incorrect identification.

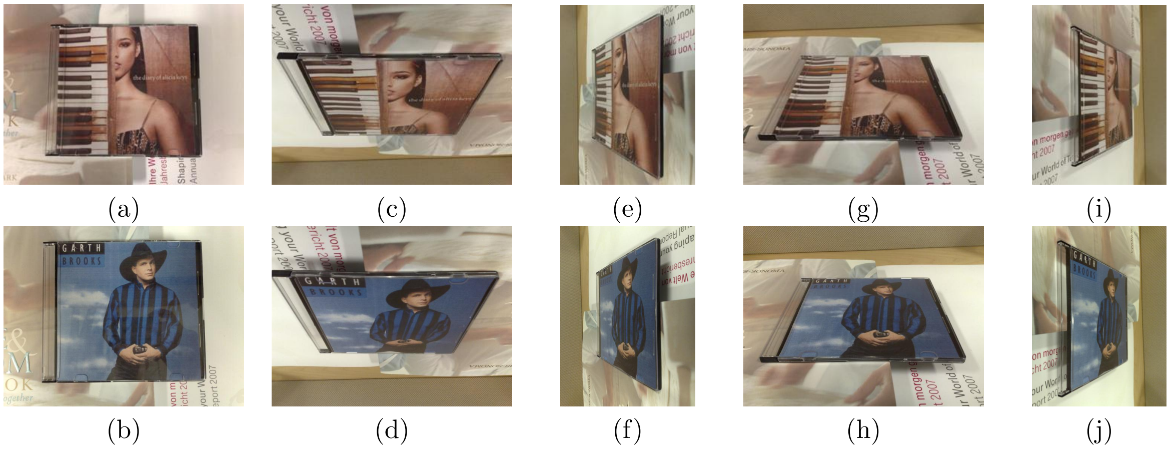

(a) (c) (e) (g) (i)
(b) (d) (f) (h) (j)

Figure 5. Query images in (a)-(b) *front*, (c)-(d) *top*, (e)-(f) *right*, (g)-(h) *bottom*, and (i)-(j) *left* views. The resolution is 640×480 pixels.

Table 1. Matching rates for different combinations of query image sets and database SVTs.

SIFT	Front SVT	Top SVT	Right SVT	Bottom SVT	Left SVT
Front Query	**0.92**	0.06	0.00	0.08	0.00
Top Query	0.00	**0.74**	0.00	0.34	0.00
Right Query	0.00	0.00	**0.60**	0.00	0.24
Bottom Query	0.00	0.28	0.00	**0.80**	0.00
Left Query	0.00	0.00	0.18	0.00	**0.76**

SURF	Front SVT	Top SVT	Right SVT	Bottom SVT	Left SVT
Front Query	**0.96**	0.00	0.00	0.12	0.02
Top Query	0.00	**0.72**	0.00	0.26	0.00
Right Query	0.00	0.00	**0.72**	0.00	0.20
Bottom Query	0.02	0.16	0.00	**0.76**	0.00
Left Query	0.00	0.00	0.24	0.00	**0.84**

Table 1 shows the matching rates for SIFT and SURF. For example, the entry in the *Left Query* row and *Front SVT* column is the matching rate for query images in the *left* view and an SVT for the *front* view. Only rates along the main diagonal are satisfactorily high, and they correspond to the situations where query and database images belong to the same view. The *Front Query-Front SVT* entry shows the highest matching rate because the *front* view does not suffer from severe perspective distortion. Additionally, the *top* and *bottom* views share similar features in the middle of the CD cover, as do the *left* and *right* views, resulting in small matching rates for the *Top Query-Bottom SVT*, *Bottom Query-Top SVT*, *Left Query-Right SVT*, and *Right Query-Left SVT* combinations. Looking across the *Front Query* row, we can observe the performance of a single fronto-parallel SVT. This SVT only works well for fronto-parallel query images and fails for query images in perspectively distorted views. A set of multiview SVTs enables significant increases in matching rates compared to any single-view SVT.

5. CONCLUSION

In this paper, we have presented multiview SVTs as an efficient server-side solution for handling perspective distortion in query images. We have shown both analytically and experimentally how statistical mismatches between query and database feature sets render a single-view SVT ineffective. Multiview SVTs can significantly reduce this statistical mismatch by preemptively mimicking the anticipated projective transformation. This solution utilizes the ample computational power of a server and shifts the complexity away from resource-limited mobile cameras. By using multiview SVTs, the image retrieval performance is much improved in the presence of perspective degradations.

APPENDIX A. DISTORTION OF FEATURE DESCRIPTORS

In this section, we analyze the effect of perspective foreshortening on a local feature descriptor, assuming that the keypoint can be repeatably identified after a projective transformation. To simplify analysis, image space is assumed to be continuous. Suppose a blob $i(x, y)$, which is a 2D-differentiable function, is centered on a keypoint identified by the SIFT or SURF detector. Without loss of generality, assume the keypoint is located at the origin. For the blob $i(x, y)$, the effect of perspective foreshortening can be approximated as geometric compression in one direction. Assume the blob is vertically compressed by factor $a > 1$, where a increases as perspective foreshortening worsens. The parameter a is not the same as the parameter a_p used in Sec. 2, although the two parameters are monotonically related, as long as compression is a good approximation for perspective foreshortening. The distorted blob is $i^a(x, y) = i(x, ay)$. Let a 1D normalized Gaussian be $g(x, \sigma) = 1/\sqrt{2\pi\sigma^2} \cdot \exp(-x^2/(2\sigma^2))$, and a 2D normalized Gaussian be $g(x, y, \sigma) = g(x, \sigma) \cdot g(y, \sigma)$. The

Gaussian-filtered second derivatives for $i^a(x,y)$ are

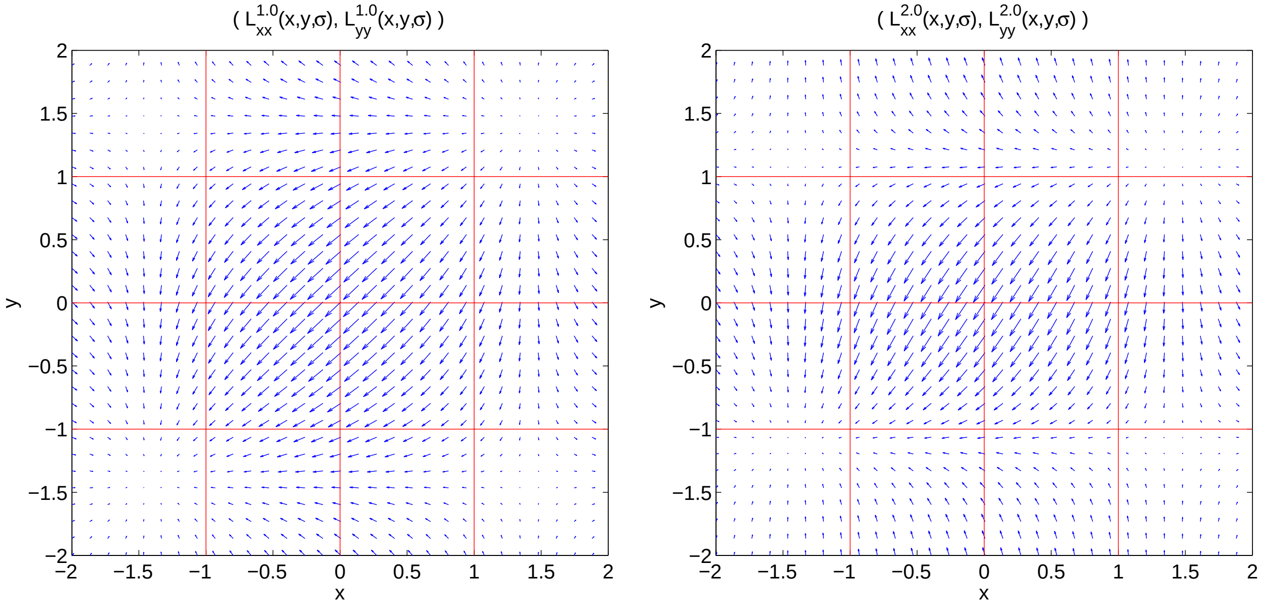
$$
\begin{aligned}
L^a_{xx}(x,y,\sigma) &= \frac{\partial^2}{\partial x^2}\, g(x,y,\sigma) ** i(x,ay)\ , &(2)\\
L^a_{yy}(x,y,\sigma) &= \frac{\partial^2}{\partial y^2}\, g(x,y,\sigma) ** i(x,ay)\ , &(3)\\
L^a_{xy}(x,y,\sigma) &= \frac{\partial^2}{\partial x \partial y}\, g(x,y,\sigma) ** i(x,ay)\ , &(4)
\end{aligned}
$$

where $**$ denotes 2D convolution and σ is the characteristic scale at which the keypoint is localized. Setting $a = 1$ yields the second derivatives of the original blob without any vertical compression.

To simplify analysis, further assume that the blob $i(x,y)$ is a 2D normalized Gaussian of scale σ: $i(x,y) = g(x,y,\sigma)$. The vertically compressed blob becomes $i^a(x,y) = 1/a \cdot g(x,\sigma) \cdot g(y,\sigma/a)$. Then, using the fact that convolution of two Gaussians of scales σ_1 and σ_2 yields another Gaussian of scale $\sqrt{\sigma_1^2 + \sigma_2^2}$, we find that

$$
\begin{aligned}
L^a_{xx}(x,y,\sigma) &= \frac{1}{a} g(x,\sigma') \cdot g(y,\sigma'') \left(\frac{x^2}{\sigma'^4} - \frac{1}{\sigma'^2} \right)\ , &(5)\\
L^a_{yy}(x,y,\sigma) &= \frac{1}{a} g(x,\sigma') \cdot g(y,\sigma'') \left(\frac{y^2}{\sigma''^4} - \frac{1}{\sigma''^2} \right)\ , &(6)\\
L^a_{xy}(x,y,\sigma) &= \frac{1}{a} g(x,\sigma') \cdot g(y,\sigma'') \left(\frac{xy}{\sigma'^2 \sigma''^2} \right)\ , &(7)
\end{aligned}
$$

where $\sigma' \equiv \sigma\sqrt{2}$ and $\sigma'' \equiv \sigma\sqrt{1 + 1/a^2}$. Now, we separate the analysis between SIFT and SURF, because these two descriptors are calculated differently, although both descriptors depend on the second derivatives in (5)-(7).



(a)

(b)

Figure 6. Gradient vector fields for (a) $a = 1.0$, and (b) $a = 2.0$.

A.1 SIFT Descriptor

The pair (L^a_{xx}, L^a_{yy}) from (2)-(3) can be regarded as a vector field. For the Gaussian blob, we can use (5)-(6) to plot the vector field in Fig. 6, contrasting two cases where vertical compression is absent ($a = 1.0$) and present

($a = 2.0$). Perspective foreshortening, modeled as geometric compression along the vertical direction, results in the vectors pointing downwards more.

The vectors (L_{xx}^a, L_{yy}^a) can also be represented in polar form as

$$m^a(x,y,\sigma) = \sqrt{L_{xx}^a(x,y,\sigma)^2 + L_{yy}^a(x,y,\sigma)^2} \tag{8}$$

$$\theta^a(x,y,\sigma) = \arctan\left(\frac{L_{yy}^a(x,y,\sigma)}{L_{xx}^a(x,y,\sigma)}\right) \tag{9}$$

For SIFT, a grid of 4-by-4 squares is laid on top of the image patch surrounding the keypoint, as shown in Fig. 6. Within the i^{th} square S_i, a histogram of the vector angles is calculated, and 8 bin centers $\{\theta_1, \ldots, \theta_8\}$ are uniformly chosen on $[0, 2\pi)$. A vector of local statistics is then formed as

$$\mathbf{v}_i = \left[\int_{B_1} m^a(x,y,\sigma)\ ,\ \int_{B_2} m^a(x,y,\sigma)\ ,\ \ldots\ ,\ \int_{B_8} m^a(x,y,\sigma)\right] \tag{10}$$

$$B_j = \left\{(x,y) \in S_i\ :\ \theta_j = \underset{\theta_k \in \{\theta_1,\ldots,\theta_8\}}{\operatorname{argmin}} |\theta_k - \theta^a(x,y,\sigma)|\right\} \tag{11}$$

The entire 128-dimensional SIFT descriptor is $\mathbf{v} = [\mathbf{v}_1,\ \mathbf{v}_2,\ \ldots,\ \mathbf{v}_{16}]$, and the normalized descriptor is $\bar{\mathbf{v}} = \mathbf{v}/||\mathbf{v}||_2$. Because the vector field is distorted by vertical compression, as shown in Fig. 6, the descriptor is also distorted. For the Gaussian blob, we can numerically evaluate (10) for all 16 squares and calculate the descriptor for different values of the compression parameter a. Fig. 7(a) plots the L_2 error in the descriptor as function of a, relative to the undistorted descriptor ($a = 1.0$). As expected, more severe vertical compression causes greater distortion in the descriptor.

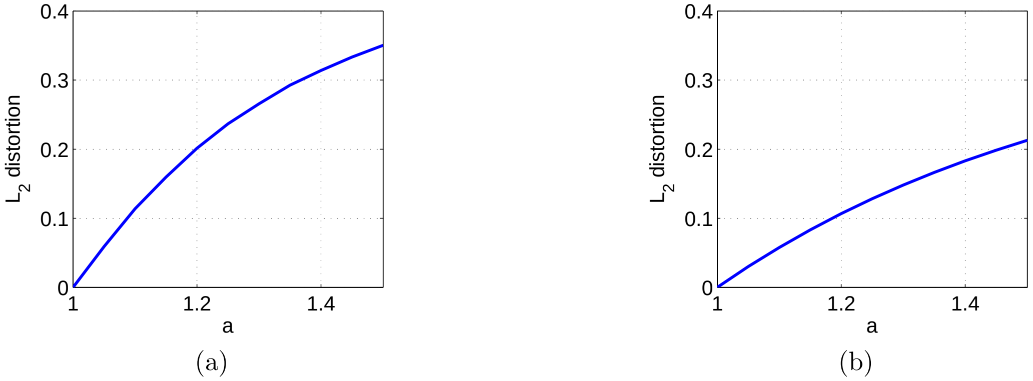


Figure 7. L_2 distortion in (a) SIFT and (b) SURF descriptor as a function of a, the severity of vertical compression.

A.2 SURF Descriptor

The region over which the SURF descriptor is calculated is also a grid of 4-by-4 squares. In the i^{th} square S_i, a vector of local statistics is computed as

$$\mathbf{v}_i = \left[\iint_{S_i} L_{xx}^a(x,y,\sigma)\ ,\ \iint_{S_i} L_{yy}^a(x,y,\sigma)\ ,\ \iint_{S_i} |L_{xx}^a(x,y,\sigma)|\ ,\ \iint_{S_i} |L_{yy}^a(x,y,\sigma)|\right]\ . \tag{12}$$

Altogether, the unnormalized 64-dimensional SURF descriptor is the concatenation $\mathbf{v} = [\mathbf{v}_1,\ \mathbf{v}_2,\ \ldots,\ \mathbf{v}_{16}]$, and the normalized descriptor is $\bar{\mathbf{v}} = \mathbf{v}/||\mathbf{v}||_2$.

Because the vector field in Fig. 6 becomes more distorted as vertical compression increases, the SURF descriptor calculated from integrals over the vector field becomes progressively more distorted. For the Gaussian blob, we can numerically evaluate (12) for all 16 squares and calculate the descriptor for different values of the compression parameter a. Fig. 7(b) plots the L_2 error in the descriptor as function of a, relative to the undistorted descriptor ($a = 1.0$). Like for SIFT, more severe vertical compression causes greater distortion in the SURF descriptor.

REFERENCES

1. D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision* **60**, pp. 91–110, November 2004.
2. H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: speeded up robust features," in *European Conference on Computer Vision*, pp. 404–417, (Graz, Austria), May 2006.
3. D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Conference on Computer Vision and Pattern Recognition*, pp. 2161–2168, (New York, NY, USA), June 2006.
4. D. Douxchamps and B. Macq, "Integrating perspective distortions in stereo image matching," in *International Conference on Acoustics, Speech, and Signal Processing*, **3**, pp. 301–304, (Montreal, Quebec, Canada), May 2004.
5. K. Koser and R. Koch, "Perspectively invariant normal features," in *International Conference on Computer Vision*, pp. 14–21, (Rio de Janeiro, Brazil), October 2007.
6. K. Mikolajczyk and C. Schmid, "Scale and affine invariant interest point detectors," *International Journal of Computer Vision* **60**, pp. 63–86, October 2004.
7. V. Lepetit, J. Pilet, and P. Fua, "Point matching as a classification problem for fast and robust object pose estimation," in *Conference on Computer Vision and Pattern Recognition*, **2**, pp. 244–250, (Washington, DC, USA), June 2004.
8. V. Lepetit, P. Lagger, and P. Fua, "Randomized trees for real-time keypoint recognition," in *Conference on Computer Vision and Pattern Recognition*, **2**, pp. 775–781, (San Diego, CA, USA), June 2005.
9. G. Schindler, M. Brown, and R. Szeliski, "City-scale location recognition," in *Conference on Computer Vision and Pattern Recognition*, pp. 1–7, (New York, NY, USA), June 2007.