Transitive Closure based visual words for point matching in video sequence

Srikrishna Bhat K K, Marie-Odile Berger, Gilles Simon, Frédéric Sur,
INRIA/LORiA/Nancy-Université, France
$\{$*bhatsrik, berger, gsimon, sur*$\}$*@loria.fr*

Abstract

We present Transitive Closure based visual word formation technique for obtaining robust object representations from smoothly varying multiple views. Each one of our visual words is represented by a set of feature vectors which is obtained by performing transitive closure operation on SIFT features. We also present range-reducing tree structure to speed up the transitive closure operation. The robustness of our visual word representation is demonstrated for Structure from Motion (SfM) and location identification in video images.

1. Introduction and Background

Recent advances in object recognition based on local region descriptors have shown the possibility of highly efficient image matching and retrieval from huge databases[9, 7]. Most existing works aim at quantizing the SIFT descriptors of the database into a vocabulary tree, where visual words are obtained using clustering techniques. Matching two images is done by comparing the lists of visual words. Recognition based matching techniques have proven to be useful for 3D reconstruction [2], location recognition [3] and pose computation [6, 5]. As in [5] we are interested in recognition based pose computation techniques with a view to augmented reality applications in large environments. In [5], SIFT features extracted from a set of reference images are used as input to bundle adjustment to obtain a 3D model of the scene. During online application, SIFT features extracted from the current frame are matched to the 2D image features of the world model using a kd-tree algorithm, thus allowing 2D-3D correspondences and finally pose computation. Bundle adjustments techniques are efficient for relatively small environments but are prone to failure when repeated patterns are present or when many small sets of corresponding points are generated. In these cases, even RANSAC procedures are unable to recover a coherent set of matches, making the procedures fail. We thus propose in this paper to use the concept of visual words for the matching stage both for model construction and for online matching. As shown in the following sections, this allows us to discard many ambiguous and erroneous matches and to noticeably increase the robustness of the model construction and the online matching.

Ideally, visual words for pose computation should contain points which are similar under visual perception and are likely to fit the same 3D point of the model. Unfortunately, visual words strongly depend on the clustering techniques and may contain a too restricted set of aspects of the same physical points. They also may contain features from several different physical points. We here propose a new way to build visual words with the idea to get groups of features which are as far as possible similar under visual perception. This is done using transitive closure techniques. Our words are more perceptively significant than classical ones and can be used to detect and to remove ambiguous words. Too large words are removed because they generally correspond to numerous similar patterns in the background. Words which contain more than one occurrence in each image are also removed because they are likely to match repetitive patterns and will induce high computational cost in the RANSAC procedure. The use of words thus allows us to reduce the set of hypothesis both for bundle adjustment and pose computation and lead to more robust procedures.

Currently, the different ways of forming the visual words can be broadly classified into two categories.

Type1[9, 7]:In the training step, the set of feature vectors extracted from the training images is divided into a pre-determined k number of groups by clustering. Each group forms a visual word represented by the cluster center. While testing, a feature vector is categorized to the word corresponding to the closest cluster center.

Type2[3]:A visual word is defined as a spherical region of radius r around a feature vector. Visual dictionary is built incrementally. For each SIFT feature f, the dictionary is searched for a visual word centered within a distance of r from f. If f does not match any of the existing words, then a new visual word centered at f is added to the dictionary. If a match is found then f is not added.

In Type1 methods, determining a suitable value for the parameter k is a difficult task. If k is small then the features corresponding to different objects merge into a single word. If k is large then the set of feature vectors from the same object will get split into multiple visual words. Another problem with Type 1 methods is that it does not provide any rejection possibility when trying to match a feature from a yet-unseen object which should be discarded based on the large distance to the nearest cluster center. On the contrary, a framework based on a distance threshold as in Type2 can be used to reject a feature vector based on distance threshold r. But it can have only spherical shaped visual words. This creates a problem when feature descriptors are extracted from smoothly varying multiple views of an object.

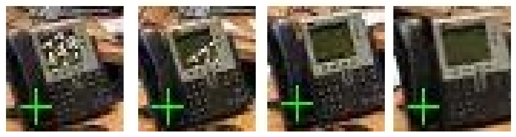


Figure 1: Smoothly varying pose

Figure 1 shows four consecutive frames from a video sequence. The '+' mark in green color in each image shows the location at which SIFT feature vectors x_1, x_2, x_3, x_4 were detected. We have observed that the Euclidean distance $d(x_i, x_{i+1}) < 125$ for $i \in \{1, 2, 3\}$, but the distance $d(x_1, x_4) > 250$. Hence for any value $r < 250$, Type2 methods will categorize x_1 and x_4 into two separate visual words. But, for $r > 200$ we have obtained many false matches between SIFT vectors.

To address this problem we propose a different way for visual word formation in section 2.

2 Visual word formation using transitive closure

Let S be the set of SIFT vectors $\{f_1.....f_N\}$ extracted from the training images. Let $d(x, y)$ be the Euclidean distance measure between two vectors x and y. As in Type 2 methods, we use a distance threshold r to match SIFT features. Two SIFT features x_1, x_2 are said to be *similar* if $d(x_1, x_2) < r$. This similarity relation is *reflexive* and *symmetric*. We perform transitive closure operation on this similarity relation on S. Each equivalence class in S obtained in this way represents a visual word. Hence our visual word v is represented by a set of feature vectors $\{f_{v_1}, .., f_{v_n}\}$ instead of a single feature vector. A vector f is said to *match* with v if there exists at least one $f_{v_k} \in v$ such that $d(f, f_{v_k}) < r$.

Our algorithm to compute visual words is as follows: Let V be the set of visual words which is initially empty. Eventually it will contain mutually exclusive subsets of S, each one of which represents one visual word. The algorithm will loop over each element of S. In each iteration $i = 1..N$ it will execute following 3 steps :

1. Find the visual words in V which have at least one element f such that $d(f, f_i) < r$

2. If any such words found in V, then merge all those sets together by union operation to form a single visual word and add vector f_i to it.

3. If no such set is found in V then a new element $\{f_i\}$ is added to V.

This algorithm, which is an adaptation of Tarjan's algorithm [8] for connected components in graphs, exactly computes the transitive closure of interest. The following section presents a range reducing technique for speeding up the similarity search (step 1).

2.1 Range-Reducing tree structure for step(1)

In i^{th} loop, the above algorithm needs to compare f_i with $i - 1$ vectors in step(1). Hence, for N vectors in S it will need $\frac{N(N-1)}{2}$ number of comparisons. To reduce the number of comparisons, we incrementally build *range reducing tree* structure which is similar to that of *M-tree*[4]. But, in addition we have structural constraints that the tree should have a fixed number of levels L and fixed *covering radius* or *range* R_l at each level l. For the sake of brevity, we present our algorithm on a running example. Figure 2 shows how new vectors $m1, m2$ and $m3$ (blue dots with a dotted blue circle of radius r around them) will be incrementally added to a tree structure which already contains 10 vectors (black dots numbered 1 to 10). The tree has 3 levels with range R_0 (red circle), R_1 (green circle) and $R_2 = 0$. Similar to M-Trees, each node at level $l = 1, .., L - 1$, contains a list of centers in which each center c has a link to a sub-tree T_c such that all the centers $c^{'} \in T_c$ satisfy the condition $d(c, c^{'}) < R_l$. The radius at the leaf level R_L is 0.

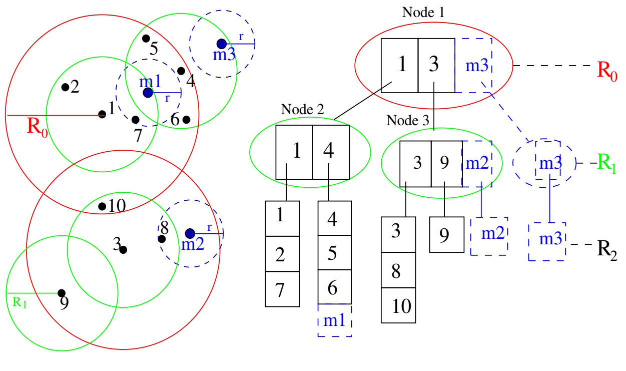


Figure 2: Range Reducing Tree structure

If we assume that such a structure is available for vectors $\{f_1.....f_{i-1}\}$ in S, then for f_i at step(1) of the algorithm, starting from the root node, at each level l, we need to consider only those sub-trees corresponding to the centers c_j satisfying *match condition* $d(c_j, f_i) \leq R_l + r$. In figure 2, for $m1$, at level 0, only the red circle centered at 1 satisfies the condition $d(c_j, m1) < R_0 + r$. Hence the search will be directed towards centers 1 and 4 at level 1. Both 1 and 4 satisfy the condition $d(c_j, m1) < R_1 + r$ and $m1$ will be compared with $1, 2, 4, 5, 6, 7$ out of which 7 is the only match.

During the search f_i will be added to the node T_{opt} corresponding to the closest center c_{opt} at level l_{opt} beyond which the *range condition* $d(c_j, f_i) \leq R_l$ fails. In figure 2, for $m2$, the condition fails in $Node3$ corresponding to center 3 in the rood node at level 0. Hence it is added to $Node3$. While adding a new feature vector to a node at level l, a subtree with a single child at each level till the leaf is added so that the tree structure remains consistent. Since the range condition $d(c, f_i) \leq R_l$ is stricter than the match condition $d(c, f_i) \leq R_l + r$, we do not need to search any additional nodes for adding f_i.

3 Experimental Results

We show our results for Location Identification and Structure from Motion (SfM). We have used the value $r = 125$ throughout our experiments. For range reducing tree we set $L = 7$ with range $800, 600, 450, 350, 250, 125$ and 0 at respective levels. Compared to an exhaustive process, we obtain 20 times reduction in the number of vector comparisons for a data set consisting of $795,000$ SIFT features. A test sift feature requires 30ms on average for finding matches.

3.1 Location Identification

For location identification we have used monocular images from ImageCLEF [1] database. It consists of three data sets *training_easy*, *training_hard* and *test*, containing 4074, 2267 and 2551 images respectively. All the datasets are acquired within an office environment, under varying illumination conditions. All the training image file names contain location id (9 locations in total). The *test* sequence contains additional locations (13 locations in total) that were not imaged in both of the training sequences. The *training_hard* sequence was acquired while moving the camera in a direction opposite to the one used for the *training_easy* and *test* sequences. It contains less number of view points of the environment compared to *training_easy*.

Figure 3 shows the SIFT features belonging to the same visual words in two different views of the stairway in *training_easy*. Each '+' mark in green color shows the location of the SIFT feature along with the visual word ID number in red color. Word IDs are assigned in the descending order of the number of features in each word. We can see that the local image patterns around SIFT features belonging to a particular visual word are similar under visual perception. We can also observe that the visual words $21, 26, 34, 41$ were detected in two views despite huge variation in the viewing angle. Hence the matching provided by our framework can be used as a good evidence for location identification. Moreover, the visual word numbered 1 mostly consists of locations with plain background. These top order words i.e. the words containing huge number of SIFT features, consist of feature vectors which belong to either repetitive or less discriminative patterns. We will see in section 3.2 that, discarding such ambiguous matches is crucial for improving the accuracy of RANSAC process for bundle adjustment. Figure 4 (a) shows all the occurrence of the words in a *test* image. Figure 4 (b) shows the matching words in one of the images in *training_easy*. We can see that all the low order visual words except 93 are good matches and the location is correctly retrieved.

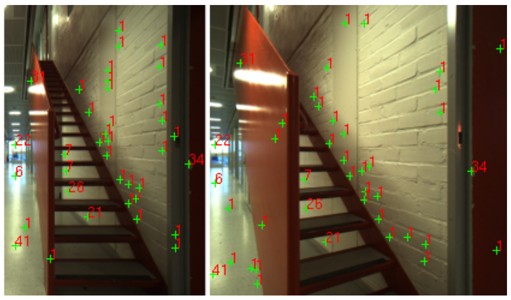


Figure 3: Words tracked through transitive closure

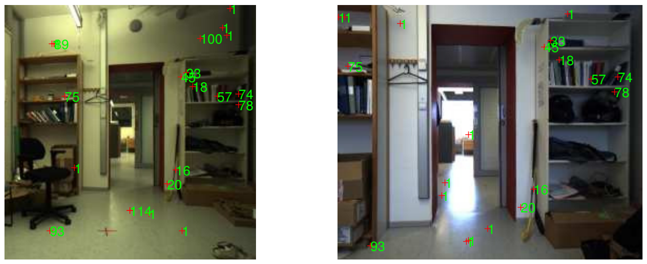
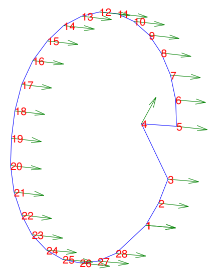

(a) (b)

Figure 4: (a)Query Image (b)Train Image

We present here a very basic decision rule for location identification. Our purpose is just to show the reliability of our visual words. Results would be improved with e.g. a Bayesian framework as in [3]. For each visual word w obtained from the training set we compute vote V_w^l which is the ratio of number of times w appears at location l to the total number of occurrence of w. We discard the words having their highest vote value less than 30% because the location information in those words is very low. For each test image I^t, votes corresponding to all the detected visual words are added for each location. Let l^{opt} be the location with highest votes. If I^t contains at least 8 different visual words and at least 30% of those words appear in l^{opt}, then I^t is labelled as l^{opt}. If I^t contains less than 5 different visual words then it is labelled as $unseen$. Otherwise I^t is not labelled due to lack of confidence. By training our algorithm using *training_easy* data set we have obtained 2109 (82.7%) correct identification, 166 (6.5%) wrong identification and 276 (10.8%) unidentified (due to lack of confidence) locations. On *training_hard* data set we have obtained 1547 (60.6%) correct identification, 602 (23.6%) wrong identification and 402 (15.8%) unidentified locations.

3.2 Structure from Motion (SfM)

We capture a short video sequence by fixing a camera on a base and moving the base in a circular path for one full round. We discard the top 5 visual words, visual words containing multiple feature vectors from a single image and visual words with less than 5 feature vectors. We use Bundler package[10] for performing SfM on two cases (a) matches obtained by our algorithm (b) matches obtained from the program included in the Bundler package. Figure 5(a) and 5(b) show the results for case (a) and (b) respectively. The green arrows show the camera orientation at subsampled camera positions numbered 1 to 28. The path and camera orientation in figure 5(a) is as expected (except at point 4) since the camera is moving in a circular path while keeping a fixed orientation. In figure 5(b) we have many abrupt jumps. To analyse this further we ran SfM 5 times for case (a) and (b). For case (a) we obtained 3800 3D points on average. Each time at least 1600 of them (around 40%) were observable in at least 10 images. For case (b) we obtained around 5200 points and at most 900 of them (around 17%) were observable in at least 10 images. This demonstrates the utility of our framework for providing reliable set of point-to-point correspondences.


(a) Proposed Matching

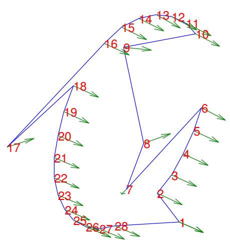

(b) Bundler Matching

Figure 5: Structure from Motion

4 Conclusion and Future Work

In this work we demonstrated the utility of transitive closure based definition of visual word for establishing reliable point-to-point correspondence between video frames. Using our framework we can discard ambiguous point correspondences between video frames, which is an important step for improving robustness of pose computation and bundle adjustment techniques. One obvious drawback in this representation is the need of storing multiple feature vectors for a single visual word. We are looking into two different strategies to address this problem (i) Improving the tree structure (ii) Finding a compact representation for visual words.

References

[1] http://www.imageclef.org/2010/icpr/robotvision.

[2] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV - 2009*.

[3] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. Fast and incremental method for loop-closure detection using bags of visual words. In *IEEE Transactions on Robotics 2008*.

[4] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *VLDB 1997*.

[5] I. Gordon and D. G. Lowe. Scene modeling, recognition and tracking with invariant image features. In *ISMAR 2004*.

[6] V. Lepetit, J. Pilet, , and P. Fua. Point matching as a classification problem for fast and robust object pose estimation. In *CVPR 2004*.

[7] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR 2006*.

[8] E. Nuutila. Efficient transitive closure computation in large digraphs. In *Acta Polytechnica Scandinavica 1995*.

[9] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV 2003*.

[10] N. Snavely. http://phototour.cs.washington.edu/bundler/.