[M2, MVA]

Object recognition and computer vision

Maha ELBAYAD
maha.elbayad@student.ecp.fr

Assignement 1
Instance-level recognition

October 20, 2015

1 Sparse features for matching specific objects in images

1.1 SIFT features detections

QIA.1

Having similarity co-variant features means we will be able to recognize and match keypoints at different scales, illumination and basic image transformations (rotation, translation..). Otherwise, we would need to extract features under different conditions then match them, which is computationally costly.

QIA.2

The *getFeatures* function adapted from the *vl_covdet* function, implements co-variant feature detectors (DoG or Hessian) and corresponding SIFT descriptors. When choosing the *Hessian* method we eliminate the candidate keypoints (points with strong DoG response) that have low contrast or are poorly localized along an edge. This is illustrated in the figure (1d) per comparison to the features in (1f)

1.2 SIFT features descriptors and matching between images

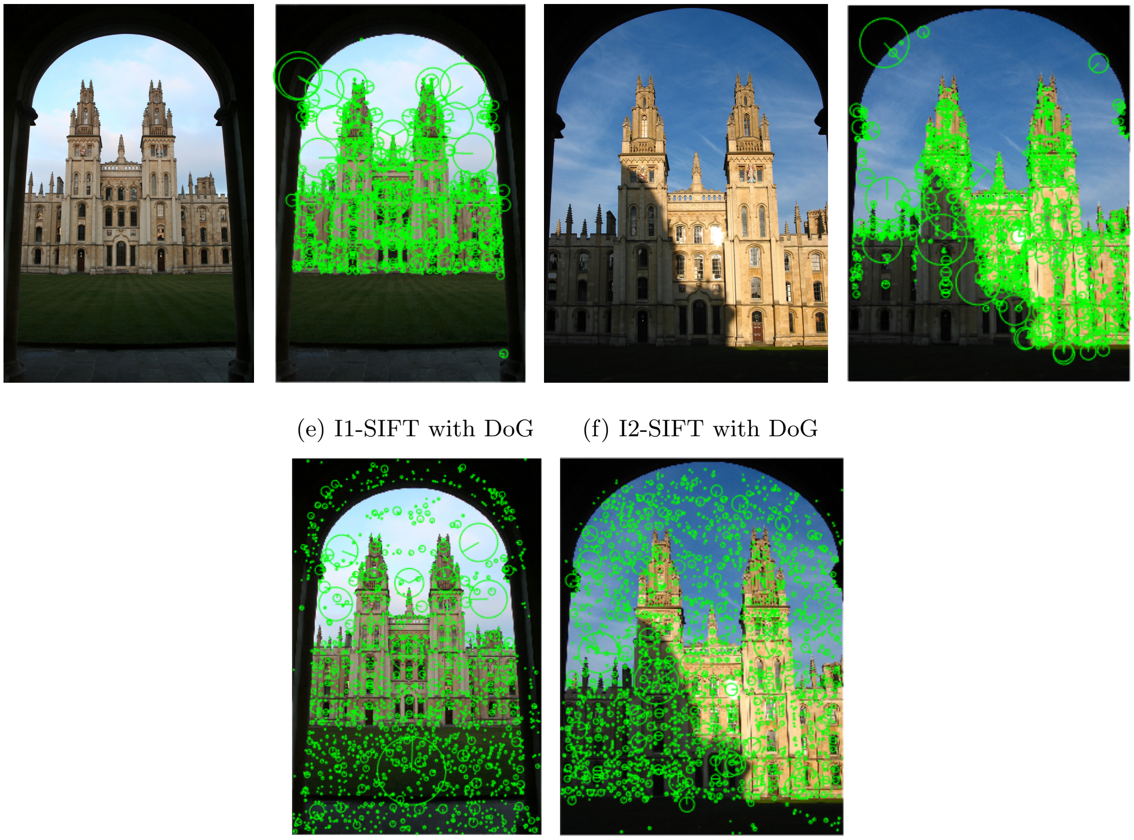
QIB.1

Computing the descriptors over larger regions is a good strategy since it would contain more signal variations and would include the edge effect area.

QIB.2

As we can see in the figure 2, mismatches occur mainly on the edges ($a_1 \to a_2$ and $b_1 \to b_2$)

Figure 1: SIFT descriptors



(a) I1 (b) I1-SIFT with Hessian (c) I2 (d) I2-SIFT with Hessian

(e) I1-SIFT with DoG (f) I2-SIFT with DoG

since the detection regions are repeatable. The mismatch $b_1 \rightarrow b_2$ is also due to the fact that the true match for b_1 is in a low contrast region where no feature has been selected.
The mismatches could be avoided by:
- Thresholding the ratio $\dfrac{\text{Distance to first NN}}{\text{Distance to second NN}}$ [Lowe04]
- Adding semi-local constraints to match not only a single feature but also it's closest neighbours under geometrical consistency.
- Considering a backward matching from image 2 to image 1.

1.3 Improving SIFT matching using Lowes second NN test

QIC.1

When implementing Lowe's second nearest neighbour test, the number of matches drops down, permetting to filter most of the mismatches. Nonetheless, when thresholding the ratio too low, most of the matches are rejected as shown in figure 3. However, some mismatches persist even with a threshold as low as .5 (figure 4):

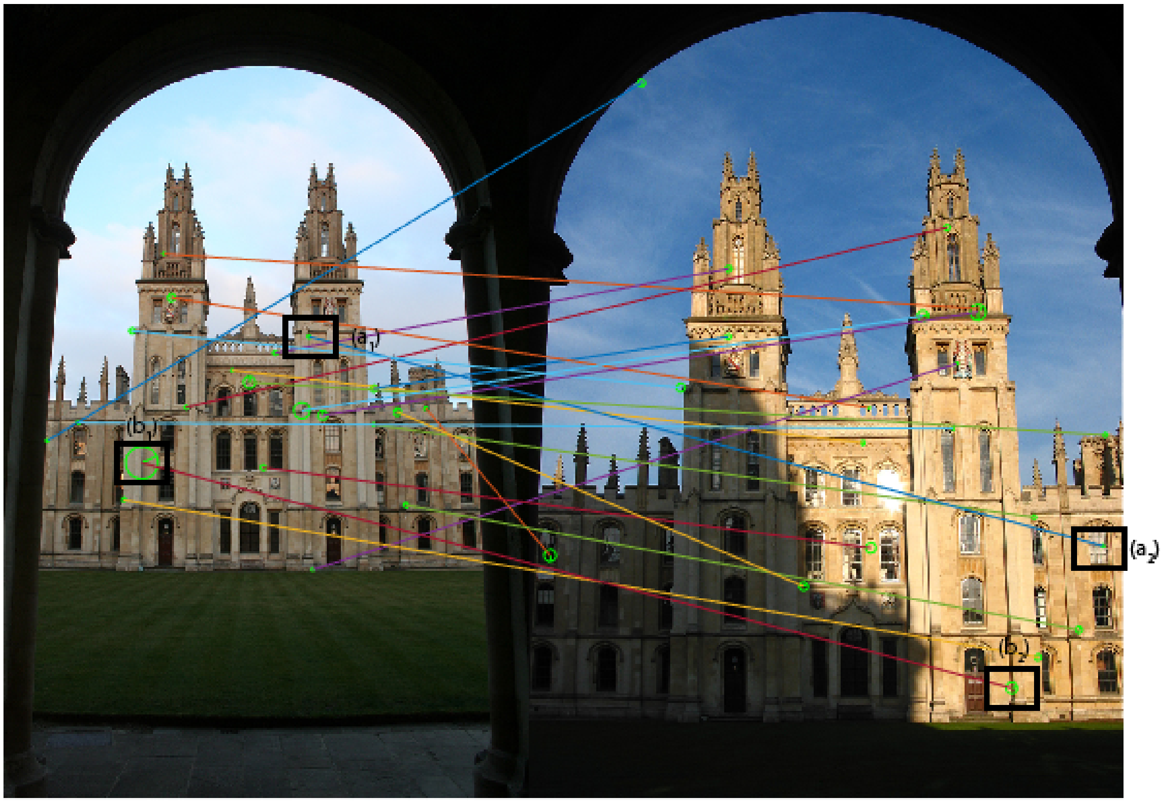


Figure 2: Some of the matches

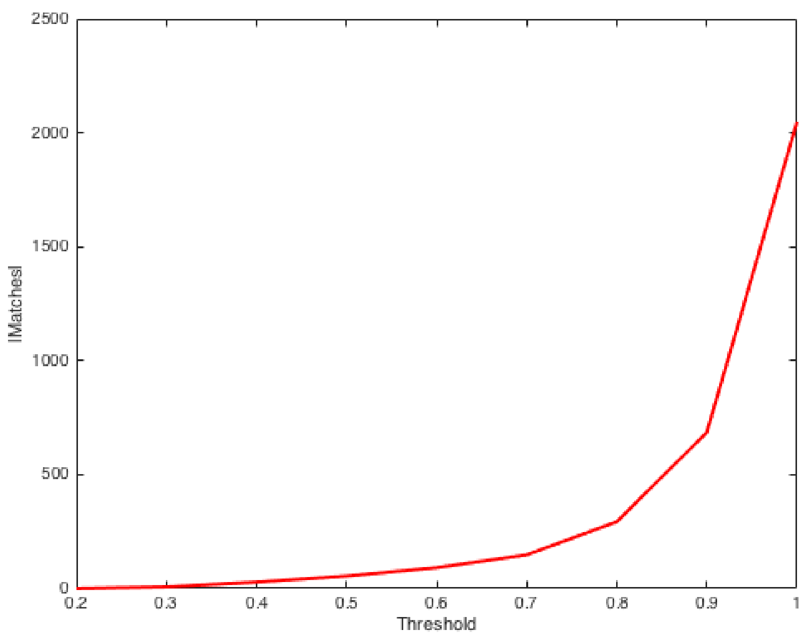
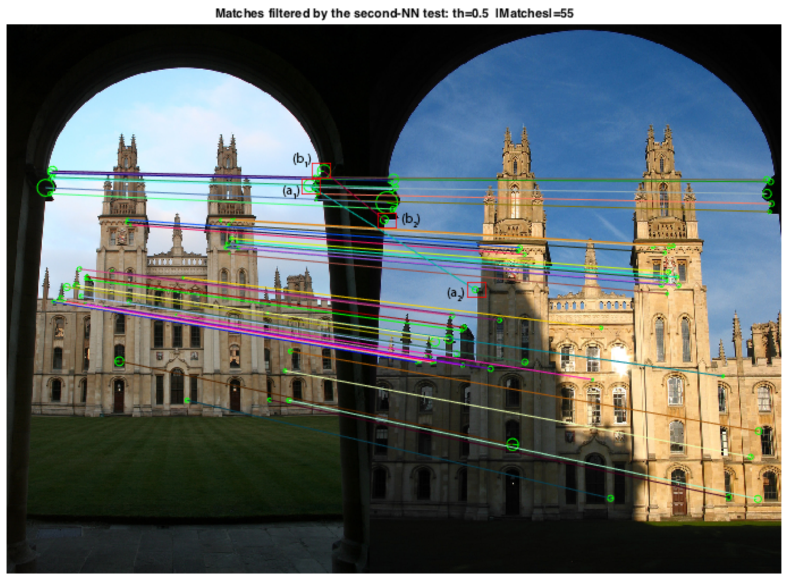


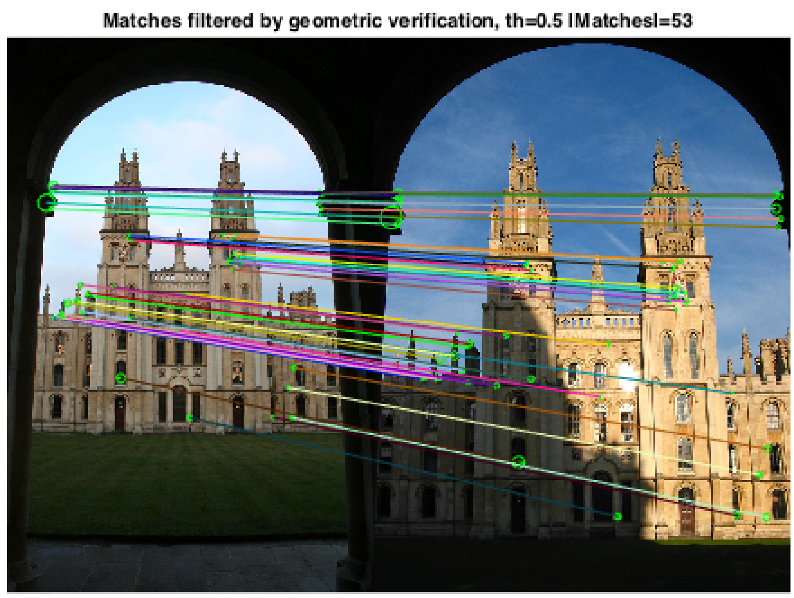
Figure 3: The number of matches as a function of the chosen threshold

Figure 4: The matches with a threshold=.5

(a) 2-NN



(b) Geometric verification



1.4 Improving SIFT matching using a geometric transformation

QID.1

From a single correspondence $(x_1, y_1, s_1, \theta_1) \leftrightarrow (x_2, y_2, s_2, \theta_2)$, we compute the similarity transformation:

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = sR(\theta) \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

as:

$$\begin{cases} \theta = \theta_2 - \theta_1 \\ t_x = x_2 - x_1 \\ t_y = y_2 - y_1 \\ s = \dfrac{s_2}{s_1} \end{cases}$$

Each correspondence yields a transformation to vote up in the 4D Hough space. We will then check the geometric tranformations with more than 3 votes.
Using the matches after Lowe's second-NN test with a .8 ratio, we get the following histograms (figure 5).

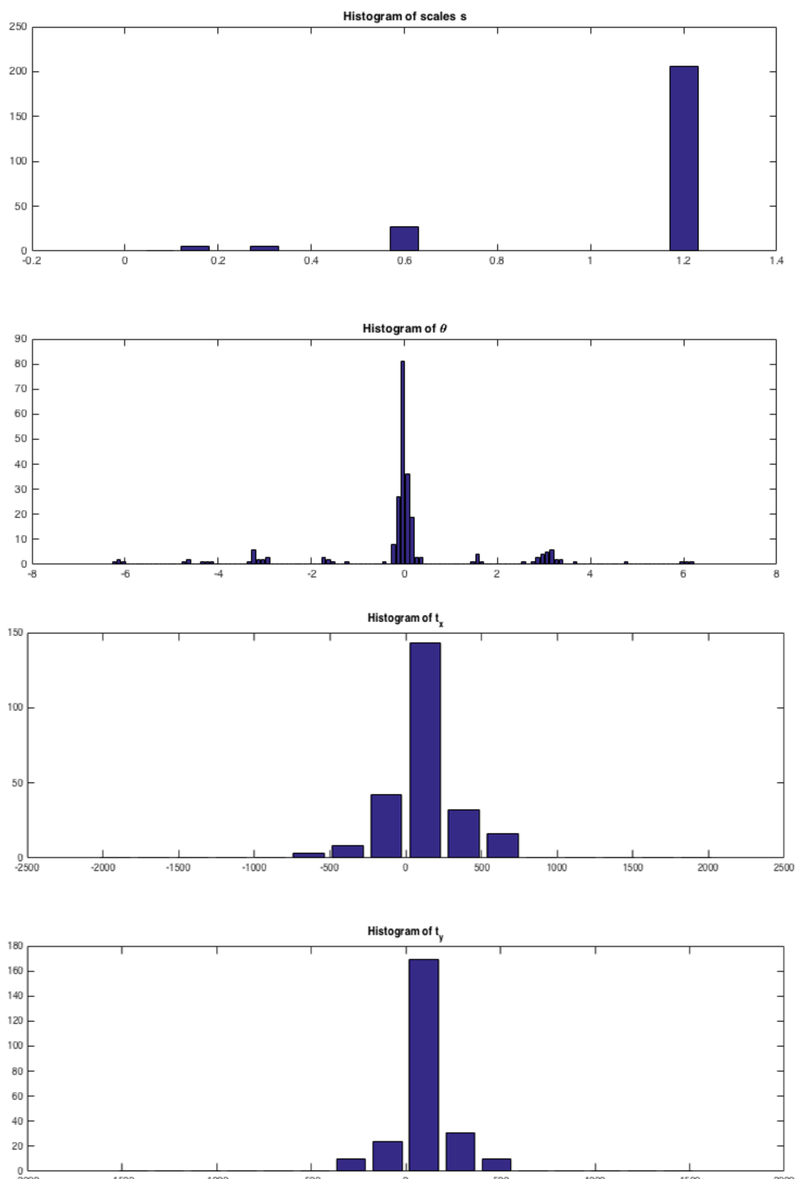


Figure 5: Histograms of the geometric transformation parameters

QID.2 (QIC.2)

The matches consistent with a similarity can be found using a RANSAC inspired algorithm. The *geometricVerification* function helps remove in the case of 2-NN/1-NN ratio of .5, the two mismatches as seen in figure 4b.

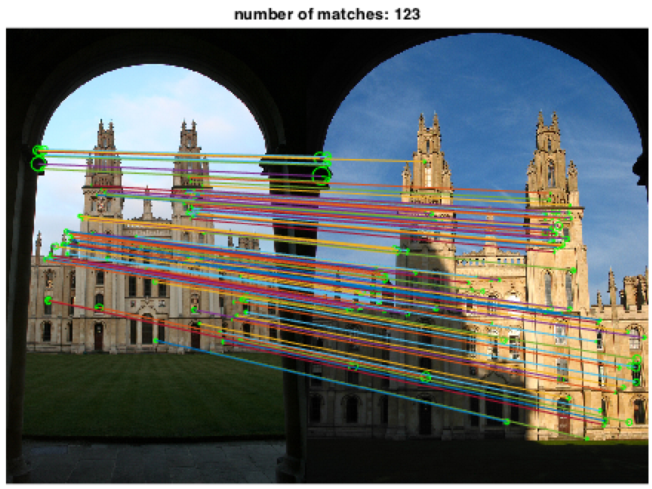
When changing the tolerance parameters for the *geometricVerification* function we control the width of the support area allowed when counting the inliers. We can see that the number of matches vary depending on witch parameter we increase (figure 6)

Another way to run geometric verification it to feed it with the closest neighbours directly without thresholding d(2NN)/d(1NN). Or better yet include the 2NN and 3NN to be geometrically verified.
We've run the geometric verification with 1NN, 1NN+2NN and 1NN+2NN+3NN and timed the geometric verification - results shown in figure :

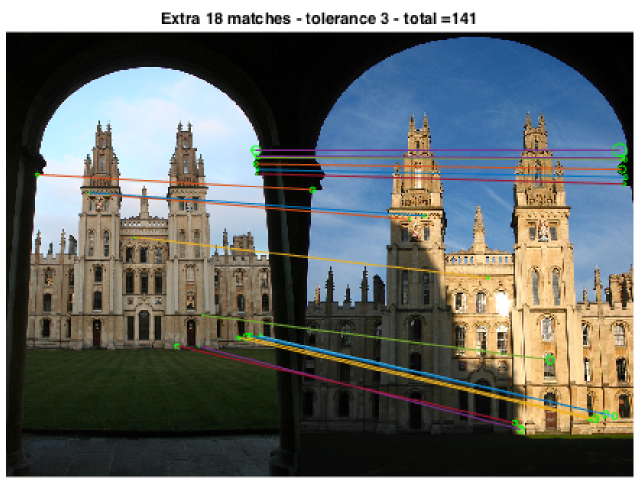
Model	number of candidates	inliers	time
1NN	2048	198	0.2739s
1-2NN	4096	257	0.6947s
1-2-3NN	6144	297	1.2693s

Figure 6: Geometric verification tuning

(a) (20,20,10) - 123 matches



(b) (20,20,50) - 141 matches - *Homography*



(c) (20,50,10) - 114 matches - *Affinity*

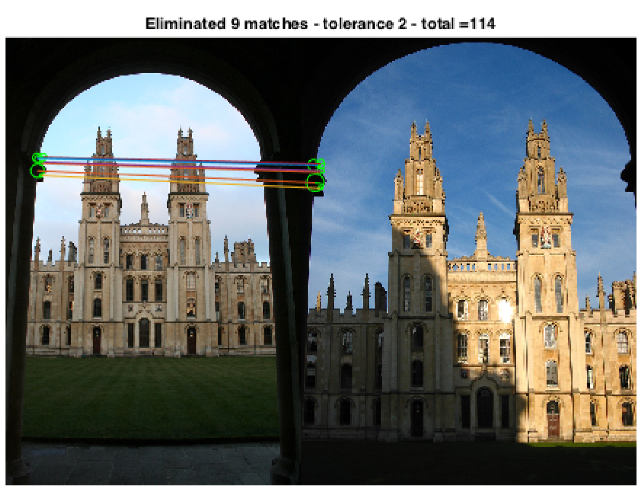
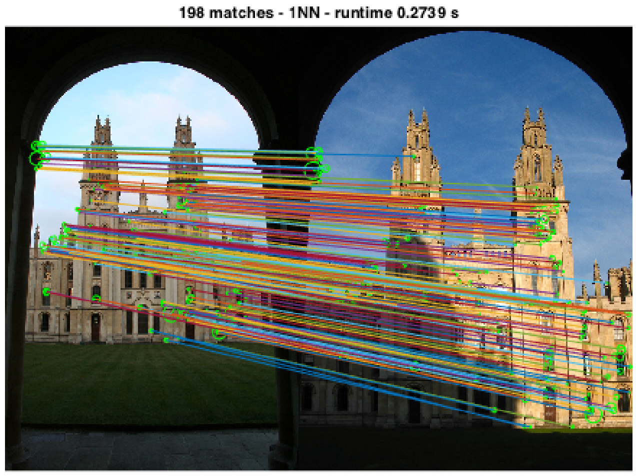
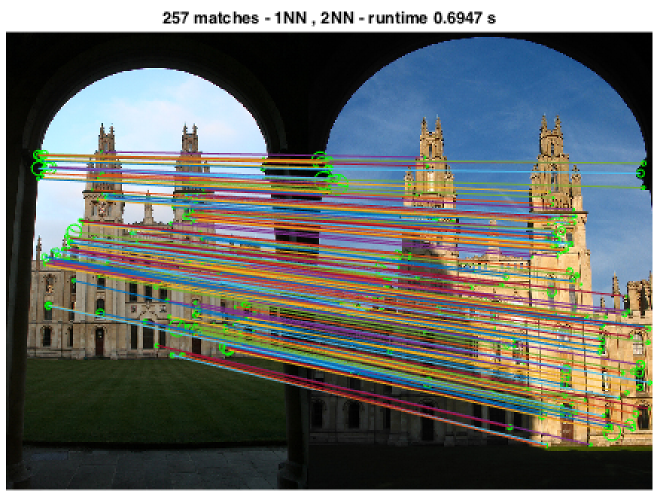


Figure 7: Geometric verification - Inclusion of 2NN and 3NN

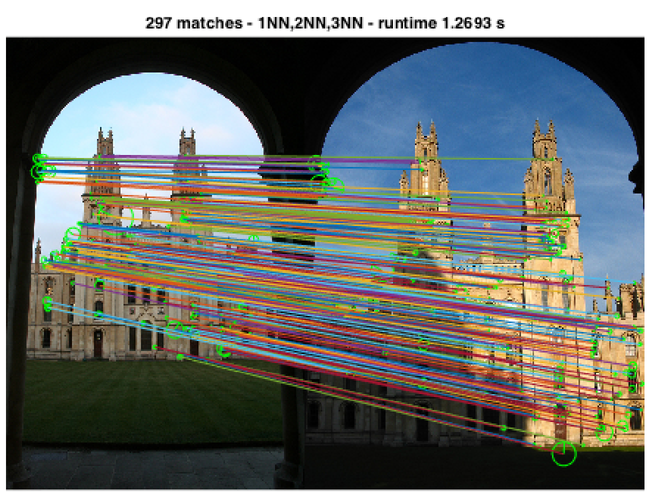
(a) 1NN



(b) 1NN -2NN



(c) 1NN - 2NN- 3NN



2 Affine covariant detectors

QII.1

When matching the six images first with similarity-covariant features then with affine-covariant ones, we note that the similarity matches decreases as the change in viewpoints gets more extreme, to the point of falling behind the affine matches as ilustrated in figure (8). In fact this is due to the dimissal of most of the similarity matches after the geometric verification stage as the circles cannot cover the same areas in the two compared images yielding matches of low quality (figure 9).

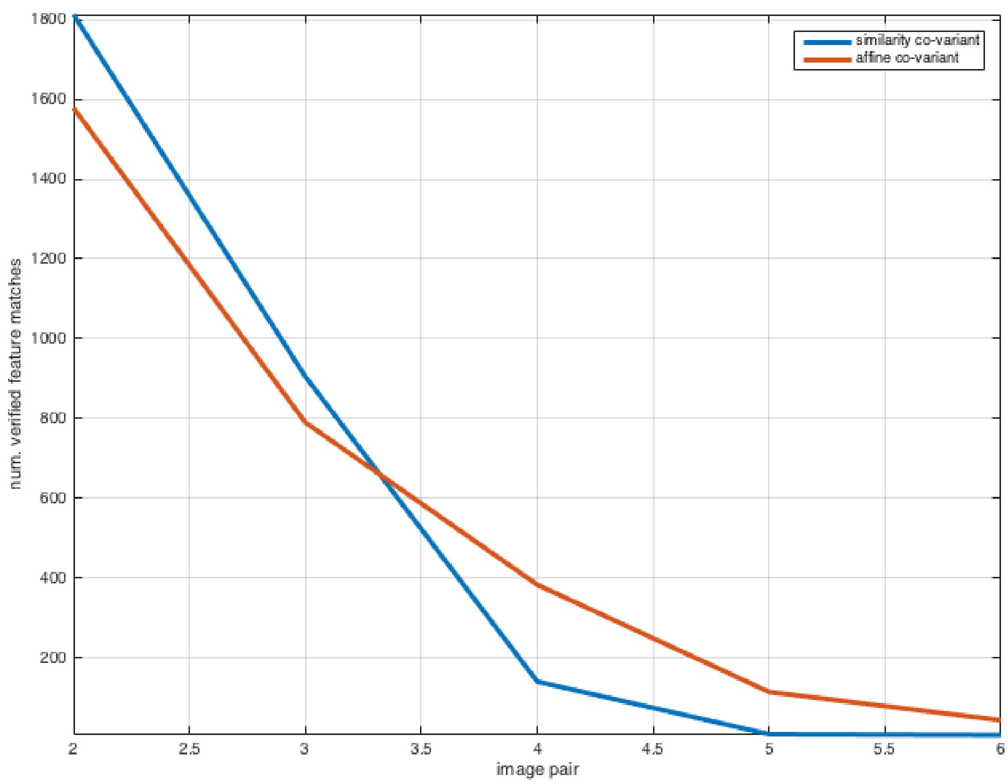


Figure 8: Number of matches with affine v. similarity co-variant features - With geometric verification

A concrete example would be the comparison of the images 1 and 5 (figure 10) where we can say that the affine quality is of $\frac{114}{282} \approx 51\%$ v. $\frac{8}{306} \approx 2.6\%$ for the similarity.

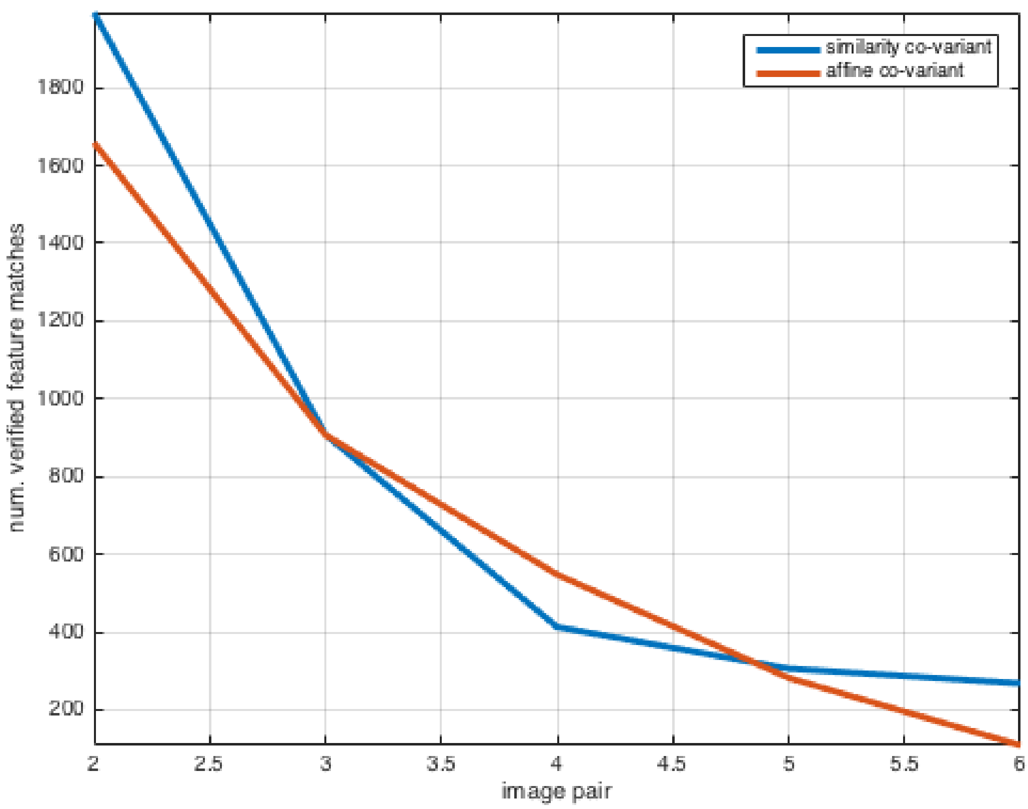


Figure 9: Number of matches with affine v. similarity co-variant features - Without geometric verification

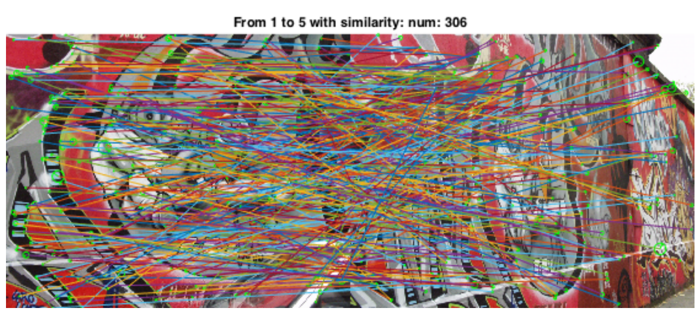
Figure 10: 1-5 affine v. similarity / G-verification v. no G-verification

(a) Affine - G

(b) Affine -noG

(c) Similarty -G

(d) Similarity -noG



3 Towards large scale retrieval

3.1 Accelerating descriptor matching with visual words

QIIIA.1

In practice, the mapping of the descriptors into visual words could be performed offline on large databases, only the query image remains (which takes approximately 1.2s), hence the speed of this step isn't that important.

QIIIA.2

Database size	speedup
10	$0.171 - 1.2 = -1.029$
100	$1.71 - 1.2 = 0.51$
1000	$17.1 - 1.2 = 15.9$

3.2 Searching with an inverted index

QIIIB.1

The top image has a score of 1 because it is the query image we're comparing to the database. A score of 1 means complete similarity between the two unit vectors.

QIIIA.3

We generate more than one match when multiple features are mapped to the same visual word and then run the matches through geometric verification - the results are shown in figure (12).

QIIIB.2

As we can see in figure 13, there are too many erroneously matched images to the query image on the top left corner (16 out of 25).

3.3 Geometric rescoring

QIIIC.1

The new computed score is the maximum of the old cosine similarity and the number of inliers in the geometric verification. This explains why the top score is much larger than 1.

QIIIC.2

After the geometric verification, the correct 9 out of 25 matches score higher than the other matches as seen in figure 15.

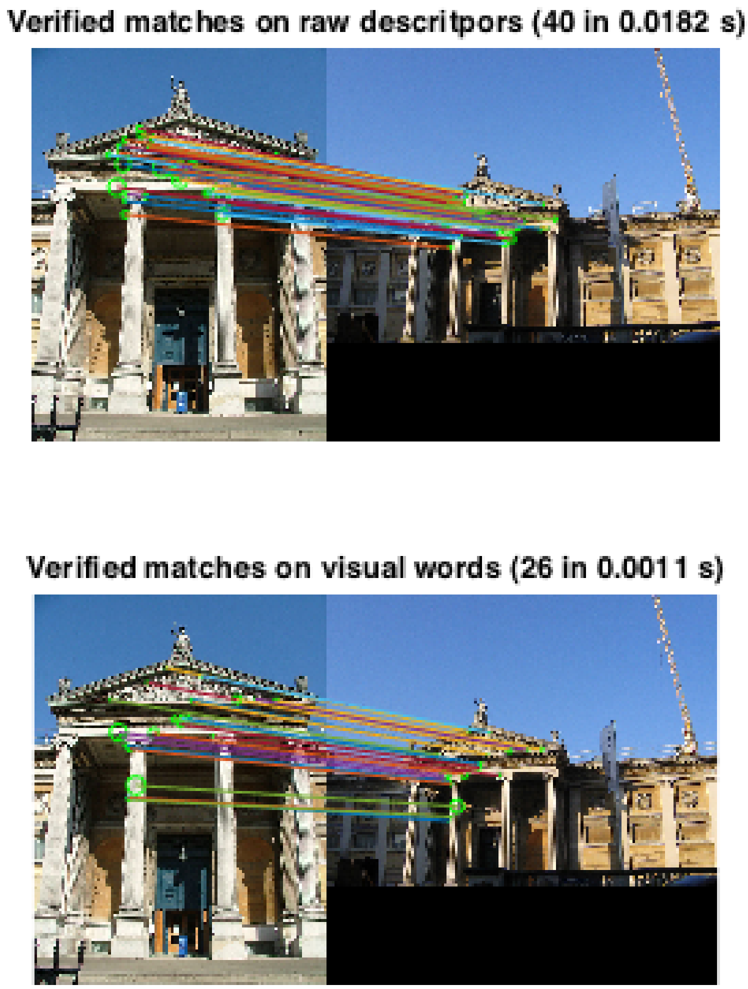


Figure 11: Visual words v. raw descriptors

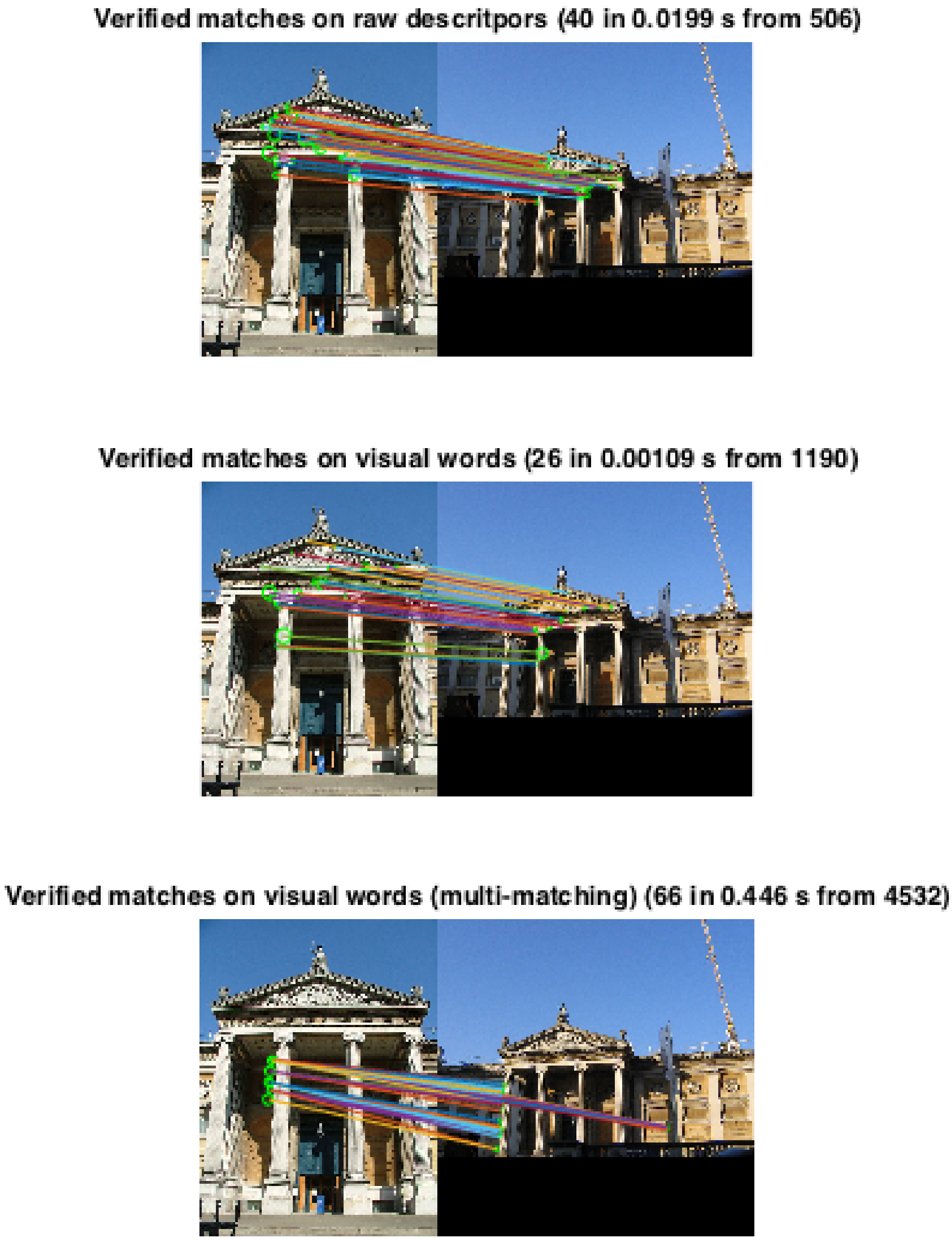


Figure 12: Visual words-1 v. Visual words - multiple v. raw descriptors

Figure 13: Top results with an inverted index

Figure 14: Top results after geometric rescoring

4 Large scale retrieval

QIV.1

The paintings database contains a total number of 9.30M features with an average of 5.36k features per image.

QIV.2

The image database takes 330 MB of memory.

QIV.3

The search is performed as follows:

(a) Loading the query image, computing its features and the corresponding histogram.

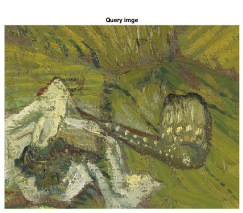
(b) Scoring the other images by similarity to the query.

(c) Running geometric verification.

query	(a)	(b)	(c)
1	0.293 s	0.023 s	0.563 s
2	0.464 s	0.026 s	0.936 s
3	1.916 s	0.033 s	3.422 s

Figure 15: Search examples: 3rd query

(a) Q



(b) Top 4 results