Geometric Verification of Image Copy Detection Results

Stefania Bergljot Stefansdottir
BSc of Computer Science and Discrete Mathematics
Reykjavik University 2016

Advisors
Bjorn Thor Jonsson and Herwig Lejsek
Abstract—Many current image recognition methods are based on identifying interest points in images and considering images a match if they share similar interest points. This does not consider the locations of the interest points, so to reduce the number of false positives, geometric verification is used to identify and reject image pairs that do not fit into an expected geometric pattern. Designing a good geometric verification requires finding a balance between false positives and false negatives, while being time efficient.

The current method used by Videntifier Technologies is based on comparing the angles between interest points. It works well but handles certain transformations incorrectly and is not very efficient.

This report details an attempt to replace it with a method based on the RANSAC algorithm, which involves searching for an underlying transformation between the images that fits most of their matching interest points. After adding some improvements, using the RANSAC method resulted in fewer false negatives, without increasing the number of false positives or reducing time efficiency.

I. INTRODUCTION

Image copy detection is important in many application domains, including multimedia forensics and copyright protection. Typically, in such application, there exists an image collection to protect or compare to, which is indexed by some means suitable for copy detection. When a potential match is considered, it is queried against the index to determine whether it is a copy of one of the images from the collection.

Unless the image is an identical copy, this is a difficult problem, as the query image may have undergone some transformation from its original copy. In natural photography, such transformations may include resizing (moving closer or zooming with a lens), affine transformations (due to different viewpoints, for example), and partial occlusion. With digital images, further transformations are possible, such as shearing, cropping, compression, etc. The typical method to handle such transformations is to use so-called local high-dimensional features, where each feature maps a small portion of the image into a sequence of numbers which is interpreted as a vector in high-dimensional space.

With large image collections, using such local features lead to an even larger collection of high-dimensional vector, so an indexing structure is needed for efficiency. As high-dimensional indexing is an intractable problem, however, the result of querying such an index structure is typically approximate. Some query features may incorrectly yield a matching feature from the collection, while other query features may incorrectly fail to return a match from the feature collection. A secondary method is then needed to analyze the results of the query to determine whether a match from the collection is indeed a match, or whether it is a false positive.

The goal of this study is to analyze and compare two approaches for such secondary analysis of image copy detection results. The image features used are SIFT (Scale Invariant Feature Transform) [1], which are considered the state of the art features for copy detection, and the NV-tree (Nearset Vector tree) [2], which is the most scalable high-dimensional indexing method from the literature. The two methods studied are a simple method based on comparing triangles of matching features, currently used by Videntifier Technologies, and b) the RANSAC method (Random Sample Consensus) [3].

The remainder of the report is organized as follows. We first briefly describe SIFT, the NV-tree, and the overall query process in Section II. Then we outline the two methods under comparison in Section III. In Section IV we present the image sets used for comparing the two methods, and present the performance of the simple method. In Section V, we then consider in detail the parameter settings for the RANSAC method, as well as two efficiency improvements. In Section VI we present compare the results of the two methods and analyze their performance in detail. Finally, we present our conclusions in Section VII.

II. BACKGROUND

When an image is entered into the database the SIFT algorithm is used to identify points of interest. In the original algorithm the features of each point of interest are encoded as a 128-dimensional vector called descriptor, but in the version used here, the descriptor has 72 dimensions. They are then stored in an NV-tree, an efficient high-dimensional index. When querying the database for matches to an image, the NV-tree is searched for descriptors similar to the query image’s descriptors. If enough descriptors have matching descriptors in an image in the database, that image is considered a potential match.

The SIFT algorithm converts an image into a set of interest points. The interest points are identified using scale-space theory and a difference-of-Gaussian function and converted to descriptors. Descriptors are invariant to scale, rotation and translation and partially invariant to changes in illumination and affine transformations. Two descriptors can be compared with each other to determine their similarity and two images are considered a potential match if they have enough similar descriptors.

False positives can occur when matching descriptors, and if enough descriptors are falsely matched between two images, they will be considered a potential match. In order to identify and eliminate these false image matches we can examine the locations of the descriptors in each image, as follows.

If two images depict the same scene, the relationship between their matching points should fit into a geometric transformation such as scaling, translation, rotation, stretching, shearing or a combination of more than one of them. Therefore a real match can be identified by finding an underlying geometric transformation that most of the matching points fit into. If the match was false, there should be no pattern to the locations of the matching descriptors, making it unlikely that a transformation that fits them will be found. The RANSAC method is based on attempting to find this underlying transformation, represented by a matrix and a vector, and accepting the images as a match only if it is found.
III. IMPLEMENTING GEOMETRIC VERIFICATION

A. Existing Method

The existing geometric verification method works by repeatedly and randomly choosing 3 descriptors from image 1 and their corresponding descriptors in image 2. The side lengths of the triangles they form in each image are computed and given a rating based on how similar the ratios of the side lengths are. Afterwards it calculates the average similarity of the 27.5% best ratings and check whether the average is under a given threshold. If so, it is considered a match. The method has been fine tuned to an extent, so things like the number of iterations, the threshold and the rating formula are slightly adjusted depending on the image pair. The method used for choosing the random descriptors also favours descriptor matches with higher similarity.

There are a few possible problems with this method:

- It only looks at 3 descriptor matches at a time. Therefore two images that have multiple small similar areas can get a high rating even though they are fairly different. The method does not make a distinction between such images and images that are overall similar.
- While computation in each iteration but the only information kept afterwards is the similarity rating of the ratios. This is not very efficient.
- The ratios of the triangles are preserved when transformations like rotation, translation and scaling are applied to an image, but not when applying stretching and shearing. When comparing natural photographs this is acceptable but digitally altered images can have any kind of transformation applied. A geometric verification method that recognises these transformations as well would be an improvement.

Note that the algorithm that compares the similarity of the descriptors themselves recognises mildly stretched and sheared versions of the same descriptor as being the same but not heavily stretched and sheared descriptors, which limits the possible improvement of the geometric verification method. However this may be improved in the future, so having a geometric verification method that already handles these cases is considered an improvement even though it cannot be demonstrated in this report.

B. General RANSAC

RANSAC is a method for estimating the parameters of a model based on data points. RANSAC is used on data sets where we expect many of the data points to fit into a specific model whose parameters are not yet known, but there may also be a large number of random/noise data points. The data points that fit the model are called inliers, the rest are outliers.

Each iteration consists of the following steps: Randomly choose a set S of data points, large enough to compute the model parameters. Then compute the parameters of a model M such that all points in S fit into M. Iterate over all points in the data set, checking whether they fit into M, within some error limit. Finally compute the percentage of points that fit into the model.

If the highest percentage meets some predefined threshold, we consider the data to fit the model and use the parameters from that model. If the highest percentage is below the threshold, then we consider the data to not fit any model.

The parameters are the number of iterations, the error limit and the vote percentage threshold.

C. RANSAC for Geometric Verification

Comparing two images involves finding matching descriptors between them. If matches are found between the images, a list of of descriptor matches is returned. The list consists of coordinate pairs, where the first coordinate is the location of a descriptor in image 1 and the second coordinate is the location of a matching descriptor in image 2. A descriptor from image 1 can be matched with multiple descriptors from image 2.

We assume that if two images are a real match then the relationship between the locations of the matching descriptors can be described by an affine transformation. An affine transformation in two dimensions can be described by a $2 \times 2$ real matrix $M$ and a $2 \times 1$ real vector $b$.

That means that if images A and B are a match, then each point $(x_a, y_a)$ in image A has a matching point $(x_b, y_b)$ in image B, so that

$$M \ast \begin{bmatrix} x_a \\ y_a \end{bmatrix} + b = \begin{bmatrix} x_b \\ y_b \end{bmatrix}$$  \hspace{1cm} (1)

In this case a model is a $2 \times 2$ matrix and a $2 \times 1$ vector, a total of 6 different values/parameters. In order to compute the parameters we need 3 sets of matching coordinates.

To test whether a data point (on the form $(x'_a, y'_a)(x'_b, y'_b)$) fits with the model parameters we have derived, we compute

$$M \ast \begin{bmatrix} x'_a \\ y'_a \end{bmatrix} + b = \begin{bmatrix} x'_e \\ y'_e \end{bmatrix}$$  \hspace{1cm} (2)

which are the expected coordinates of the matching descriptor based on the model parameters. Then the euclidean distance between the expected coordinates $(x'_e, y'_e)$ and the actual coordinates $(x'_b, y'_b)$ is computed. If the distance is within the error limit, this data points is considered to fit with the model parameters.

The RANSAC implementation used the same method for picking the random descriptors as the existing version, where descriptor pairs with high similarity were slightly more likely to be chosen.

IV. COMPARISON COLLECTIONS

To test the quality of different geometric verification methods used data sets of similar images. Each data set consists of several subsets, where each subset contains multiple similar images. In most of the data sets the images were real matches, but one of them had false positives. In some cases, the similar images were filtered or geometrically transformed versions of the original image, while in other cases they were natural...
Algorithm 1 Basic RANSAC for geometric verification

1: procedure RANSAC(image1, image2, iterations, votes, distance)
2:   bestVotes := 0
3:   total := the number of matched descriptors in image1
4:   for i from 0 to iterations - 1 do
5:     Randomly choose three descriptors A, B, C
6:     From their matches choose random descriptors a, b, c
7:     Compute the parameters of the model M
8:     yesVotes := 0
9:     for each descriptor D in image1 with a match do
10:        d' := expected location of D's match using M
11:        for each descriptor d that matches D do
12:           if euclideanDistance(d,d') < distance then
13:              yesVotes := yesVotes + 1
14:              break
15:           end if
16:        end for
17:     best := max(best, yesVotes)
18:   end for
19:   return best \geq total * votes
20: end procedure

photographs taken of the same scene. The methods were tested by running them on every pair of images in each subset and counting how many pairs were correctly classified. The following data sets where used for testing:

- **Clusters** contains sets of similar photographs. Most of them are taken outdoors and feature buildings. They are unedited and taken from roughly the same perspective, but the light conditions, proximity to the subject and minor details vary.

- **Scaling And Aspect** contains 2 sets, each features a photograph and many edited versions of it, obtained by scaling, stretching and filtering the original image.

- **StirMark120** contains 120 sets, each featuring one photograph and 15 edited versions of it: rotated, sheared, scaled and filtered.

- **WeakMatches** is a small data set that contains similar but fairly difficult to match pictures.

- **FalsePositives** is a small data set that contains sets of false positives: images with at least one matching descriptors but should not be considered a match. In many cases the false descriptor matches are caused by similar text appearing in the images. For example, two images featuring close ups of street signs with similar fonts may have a large number of matching descriptors. Since these cases are quite different, the results for the text-based and non-text based false positives are shown separately.

An ideal geometric verification method would correctly identify all image pairs in Clusters, Scaling, StirMark and WeakMatches as matches, while rejecting all image pairs in the FalsePositives data set.

Some of the potential matches have fewer than five matching descriptors. Those are considered false positives and are immediately rejected both by the existing method and the RANSAC implementation. Since they are always rejected regardless of the geometric verification method used, we will not consider them when computing the ratio of correctly identified images for each method.

Table I shows the total number of images in each data set, how many subsets of similar images they contain, how many pairs of similar images there are, and how many such pairs have at least five descriptor matches.

Table II shows the quality of the existing method when applied to these data sets. The columns of the table show how many image pairs in each data set are accepted as a match or rejected. The Ratio column shows the percentage of image pairs that are correctly classified by the geometric verification.

V. PARAMETERS AND IMPROVEMENTS

A. Parameters

RANSAC has three parameters: the number of iterations, the vote threshold and the distance threshold. When analyzing a pair of images that are a real match, there should exist an affine transformation that all of their matching descriptors fit into. However there may be some falsely matched descriptors between the images, even though they are a real match. These will not fit the underlying transformation. Additionally, there may be small errors in the coordinates of the descriptors, especially if the images are not very similar. Therefore setting the vote threshold to 100% and the distance threshold close to 0 will result in many real matches being rejected. To avoid wasting time on computing square roots, the algorithm compares the squared distance to the distance threshold. Therefore the value of the distance threshold parameter is actually the square of the allowed distance from the expected locations of the descriptor.

![Table I: Data sets](image-url)

<table>
<thead>
<tr>
<th>Data set</th>
<th>Images</th>
<th>Subsets</th>
<th>Pairs</th>
<th>Pairs with 5+ descriptors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clusters</td>
<td>1082</td>
<td>71</td>
<td>10821</td>
<td>9860</td>
</tr>
<tr>
<td>Scaling</td>
<td>272</td>
<td>2</td>
<td>18360</td>
<td>9275</td>
</tr>
<tr>
<td>StirMark</td>
<td>3120</td>
<td>121</td>
<td>39000</td>
<td>38879</td>
</tr>
<tr>
<td>WM</td>
<td>31</td>
<td>11</td>
<td>38</td>
<td>38</td>
</tr>
</tbody>
</table>

| FP-text   | 23     | 6       | 37   | 37                        |
| FP-other  | 28     | 7       | 91   | 28                        |

![Table II: Performance of existing method](image-url)

<table>
<thead>
<tr>
<th>Data set</th>
<th>Accepted</th>
<th>Rejected</th>
<th>Correct (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clusters</td>
<td>9309</td>
<td>551</td>
<td>94.41</td>
</tr>
<tr>
<td>Scaling</td>
<td>7571</td>
<td>1704</td>
<td>81.63</td>
</tr>
<tr>
<td>StirMark</td>
<td>38844</td>
<td>35</td>
<td>99.91</td>
</tr>
<tr>
<td>WM</td>
<td>36</td>
<td>2</td>
<td>94.74</td>
</tr>
<tr>
<td>FP-text</td>
<td>5</td>
<td>32</td>
<td>86.49</td>
</tr>
<tr>
<td>FP-other</td>
<td>0</td>
<td>28</td>
<td>100</td>
</tr>
</tbody>
</table>


We call the set of descriptors that satisfy the strongest model the "consensus set". To be able to choose reasonable vote and distance thresholds we need to know what the size of the consensus set tends to be in a typical real match, what the size tends to be in a typical false positive, and how the size of the consensus set corresponds to the distance threshold.

To determine this, an adjusted version of the RANSAC algorithm was used. The adjusted version did not have a vote threshold but instead returned the percentage size of the largest consensus set found in 100 iterations. Image pairs with fewer than 5 descriptor matches were not included. The number returned is a lower bound on the true size of the consensus set. This version of the algorithm was used with varying distance thresholds on every image pair in each data set. The distance thresholds used were 0.125, 0.25, 0.5, 1, 2, 4, 8, 16 and 32.

For each data set and distance threshold, this process returned the sizes of the consensus sets for every image pair in the data set (the number of image pairs in the data sets is between a few dozens and tens of thousands). From those numbers, the values at each percentile were computed. For example, when running this on the Scaling And Aspect data set with a distance threshold of 4, the image pair with the smallest consensus set had a set of size 0.238, the pair at the 5% percentile had a set of size 0.42156, the median image pair had a consensus set of 0.7391, the pair at the 95% had a set of size 0.9736 and the pair with the largest set had a size of 1.

This means that if RANSAC were used on the Scaling And Aspect data set with the parameters iterations = 100, votes threshold = 0.7391 and distance threshold = 4, the method would accept half the images as a match and reject half (excluding the ones with less than 5 descriptor matches). Generally, if the image at the k-th percentile has a consensus set of size c when using distance threshold d, then running regular RANSAC with distance threshold d and vote threshold c should result in 1 − c of the data set being identified as a match.

The graphs in Figure 1 show the results for all the data sets and distance thresholds used. The x axis is the distance threshold, the y axis is the size of the consensus set found and the curves represent the values at certain percentiles. In the graphs the curves are labelled with the inverted percentile, which corresponds to the percentage of images that are accepted when those parameters are used. For example, if we want to know what kind of parameters to choose in order to accept to at least 90% of the images in a certain data set, we would look at the curve labelled with 90% and choose a distance and vote threshold that falls on or below that curve. Those parameters should result in at least 90% of the data set being classified as a match.

However, these numbers are based on 100 iterations of RANSAC, which is too slow for general use. Therefore, when running RANSAC with fewer iterations, the number of accepted image pairs is likely to be slightly lower than the graphs suggest.

The results for the data set FP-other is not shown here because the number of matching descriptors is too low between the images. For example, if two images share 5 descriptors, the consensus set will always have a size of at least 0.6. See the section on scaling the vote threshold for details.

The graphs for the real match data sets are fairly different from one another. For example, the 99% curve is somewhat higher in Clusters than in Scaling And Aspect. This indicates that Clusters has fewer matches that are difficult to handle than Scaling And Aspect. Additionally, the curves in Scaling And Aspect and StirMark do not even out as quickly when using higher distance thresholds. These data sets include geometrically transformed images while the other two do not, so this could mean that there is more error in the locations of the descriptors when working with geometrical transformations.

The curves FalsePositives-text are much lower than in the real match data sets. The reason for this is that the descriptor matches between the images do not have an underlying geometric transformation. Instead their locations are mostly random. The 1% curve is surprisingly high. It is caused by one pair of images that feature very different scenes but have a large, nearly identical watermark. Since almost all of the matching descriptors between them are on the watermark, the consensus set found is very large. It is probably difficult to fine-tune the geometric verification method to reject pairs like these without negatively impacting the overall performance.

The graphs can be used to estimate the results of choosing certain parameters. For example, the intersection of distance threshold = 8 and voting threshold = 0.4 is close to or slightly below the 99% curve in all of the real match data sets, and close to the 10% curve in the FP-text data set. That means that running RANSAC with those parameters and 100 iterations should result about 99% of the real matches and 10% of false positives being accepted, which would be an improvement over the existing method in most of the data sets.

B. Varying Vote Thresholds

The existing algorithm considers two images to be a potential match if there are at least five descriptor matches between them. Since the RANSAC algorithm computes the model parameters from three matching descriptors, a pair of images with five descriptors matches are guaranteed to have a consensus set of at least 0.6.

If such a pair has a consensus set of 0.6 it is very unlikely that the pictures are a match. However if a pair of pictures with images of matching descriptors has a consensus set of 0.6 it is very likely that they are a match. Therefore it is necessary to scale the vote threshold so that images with few matching descriptors are held to a higher standard than those with many.

A straightforward way of scaling the vote threshold would be to subtract three from the consensus set and the total data set when computing the size of the consensus set. Example: if two images have 10 matching descriptors and the algorithm finds a consensus set of size 7, we consider that to be a consensus of \((7 - 3)/(10 - 3) = 0.57\) instead of 0.7. If the images had 100 matches and a consensus set of 70, the consensus would be \((70 - 3)/(100 - 3) = 0.69\). That way the vote threshold for
Figure 1: The relationships between the distance threshold and the size of the consensus set in different data sets.
images with many matches will remain relatively unchanged while images with few matches will have a significantly higher threshold.

Looking at the False Positives data set using the new version of the votes threshold and distance threshold, the highest consensus found for the non-matches was 0.5 [except for two text-based pairs that are very similar, talk about them elsewhere].

Looking at the figures from the previous section, we can estimate that with these parameters the algorithm would reject at least 1% of the matches in Clusters, 5% of the matches in Scaling and Aspect and 1-5% of the matches in WeakMatches. While many of the matches that failed under those parameters had few matching descriptors, some of them had 50 - 100 matching descriptors and consensus sets of over 0.45. Ideally the algorithm would accept such matches. Therefore even stronger scaling is needed.

An alternative is to adjust the vote threshold with the formula:

\[
\nu' = \nu + (1 - \nu)/(d - 4)
\]

where \(\nu'\) is the new threshold, \(\nu\) is the original vote threshold and \(d\) is the number of descriptor matches. This results in a very high threshold for images with very few matches, but decreases faster than the −3 method. Figure 2 shows the difference between these two scaling methods when the unscaled vote threshold is set to 0.5.

To test the effect of scaling the vote threshold, both versions on RANSAC were tested on the False Positive data set and a subset of both ScalingAndAspect and StirMark. 20 iterations and different vote and distance thresholds were used. Using a scaled vote threshold should result in a lot fewer image pairs in FalsePositives being incorrectly identified as a match, since most of its image pairs share few descriptors between them, which will result in the vote threshold being significantly increased. ScalingAndAspect and StirMark should both see a small decrease in the number of image pairs identified as real matches, because real matches tend to have many descriptors in common, so the scaling factor is small.

Figure 3 shows the result of running RANSAC with and without scaling on these data sets. The non-scaled graphs show that it is impossible to choose parameters that result in both most of the false positives being rejected and most of the true positives being accepted. As expected, the scaled version accepts much fewer of the false positives, while accepting only slightly fewer pairs from the other data sets. This shows that scaling the vote threshold allows us strongly reduce the number of false positives at the expense of slightly increasing the number of false negatives.

### C. Early Stops

In the given implementation of RANSAC, each iteration consists of generating the model parameters and testing all pairs of matching descriptors to see if they fit the model parameters. The time complexity is thus the number of iterations times the number of matches. In some iterations all of the three pairs will be inliers, so the model parameters should fit most of the other inliers and it will receive a high number of votes. In other iterations one or more of the matches chosen will be an outlier so the model parameters generated will be inaccurate. Then only a small number of the matches will fit into the model by random chance. The time efficiency of the algorithm can be improved by detecting the latter type of iterations and terminating them early.

To do this we need to define some condition on which to stop the iteration. Let \(C\) be the event that the condition is true and \(G\) be the event that three inliers were chosen. Ideally \(P(G|C)\) should be low in order to minimize the chance of terminating good iterations, but \(P(C)\) should be high enough that many iterations will be stopped so the time improvement will be significant. Computing the condition also needs to be fast.

\[
P(\neg G|C) = \frac{P(C|\neg G) * P(\neg G)}{P(C|\neg G) * P(\neg G) + P(C|G) * P(G)}
\]

Let \(c\) be the ratio of the descriptor matches belonging to the consensus set and \(total\) be the number of descriptor matches. Since computing the model parameters requires 3 matches, the probability of choosing only inliers for the model parameters is \(P(G) = c^3\). Then

\[
P(\neg G|C) = \frac{P(C|\neg G) * (1 - c^3)}{P(C|\neg G) * (1 - c^3) + P(C|G) * c^3}
\]

As an example, let us choose having the first three votes be negative as the condition. Then \(P(G) = (1 - c)^3\), because method would have to randomly pick three descriptors not belonging to the consensus set. \(P(C|\neg G)\) is difficult to estimate but it should be most \((1 - \frac{3}{total})^3\), because there must be at least three descriptors that fit this model.

Assume that an image pair has 60 matching descriptors, out of which 45 are inliers. Then \(P(C) = 0.5023\). Therefore this condition would occur about half the time, so about half
Figure 3: RANSAC results, with and without scaling
87.56% of the matches in ScalingAndAspect, while accepting RANSAC-es accepted 97.69% of the matches in Clusters and it may slightly reduce the quality of the method. In this case, iteration will meet the condition and be terminated, so using improvement in the time efficiency. Choosing a different condition could lead to a more significant analysis of the data sets and the method. It is possible that are all negative, was a guess instead of being based on any stop condition used here, checking if the first three votes using early stops is an improvement or not. However the differences here are not large enough to conclude whether

the iterations would be cut short. Furthermore, $P(\neg G|C) = 0.9869$, so the likelihood of losing a good model would be very low.

### VI. Comparison

#### A. Quality

The results of using RANSAC on the data sets are shown in Table III. The parameters used were 20 iterations, distance threshold of 8 and vote thresholds of 0.4 with scaling. RANSAC performed equally well or better than the existing method in all the data sets except StirMark. The largest improvement was in ScalingAndAspect, which includes many stretched images which are difficult for the existing method to handle.

Table IV shows the time needed by the different methods to classify the data sets. Only the time spent on the geometric verification method was counted. RANSAC is slower at handling the false positives, but is faster with the other data sets. The table shows that on the largest timed data set, Clusters, RANSAC was over 9% faster than the existing method.

The effect of using early stops with RANSAC (RANSAC-es) was also measured. It was slower than regular RANSAC on the Clusters data set but slightly faster on the others. The differences here are not large enough to conclude whether using early stops is an improvement or not. However the stop condition used here, checking if the first three votes are all negative, was a guess instead of being based on any analysis of the data sets and the method. It is possible that choosing a different condition could lead to a more significant improvement in the time efficiency.

When using early stops, there is a chance that a good iteration will meet the condition and be terminated, so using it may slightly reduce the quality of the method. In this case, RANSAC-es accepted 97.69% of the matches in Clusters and 87.56% of the matches in ScalingAndAspect, while accepting

the same as regular RANSAC in WeakMatches and FalsePositives.

#### B. Shearing and Stretching

As discussed earlier, the existing method does not accept heavily sheared or stretched images as matches, but we expect RANSAC to handle those transformations easily. To test if there was a difference in how the methods performed, a new dataset was created. It contained a photograph from the StirMark data set along with multiple sheared versions of it. The matrix used for the transformation was

$$
\begin{bmatrix}
1 & 0 \\
-n & 1
\end{bmatrix}
$$

where $n$ took the values 0.01, 0.02, 0.03, ..., 0.99, 1.00. For each value of $n$, a new sheared image was generated, resulting in a data set containing 101 images and 5050 image pairs. Shearing with $n = 1$ results in a 45 between the bottom and left sides of the image.

The data set was tested with the existing method, with RANSAC with and without scaling. Additionally, it was tested using pure SIFT where no geometric verification is performed, and all image pairs with enough matching descriptors are accepted. If our hypothesis is correct RANSAC should accept almost all matches that pure SIFT accepts, while the existing method should accept significantly fewer matches.

Table V shows the results of running the different methods. The RANSAC methods had 20 iterations, distance threshold 8 and votes threshold 0.4. RANSAC without scaling accept all except 10 of the image pairs accepted by SIFT. The failures might be caused by false positive descriptor matches, or by by increased error in the locations of the descriptors. RANSAC with scaling accepts fewer but still outperforms the existing method.

I also tested the methods by using stretching and shearing on 10 different images from the StirMark data set using the same shearing transformation as above and

$$
\begin{bmatrix}
1 & 0 \\
0 & 1 + n
\end{bmatrix}
$$

for the stretching, where $n$ took the values of 0.01, 0.02, 0.03, ..., 0.99, 1.00, where $n = 1$ doubles the image’s height. In this case, each of the 100 images was compared with the original version of it. Figure 4 shows how the different methods performed. The x axis shows the level of stretching and shearing, and the y axis shows for how
many of the 10 images the images were correctly identified as a match.

As expected, the overall the quality goes down for all methods as the level of stretching is increased.

Figure 5 show examples of shearing. This image is the one used in the shearing and stretching data set. Both the existing method and RANSAC correctly identify the original image and the mildly sheared one as being a match. However only RANSAC recognizes the original and the moderately sheared one as being a match. When comparing the original and the heavily sheared image, the descriptors have been distorted too much for SIFT to recognize the descriptor matches between the images. For that reason both methods fail to recognize them as a match. A stretched version of this image can be seen in figure 6. Like with the shearing, the mildly stretched image is recognized by both RANSAC and the existing method, the moderately stretched one is only recognized by RANSAC and the heavily stretched one is recognized by neither.

This shows that using RANSAC is a great improvement over the existing method when working with stretched and sheared images.

VII. CONCLUSIONS AND FUTURE WORK

Using RANSAC for geometric verification resulted in fewer false negatives in most of the data sets, without increasing the number of false positives or sacrificing time efficiency. As expected, the improvement was especially big for image pairs generated by shearing and stretching.

Scaling the vote threshold upwards when the descriptor matches are few improved the RANSAC method significantly, by making it stricter on image pairs that are unlikely to be a real match. The other attempted improvement, stopping bad iterations early, did not lead to a conclusive improvement in the time efficiency. However the stopping condition that was tested was chosen as a guess, rather than being based on analysis of the data sets, so it is possible that a better stopping condition could lead to more time improvement.

One possible additional improvement is to weight the votes so that descriptor matches that are highly similar count for more than matches that are only mildly similar.

REFERENCES

Figure 5: Shearing Example

(a) An image from the Stirmark data set
(b) Shearing with $n = 0.35$
(c) Shearing with $n = 0.55$
(d) Shearing with $n = 0.85$

Figure 6: Stretching Example

(a) Stretching with $n = 0.35$
(b) Stretching with $n = 0.65$
(c) Stretching with $n = 0.85$