Decade of Lost Precision: A General Framework For Image Feature Matching Without Geometric Constraints

Jonas Toft Arnfred[a], Stefan Winkler[a]

[a] *Advanced Digital Sciences Center (ADSC), University of Illinois at Urbana-Champaign (UIUC), Singapore*

Abstract

Computer vision applications that involve the matching of local image features frequently use *Ratio-Match* as introduced by Lowe and others, but is this really the optimal approach? We formalize the theoretical foundation of *Ratio-Match* and propose a general framework encompassing *Ratio-Match* and three other matching methods. Using this framework, we establish a theoretical performance ranking in terms of precision and recall, proving that all three methods consistently outperform or equal *Ratio-Match*.

We confirm the theoretical results experimentally on over 3000 image pairs and show that we can increase matching precision by up to 30 percentage-points without further assumptions about the images we are using. These gains are achieved by making only a few key changes of the *Ratio-Match* algorithm and do not increase computation times.

Keywords: Feature matching, ratio match, mirror match, self match

1. Introduction

Matching image points is a crucial ingredient in almost all computer vision applications that deal with sparse local image features, such as image categorization [1], image stitching [2], object detection [3], and near duplicate detection [4], to mention just a few examples. All of these rely on accurately finding the correspondence(s) of a point on an object in a *query image* given one or more *target images* that might contain the same object. In many applications the

target images have undergone transformations with respect to the *query image*; in stereo vision, the viewpoint is different, while in object recognition and near duplicate detection both the lighting and even the object itself may also be transformed.

In the literature two approaches to feature point matching have been pursued and later merged, namely the *geometric approach* and the *descriptor-centric approach*.

In the purely geometric approach, feature points are matched based on their location in the images. Scott and Longuet-Higgins [5] and Shapiro and Brady [6] introduced the use of spectral methods by deriving a coherent set of matches from the eigenvalues of the correspondence matrix. Other examples of this approach include [7, 8].

The descriptor centric approach on the other hand finds matches by pairing similar keypoints. The first examples of this approach used the correlation of the raw image data immediately surrounding the feature point [9, 10] to calculate this similarity. Later algorithms were enhanced by invariant feature descriptors as first introduced in [11] and later popularized by the work of Lowe introducing SIFT [12] and Bay et al. introducing SURF [13].

Many descriptor variations have since been proposed, and a few surveys have emerged comparing them. Mikolajczyk and Schmid [14] evaluate 11 such feature descriptors using images of planar scenes. Moreels and Perona [15] evaluate 5 descriptors with different keypoint detectors using a large image set consisting of 3D models. Heinly et al. [16] take a closer look at binary features such as BRIEF [17], BRISK [18], and ORB [19], comparing them to SIFT and SURF.

A straightforward way to find a set of correspondences using only feature points is to apply a threshold to the similarity measure of the feature vectors, accepting only correspondences that score above a certain level of similarity [20]. When we match images with the assumption that the correspondence between two feature points will be unique, we can further increase precision by only matching a feature point to its nearest neighbor in terms of descriptor similarity. Instead of thresholding based on similarity, Deriche et al. [9] and

Baumberg [10] propose using the *ratio* of the similarity of the best to second best correspondence of a given point to evaluate how unique it is. Their finding has later been tested by several independent teams, all concluding that thresholding based on this ratio is generally superior to thresholding based on similarity [12, 14, 15, 21].

Brown and Lowe [22] extend ratio match to deal with a set of images by using not the ratio of the best and second best correspondence, but the average ratio of the best and the second best correspondences across a set of images. Rabin et al. [21] try to enhance descriptor matching by looking at the statistical distribution of local features in the matched images, and only return a match when such a correspondence would not occur by mere chance. Finally, a precursor of the algorithms discussed in this paper was introduced by the authors as *Mirror-Match* [23], which makes use of the feature points in both images to decide if a match is valid.

A plethora of hybrid solutions have combined descriptor matching with various geometric constraints to improve matching. These constraints are based on assumptions regarding the transformation between the query and *target images*. At the stricter end we have epipolar constraints, assuming that the images can be tied by a homography [24, 25], and angular constraints, assuming the correspondences are angled similarly [26, 11]. Often these approaches are made computationally feasible by modeling feature correspondences as an instance of graph matching, where each feature is a vertex, and edge values correspond to a geometric relation between two features. Approximate graph matching algorithms can then be used to efficiently establish an isomorphism between the graph of features in two images [27, 28, 29, 30]. Others define image regions and reject or accept correspondences based on the regions they connect [31, 32].

Any matching method relying on geometric constraints is limited by inherent assumptions about the geometric relationship between the two images. Broad assumptions such as the epipolar constraint only apply in simple image transformations. For more complex transformations we need models suitable for each particular case, which limits the algorithm to the subset of images that fit the

model. In the case of object recognition for example, the transformations from one scene to another often feature a change in perspective, background, and sometimes variations within the object itself: a person can change pose, a car model can have different configurations, a flower can bloom etc.

When matching these instances we are forced to either create a sophisticated model that represents the variables of transformation within the object, or alternatively find correspondences using an algorithm with no inherent geometric assumptions. Besides, any geometric method acts as a filter on a given set of correspondences. Therefore, if the initial set of purely descriptor-based matches contains fewer incorrect correspondences, the final set can be calculated faster and more accurately.

The methods we propose in this paper are designed to be free from assumptions about image geometry. They extend and improve on Lowe's *Ratio-Match* [12] and the authors' *Mirror Match* [23] by generalizing both algorithms to a framework of matching methods. We go on to formally establish a ranking based on how well different methods within the framework compare in terms of precision and recall. Our experimental evaluations confirm the theoretical results and show that *Ratio-Match* on all counts is generally a sub-optimal choice as a matching algorithm.

In our previous paper [23], we introduced *Mirror-Match* and *Mirror-Match with Clustering*, two algorithms that outperform the state of the art. The novel contributions of the present paper consist of presenting these algorithms together with several related existing algorithms in a general and comprehensive framework. We further develop the theoretical foundations for comparing the algorithms and use these to formally prove a ranking in terms of performance for the different algorithms. This also enables us to understand why *Mirror-Match* performs better than *Ratio Match* in the first place. In addition we benchmark all algorithms within the framework extensively on a much larger dataset containing over 3000 image pairs.

The paper is organized as follows. In Section 2 we introduce the original *Ratio-Match*, extend on this method to introduce the proposed framework and

go on to compare the members of the framework theoretically. In Section 3 we present evaluations and discuss the results obtained. Section 4 concludes the paper.

2. Matching Framework

2.1. Definitions

The proposed framework is inspired by *Ratio-Match* as introduced by Deriche et al. [9] and later used by Baumberg [10] and Lowe [12]. *Ratio-Match* is motivated by the observation that nearest neighbor feature matching is not necessarily the best strategy as illustrated by [12, 14]. The distance between the feature descriptors of two nearest neighbors might tell us on a global level how much they resemble each other, but it does not tell us if other feature points are equally similar. *Ratio-Match* makes use of the ratio between the nearest and second nearest neighbor as a heuristic to determine the confidence of the match. Matches are returned only if this ratio is lower than a given threshold τ, filtering out feature points that are ambiguous because others match almost equally well.

The underlying assumption in ratio match is that the point we seek to match in the *query image* has only one true correspondence in a given *target image* or no matches at all. In both cases we can infer that the *second* nearest neighbor in the *target image* is not a true correspondence. We look at the distance between the second nearest neighbor and the feature point as a *baseline*. It tells us how similar the descriptors of two feature points can be when they are not a true correspondence. Some feature points might have very unique descriptors with large distances to false correspondences, while others may be generic with plenty of similar points. Knowing the baseline for all features allows us to be lenient in the first case and cautious in the second. In practice *Ratio-Match* scores a match by dividing the distance to the nearest neighbor with the distance to the second nearest neighbor (the baseline) to estimate how distinct the correspondence is from a false match.

In what follows we will use the following nomenclature:

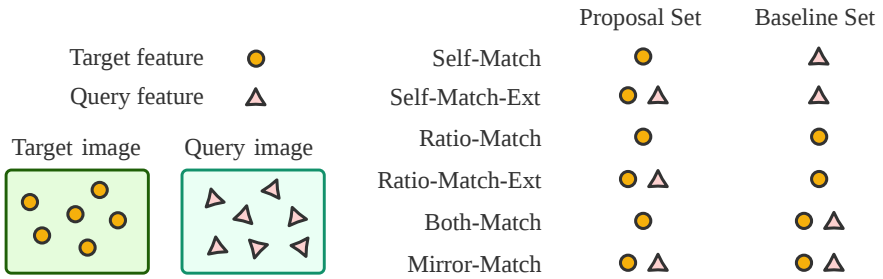


Figure 1: Graphical representation of the *baseline set* and *proposal set* for different methods in the proposed framework.

- Let f_q be a feature point in the *query image*.
- Let F be a set of features. F_t denotes all features from the *target image*.
- Let $\tau \in [0 \dots 1]$ be a threshold used to decide whether to keep a match.
- Let the *proposed match* be the nearest neighbor of a query feature f_q picked from a set of feature points that we will call the *proposal set*, which does not contain the query feature.
- Let the *baseline match* be the nearest neighbor of the query feature f_q picked from a set of feature points that we will call the *baseline set*, which contains neither the *proposed match* nor the query feature. In *Ratio-Match* the *baseline match* is the second nearest neighbor.

2.2. A framework of matching methods

We can generalize *Ratio-Match* by expanding on the idea of a *baseline* and *proposal set*. With *Ratio-Match* these two sets are created from features in the *target image*, but they need not be. If we use the features in the *query image* as well as the combined features of both images, we end up with six possible permutations of a *Ratio-Match*-like algorithm. We illustrate these variants in Figure 1 and will go on to both prove theoretically and demonstrate empirically that *Ratio-Match* is among the least performant of the pack.

The algorithms *Ratio-Match*, *Self-Match* and *Both-Match* all find the best match to a given *query feature* only in the *target image*. They differ by the

feature set used as the *baseline set*. While *Ratio-Match* uses features from the *target image*, *Self-Match* draws the *baseline set* from the *query image*. Finally *Both-Match* uses the conjunction of features from both images.

Each of these three algorithms has an 'extended' case where the proposal set is replaced with the conjunction of features from both the *target image* and the *query image*. The extended version of *Self-Match* is *Self-Match-Ext* and the extended version of *Ratio-Match* is *Ratio-Match-Ext*. Finally we call the extended version of *Both-Match* for *Mirror-Match* because it was under this name that the algorithm was originally introduced by the authors in [23].

We will not discuss the *Self-Match-Ext* and *Both-Match* variants in this paper since we prove them to be equivalent with *Self-Match* and *Mirror-Match* respectively in Appendix A and Appendix B.

Algorithm 1 Generalized matching algorithm for two images

Require: I_{query}, I_{target} : images, $\tau \in [0, 1]$
  $F_q = get\_features(I_{query})$
  $F_t = get\_features(I_{target})$
  $F_{proposal-all} = get\_proposal\_features(F_q, F_t)$
  $F_{baseline-all} = get\_baseline\_features(F_q, F_t)$
  $M = \varnothing$
  for all $f_q \in F_q$ **do**
    $F_{proposal} = F_{proposal-all} \setminus \{f_q\}$
    $f_p \leftarrow getNearestNeighbor(f_q, F_{proposal})$
    $F_b = F_{baseline-all} \setminus \{f_q, f_p\}$
    $f_b \leftarrow getNearestNeighbor(f_q, F_b)$
    $r \leftarrow distance(f_q, f_p)/distance(f_q, f_b)$
    if $(r < \tau) \wedge (f_p \in F_t)$ **then**
      $matches \leftarrow matches \cup (f_q, f_p)$
    end if
  end for
  return M

Algorithm 1 shows the generalized matching framework, where the *proposal set* and *baseline set* are defined according to Figure 1. Figure 2 illustrates the structure of this algorithm.

For some methods it is possible that the *proposal match* is a feature from the *query image*, in which case we discard the match as a false correspondence.

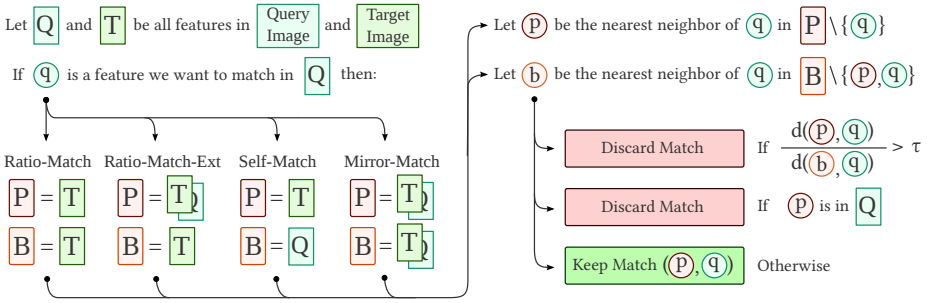


Figure 2: Feature matching flow chart. τ is the ratio threshold, and $d(x, y)$ is the distance between two feature descriptors x and y.

It also happens that we encounter correspondences with $r > 1$, in which case the match is also discarded. Take for example the case of *Self-Match* where we might find that the nearest neighbor of a feature in the *target image* is further from the query feature than the nearest neighbor in the *query image*. In this case the match is discarded.

2.3. Uniqueness Ratio

The main distinguishing factor between the algorithms in the framework is the final ratio between the distance of the two nearest neighbors of a query feature, which determines if we keep or discard a match. We illustrate this process in Figure 2.

We can define a calculation of this ratio that is common to all algorithms in the framework, which allows us to compare the algorithms theoretically. To do so, we introduce the *uniqueness ratio* r, based on the concept of a nearest neighbor.

Given a feature f_i and a set of features F, the nearest neighbor of f_i in F is calculated as follows:

$$\arg\min_{f_j \in F} d(f_i, f_j)$$

Here, $d(f_i, f_j)$ is the distance between feature descriptors. With SIFT and SURF this is the Euclidean distance of the feature vectors [12, 13], whereas BRIEF, BRISK, and FREAK use the Hamming distance [18, 17, 33].

Let $f_p \in F_p$ and $f_b \in F_b$ be the nearest neighbors of a feature f_q in F_p and F_b. The *uniqueness ratio* is defined as follows:

$$\begin{aligned} r &= \mathrm{r}(f_q, F_p, F_b) \\ &= \frac{d(f_q, f_p)}{d(f_q, f_b)}. \end{aligned}$$

The performance in terms of precision and recall of any algorithm in the proposed framework is uniquely identified by the *uniqueness ratio* r. To show this, let K be the number of possible true correspondences between *query* and *target image*. *Precision* and *recall* are defined as:

$$\begin{aligned} \text{Recall} &= \frac{\#\text{Correct}}{K} \\ \text{Precision} &= \frac{\#\text{Correct}}{\#\text{Correct} + \#\text{Incorrect}} \end{aligned}$$

Given $f_p \in F_p$ as the nearest neighbor of some $f_q \in F_q$, we define the set of features F_{true} as $\{f_q \in F_q \mid f_p \text{ is a true correspondence of } f_q\}$ and F_{false} as $F_q \setminus F_{true}$. We can then define the number of correct and incorrect matches as follows:

$$\begin{aligned} \text{keep\_match}(f_q) &= \begin{cases} 1 & \text{if } \mathrm{r}(f_q, F_p, F_b) < \tau \\ 0 & \text{if otherwise} \end{cases} \\ \#\text{Correct} &= \sum_{f_q \in F_{true}} \text{keep\_match}(f_q) \\ \#\text{Incorrect} &= \sum_{f_q \in F_{false}} \text{keep\_match}(f_q) \end{aligned}$$

If for every query feature f_q, r is identical across two matching methods, then they return identical results.

2.4. Proofs of Algorithm Performance

Under the following three assumptions, which largely reflect the performance of matching feature points in practice, we can theoretically compare the performance of the different algorithms shown in Figure 1:

1. For any point in the *query image* there is at most one real correspondence in the *target image* and no real correspondences in the *query image* itself, as assumed by *Ratio-Match*.

2. The distance between two feature descriptors within the *query image* is larger than their distance to a real correspondence in the *target image*. More precisely, given f_q, a feature from the *query image* for which a real correspondence, f_{match}, exists in the *target image*, we assume that $\forall f_i \in F_q : d(f_q, f_{match}) < d(f_q, f_i)$.
3. For a set of query features F_q with a true correspondence in the *target image*, the distribution of *uniqueness ratios* using F_t as the baseline set is similar to using F_q, since the two images in the case of true correspondences are bound to share part of the same scene.

Based on these assumptions we prove that *Ratio-Match-Ext* is equal to or better than *Ratio-Match* in terms of both precision and recall. Consider the nearest neighbor f_p of a query feature f_q and the two cases where we either have $f_p \in F_q$ or $f_p \in F_t$. For $f_p \in F_t$ the uniqueness ratio of *Ratio-Match-Ext* is:

$$\begin{aligned} r &= \mathrm{r}(f_q, F_p, F_b) \\ &= \mathrm{r}(f_q, F_q \cup F_t, F_b) \\ &= \mathrm{r}(f_q, F_t, F_b) \end{aligned}$$

Since $F_p = F_t$ for *Ratio-Match*, when the nearest neighbor is found in the *target image*, the two algorithms behave identically. For $f_p \in F_q$ *Ratio-Match* gives us the following ratio:

$$\begin{aligned} r &= \mathrm{r}(f_q, F_p, F_b) \\ &= \mathrm{r}(f_q, F_t, F_t). \end{aligned}$$

That is, *Ratio-Match* calculates the ratio based on the two nearest correspondences in the *target image*. *Ratio-Match-Ext* on the other hand does not return any correspondence, because the nearest neighbor is in the *query image*. Since query features with a nearest neighbor in the *query image* are false correspondences per assumption #1 and #2, this proves that *Ratio-Match-Ext* has superior *Precision* to *Ratio-Match* while maintaining equal *Recall*.

Next we show that *Mirror-Match* is equal or better than *Ratio-Match-Ext* in terms of precision and recall. Consider as before the nearest neighbor f_p of a query feature f_q. When f_p resides in the *query image*, both algorithms behave alike and discard the match. However, consider the uniqueness ratio of *Mirror-Match* for the case where f_p resides in the *target image*:

$$\begin{aligned} r &= \mathrm{r}(f_q, F_p, F_b) \\ &= \mathrm{r}(f_q, F_p, F_q \cup F_t) \\ &= \max(\mathrm{r}(f_q, F_p, F_t), \mathrm{r}(f_q, F_p, F_q)). \end{aligned}$$

For a true correspondence the *uniqueness ratio* using F_q as a baseline is distributed similarly to the ratio when using F_t according to assumption #3. In this case the algorithm performs like *Ratio-Match-Ext*. However, for an untrue correspondence with a baseline match in the *query image* which is closer than the baseline match in the *target image*, *Mirror Match* will return a worse *uniqueness ratio*. This in turns means that *Mirror Match* is equal or better than *Ratio-Match-Ext* in terms of *Precision* while maintaining equal *Recall*.

Using a similar procedure we can prove that *Self-Match* is equal to *Self-Match-Ext*, and *Both-Match* is equal to *Mirror Match*. The details can be found in Appendix A and Appendix B, respectively.

We have thus proven that – based on assumptions #1-3 – for any given recall rate, the precision of the algorithms in the framework compares as follows:

$$\textit{Ratio-Match} \leq \textit{Ratio-Match-Ext} \leq \textit{Mirror-Match}$$

Self-Match is most closely related to *Ratio-Match* in that both the *baseline* and *proposal* set are created based on only one image. While the *proposal set* for both algorithms is based on the *target image*, *Self-Match* uses the *query image* for the *baseline set*, whereas *Ratio-Match* sticks with the *target image*. For cases with a lot of overlap between the *target* and *query image* they should perform similarly. However when we match images where the *target image* may not overlap at all with the *query image*, using the *query image* and not the *target image* as a *baseline set* seems like a reasonable choice, given that the baseline

set would be closer to the feature matched in this case and more strictly rule out false correspondences.

2.5. Discussion of Assumptions

The three assumptions underlying the formal ranking of algorithms presented above have been chosen to reflect conditions under which we would ideally match feature points. However, for each assumption there exist corner cases where it is no longer valid. In this section we discuss these corner cases in order to review the circumstances under which the absolute or relative performance of the algorithms might differ.

The first assumption states that there is at most one unique match to every feature point, which is often the case for natural images. However, for the recognition of object classes or in images with repetitive content, a point in the *query image* might have several true correspondences in the *target image*, so using a *baseline match* from the *target image* will lead to a much higher *uniqueness ratio*. However, as long as the *query image* does not contain repetitive objects, we can still use *Self-Match* without loss of precision. Conversely the query image could have several similar objects which could negatively impact performance for all algorithms except *Ratio-Match* in cases where the target image only contains one such object. However when applied to scene matching we would normally expect there to be several similar objects in the target image too. In this case it could easily be ambiguous to match any of these points together. Here the uniqueness assumption plays to our advantage by forcing us to ignore features that will lead to ambiguous matches.

According to the second assumption, feature descriptors behave such that when matched, a true correspondence to a *query feature* will always be closer in distance than any other feature in the query image. For this assumption to be violated, either the *query image* has to contain repeating patterns (violating assumption #1), or the two images are so different (i.e. taken from very different viewpoints or under different lighting conditions) that the descriptor distance between two unrelated features ends up being closer than the distance between

a feature and its true correspondence. In this case *Ratio-Match-Ext* and *Mirror-Match* would discard the match while *Ratio-Match* would still attempt to match the descriptor to a feature in the *target image*.

The empirical results of *Self-Match* in Figure 6 and Figure 9 support the second assumption. *Self-Match* discards any match where a feature in the query image is more similar to the query feature, yet the algorithm consistently performs equally well or better than *Ratio-Match*, providing further evidence to the notion that for almost all true correspondences the best match is found in the *target image*. As we would expect this assumption breaks down as we increase τ to accept matches with higher uniqueness ratios, illustrated by the higher recall rates of *Ratio-Match* compared to *Self-Match* in most plots. Once we accept almost all proposed nearest neighbors as matches, we are bound to encounter more non-distinct feature points for which even true correspondences might have better matches amongst the *query features*.

The third assumption states that the *uniqueness ratios* using either the *query* or *target image* as the baseline set would be similar in distribution, given that the feature point we are matching has a true correspondence. We can support this assumption by looking at the *uniqueness ratios* returned by *Self-Match* and *Ratio-Match*, since they use the *query* and *target image* respectively to calculate them. Figure 3 shows the actual *uniqueness ratios* measured from 3024 image pairs featuring 3D objects (see Section 3.1 for more on this dataset). For ratios lower than 0.7, the *uniqueness ratios* are similar, but as we approach more lenient thresholds, the ratios based on the *query image* are higher than those for the *target image*. This means that the third assumption becomes invalid for more lenient thresholds, and we can no longer expect *Mirror-Match* to outperform *Ratio-Match-Ext*.

3. Experiments and Results

In this section we present experiments and evaluation results comparing *Self-Match*, *Ratio-Match*, *Ratio-Match-Ext*, and *Mirror-Match*.

We evaluate the matching algorithms of our framework on two sets of data:

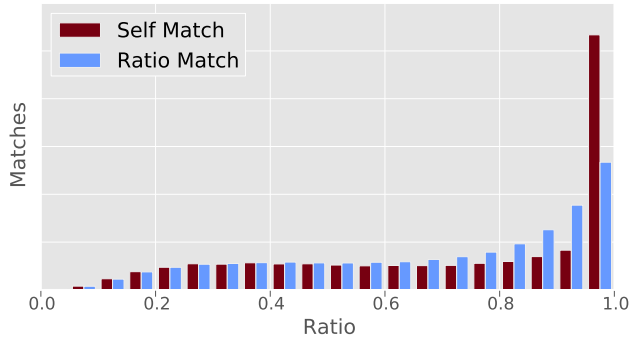


Figure 3: Normalized histogram of the *uniqueness ratios* of true correspondences for 3000 image pairs.

- 3D Objects database released by Moreels and Pietro [15] (Section 3.1). It contains a set of 86 3D objects photographed from all sides at 5 degree intervals from two different elevation angles and under three different lighting conditions.

- Graffiti dataset released by Mikolajczyk et al. [14] (Section 3.2). It contains eight sets of six images, each highlighting a different challenge when matching images. We evaluate the framework on four particularly challenging cases.

Unless mentioned otherwise, we use the SIFT descriptor and keypoint detector with default parameters as implemented in the OpenCV library 2.4.6. To further test how the framework performs across descriptors, we also compare SIFT with SURF [13], BRISK [18], BRIEF [17], and FREAK [33] in Section 3.3.

Across all experiments, only the luminance channel is used in the feature detection and description stage. Images larger than 1024×768 are resized to fit within those dimensions using ImageMagick with default parameters.

3.1. Evaluation on 3D Objects

The 3D objects dataset by Moreels and Pietro [15] allows us to experimentally compare matching algorithms over a large range of object and surface types rotated on a turnstile and photographed from every 5 degree turn. 15 objects

from the dataset are shown in Figure 4. We use images of 84 objects under three different lighting conditions at 12 different angle intervals, leading to a total of 3024 image pairs.

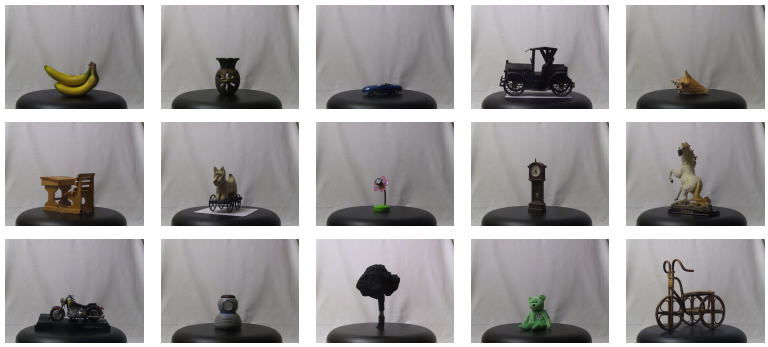
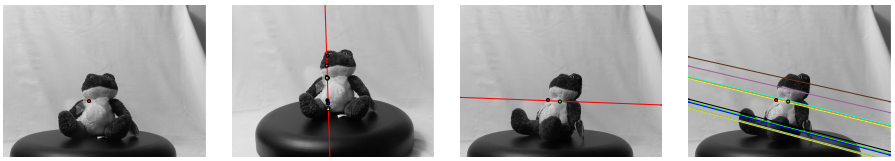


Figure 4: 15 objects from the 3D Objects dataset by Moreels and Pietro [15].

To validate matches, Moreels and Pietro propose a method using epipolar constraints [15, p.266], which is outlined in Figure 5. According to their experiments, these constraints are able to identify true correspondences with an error rate of 2%. We use their proposed method to generate the ground truth for the evaluation of our framework.

To compute the total number of possible correspondences, we take each feature in a *query image* and count how many have a feature in the *target image* which would satisfy the epipolar constraints outlined above. Features with no correspondences were not included in the set of features for testing. This was necessary because a relatively small number of actual true correspondences between folds in the background material were mistakenly counted as false positives when evaluated, which impacted the precision of the test in particular in cases with few true correspondences. In practice very few feature points were excluded for this reason.

We evaluate all matching algorithms from our framework on the 3D Objects dataset by matching images at different angular intervals. For each object we pick the *query* image as the image taken at 10 degrees rotation for calibration stability. We then match this image with the same object turned an additional Δ



(a) Query Image (b) Auxiliary Image (c) Target Image (d) Target Image with epipolar constraints

Figure 5: Creating epipolar constraints based on three source images [15]. (a) *Query image* marked with the position of the feature we are attempting to match. (b) Auxiliary image, taken at the same rotation as the *query image* but from a higher elevation angle. The line going through the image is the epipolar line of the feature point in the *query image*. The markers indicate all feature points in the image found near the epipolar line. In c) we show the *target image* which in this case is rotated 45 degrees from the *query image*. The line overlaid on the image represents the epipolar line corresponding to the feature point shown in the *query image*. The markers indicate all feature points in the image found near the epipolar line. In d) we show the *target image* again, this time overlaid with the epipolar lines corresponding to all the features shown in the *auxiliary image*. A true correspondence should be found within a small distance of one of the intersections of the line in c) and the lines in d). In this particular case both feature points shown in c) and d) could possibly be true correspondences.

degrees, $\Delta \in \{5, 10, \ldots, 60\}$. For every angle interval we compare images taken under 3 different lighting conditions as provided by the dataset. We include all objects in the database for which photos at 5 degree angle intervals are available, except for the "Rooster" and "Sponge" objects due to image irregularities.

Figure 6 shows the performance of the different matching methods in our proposed framework for 12 increasingly bigger angle differences. The results are shown in a precision-recall plot to make it easy to compare performance in terms of precision at similar levels of recall. For each plot we show the averaged results on all 3D objects, with equal weight given to each image pair.

Ratio-Match and *Self-Match* exhibit similar results, while *Mirror-Match* and *Ratio-Match-Ext* outperform both of them, showing the advantage of composing the *proposed set* of features from both images. *Mirror-Match* fares slightly but consistently better than *Ratio-Match-Ext*. In general we see the largest performance improvements at lower recall and a convergence of precision at higher recall. This is expected since a higher recall is a direct consequence of a more lenient threshold; as we let the threshold approach 1, we lose the benefits of thresholding on the *uniqueness ratio*, and all methods start approximating

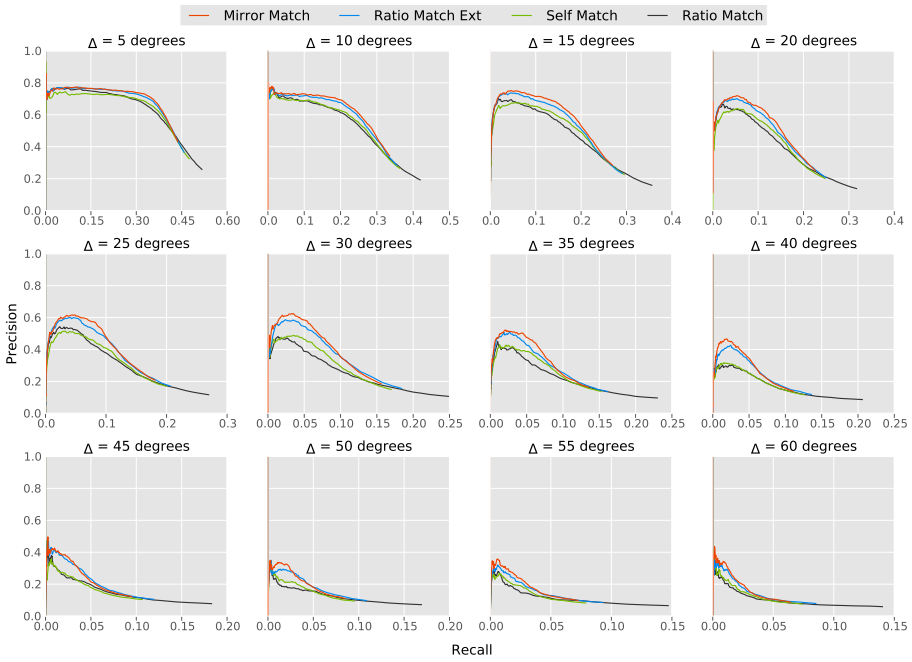


Figure 6: Results for the 3D objects dataset. Each plot contains the results of 84 objects photographed under 3 different lighting conditions averaged over each image pair.

the results of a simple nearest neighbor match. For all but *Ratio-Match*, the features with better matches within the same image are still weeded out, which explains the tail end of *Ratio-Match* at high recall where the other algorithms no longer have results. As we demonstrated in [23], the removal of within-image matches improves performance on image pairs with partial or no overlap. For no overlap we would expect a matching algorithm to reject matches within regions that have no matching counterparts in the other image.

The performance gap between methods increases gradually with angle, reaching its maximum between 25 and 40 degrees, where *Mirror-Match* exhibits about 20 percentage-points higher precision than *Ratio-Match*. At larger viewpoint differences above 45 degrees, the performance gain decreases. We suspect this is partly because assumption #2 stops being valid when the images are transformed beyond a certain level of recognizability.

To summarize, the methods perform as predicted on the 3D Objects dataset,

with *Mirror-Match* in general performing better than *Ratio-Match-Ext*, which in turn outperforms *Ratio-Match*. The experimental results also show that the overall performance of *Self-Match* is equal to or better than *Ratio-Match*.

3.2. Evaluation on partially overlapping scenes

To evaluate how the different algorithms in the framework handle partially overlapping scenes, we make use of four images from the Graffiti dataset by Mikolajczyk et al. [14], all featuring challenging perspective changes as shown in Figure 7. For each of the four image pairs we generate a hundred pairs of smaller square patches, each patch randomly positioned within the original image. Five examples of this process can be seen in Figure 8. This ensures that image pairs that do not contain any overlap are still visually similar, making it more challenging for a matching algorithm to avoid false positives, i.e. matching local image features that are not true correspondences.

In practice the amount of overlap between pairs in a test set will depend on the overlap and viewpoint change in the source image pair. To give a rough idea, Table 1 shows the overlap of patch pairs created from images 1 and 3 of the *Graf* image.

Table 1: Overlap in the set of 100 patch pairs created from the 'Graf' image pair (Figure 7).

Amount of overlap:	0%	$< 50\%$	$> 50\%$
Number of patch pairs:	21	54	25

Figure 7: Four image pairs from the Graffiti dataset by Mikolajczyk et al. [14]. From left to right the images are 'Boat' (rotation), 'Graf' (perspective), 'Bikes' (blur) and 'Wall' (perspective).

Figure 8: Five examples of image patches extracted from the 'graf 1-3' image pair shown in Figure 7. Each image is a 300x300 pixel crop taken from a random position in 'graf 1' (top row) and 'graf 3' (bottom row).

Given a potential match between two points p_1 and p_2, $m = (p_1, p_2)$, and a homography H relating the two images I_1 and I_2, we can determine if m is an inlier by checking if the two points satisfy the following criteria:

$$|Hp_1 - p_2| + \left|H^{-1}p_2 - p_1\right| < d_{\max}$$

That is, the distance between p_1 translated to I_2 and p_2 *plus* the distance between p_2 translated to I_1 should be less than a certain threshold (we use $d_{\max} = 5$ pixels here).

As with the 3D Objects dataset, we accumulate results across the hundred image patch pairs using an average weighted by the amount of true correspondences between each patch pair. This ensures that difficult to match and trivial image pairs have an equal impact on the evaluation result despite the difficult image pairs yielding far fewer actual matches.

In Figure 9 we see the performance of the three different matching methods over 400 crop pairs based on four image pairs. The results are again shown in a precision-recall plot weighted by the number of possible true correspondences for the individual patch pairs.

In the plots for 'boat', 'graf' and 'bikes' we see the same pattern unfold. *Ratio-Match* performs equally well as the other algorithms at low and high recall rates, but is otherwise outperformed by *Self-Match*, *Ratio-Match-Ext*, and *Mirror-Match*. Out of these three algorithms, *Mirror-Match* performs consis-

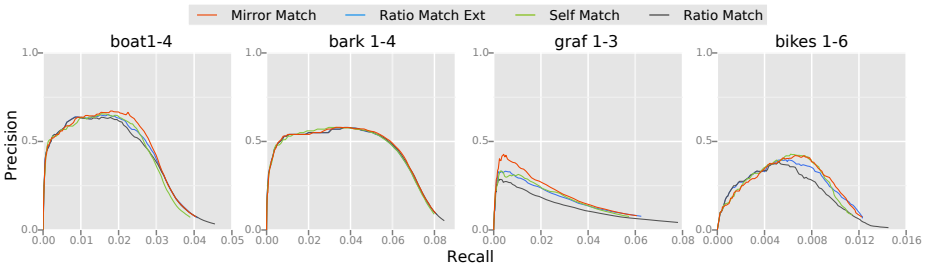


Figure 9: Results for the Graffiti dataset. Each plot contains the average result over 100 partially overlapping image patches based on the same two images weighted over the number of true correspondences for each pair.

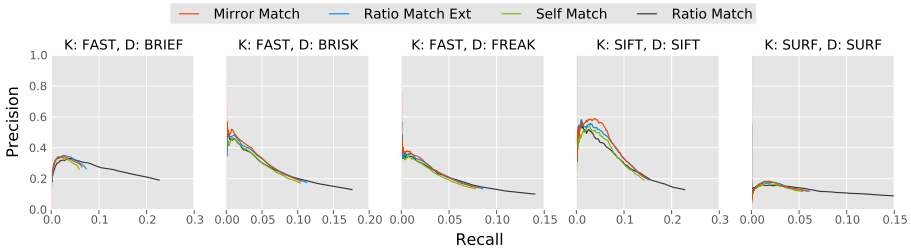


Figure 10: Keypoint / Descriptor combinations measured on 15 pairs of photos of 3D objects taken 25 degrees apart.

tently better than the other two in the case of 'boat' and 'graf' while being matched by *Self-Match* when testing for blur on 'bikes'. Finally 'bark' demonstrates a case where there is no discernible difference between the matching algorithms. These results are in agreement with the theoretical findings and showcase a range of improvements by *Mirror-Match* over *Ratio-Match* varying from about 30 percentage points increase in precision at low recall rates for 'graf' to an almost identical performance in the case of 'bark'.

For all four image pairs we see a dip in precision at very low recall, which is explained by the presence of image patch pairs with no or little overlap. When matching these pairs we might often find matches even though there are no or few possible true correspondences, which affects the results at very low recall rates.

3.3. Evaluation of Descriptors

To evaluate whether the improvement gains with the SIFT descriptor translate to other feature descriptors such as SURF [13] or binary features such as BRIEF [17], BRISK [18], or FREAK [33], we evaluated the different descriptors on the 3D Objects data set. The binary features were each paired with the FAST feature detector for consistency, while SIFT and SURF were tested with their respective feature detectors. For each keypoint detector and feature descriptor pair we used the standard implementation from OpenCV 2.4.6.

The descriptors were evaluated on the 15 images of the 3D Objects dataset pictured in Figure 4, with a fixed angle offset of 25 degrees. Otherwise the evaluation is performed in the same manner as the general evaluation of 3D Objects in Section 3.1, and the data presented is accumulated over the 15 objects, each set of results weighted by the possible number of correspondences.

The results in Figure 10 are again in line with the theoretical performance analysis of Section 2.4. They also show a clear difference in the variance of matching method performance between SIFT and the rest of the descriptors. The performance of *Mirror-Match* using SIFT shows a clear increase in precision over *Ratio-Match* at equal recall levels. For other descriptors the improvement is much less pronounced. For SURF, BRISK, and BRIEF we see a small improvement in precision using *Mirror-Match* over *Ratio-Match* and *Ratio-Match-ext*, whereas FREAK yields similar results no matter what algorithm is used. For all descriptors *Self-Match* performs on par with *Ratio-Match*.

Part of this difference between SIFT and the other descriptors can be explained by looking at the recall rate with for example SURF. Given the same amount of possible matches, *Ratio-Match* with SURF spans a larger range of recall. Since all other methods filter out matches that are better matched within the same image, but show no increase in precision, we can conclude that correct matches are discarded because there are better matches within the same image. This violates assumption #2 stating that a descriptor is well behaved and matches best with a descriptor of a true correspondence. This means that for this particular evaluation case, SURF and the binary features tested are not

sufficiently discriminative to provide a viable case for preferring *Mirror-Match* over *Ratio-Match*.

3.4. Computational Performance

In terms of computational complexity, the algorithms can be implemented in $O(n \log n)$, if we assume that both the *query* and the *target* image have n feature points. For a *target image* with m features where m is significantly different from n, the complexities are as noted in Table 2. In practice the constant factors involved in the actual matching of two images with approximately the same number of feature points are usually small enough that all the algorithms run at very similar speeds. This is also shown in Table 2, which contains the running times as measured while matching 15 3D objects under three different lighting conditions (42 image pairs were matched in total). The running times are averaged over three separate runs of each algorithm implemented using the same data structures and libraries in Python.

Table 2: Complexity and average running times over 45 different image pairs with average $n = 237$ and average $m = 247$ feature points as tested on a Intel® Core™ i5-3550 CPU @ 3.30 GHz with 8 GB memory.

Algorithm	Complexity	Avg Running Time
Self-Match	$O(n \log(nm))$	2.62s
Ratio-Match	$O(n \log(m))$	2.53s
Ratio-Match-Ext	$O(n \log(n+m))$	2.49s
Mirror-Match	$O(n \log(n+m))$	2.44s

4. Conclusions

We have proposed a general framework of feature matching methods, building on the ideas behind *Ratio-Match* and *Mirror-Match*, and introducing the additional variants *Self-Match* and *Ratio-Match-Ext*. We formally proved under three assumptions that *Mirror-Match* performs better than or equal to *Ratio-Match-Ext* in terms of precision and recall, which in turn performs better than or equal to *Ratio-Match*.

From an evaluation using images of rotated 3D objects, we have shown that the theoretical conclusions are confirmed by our experimental evaluation, with

Mirror-Match often outperforming *Ratio-Match* significantly over 3024 image pairs. With further evaluation on partially overlapping scenes we have shown that this also holds true for cases where we can't be sure to find the queried object in the other image. Finally we show that the theoretical conclusions hold across a variety of popular feature descriptors (SIFT, SURF, BRISK, BRIEF, and FREAK). These performance gains come for free in terms of both computational complexity as well as actual computing time.

Appendix A. Proof of equivalence of Self-Match and Self-Match-Ext

To prove the equivalence between *Self-Match* and *Self-Match-Ext*, we look at the nearest neighbor f_{nn} of every query feature f_q and consider the two cases of $f_{nn} \in F_q$ and $f_{nn} \in F_t$. For $f_{nn} \in F_t$ the uniqueness ratio of *Self-Match-Ext* is:

$$\begin{aligned} r &= \mathrm{r}(f_q, F_p, F_b) \\ &= \mathrm{r}(f_q, F_q \cup F_t, F_b) \\ &= \mathrm{r}(f_q, F_t, F_b) \end{aligned}$$

Since $F_p = F_t$ for *Self-Match*, the two algorithms behave identically when the nearest neighbor is found in the *target image*. For $f_{nn} \in F_q$ we know that the best match for the query feature is in the *query image* itself. Since neither algorithm returns a correspondence within the same image, they also behave identically for this case. This proves that for any set of *query features Self-Match* returns the same matches as *Self-Match-Ext*.

Appendix B. Proof of equivalence of Both-Match and Mirror-Match

The proof of the equivalence between *Both-Match* and *Mirror-Match* is almost identical to the equivalence proof between *Self-Match* and *Self-Match-Ext*. We look at the nearest neighbor f_{nn} of every query feature f_q and consider the two cases of $f_{nn} \in F_q$ and $f_{nn} \in F_t$. For $f_{nn} \in F_t$ the uniqueness ratio of

Mirror-Match is:

$$\begin{aligned} r &= \mathrm{r}(f_q, F_p, F_b) \\ &= \mathrm{r}(f_q, F_q \cup F_t, F_b) \\ &= \mathrm{r}(f_q, F_t, F_b) \end{aligned}$$

Since $F_p = F_t$ for *Both-Match*, the two algorithms behave identically when the nearest neighbor is found in the *target image*. For $f_{nn} \in F_q$ we know that the best match for the query feature is in the *query image* itself. Since neither algorithm returns a correspondence within the same image, they also behave identically for this case. This proves that for any set of *query features Both-Match* returns the same matches as *Mirror-Match*.

References

[1] A. Bosch, A. Zisserman, X. Muoz, Scene classification using a hybrid generative/discriminative approach, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (4) (2008) 712–727.

[2] M. Brown, D. G. Lowe, Automatic panoramic image stitching using invariant features, International Journal on Computer Vision 74 (1) (2007) 59–73.

[3] J. Zhang, M. Marszałek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: A comprehensive study, International Journal on Computer Vision 73 (2) (2007) 213–238.

[4] W.-L. Zhao, C.-W. Ngo, Scale-rotation invariant pattern entropy for keypoint-based near-duplicate detection, IEEE Transactions on Image Processing 18 (2) (2009) 412–423.

[5] G. L. Scott, H. C. Longuet-Higgins, An algorithm for associating the features of two images, Proceedings of the Royal Society of London. Series B: Biological Sciences 244 (1309) (1991) 21–26.

[6] L. S. Shapiro, M. J. Brady, Feature-based correspondence: An eigenvector approach, Image and Vision Computing 10 (5) (1992) 283–288.

[7] S. Sclaroff, A. P. Pentland, Modal matching for correspondence and recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (6) (1995) 545–561.

[8] M. Carcassoni, E. R. Hancock, Spectral correspondence for point pattern matching, Pattern Recognition 36 (1) (2003) 193–204.

[9] R. Deriche, Z. Zhang, Q.-T. Luong, O. Faugeras, Robust recovery of the epipolar geometry for an uncalibrated stereo rig, in: Proc. European Conference on Computer Vision (ECCV), 1994, pp. 567–576.

[10] A. Baumberg, Reliable feature matching across widely separated views, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2000, pp. 774–781.

[11] C. Schmid, R. Mohr, Local grayvalue invariants for image retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (5) (1997) 530–535.

[12] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal on Computer Vision 60 (2) (2004) 91–110.

[13] H. Bay, T. Tuytelaars, L. Van Gool, SURF: Speeded up robust features, in: Proc. European Conference on Computer Vision (ECCV), 2006, pp. 404–417.

[14] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Trans. Pattern Analysis and Machine Intelligence 27 (10) (2005) 1615–1630.

[15] P. Moreels, P. Perona, Evaluation of features detectors and descriptors based on 3D objects, International Journal on Computer Vision 73 (3) (2007) 263–284.

[16] J. Heinly, E. Dunn, J.-M. Frahm, Comparative evaluation of binary features, in: Proc. European Conference on Computer Vision (ECCV), 2012, pp. 759–773.

[17] M. Calonder, V. Lepetit, C. Strecha, P. Fua, BRIEF: binary robust independent elementary features, in: Proc. European Conference on Computer Vision (ECCV), 2010, pp. 778–792.

[18] S. Leutenegger, M. Chli, R. Y. Siegwart, BRISK: Binary robust invariant scalable keypoints, in: Proc. International Conference on Computer Vision (ICCV), IEEE, 2011, pp. 2548–2555.

[19] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, ORB: An efficient alternative to SIFT or SURF, in: Proc. International Conference on Computer Vision (ICCV), IEEE, 2011, pp. 2564–2571.

[20] R. Szeliski, Computer Vision: Algorithms and Applications, Springer, 2010.

[21] J. Rabin, J. Delon, Y. Gousseau, A statistical approach to the matching of local features, SIAM Journal on Imaging Sciences 2 (3) (2009) 931–958.

[22] M. Brown, R. Szeliski, S. Winder, Multi-image matching using multi-scale oriented patches, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2005, pp. 510–517.

[23] J. T. Arnfred, S. Winkler, S. Süsstrunk, Mirror Match: Reliable feature point matching without geometric constraints, in: Proc. IAPR Asian Conference on Pattern Recognition (ACPR), 2013, pp. 256–260.

[24] P. H. S. Torr, A. Zisserman, MLESAC: A new robust estimator with application to estimating image geometry, Computer Vision and Image Understanding 78 (1) (2000) 138–156.

[25] O. Chum, J. Matas, Matching with PROSAC – progressive sample consensus, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, San Diego, CA, 2005, pp. 220–226.

[26] J. Kim, O. Choi, I. S. Kweon, Efficient feature tracking for scene recognition using angular and scale constraints, in: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Nice, France, 2008, pp. 4086–4091.

[27] M. Leordeanu, M. Hebert, A spectral technique for correspondence problems using pairwise constraints, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 2, San Diego, CA, 2005, pp. 1482–1489.

[28] L. Torresani, V. Kolmogorov, C. Rother, Feature correspondence via graph matching: Models and global optimization, in: Proc. European Conference on Computer Vision (ECCV), Springer, 2008, pp. 596–609.

[29] J. Yarkony, C. Fowlkes, A. Ihler, Covering trees and lower-bounds on quadratic assignment, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2010, pp. 887–894.

[30] Y. Yuan, Y. Pang, K. Wang, M. Shang, Efficient image matching using weighted voting, Pattern Recognition Letters 33 (4) (2012) 471–475.

[31] M. Cho, J. Lee, K. M. Lee, Feature correspondence and deformable object matching via agglomerative correspondence clustering, in: Proc. International Conference on Computer Vision (ICCV), IEEE, 2009, pp. 1280–1287.

[32] L. Wu, Y. Niu, H. Zhang, H. Zhu, L. Shen, Robust feature point matching based on local feature groups (LFGs) and relative spatial configuration, Journal of Computational Information Systems 7 (9) (2011) 3235–3244.

[33] A. Alahi, R. Ortiz, P. Vandergheynst, FREAK: Fast retina keypoint, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 510–517.

[34] T. Tuytelaars, K. Mikolajczyk, Local invariant feature detectors: a survey, Foundations and Trends® in Computer Graphics and Vision 3 (3) (2008) 177–280.