Universiteit Leiden
The Netherlands

Opleiding Informatica

Interest Point Detectors and Descriptors: an Evaluation

Berend van Starkenburg

Supervisors:
Prof.dr. M.S.K. Lew

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

16/07/2021

Abstract

Applications for interest point detectors and descriptors are just as broad as the field of computer vision as a whole. Each technique has a slightly different approach to cover its niche. With the growing number of new detectors and descriptors, it is crucial to evaluate in which context the detector is most potent and where its struggles lie. In this paper, we will evaluate a collection of novel and established local detectors and descriptors. First, we design a dataset so that the methods are exposed to a diverse set of conditions. With this dataset, the stability of interest point detectors is tested, after which the matching capabilities of the descriptors are put to the test. The methods will also be evaluated on their capability to detect copies from a large set of images. Finally, an existing method will be improved to make it more robust to transformations. This study aims to help researchers identify which method is best appropriate for their purposes.

Contents

1 Introduction

One of the two ruling paradigms in computer vision is salient feature extraction. At the base of the process lies the localization of interest points, regions in an image that capture the most distinctive patterns of the image and are stable under different types of deformations, and the description of the characteristics of these regions. Therefore, many of the algorithms responsible for feature extraction exist in two parts: A detector part that localizes the interest points and a descriptor part that stores the characteristics of the interest points in a feature vector. However, pure detectors and descriptors also occur. While the process in essence is simple, the technique has a vast range of applications: image stitching[EA14], content-based retrieval[THBL08], 3D-reconstruction[ACM11], and object recognition[APAJ12], to name a few.

Because of this diversity, we would also expect the different types of salient point techniques to be diverse. Many different implementations exist, each with its forte and drawbacks. A challenge that remains is accurately mapping the performance of the still-growing lists of extraction methods. A descriptor designed for texture recognition is likely to perform worse for visual tracking. Therefore, it is essential that the benchmarks used to evaluate these methods can capture their performance on this variety of applications, and the evaluation criteria used to present the performance are unbiased towards certain types of detectors or descriptors.

In this paper, the strengths and limitations of established methods like SIFT[Low04] and KAZE[ABD12], and novel methods like FFD[GLT21] and BEBLID[SSBB20] will be explored in the context of matching and object recognition. Each image in a customized dataset will be matched against a similar image under different circumstances, such as transformations or other modifications. A trade-off between sound localization and stability will be explored by evaluating the redundant and non-redundant repeatability rate of interest point detectors. With the help of the computed descriptors, corresponding regions in the two images are found and verified with ground truth to evaluate the matching performance. In this process, it is found out what dataset design is best to draw out the strengths and weaknesses of the techniques and which evaluation criteria best capture this information.

1.1 Thesis overview

The paper has the following structure: First, we will present a combination of novel and established interest point techniques, and we will briefly explain their strategies. Next, a dataset containing different types of image deformations will be designed from an existing dataset. For the evaluations, the detectors will first be tested by measuring the repeatability score. Descriptor performance will be measured by evaluating the recall and precision scores. The ability to detect copies in a large dataset will be tested next. Furthermore, we will examine if an addition to the FFD detector can make it invariant to rotations or other transformations and improve its performance on detecting copies.

2 Related Work

Efficient interest point detectors should have the ability to detect key points that remain stable under different types of transformations. Earlier methods relied on detecting corners in the image, such as the Harris Corner detector[HS88], which is invariant to photometric transformations, scales, and rotations. A more recent implementation of a corner detector is FAST[TH98]. While efficient, viewpoint changes prove to be a challenge to such detectors. In response, affine invariant detectors like Harris Affine and Hessian Affine[KM02] were proposed that use the earlier methods and make the regions affine invariant using affine shape adaptation. In this paper, detectors using a different method will be evaluated, namely the multiscale keypoint detectors. We can describe this method in two steps:

1. Construct a scale-space pyramid

2. Localize interest points by searching for extrema in the pyramid

More current detectors, such as superpoint[DMR17] disregard manual keypoint detection and use learning-based methods to detect interest points.

Just as there are plenty of different detectors, a large variety of interest point descriptors exist. The most straightforward descriptor uses features that represent a vector of pixels. The dimensionality of these features is vast; thus, different methods have been proposed. Some descriptors use histograms to represent specific characteristics of pixels in the detected regions. A noteworthy example is SIFT, which describes the keypoints as a 3D histogram of gradient locations and orientations. Other descriptors use learning-based method. Properties Optimization Point(POP)[YTX+19] uses a fully connected convolutional neural network for detecting and describing keypoints. BEBLID uses an ensemble of weak learners to compute the keypoints.

Performance evaluation of local detectors has been done several times[CS00][STLL03]. Interest point evaluations were carried out in the context of texture recognition[RH99][LSP05], visual tracking[SG11], and matching[MS05].

The following detectors and descriptors will be evaluated in this paper:

2.1 SIFT(detector/descriptor)

SIFT[Low04] is one of the most established interest point techniques. It detects keypoints using a cascade filtering approach. First, the algorithm creates a scale space of images by constructing progressively Gaussian blurred images. For each adjacent Gaussian image, the scale difference is taken to get a difference of Gaussian Pyramid. Spatially and across scales, the algorithm looks for extrema in the pyramid to find potential interest points. A quadratic surface fits the values in the neighborhood of the found extremum, which allows the algorithm to reject candidates with low contrast or poorly localized along the edge. In the meantime, the Taylor series expansion of DoG, D, is used to accurately localize at which scale the extrema are located. If the value of D at the extrema is too small, it is discarded. Finally, the hessian of the extrema's D that are sensitive to edge responses are rejected if the ratio of the trace and determinant exceeds a certain threshold.

The following steps are to make the algorithm invariant to rotations and to extract the features of the keypoints. A histogram of local gradient directions is computed for the remaining extrema. The orientation assigned to the peak is determined by the most occurring orientation in the neighborhood of the peak. For forming the descriptor, the gradient directions of the 16x16 neighborhood are grouped in a 4x4 histogram array with eight orientations, which results in a descriptor of 128 dimensions.

2.2 FFD(detector)

FFD [GLT21] is a recently developed robust feature detector that aims to reduce the computational costs of the scale-space analysis other multiscale detectors struggle with by analyzing a new relationship between the difference of Gaussian and the Laplacian of Gaussian by formulating the superimposition that occurs during edge transmission. FFD starts with constructing its scale-space by first blurring the input image with a Gaussian and next extracting a coarse image from the previous scale with a cubic spline kernel with varying $\sigma's$ at each scale level, after which the fine scale-space is created by taking the difference of Gaussian of adjacent coarse images. A coarse scale-space pyramid is created by taking the difference of each adjacent image. In contrast, to SIFT, this pyramid contains no octaves, and the images are up-sampled instead of down-sampled, intending to provide more robust responses along the edges to detect potential edges SIFT would otherwise miss. Non-maximum suppression is applied to the fine scale-space to find interest point candidates. Similar to SIFT, extrema with low contrast are discarded, and the quadratic Taylor expansion is applied to locate the peak's scale.

2.3 KAZE(detector)

KAZE, proposed by P. Alcantarilla[ABD12], being another multiscale method, also aims to detect interest points at multiple scales by creating a scale-space. The method avoids constructing the scale-space with linear diffusion, like a Gaussian, because of the following disadvantages: While blurring reduces noise and emphasizes interesting features such as corners and textures, the blurring occurs to the same degree across the image. For this reason, noise is reduced to the same degree as details, which results in a loss of localization. KAZE opts to construct the scale-space with nonlinear diffusion filtering. KAZE first constructs a set of blurred images with AOS schemes. To detect keypoints, the response of the scale-normalized determinant of the Hessian is computed. The extrema are found with a sliding window technique and considered as keypoints similarly to SIFT.

2.4 DAISY(descriptor)

DAISY, a dense descriptor proposed by Tola et al.[TLF10], was designed to yield good results in wide-baseline stereo matching. Orientation maps are computed for each quantized direction, comparably to SIFT. However, to efficiently compute the descriptor at each pixel for dense matching, the weights of the SIFT orientation histogram are replaced with convolutions of the gradients in specific degrees with several Gaussian filters. This way, DAISY keeps the exact invariance as the SIFT descriptor while only having to compute the orientation histogram once per region.

2.5 LUCID(descriptor)

LUCID (Locally Uniform Comparison Image Descriptor) is a binary descriptor proposed by Ziegler et al[ZCKB12]. The algorithm constructs the descriptor as the order permutations of the concatenated RGB color channel vector of an nxn image as seen in Image 1.

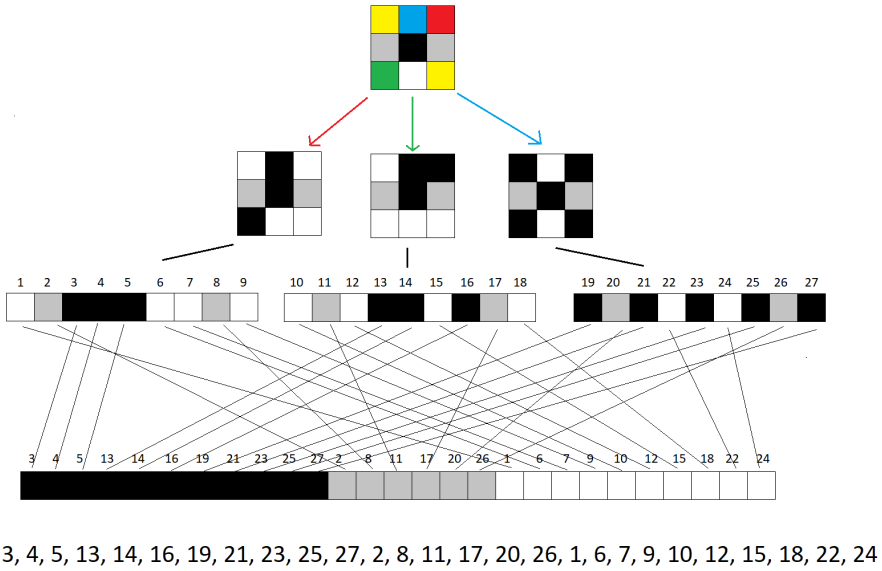

Figure 1: LUCID feature construction

Since a binary descriptor is generated, and the feature dimensionality is very low, LUCID is highly suitable for real-time applications.

2.6 BEBLID(descriptor)

Another, more recent, binary descriptor is BEBLID (Boosted Efficient Binary Local Image Descriptor)[SSBB20]. First, a real-valued descriptor is generated with a weak learner selection strategy using AdaBoost. To generate an efficient descriptor, the feature extraction the weak learner depends on is balanced to be discriminative and fast to compute. The real descriptors are simplified to a binary descriptor when the weight of all weak learners is equal.

3 Experimental Setup

The experiments were conducted on a 64-bit operating system, with an Intel® Core™ i7-8700k CPU @ 3.70GHZ processor and 32-GB of RAM. The implementation for the FFD detector and the code for the non-redundant repeatability was provided by the author. The OpenCV implementations were used for the remaining detectors and descriptors and matching strategies.

3.1 Datasets

The datasets used for the evaluations are based on the Microsoft COCO: Common Objects in Context dataset[LMB+14].

First, 1000 images were randomly selected from this dataset to serve as a ground truth. For each image, data augmentation was applied to create new images by applying the following transformations to the original image:

Viewpoint change: It is possible to emulate a viewpoint change in an image with an affine transformation, which can be interpreted as a composition of a rotation and a deformation such that:

$$x' = \begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix} * x$$

, with x' being the distorted image, x being the original image, t a translation and A the affine matrix:

$$A = R(\psi) * R(-\phi) * D * R(\phi)$$

, with R(ϕ) being a rotation, R($-\phi$) and R(ϕ) being a deformation, and D being a scaling matrix that models a tilt, t[WOBL17][RH04].

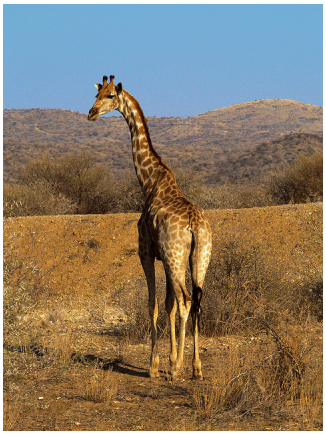

Figure 2: Reference Image

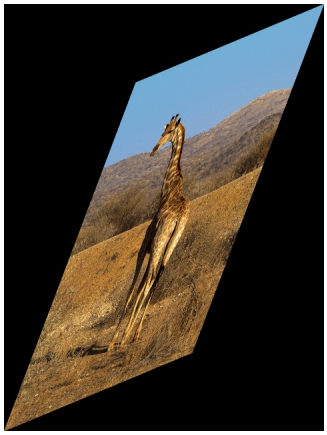

Figure 3: Affine transformed query image

Scale changes: New images are created by incrementally scaling the original image.

Rotations: The original image is rotated by ϕ degrees to create new images.

Image zooms: New images are generated by progressively zooming in to the center of the original image.

Other images are generated from the original image by adding salt&pepper noise to the image and compressing or Gaussian blurring the original image.

3.1.1 Dataset for Detector and Descriptor Evaluation

The dataset used for the detector and descriptor evaluation contains 40 affines, nine blurred images, 19 compressed images, 15 cropped images, 11 images with added noise, 23 rotated images, and ten scaled images per original image in a dataset with 127000 images.

3.1.2 Dataset for Copy Detection Evaluation

For the purpose of copy evaluations, the image must copy closely resemble the original image to emulate a practical situation where copy detection would be used. While image croppings and scales are the most likely transformations contributing to the image copies found on the WWW[LTCJ+07], all other mentioned transformations are also evaluated. For each transformation, five slightly transformed images are generated per original image, resulting in a dataset of 35000 images.

3.2 Matching Strategy

Since we are evaluating detectors and descriptors in the context of matching and recognition, we are interested in how well descriptors can match corresponding regions in images. A commonly used approach is based on evaluating the number of correct matches, and incorrect matches are established for a reference image and a query image. The process is summarized as follows:

1. Keypoints are detected in a reference image
2. Descriptors are computed for the detected keypoints and stored in a feature vector
3. For each query image in the transformed set of the reference image, detect keypoints, compute descriptors, and store descriptors in a feature vector
4. For each feature in the reference image's feature vector, find the feature in the query image's feature vector with the closest distance
5. Filter bad matches from good matches

Step 4 is where the matching starts. To find the closest neighbor most efficiently, the OpenCV implementation of Fast Library for Nearest Neighbors is used for real descriptors. Constructing a KD-tree index is also a used approach, but this would give no considerable improvement compared to an exhaustive search for features with a high dimensionality[Low04]. For binary descriptors, an LSH index is constructed, and the hamming distance is used to find the closest neighbor, which is an efficient approach in and of itself.

Step 5 is where we can evaluate the competency of a descriptor. Since the image matching only looks at the distance between vectors and not at a distance between their corresponding keypoints, the closest neighbor could happen to be a feature that corresponds to a keypoint far away from the original keypoint. First, what separates a good match from other matches is dependant on the matching definition. In this paper, Lowe's ratio test is used to extract good matches from all matches[Low04]: A match is a correct match `if d1 < d2 * r`, where `d1` is the distance between a feature of the reference image and the closest feature of the query image, `d2` the second closest, and `r` a ratio set on 0.7 for this paper.

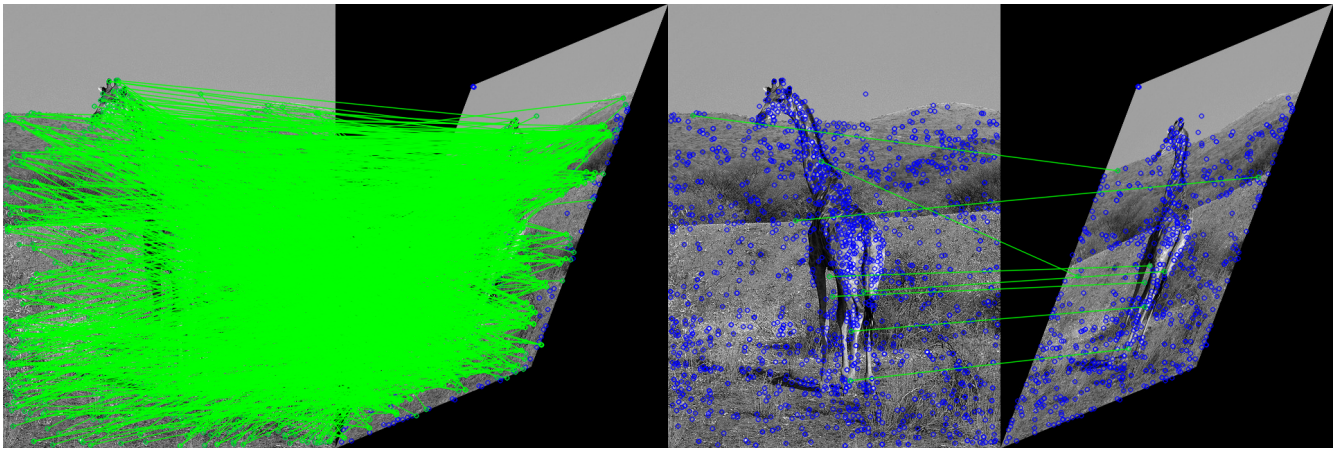

Figure 4: Filtering bad matches with ratio test

In Image 4, an example of match filtering with We illustrate Lowe's ratio test with an affine transformed query image. Initially, each detection in the two images has a match. After the ratio test, only the matched detection that is also spatially in the image close to each other are kept.

3.2.1 Copy Detection

The dataset to evaluate the copy detection is employed as follows: First, the keypoints are detected, and descriptors are computed for each image in the dataset. Next, the copies are extracted from the dataset. To determine whether an image matches as a copy or not, we used the following heuristic: If the amount of correct matches between two images is larger than a certain percentage of total matches, the query image is classified as a copy of the reference image. This can be summarized as:

```
if #correspondences > threshold * min(kps1, kps2):
    image = copy
```

Each original image is matched against the whole set of transformed images. The threshold variable first starts at 100, so all matches need to be correct for the image to be classified as a copy. The threshold is slowly decreased until it hits 0. This gives us the number of copies a descriptor was able to retrieve at each rank. Since we know a priori if the query image is an actual copy, we can get the number of true positives and false positives at each rank to use as evaluation criteria.

3.3 Evaluation criteria

The following section gives an overview of the evaluation criteria used to measure the performance of local detectors and descriptors.

3.3.1 Repeatability

To evaluate the performance of keypoints detectors, the most widely used criterion is the repeatability[CS00]. It can be understood as the proportion between interest points detected at the same relative position across two images related by a homography H, and the minimal amount of detections in the images:

$$rep = \frac{\#repeatable\_keypoints}{min(kps1, kps2)}$$

Multiple definitions on when keypoints are repeated exist. The ones used in this paper are:

- Using an overlap error[KM04]: A pair of detections are considered repeated if:

$$1 - \frac{A_{xb} \cap A_{xab}}{A_{xb} \cup A_{xab}} \leq \epsilon$$

  where A_{xb} is, the area of the detected region of image B and A_{xab} is the area of the projection of the detected region in image A on image B with homography H. ϵ is the maximum overlap error allowed for a correspondence. For most purposes. ϵ is set to 0.4.

- Distance between keypoint centers: A detector that arbitrarily selects a large scale for keypoints could attain a higher repeatability rate. Therefore, the following definition will also be used to measure the repeatability score:

$$||x_b - H * x_a|| < \epsilon$$

  x_a and x_b are the keypoints of image A and B, respectively, and H is the homography that maps x_a to image B. According to [CS00], ϵ is set to 1.5.

3.3.2 Non Redundant Repeatability:

As we have seen with the overlap criterion, the repeatability can be biased towards certain detectors that select keypoints in such a way, the repeatability score gets inflated. Detectors that select a large number of keypoints, that may be poorly localized could also inflate the repeatability score. The problem with this is that more storage is required, matching takes longer, and the extra descriptors will not be unique[ZR11]. The repeatability no longer reflects the performance of the detector.

Several methods such as supervised regression to rank keypoints, and using entropy to penalize keypoints that are poorly localized[ZR11] have been suggested to eliminate this bias. However, in this research, alongside the conventional repeatability, the non-redundant repeatability proposed by [RDM14] will be evaluated:

First, a mask function with a truncated Gaussian is assigned to each detector pair. The masks are normalized so that we can get the number of keypoints by taking the sum of the integral over the image domain of each mask. In essence, the masks indicate how much each pixel contributes to a detection. One pixel may contribute to multiple detections. If this is the case, only one keypoint would be necessary, while the other keypoints are redundant. Because of this, we can get the number of nun redundant keypoints by taking the integral over the image domain of the maximum of the masks. This way, overlapping keypoints will count as one non-redundant detection.

Now we can formulate the non redundant repeatability as:

$$nr\_rep = \frac{\int_\Omega max_{k \in K_r}, f_k(x) dx dy}{min(kp1, kp2)}$$

where K_r is the set of repeated detections, and f_k is the mask function:

$$f_k(x) = K e^{-\frac{1}{2\zeta^2}(x - x_k)^T \Sigma_k^{-1}(x - x)}$$

where $(x - x_k)^T \Sigma_k^{-1}(x - x)$ encodes the elliptical region of a detection, K the set of detections, and ζ a control variable dependant on the type of descriptor used.

3.3.3 Recall and Precision

For evaluating the performance of descriptors and copy detection, we use recall and (1-) precision[MS05]. Recall describes the descriptors ability to match corresponding regions of two images correctly, while precision is an indicator on how well the descriptor can compute descriptors that correctly match in proportion to all computed descriptors:

$$recall = \frac{TP}{TP + FN}$$

$$precision = \frac{TP}{TP + FP}$$

The way TP, FP, and FN are defined differs for descriptor evaluation and copy detection evaluation:

Descriptor evaluation:

- TP = #correct_matches
- FP = #false_matches
- FN = #unmatched_correspondences

Copy detection evaluation:

- TP = #correct_copies
- FP = #false_copies
- FN = #undetected_copies

3.4 Improving the FFD Detector

The FFD detector was proposed to be an interest point detector with a performance as well or even better than SIFT while being faster in detecting keypoints. The method it uses to detect keypoints also resembles SIFT's method. The SIFT detector indicates the keypoint orientation for the descriptor to achieve rotation invariance. FFD only indicates the position and scale. Due to the already low complexity, FFD is a suitable candidate to test if it is possible to achieve rotation invariance by assigning orientations to keypoints.

Three different versions based on the baseline FFD detector that each use a different orientation assignment technique will be evaluated on rotated images with the same criteria as the descriptor evaluation. One of these versions also will be included in the copy detection evaluation. The three techniques used are the following:

3.4.1 Histogram Orientation Assignment

The first method to assign orientations to keypoints is the same method the SIFT detector uses to assign orientation, as explained in Section 2.1. First, orientations are computed for each keypoint in a neighborhood that depends on the keypoint's scale. An orientation can range from 0 to 360 degrees. From these orientations, a histogram with 36 bins is constructed. For example, an orientation of 0 degrees will be put in bin 1, 10 degrees in bin 2, and so on. The importance of each orientation depends on the gradient magnitude between the keypoint and the location of the current orientation. This results in a histogram where the peak is the most dominant orientation of the neighborhood. This orientation is assigned to the keypoint. If other bins are greater than $0.8 * peak$, the keypoint is duplicated, and that orientation is assigned to the newly created keypoint.

The following two orientation methods were used in [GTH11] to evaluate different types of orientation assignments for different detectors

3.4.2 Taylor's Method

A straightforward method proposed by [TD11] that is also fast to compute but is affected more by noise than the other techniques:

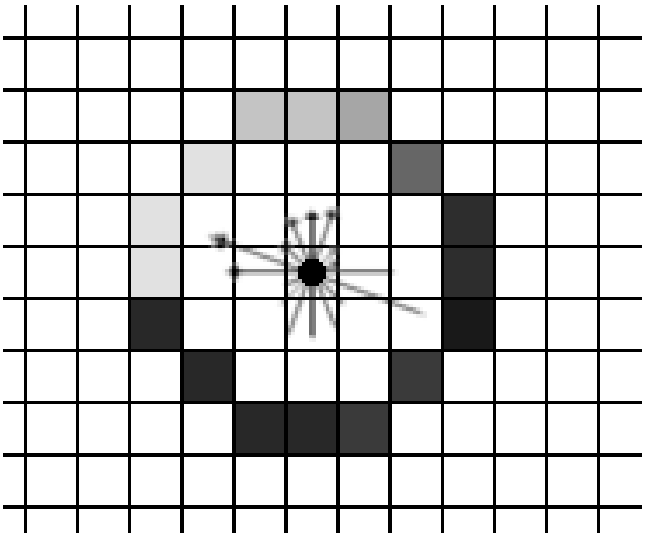

Figure 5: Intensity difference between opposite pixels

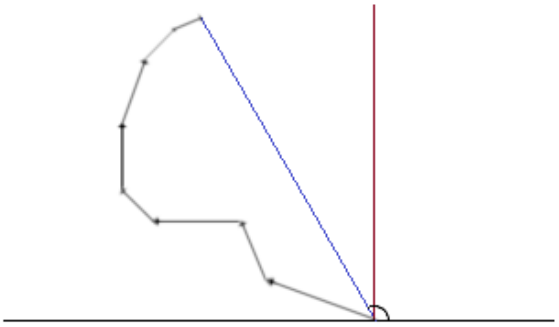

Figure 6: Appended difference vectors

The method starts by computing vectors for the opposite pixel pairs (radius 7 in this paper) as seen in Image 5. Then, the vectors are weighted by the intensity difference between two opposite pixels. Next, the vectors are appended starting from (0,0). Then, a line is drawn from the origin to the point where the appended vector ends up. The orientation assigned to the keypoint is the degrees of this line and the positive x-axis.

3.4.3 Center of Mass

This technique aims to compute the balance point that uses the weighted intensity of each pixel as the mass of a circular image patch and uses its direction relative to the keypoint as the orientation. More formally the location c_x, c_y, of the center of mass is calculated by the following two sums:

$$cx = \frac{1}{S} \sum_{(x,y) \in R} w(r) * (x - x_p) * I(x, y) \; ; \; c_y = \frac{1}{S} \sum_{(x,y) \in R} w(r) * (y - y_p) * I(x, y)$$

Where $I(x, y)$ is the gray-value intensity of that pixel, $w(r)$ is a weighting function that determines the weight of the intensity based on the distance of the current pixel and the keypoint r. The further away the current pixel is from the keypoint, the lower the weight, ans S is determined by:

$$S = \sum w(r) * I(x, y)$$

Since FFD is a detector that detects keypoints at sub-pixel locations, for both methods, the intensity of sub-pixel locations is used as well, which are computed with interpolation.

For the evaluation, the detector variants must be paired with a descriptor that uses the orientation information to compute its descriptors. Therefore, we chose to pair the detectors with the SIFT descriptor. For the regular descriptor evaluation, only rotated images are used from the dataset since we are interested in the impact of the orientation assignment on rotated images alone. The recall and precision will be plotted. The `AUC` of the plots and the time it takes on average to assign the orientation for each keypoint per image will be presented in a table. For Taylor's method, a radius of 7 pixels was used, and for the center of mass method, a radius of 9 was used for the pixel neighborhood.

For the copy detection, only the histogram assignment will be evaluated. However, this method will be evaluated for all images in the copy detection dataset in the same way the other descriptors are evaluated for copy detection.

4 Results and Discussion

4.1 Detector Evaluation

In this section, the results of the detector evaluation will be presented and discussed. The overlap error and the distance error ϵ, as discussed in Section 3.3.1, will be set at 0.4 and 1.5 respectively. The figures are using the overlap criterion to compute the `repeatability`.
First, the regular `repeatability` per transformation in Images 7 and 8.

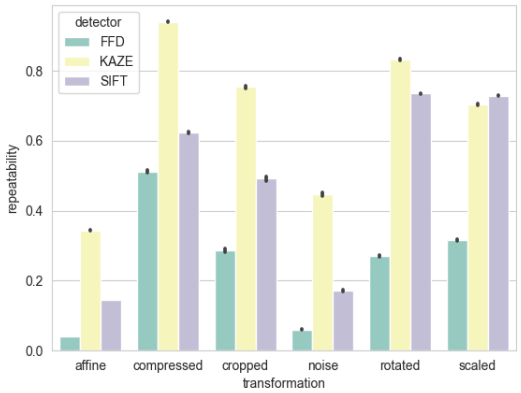

Figure 7: Repeatability rate per transformation

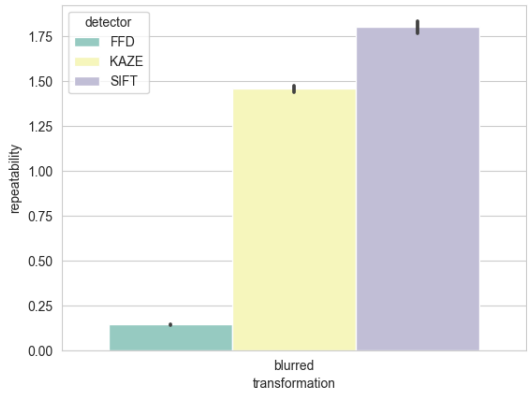

Figure 8: Repeatability rate for blurred images

First of all, the FFD detector attains considerably smaller rates than the other two detectors. In Table 2, the detectors' `repeatability` with the distance error is shown as well. The values the FFD detector achieves is far more competitive in comparison. This has the following explanation. The FFD detector selects a smaller scale for the keypoints in comparison to the other detectors. The overlap with keypoints of the query image that would normally be in the neighborhood of the original keypoint, if the distance error was used, is too small to result in a correspondence. This could also be the explanation for the inflated `repeatability` rates for blurred images: The SIFT and KAZE detectors select a large scale for every keypoint so that the amount of overlapping keypoints will be larger than the minimum amount of keypoints in the reference image or query image. This hypothesis is backed up by Table 1.

detector	affine	blurred	compress	cropped	noise	rotated	scaled
FFD	2.214	5.417	2.263	2.436	2.381	2.244	2.070
KAZE	12.443	17.963	8.432	7.676	5.832	9.677	9.919
SIFT	6.311	22.226	4.458	4.667	3.651	4.792	5.094

Table 1: Average keypoint scale per transformation

The reason for still using the overlap method over the distance method for creating the figures is since the distance criterion is not scale-invariant[RDM14].
Next, the non redundant repeatabilty per transformation in Image 9.

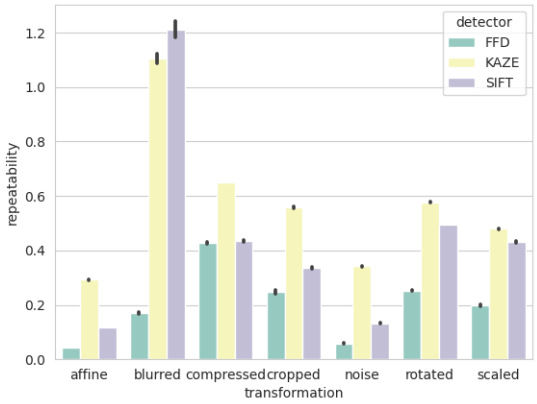

Figure 9: Non redundant repeatability rate

Only measuring the `repeatability` with non-redundant keypoints impacts the KAZE detector the most. The likely reason for this is the higher `repeatability` the detector attains in comparison to FFD and SIFT. The proportion of keypoints that remain repeatable with the KAZE detector is smaller than the proportion of FFD and SIFT, which could indicate that KAZE is prone to selecting overlapping keypoints.

For now, we only looked at the overall repeatability rate per transformation. To see if certain viewpoint changes, or certain rotations affect the repeatability more than others, Figure 10 and 11 are presented.

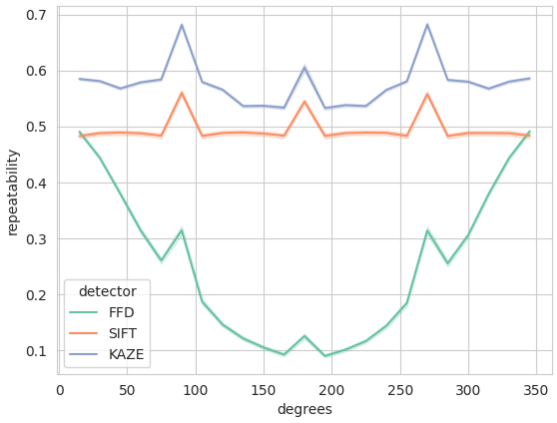

Figure 10: Repeatability rate per degrees

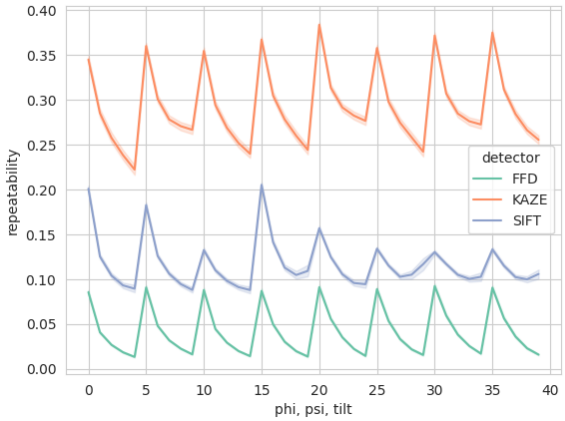

Figure 11: Repeatability rate for different viewpoints

The values on the x-axis of Figure 11 corresponds to all permutations of ϕ, ψ and t, with $\phi \in \{0, 1, 2\}$, $\psi \in \{0, 1, 2\}$, and $t \in [2, 6]$.

Finally, Table 2 containing the results of the detector evaluation.

	overlap = 0.4				*dist* = 1.5	
	#rep. kps	rep.	#nr-rep. kps	nr-rep.	#rep. kps	rep.
overall						
FFD	166.97157	0.2376734	135.7312	0.1996715	278.714483	0.458073
KAZE	610.53	0.78191	438.3519	0.57164	374.5302	0.3938
SIFT	516.990857	0.6704241	343.69814	0.45103	491.07417	0.50673
affine						
FFD	36.3484	0.0425155	36.0769	0.042033	137.078	0.167835
KAZE	365.213	0.342425	317.102	0.291897	108.291	0.0974645
SIFT	177.472	0.144875	143.126	0.116743	250.523	0.201091
blurred						
FFD	52.3876	0.175841	48.8114	0.170414	81.9396	0.236964
KAZE	270.387	**1.45613**	200.719	1.10534	145.168	0.316967
SIFT	201.485	**1.79936**	143.847	1.21143	123.665	0.373325
compressed						
FFD	420.334	0.512681	345.174	0.42846	516.813	0.631823
KAZE	1031.62	0.938747	706.787	0.649803	714.483	0.603008
SIFT	799.534	0.622841	545.879	0.434184	892.386	0.69102
cropped						
FFD	128.765	0.279279	110.485	0.247958	297.03	0.900166
KAZE	400.821	0.754206	289.075	0.557304	332.693	0.580305
SIFT	324.011	0.490533	212.789	0.335309	508.063	0.87685
noise						
FFD	40.062	0.0608043	38.1881	0.0581252	88.2153	0.140304
KAZE	489.798	0.446638	372.633	0.341295	341.607	0.27455
SIFT	204.861	0.171351	156.97	0.131674	245.24	0.193503
rotated						
FFD	220.951	0.276136	200.07	0.252676	551.211	0.671348
KAZE	927.391	0.832065	640.305	0.576969	604.939	0.490303
SIFT	959.283	0.735301	629.959	0.495245	926.568	0.704561
scaled						
FFD	269.953	0.316457	171.313	0.198034	-	-
KAZE	788.48	0.703151	541.842	0.478876	-	-
SIFT	952.29	0.728708	573.317	0.432636	-	-

Table 2: Results of the detector evaluation

Where #`rep. kps` means the number of detected repeatable keypoints, `rep.` the repeatability, #`nr-rep. kps` the number of non redundant detected repeatable keypoints, and `nr-rep.` the non redundant repeatability.

4.2 Descriptor Evaluation

Next, the results of the descriptor evaluation. The same dataset has been employed. To obtain the `recall` vs `1-precision` curves, the threshold values are first set to 0, and slowly incremented. Because of the stringent thresholds early in the process, virtually no matches are kept. When the thresholds start rising, more and more matches will be deemed correct. A competent detector hits a `recall` value of 1 early in the process, when the `1-precision` is still low. A less competent detector will only achieve correct matches with a high threshold when many false positives are kept as well.

The curves of each separate transformation will be compared, to investigate the effect of each transformation on the descriptors' exactness.

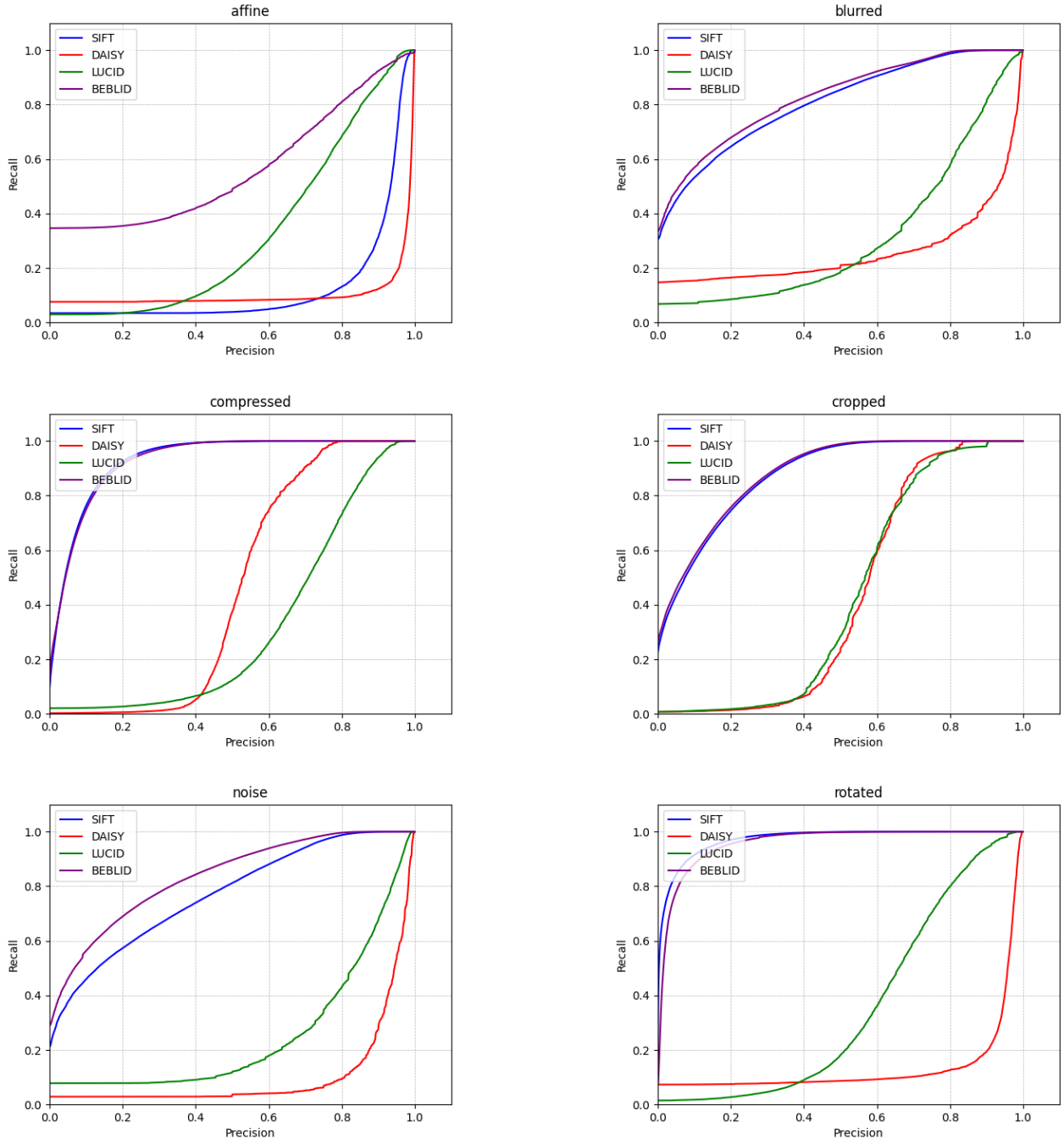

Figure 12: Recall vs 1-precision for viewpoint(a), blurred(b), compressed(c), cropped(d), noise(e), and rotations(f)

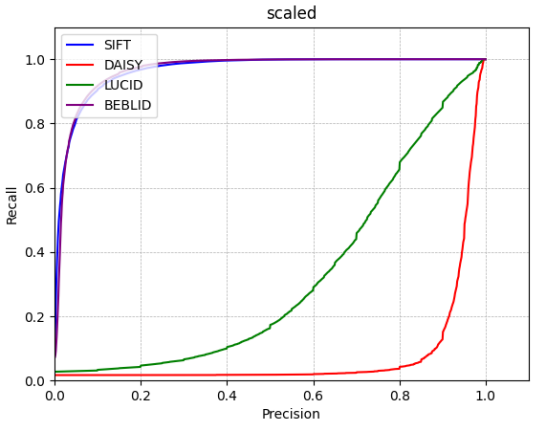

Figure 13: Recall vs. 1-precision scaled

All descriptors suffer the most from viewpoint changes. For SIFT and BEBLID, the descriptors seem to perform well on scales, rotations, and image compressions and reasonably for the other modifications. LUCID and DAISY seem sensitive to all modifications, while LUCID performs slightly better under transformations, and DAISY outmatches LUCID for compressed images.

4.3 Copy Detection

For the next section, the same detector/descriptor pairs are evaluated, this time on the copy detection dataset. In addition, two other pairs are evaluated. First, the baseline FFD detector with the SIFT descriptor, and secondly, the modified FFD detector with the SIFT descriptor. The reason for this is to test the efficacy of the modified FFD detector compared to the baseline method. Later on in this section, the two will be evaluated separately from the other descriptors.

To test how well a descriptor performs on detecting copies, we will look at the percentage of detected copies at each threshold rank. Since varying this rank also lowers the specificity of the descriptor, it is also helpful to investigate how the `precision` is affected by varying the rank.

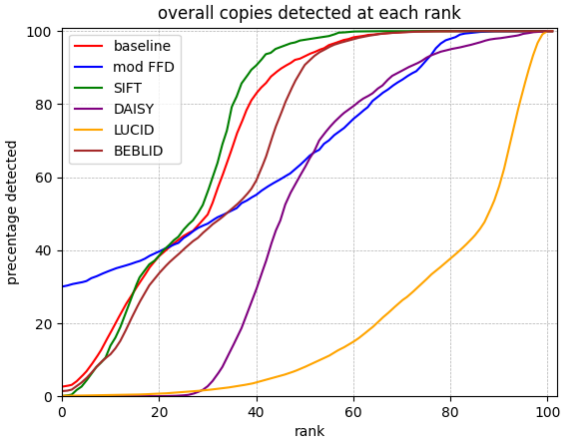

Figure 14: Percentage of copies detected per rank

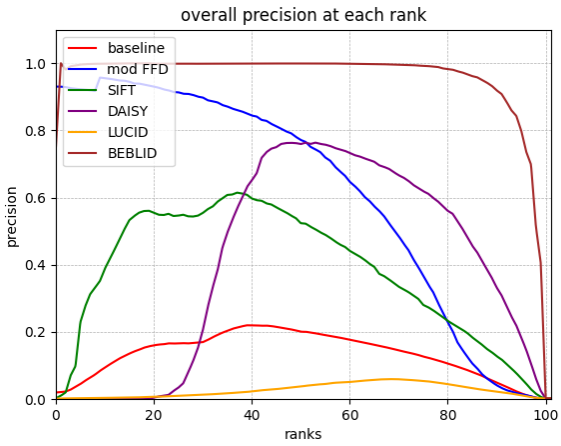

Figure 15: Precision per rank

Figure 14 shows the percentage of copies found at each rank. Again, we determine if a copy is found if $\#correct\_matches > (100 - i) * min(kp1, kp2)$, where i is the rank where the copy is found. Figure 15 shows the `precision` at each rank. The SIFT descriptor is able to find all copies at the lowest rank, followed by BEBLID. LUCID is only able to find all copies when the threshold to determine an image match is very low. For the `precision` at each rank, we see that the initial specificity of most descriptors is high at the rank the first copies are found. LUCID is the exception since it only starts finding copies at a high rank, where the threshold of determining a copy is less stringent, and more non-copy images are being labeled as copies. The BEBLID descriptor performs very well based on its `precision` across the ranks being around the max in the range the first copies are found, and the rank hits 100.

Finally, to show how well each descriptor performs per transformation, Table 3 shows the AUC score for each transformation.

descriptor	affine	blurred	compress	cropped	noise	rotated	scaled
BEBLID	57.615	69.042	92.407	83.510	74.292	57.908	55.294
DAISY	45.328	44.402	65.509	60.756	59.447	45.898	41.730
FFD	63.778	77.717	93.459	86.912	73.421	63.851	60.018
LUCID	10.458	5.793	45.467	40.395	26.381	10.821	8.359
modFFD	53.871	71.899	87.728	81.567	73.689	55.485	49.103
SIFT	66.516	74.915	93.241	86.130	77.775	65.466	67.104

Table 3: Average keypoint scale per transformation

To demonstrate what image types different descriptors commonly confuse with the query image while detecting copies, Figure 16 is presented:

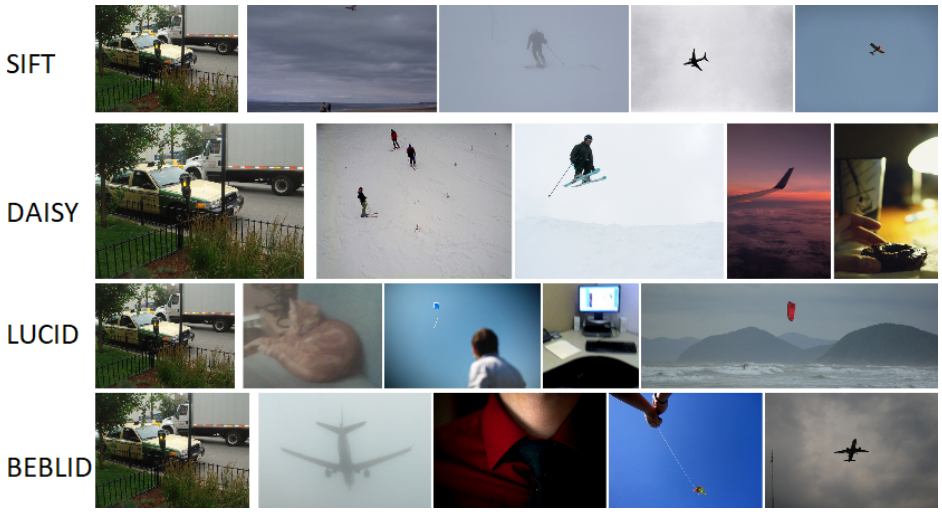

Figure 16: Images typically confused with a query image. From left to right: the first image is the query image, the next two images are deemed identical to the query image (100% of matches are correct), the fourth image is an image found at a low rank (10-20), the last image at a high rank(65-70).

The central theme with images confused as copies is foggy images with few distinctive characteristics. LUCID is the main exception. While the confused images are somewhat blurry, recognizable entities are present in the images, even at low ranks. With DAISY, this starts happening with images marked as a copy at higher ranks. BEBLID confuses an image with contrasting color to the query image with a copy, which is remarkable, since it is the descriptor with the highest overall precision. A reason for this could be that the BEBLID descriptor does not put a high emphasis on color.

4.4 Evaluating modFFD

4.4.1 Descriptor Evaluation

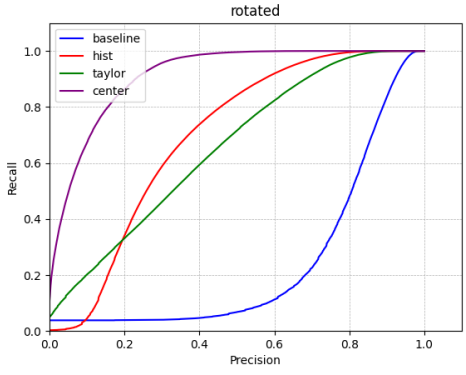

Figure 17: Recall vs. 1-precision FFD variants

technique	AUC	average time per image(ms)
baseline	0.2414	57.0373
histogram	0.6945	240.794
Taylor's	0.6558	61.9887
center of mass	0.9134	106.899

Table 4: Detector metrics

In Figure 17, the `recall` vs `1-precision` of the modified FFD versions are shown, while in Table 4, the area under curve and average time to detect all keypoints in an image are shown. The first thing that becomes clear is that the baseline method is in no way rotation invariant, which was to be expected. The center of mass method performs the best by far and hits a `recall` value of 1 at a relatively low precision value. The other two methods perform better than the baseline, but the histogram method does not provide enough stability for matching rotating images compared to the extra time cost. A case can be made for Taylor's method, while it performs worse than the histogram method. With Taylor's method, only 4 ms are required extra to provide a somewhat robust orientation assignment, which makes it preferable to use for real-time matching, especially if combined with a binary descriptor that can use the orientation assignment. A reason for the lacking performance of the histogram orientation assignment could be since the method is developed for usage with SIFT keypoints. SIFT detects keypoints at certain octaves, and when constructing the orientation histogram, it uses the Gaussian image of the octave where the keypoint is detected. This feature is added to make the orientation assignment more resistant to noise[Low04]. FFD does not detect keypoints at octaves, and therefore calculating the orientation this way does not provide any benefits or even hamper providing stability for matching rotated images.

4.4.2 Copy Detection

The first peculiar thing is the amount of copies modFFD can find from rank 0. The baseline method quickly catches up to modFFD and can find each copy at a relatively low rank. At first glance, the baseline seems preferable to the modified version; however, when we look at the specificity of the two detectors, a case can be made for the modified version. The modified version retains a very high specificity, while it can find a decent amount of copies before the `precision` starts to drop. The FFD detector has initially been modified to make the detector rotation invariant with the added orientation of the SIFT detector. If we take a look at Table 3, the AUC for rotations is lower than the original implementation. At least for detecting rotated copies, the orientation implementation did not succeed.

4.5 General Statistics

Finally, some miscellaneous metrics for the detectors and descriptors:

detector	average kps detected	average time per 1000 kps(ms)
FFD	1879.579	63.929
KAZE	1958.259	309.673
SIFT	2460.642	92.361

Table 5: Detector metrics

Table 5 shows the number of keypoints each detector detects on average for each image in the datasets and the average time it takes to detect 1000 keypoints in milliseconds.

detector	descriptor type	descriptor size	description time	matching time
BEBLID	Binary	32/64	14.038	46.643
DAISY	Real	800	250.032	25.604
LUCID	Binary	3.375	2.937	47.935
SIFT	Real	512	210.813	23.521

Table 6: Descriptor metrics

Table 6 shows the type of the descriptors that are computed (binary or real), the storage in bytes per descriptor, the time it takes on average to describe all keypoints detected in the image in ms, and the average time it takes to match descriptors and filter suitable matches in ms. What is interesting to note, is that the description time is heavily associated with the size of the produced descriptor. The matching time, however, is faster than those of binary descriptors, probably due to the efficient implementation of FLANN compared to the bruteforce LSH index matching.

5 Conclusions and Further Research

In this study, a set of interest point detectors and descriptors was evaluated on a dataset containing multiple transformations ranging from slight to highly disruptive. The KAZE detector seemed to provide the highest amount of stable keypoints based on the overlap repeatability criterion, while SIFT outmatched KAZE if the distance criterion was used. The non-redundant repeatability rate proved to be a helpful criterion for determining the amount of distinct stable keypoints a detector could localize. For the descriptor evaluation and copy detection evaluation, the SIFT descriptor proved to be the most efficient regarding most transformations, closely followed by BEBLID. BEBLID showed to be more robust under affine transformations compared to other descriptors. SIFT outperforming the other descriptors overall might seem strange considering the age of the method. The characteristics of the other descriptors can explain the reason for this. First of all, LUCID is a

binary descriptor that only uses gray-scale intensity to compute its descriptors. While it provides a very fast descriptor, it disregards scales the keypoints are detected at and keypoints' orientations, making it not invariant to these transformations. DAISY performing worse than SIFT could be because it is designed for dense matching, and the context of this research was matching using well-localized keypoints. If it were to be used with dense keypoints, it might improve the performance of DAISY. Finally, BEBLID is outperformed by SIFT but only slightly. It shows the exact invariance as SIFT and is more efficient since it is a binary descriptor. It also performs better on affine transformations, making it an excellent descriptor when real-time matching is required and a good performing descriptor otherwise. An existing method, namely the FFD detector, was modified to include an orientation of the interest point region as an indication for the descriptor to use. Rotation invariance has been achieved using the center of mass method, and the performance of rotated images has been improved with the orientation histogram method and Taylor's method. While it was not established that the orientation histogram method improved copy detection for rotated images, it helped detect copies if the threshold determining a matched image is a copy was stringent.

For future research, the following concepts would be interesting to investigate. First of all, a more theoretical evaluation could give more insight into how detectors and descriptors operate. For instance, why do interest point detectors find keypoints at a larger scale for blurred images than for other transformations? Also, why does the component that makes the SIFT detector invariant to rotations not help make a detector that works similarly to SIFT robust to rotations? Finally, a follow-up study could be carried out to investigate how the orientations assigned to the keypoints change for different rotations. Secondly, for designing the dataset, the parameters for changing the viewpoint were chosen to be broad without having too many permutations. It is a real possibility that some of the resulting images have a viewpoint that is unlikely to occur in real life. A new study investigating which parameters result in plausible viewpoints would help design new datasets for comparative studies.

References

[ABD12] Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J. Davison. Kaze features. *Computer Vision – ECCV 2012*, pages 214–227, 2012.

[ACM11] Alberto Argiles, Javier Civera, and Luis Montesano. Dense multi-planar scene estimation from a sparse set of images. *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4448–4454, 2011.

[APAJ12] R. Ahila Priyadharshini, S. Arivazhagan, and S. Jeypriya. Object recognition with wavelet-based salient points. *Global Trends in Information Systems and Software Applications*, pages 532–541, 2012.

[CS00] Christian Bauckhage Cordelia Schmid. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37:151–172, 2000.

[DMR17] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. *CoRR*, abs/1712.07629, 2017.

[EA14] H. El-Bakry E. Adel, M. Elmogy. Image stitching based on feature extraction techniques: A survey. *International Journal of Computer Applications*, 99(6):1–8, 2014.

[GLT21] Morteza Ghahremani, Yonghuai Liu, and Bernard Tiddeman. Ffd: Fast feature detector. *IEEE Transactions on Image Processing*, 30:1153–1168, 2021.

[GTH11] Steffen Gauglitz, M. Turk, and Tobias Höllerer. Improving keypoint orientation assignment. *BMVC*, 2011.

[HS88] C. G. Harris and M. Stephens. A combined corner and edge detector. *Alvey Vision Conference*, 1988.

[KM02] Cordelia Schmid Krystian Mikolajczy. An affine invariant interest point detector. *European Conference on Computer Vision (ECCV '02),*, 7:128–142, 2002.

[KM04] Cordelia Schmid Krystian Mikolajczy. Scale affine invariant interest point detectors. *International Journal of Computer Vision*, 60:63–86, 2004.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. *Computer Vision – ECCV 2014*, pages 740–755, 2014.

[Low04] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[LSP05] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, 2005.

[LTCJ+07] Julien Law-To, Li Chen, Alexis Joly, Ivan Laptev, Olivier Buisson, Valerie Gouet-Brunet, Nozha Boujemaa, and Fred Stentiford. Video copy detection: A comparative study. *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, page 371–378, 2007.

[MS05] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[RDM14] Ives Rey-Otero, Mauricio Delbracio, and Jean-Michel Morel. Comparing feature detectors: A bias in the repeatability criteria, and how to correct it. *CoRR*, abs/1409.2465, 2014.

[RH99] T. Randen and J. H. Husøy. Filtering for texture classification: A comparative study. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21:291–310, 1999.

[RH04] Andrew Zisserman Richard Hartley. *Multiple View Geometry in Computer Vision.* Cambridge, 2004.

[SG11] Tobias Höllerer Steffen Gauglitz. Evaluation of interest point detectors and feature descriptors for visual tracking. *International Journal of Computer Vision*, 94:335–360, 2011.

[SSBB20] Iago Suárez, Ghesn Sfeir, José M. Buenaposada, and Luis Baumela. Beblid: Boosted efficient binary local image descriptor. *Pattern Recognition Letters*, 133:366–372, 2020.

[STLL03] Nicu Sebe, Qi Tian, Etienne Loupias, and Michael Lew. Evaluation of salient point techniques. *Image and Vision Computing*, 21:1087–1095, 01 2003.

[TD11] S. Taylor and T. Drummond. Binary histogrammed intensity patches for efficient and robust matching. *International Journal of Computer Vision*, 94:241–265, 2011.

[TH98] Miroslav Trajković and Mark Hedley. Fast corner detection. *Image and Vision Computing*, 16(2):75–87, 1998.

[THBL08] Bart Thomee, Mark J. Huiskes, Erwin Bakker, and Michael S. Lew. Large scale image copy detection evaluation. *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, page 59–66, 2008.

[TLF10] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):815–830, 2010.

[WOBL17] Song Wu, Ard Oerlemans, Erwin M. Bakker, and Michael S. Lew. A comprehensive evaluation of local detectors and descriptors. *Signal Processing: Image Communication*, 59:150–167, 2017.

[YTX+19] Pei Yan, Yihua Tan, Yuan Xiao, Yuan Tai, and Cai Wen. Unsupervised learning framework of interest point via properties optimization. *CoRR*, abs/1907.11375, 2019.

[ZCKB12] Andrew Ziegler, Eric Christiansen, David Kriegman, and Serge Belongie. Locally uniform comparison image descriptor. *Advances in Neural Information Processing Systems*, 25, 2012.

[ZR11] C. Lawrence Zitnick and Krishnan Ramnath. Edge foci interest points. *2011 International Conference on Computer Vision*, pages 359–366, 2011.