Place Recognition with ConvNet Landmarks

Wael Mousa and Ali Khalayly

June 2020

1 Introduction

Place recognition and object detection use has expanded in recent years, with a variety of techniques used to solve a large scale of problems, from simple to more complex ones. However, all the different approaches that are used involve significant compromises, and cannot solve many of the challenges that stand in its way, including robustness to environmental changes, different viewpoints of a certain scene and dynamic objects, so machines that are built to solve this issue do not provide accurate results as we might expect.

The project we are discussing in this article implements the approaches that were presented in "Place Recognition with ConvNet Landmarks: Viewpoint-Robust, Condition-Robust, Training-Free" [1], including minor changes and several additions that will be explained later on. We found out, that the approaches in this article lead to positive results in terms of robustness to environmental changes, different viewpoints and dynamic objects, compared to other approaches. We depend on the astonishing power of convolutional neural network (CNN) features to identify matching landmark proposals between images to perform place recognition over extreme appearance and viewpoint variations. By conducting several experiments on the data sets that are used in [1], we achieved similar results, which backs up the idea that the use of convolutional neural networks and this state of the art technique is superior to the previous approaches that we mentioned before.

Our project contributions are the following:
1) Implementing a place recognition system that is robust to viewpoint and appearance variation, and testing it on different data sets.
2) New challenging addition, which limits the size of the boxes that

we extract, and use only these boxes to solve the place recognition problem, in order to test the robustness of CNNs depending only on tiny objects in the photos.

2 System Components

In this section, we will present the four critical components of our module:

1. Edge Boxes

In order to extract the landmarks of each image, we apply Edge Boxes [2] algorithm on it. Edge boxes finds a set of boxes, calculates the score of each box by "measuring the number of edges that exist in the box minus those that are members of contours that overlap the box's boundary", which indicates the likelihood of it containing an object, and returns a set of the highest scored bounding boxes which surround each landmark, its coordinates on the image and its score. In our implementation, we extracted 50 or 100 boxes for each image, and the whole set of boxes was the input of the CNN.

2. AlexNet layer

After extracting the boxes from the images, the boxes (as vectors) are resized to the expected input size of 231 x 231 x 3 pixels and then inserted into a convolutional neural network (AlexNet) in order to calculate a feature vector to describe the landmarks' appearance. The whole idea was that this vector should be robust to changes in weather, light, time of day, movement of dynamic objects and changes in perspective.
[1] has shown that the third convolutional layer of AlexNet is highly invariant against environment changes, therefore we used an implementation of AlexNet network that was pre-trained on ImageNet data set, and modified it so that only the layers up to the third layer (conv3) are calculated. Each one of the feature vectors is a vector of 64,896 dimensional feature.

3. Gaussian Random Projection

As we have seen earlier, the operations which were executed on each box resulted to a 64,896 dimensional feature. Afterwards, we must calculate the pairwise cosine distance between each two im-

ages' vectors. Working with such high dimensions is a very expensive operation and slows the process, and here stands the importance of Gaussian Random Projection, which is one approach to reduce the number of the dimensions of a vector without changing the distance between pairs of existing vectors drastically. We applied the Gaussian Random Projection to transform the original vectors to a lower-dimension vectors, we relied on the Johnson-Lindenstrauss-Lemma [3] which says that a set of points in high dimensional space can be transferred to a lower dimensional space while maintaining the euclidean distances up to a marginal factor.
The outcome of this step is a projection of each original 64,896 dimensional feature to 1024, 4096 dimensions, which were later compared.

4. Landmark matching and cosine distance calculation

Our goal was to match between pairs of similar images. To achieve that, we need to perform matching between each image and their 100 boxes. For each box (projected vector) of image A, we perform a nearest neighbor search based on cosine distance to find the best matching vector in image B, and vice versa. We do the same operation on each vector of image B to find the best matching vector in Image A. Afterwards, we perform a crosschecking and only mutual matches are accepted (as explained in article [1]).
After finding the matched vectors pairs, we score each pair by calculating shape similarity and overall similarity according to two given formula from article [1], and then we sum the similarity of all matched pairs of those two images and divide by a constant factor (explained in [1] article).
To retrieve the best matching image C from our database for a source image A, we calculate the similarity between A and all the images in the database, and return image C such that the pair (A,C) has the highest similarity score of all pairs.

3 Reproducing article results and comparisons

The implementation that is discussed in [1] was tested on many different datasets. In order to develop the current model and add more

extensions to it, we had to make sure that we can achieve similar results compared to the article. we performed tests on some of the datasets which were mentioned above:

*here we draw the graph of Precision as a function of Recall, where Recall is the number of True Positive images divided by the sum of True Positive plus False Negative images, and Precision is the number of True Positive images divided by the sum of True Positive plus False Positive images. An image is labeled true if it matches the original photo it was compared to (same place), and is labeled positive if it's similiraty to the original image is higher by a factor than the second most similar image to the orignal image.

1) GardensPointWalking dataset:

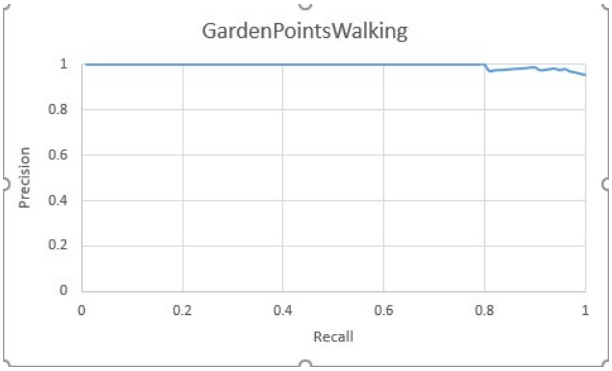

Figure 1: Our results

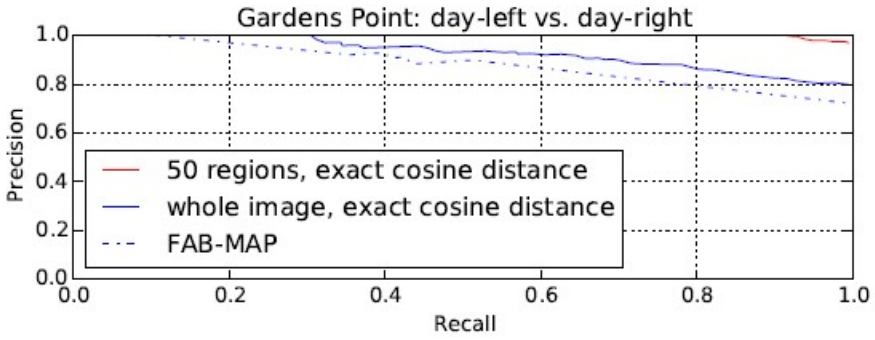

Figure 2: Article results

We performed the test with a limitation of 50 boxes per image, 1024 GRP, and we can see from the graph that our results are similar to the matching plot.

2) Kurfürstendamm dataset:

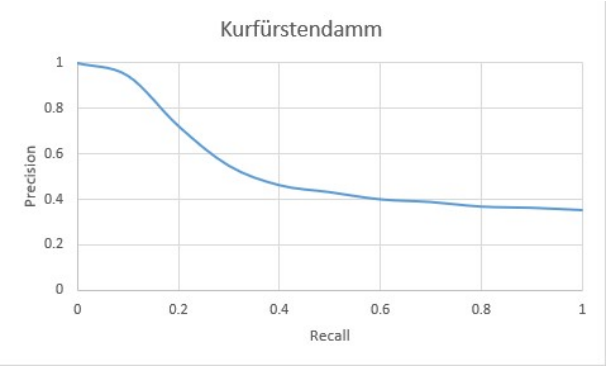

Figure 3: Our results

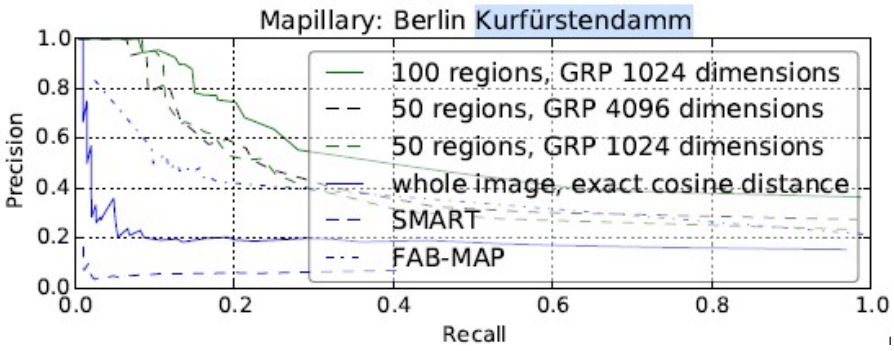

Figure 4: Article results

We performed the test with a limitation of 100 boxes per image, 1024 GRP, and we can see from the graph that our results are similar to the matching plot.

3) Nordland Train dataset:

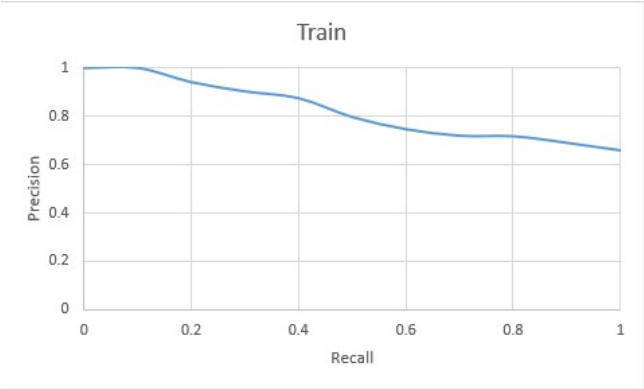

Figure 5: Our results

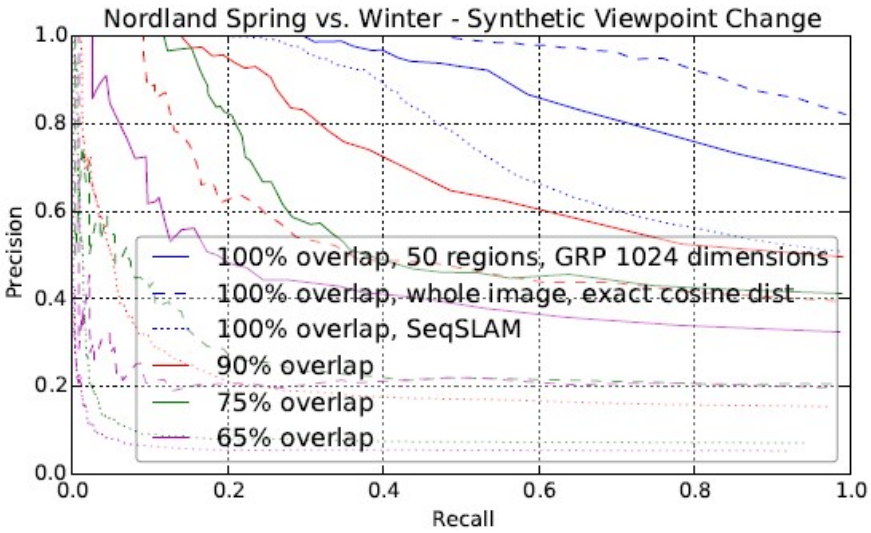

Figure 6: Article results

We performed the test with a limitation of 50 boxes per image, 1024 GRP, and we can see from the graph that our results are similar to the matching plot.
*In this test, we did not use the whole dataset due to the size of the dataset, therefore we did not get 100 percent matching results.

4 Additional features to the article's module

We wanted to check whether we can limit the size of the bounding boxes and still get accurate results. This was made in order to test the robustness of CNNs depending only on tiny objects; this addition will help us identify real world places relying only on its tiny objects, which will lead to a significant decrease in runtime compared to the runtime of operations made on bigger boxes. We performed several tests on the "GardenPointWalking" dataset. In each test, we limited the maximum area of the boxes which are returned by the EdgeBoxes algorithm, and then followed the same steps as explained in Section 2. The following graph shows the results that were received (each plot represents a test that run with a limit of X area value whereas X is the value that matches the color of the plot):

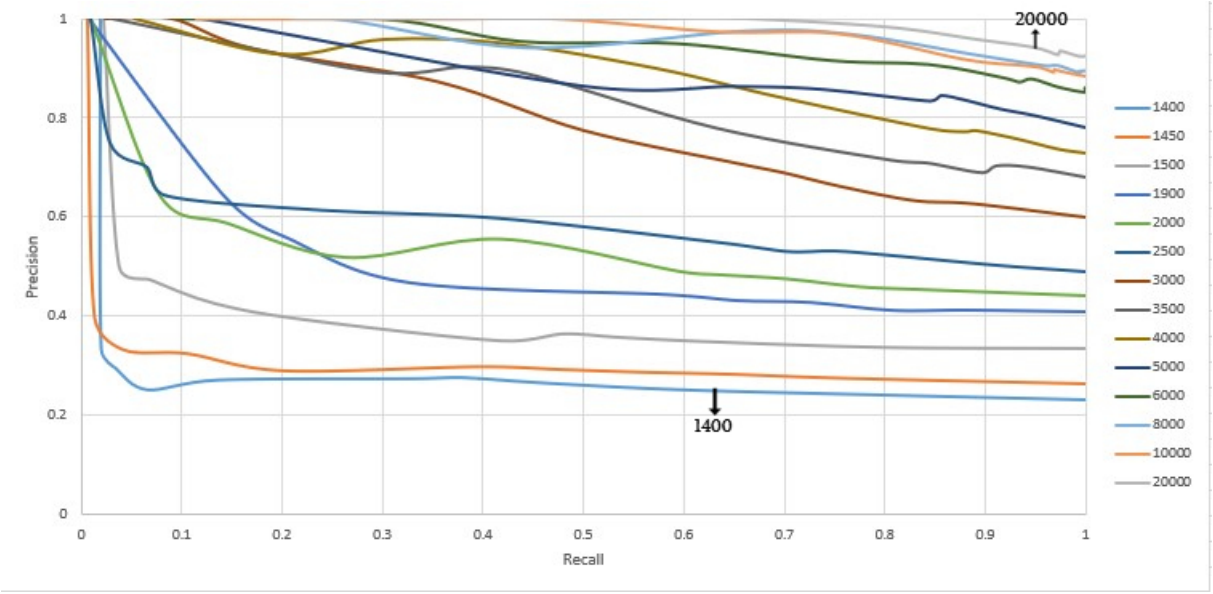

Figure 7: Results

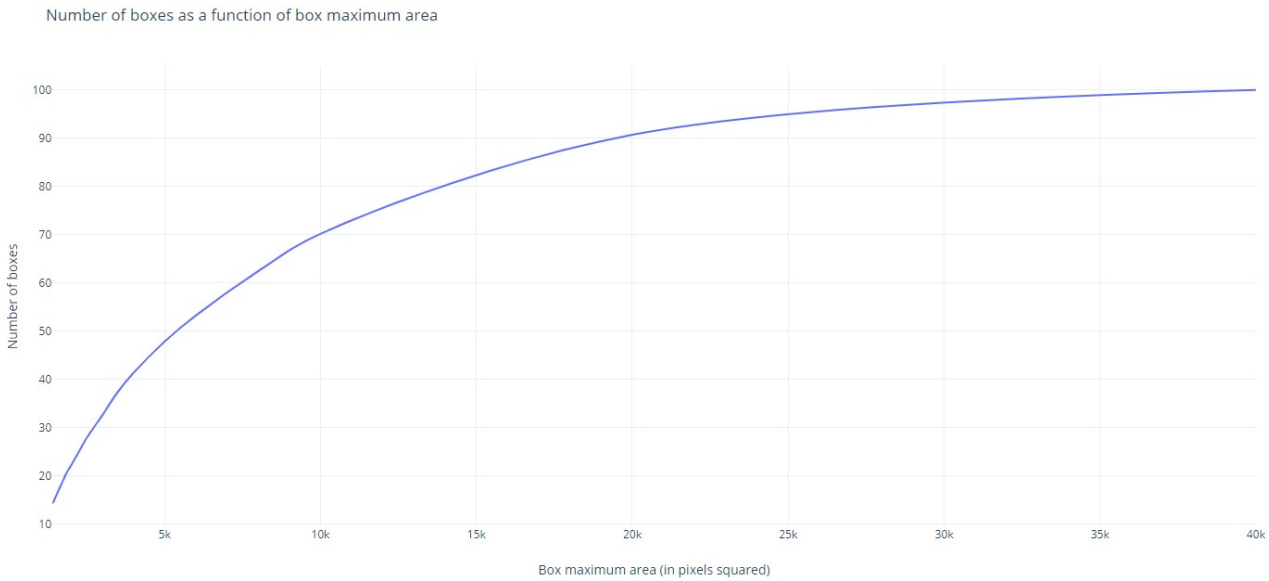

Figure 8: Number of boxes as a function of box maximum area

As we can notice in figure 8, when we decreased the maximum box area, we received less boxes which has a major impact on our results in figure 7 and is the main reason why we get lower precision when we set a limit on the maximum box area.

5 Summary

In our project, we managed to get similar results to [1] as described in previous sections. Regarding the additional feature which we are presenting here, as we can see in figure 7 and figure 8, the results got worse (lower precision) as we decreased the maximum area limit as expected, because we received less boxes to compare (according to figure 8). On the other hand, we can notice that there were some plots with a low area limit which yielded great results. These results provide a positive starting point for future additions in the field of place recognition.

6 Alternatives to Edge Boxes

In our project, we used Edge Boxes as an algorithm to extract boxes which contain objects. As we mentioned in the previous paragraph, the smaller the limitation on the boxes area, the less boxes we manage to extract.
This issue lead to a degradation of the precision metric, as the small number of boxes from each image does not provide enough data. In addition, at some point (when the area of the boxes was limited to a maximum of less than 1400), we stopped receiving boxes at all. This prevented us from testing the robustness of CNNs based on tiny objects as described before. Therefore, we wanted to find an alternative to Edge Boxes which may solve either of these two issues.

6.1 SIFT - Scale-Invariant Feature Transform

The main algorithm that we focused on was SIFT:
SIFT is invariance to image scale and rotation and there are mainly four steps involved in the SIFT algorithm:
1) Scale-space peak selection: Real world objects are meaningful only at a certain scale. Scale-space is separated into octaves and the number of octaves and scale depends on the size of the original image. So we generate several octaves of the original image. Each octave's image size is half the previous one. Within an octave, images are progressively blurred using the Gaussian Blur operator. Now we use those blurred images to generate another set of images, the Difference of Gaussians (DoG). These DoG images are great for finding out interesting keypoints in the image. One pixel in an image is compared with its 8 neighbors as well as 9 pixels in the next scale and 9 pixels in previous scales. This way, a total of 26 checks are made. If it is a local extrema, it is a potential keypoint. It basically means that keypoint is best represented in that scale.
2) Keypoint Localization: The amount of Keypoints generated in the previous step is substantial. Some of them lie along an edge, or they don't have enough contrast. In both cases, they are not as useful as features, so we get rid of them. Taylor series expansion of scale

space is used to get a more accurate location of extrema, and if the intensity at this extrema is less than a threshold value, it is rejected.
3) Orientation Assignment: Now we have legitimate keypoints, they've been tested to be stable. We already know the scale at which the keypoint was detected. A neighborhood is taken around the keypoint location depending on the scale, and the gradient magnitude and direction is calculated in that region. An orientation histogram with 36 bins covering 360 degrees is created. Let's say the gradient direction at a certain point (in the "orientation collection region") is 18.759 degrees, then it will go into the 10–19-degree bin. And the "amount" that is added to the bin is proportional to the magnitude of the gradient at that point. The highest peak in the histogram is taken and any peak above 80 percentage of it is also considered to calculate the orientation. It creates keypoints with same location and scale, but different directions. It contributes to the stability of matching.
4) Keypoint descriptor: At this point, each keypoint has a location, scale, orientation. Next is to compute a descriptor for the local image region about each keypoint that is highly distinctive and invariant as possible to variations such as changes in viewpoint and illumination. To do this, a 16x16 window around the keypoint is taken. It is divided into 16 sub-blocks of 4x4 size. For each sub-block, 8 bin orientation histogram is created. So 4 X 4 descriptors over 16 X 16 sample array were used in practice. 4 X 4 X 8 directions give 128 bin values. It is represented as a feature vector to form keypoint descriptor. To achieve rotation independence, the keypoint's rotation is subtracted from each orientation. If we threshold numbers that are big, we can achieve illumination independence. So, any number (of the 128) greater than 0.2 is changed to 0.2. This resultant feature vector is normalized.

We performed several tests (one test for each area size of: 100, 200, 400, 600, 900, 1400) using this algorithm on our dataset ("GardenPointWalking") as follows:
First we extracted the keypoints of each image with its descriptors, then we limited the radius of each keypoint to a maximal bound, and then we sorted the keypoints by their 'response' values (this parameter implies how likely this keypoint will contain an object), and chose the highest scored 150 keypoints. We converted each key-

point into a square shaped box relying only on the keypoint's radius and centre point. From this stage and on, we performed the same steps as mentioned and explained before. We received the following results:

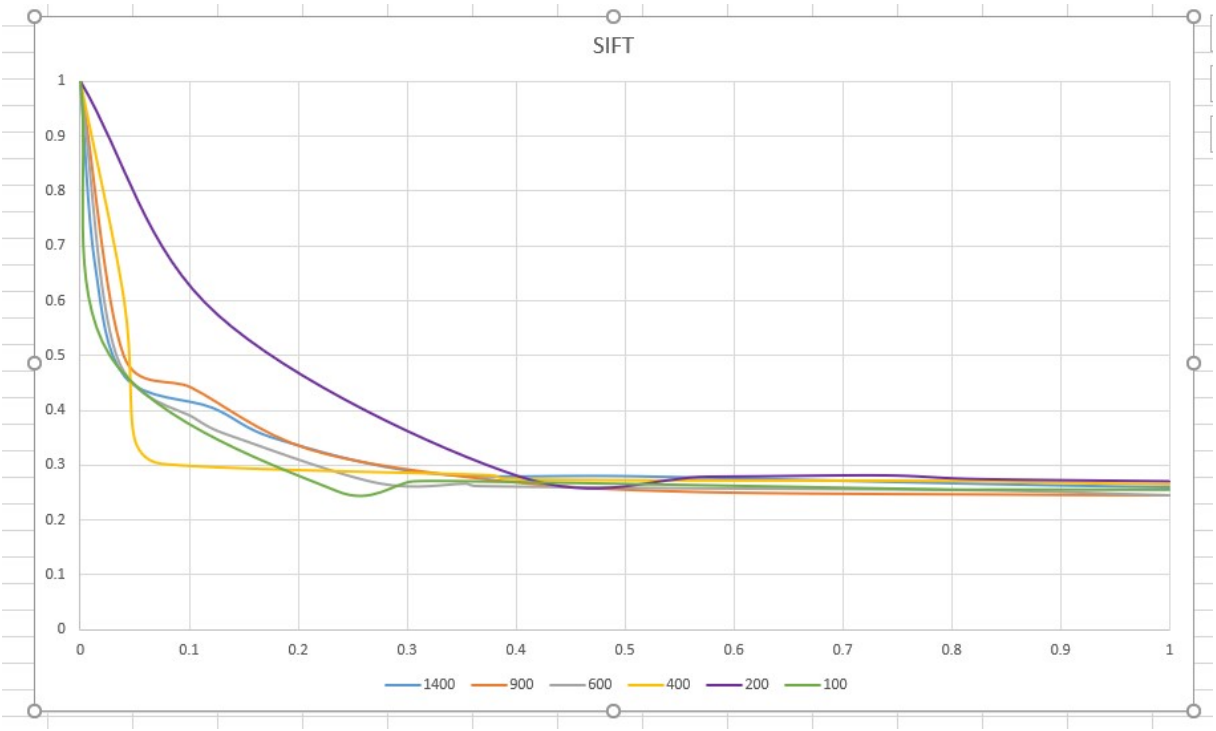

Figure 9: SIFT Results

As we can notice in figure 9, we achieved slightly better precision result (for the 1400 area limitation) and the precision remained stable regardless of how small the box was. As we can see, although we did not improve the precision results significantly, we did managed to get satisfying results for the second issue mentioned before. The results above show that SIFT has a great potential in detecting tiny objects and replacing Edge Boxes for this matter.

7 Future Thoughts

On the one hand, as we saw from the results, using Edge Boxes as an algorithm which extracts the image's features in our flow will lead to astonishing matching accuracy as long as the size of the boxes is not limited to a small number. On the other hand, we saw that this algorithm is weak when the boxes we want to extract are small, as the number of extracted boxes is not enough and the boxes don't provide sufficient information of the images we are matching. In contrast, replacing Edge Boxes with SIFT to extract keypoint features gave us better results for small boxes, as it both produced more boxes for each image, and the matching on the final stage was more accurate. Therefore, we think that building a dynamic matcher, which mixes between Edge Boxes and SIFT for extracting keypoints and features from the images, can improve the accuracy and be more useful especially when the boxes in the image are small. This matcher can start by running Edge Boxes to collect big boxes from the image, and if there is a small number of boxes which can be extracted from this approach, SIFT will be used to find smaller boxes which can give additional data on the images we are testing.

8 References

[1]: Niko Sunderhauf, Sareh Shirazi Adam Jacobson, Feras Dayoub, Edward Pepperell, Ben Upcroft, and Michael Milford. Place Recognition with ConvNet Landmarks: Viewpoint-Robust, Condition-Robust, Training-Free. ARC Centre of Excellence for Robotic Vision, Queensland University of Technology, Brisbane QLD 4001, Australia

[2]: C. Lawrence Zitnick and Piotr Dollar. Edge boxes: Locating object proposals from edges. In ECCV. European Conference on Computer Vision, September 2014.

[3]: William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. Contemporary mathematics, 26(189-206):1, 1984.

[4]: Introduction to SIFT (Scale-Invariant Feature Transform), OpenCV