Feature detection and matching

SIFT DESCRIPTOR FOR DETECTION, MATCHING AND PANORAMA CREATION

Aleix Solanes and Pablo Martínez | Computer Vision - MAI | November 2015

FEATURE DETECTION

Question. *Take at your own choice several keypoints that have been detected at different scales. Using the theory given in lectures, comment on the reasons of why do think that a keypoint has been detected at that position and at that particular scale. You may repeat the experiment with another image (such as 'river1') to understand what a significant keypoint is.*

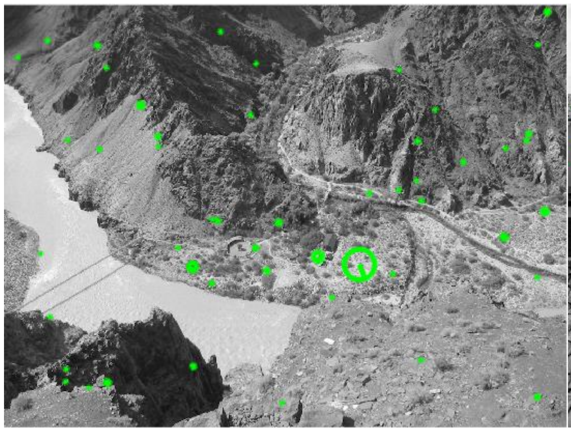
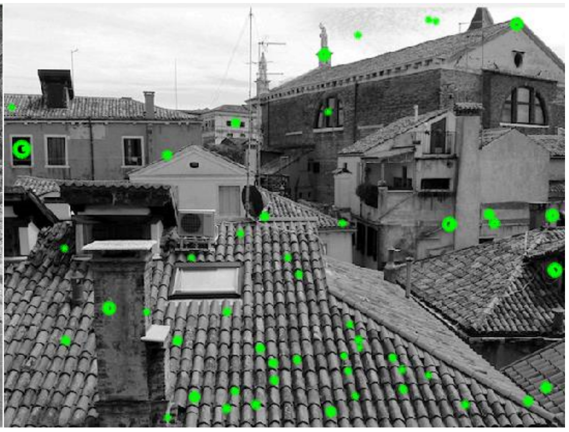
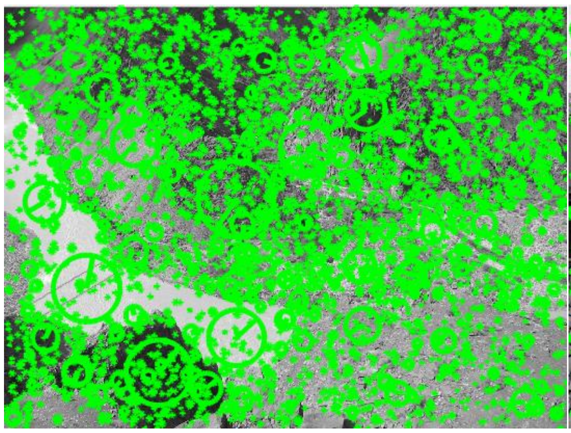
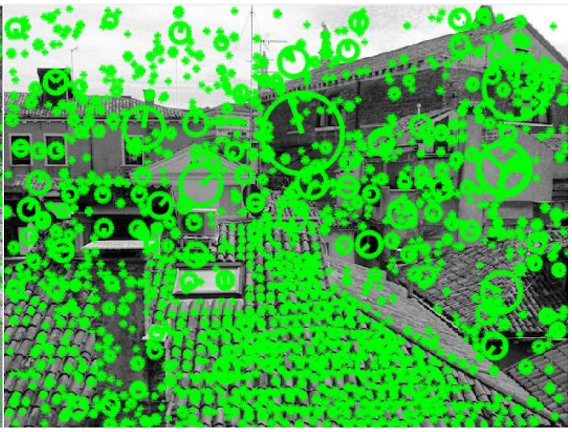



Figure 1 and 2: River and Roof image with 50 random keypoints

Figures 3 and 4: River and Roof images with all keypoints

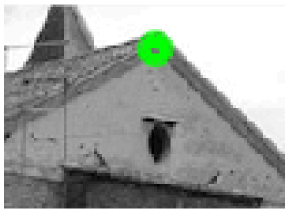


Figure 5: A keypoint from the roofs image.

The previous image (figure 5), shows a keypoint from the figure 2, that is an edge of the roof. This keypoint is selected because it is a stable corner. At this scale (the algorithm checks different scales to be able to detect corners and differentiate from edges with the aperture problem) the corner has a good unique minimum, and is a good candidate to be a keypoint.

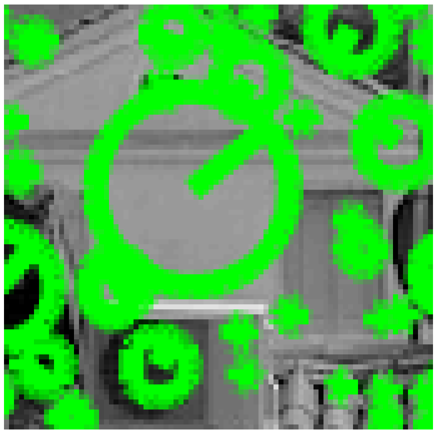


Figure 6: A big keypoint located at a higher scale

In the keypoint from figure 6, we can see that it was located at a higher scale, and that computing the orientation of the keypoint (by using the dominant orientation of the gradient orientation histogram) shows that the direction points to the upper right side. By determining the orientation of the keypoint, we can assure that if the properties of the description are measured relative to this orientation, this will provide invariance to rotation.

Test different Peak threshold values

Question*. Try to slowly increase or decrease the threshold. Comment why the number of detected keypoints decreases when the threshold is increased. Is this the expected behavior according to the way the threshold is defined?*

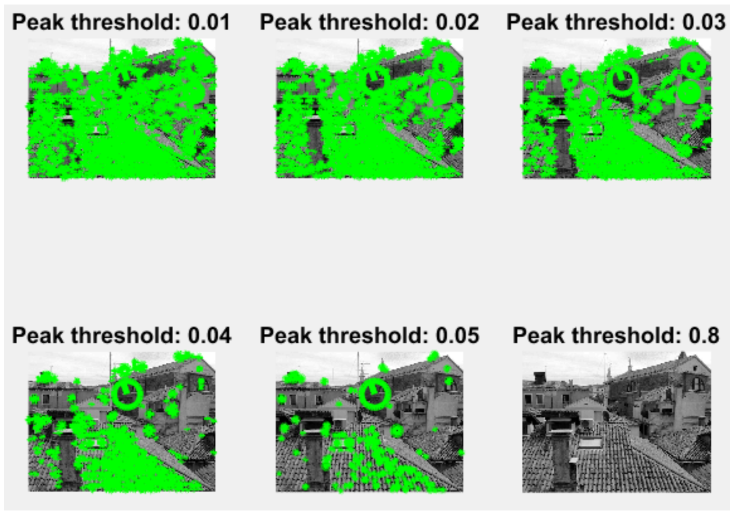


Figure 7: Testing different peak threshold

In the figure 7, we can see how the number of detected keypoints decreases when the peak threshold is increased. This happens because in order to check whether a point is significant or not, the candidates are tested by a two-step procedure, and the first one, is to test if the absolute value of the Difference-of-Gaussian image at the minimum or maximum X is greater than the given threshold. So, it is normal to think that it is more probable that there are a lot more key points that are higher than 0.01 comparing the ones that are higher than 0.8 which is a much higher value.

This threshold allows the algorithm to reduce the sensitivity to the noise of an image.

Test different Edge threshold values

Question. *Try to slowly increase or decrease the threshold. Comment why the number of detected keypoints increases when the threshold is increased. Is this the expected behavior according to the way the threshold is defined?*

Considering a peak threshold of 0.04:

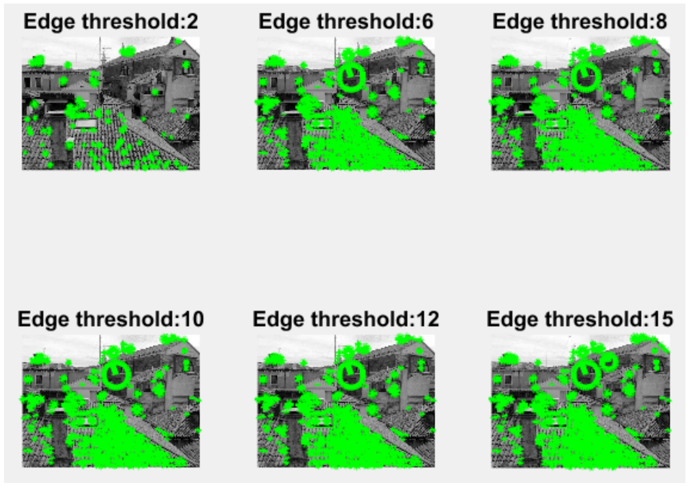


Figure 8: Different edge thresholds

In the above figure it is easy to see that the edge threshold produces a low number of keypoints, and a threshold of 15 produces a higher number of key points.

In order to see it more clearly the following table shows the number of keypoints for each plot of figure 8.

Edge threshold	Number of keypoints
2	165
6	680
8	743
10	771
12	801
20	838

Table 1: Number of keypoints for each edge threshold tested

To eliminate peaks of the Difference of Gaussian scale space which has a small curvature it is used the following formula, which defines a value K:

$$\kappa = \frac{\mathrm{Tr}(\mathrm{H})^2}{\mathrm{Det}(\mathrm{H})} = \frac{(\lambda_{max} + \lambda_{min})^2}{\lambda_{max}\lambda_{min}} = \frac{(r+1)^2}{r}$$

The value K will be lower when the two eigenvalues are equal, and it will increase with r. So when we has a value that accomplishes the following formula, the value will be accepted:

$$\frac{\mathrm{Tr}(\mathrm{H})^2}{\mathrm{Det}(\mathrm{H})} < \frac{(r+1)^2}{r}$$

So, when we have an edge threshold with a higher value, more values can reach the condition, so more keypoints will be detected.

FEATURE MATCHING

Question*. Modify the previous threshold. What is the effect of this threshold? Why do results improve as the threshold is increased? Comment your response.*

We have used the following code, and it is asked to modify the 2.0 value, which corresponds to a threshold which filters matches. The uniqueness between a pair is measured by dividing the distance between the best matching keypoint and the distance to the second best matching keypoint.

```
[matches2, scores2] = vl_ubcmatch(da, db, 2.0);
show_matches(Ia,Ib,fa,fb,matches2);
```

We have the following images:

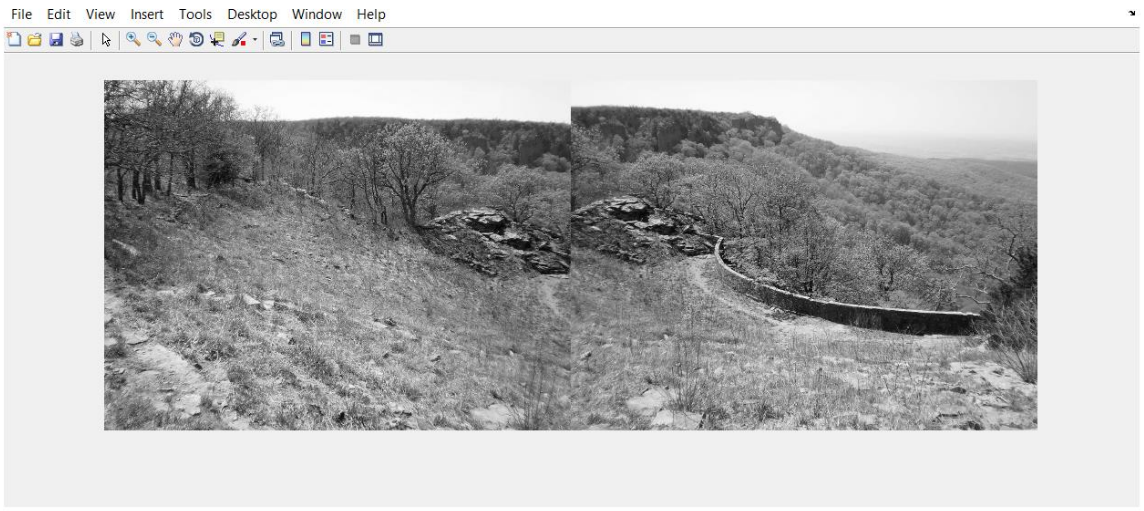


Figure 9: Both images that are going to be converted into a panoramic image

And we want to find the matches between the two images.
By modifying the threshold, the matching of both pairs are more or less equal. So a higher matching will result in a higher similarity between the two matches.
By applying a threshold of 2.0:

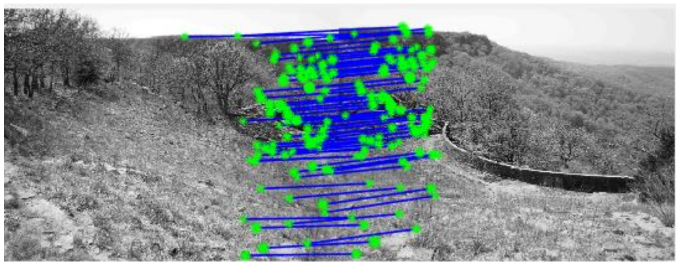


Figure 10: Applying a threshold of 2.0

If we increase this value to 3.0:

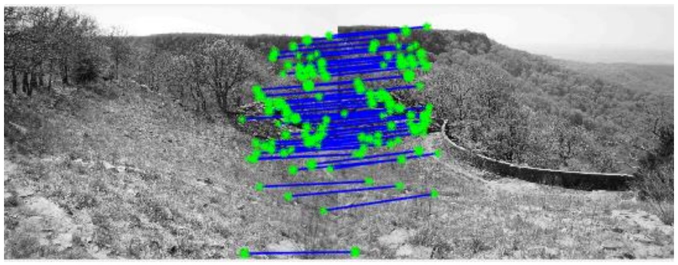


Figure 11: Applying a threshold of 3.0

With a value of 5.0:

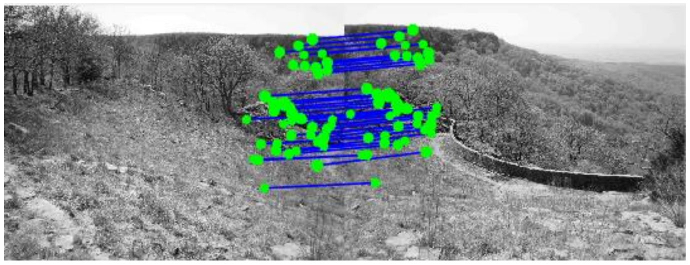


Figure 12: Applying a threshold of 5.0

If d1 is the nearest neighbour distance, and d2 the second nearest one:

- If the ratio d1/d2 is small, the match will be correct with the nearest neighbor
- If the ratio d1/d2 is high, the match won't be correct with the nearest one.

In function vl_ubcmatch(Descriptor 1, Descriptor 2, Threshold), this threshold forces the following:

- A descriptor D1 will be matched to a descriptor D2, only if the distance between both descriptors multiplied by the threshold is not greater than the distance of D1 against all other descriptors.

So, what we make by increasing the threashold, is to ensure the certainty of a value being the best matching for another descriptor. As a result, as it gets higher the threshold, the more we can assume the matching is the best one.

LINEAR MODEL

Question. *Comment on the obtained result. Is the resulting model (more or less) correct? By looking at the original images, do you think a linear model is enough to model the transformation between the two images?*

When considering a translational model, the distance of the translation will be the average over the observed displacements. If the difference between the two images is a translational motion, then this model would be OK. In this case the resulting model is not terrible, in fact it is more or less correct.

Despite the result being more or less correct, it is not enough to model the transformation between the two images, it is necessary to consider other transformations between both images.

By applying only the consideration of a translation motion, the result is the following:

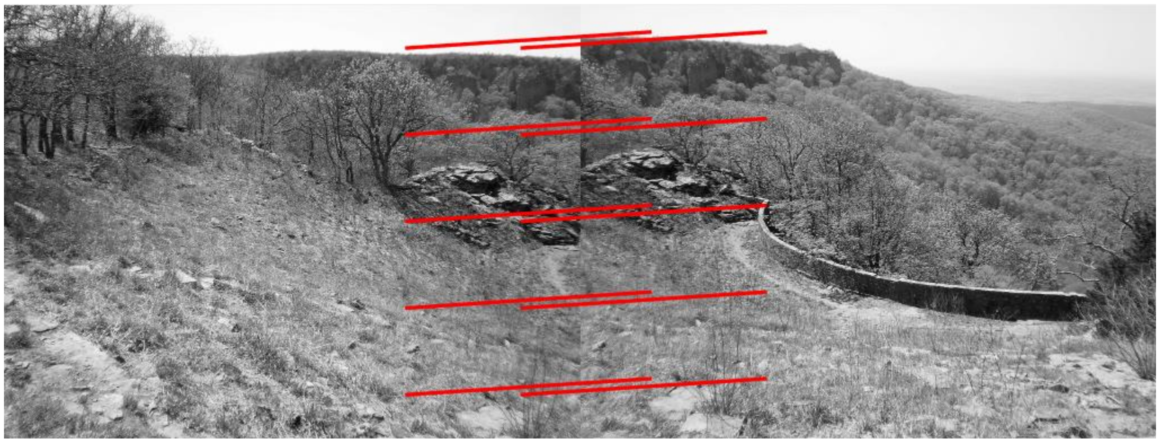


Figure 13: Applying only a translation motion

AFFINE MODEL

After the linear model, we tried an affine model. First an approach that tries to minimize the error between the model and the corresponding. In this case we have a higher number of degrees of freedom, so the result is worse than before. The result is as follows:

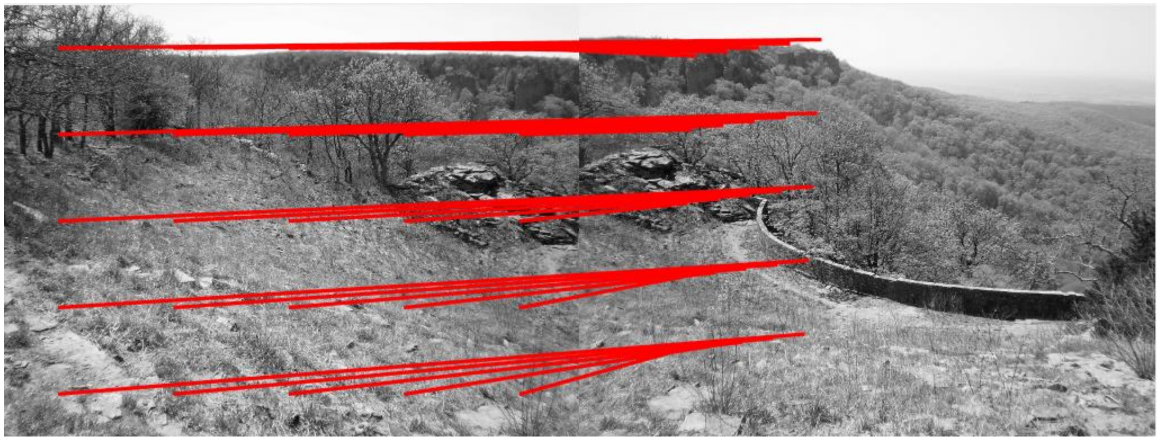


Figure 14: Apply the affine model

In order to improve the solution, now we consider only some of the best matchings.

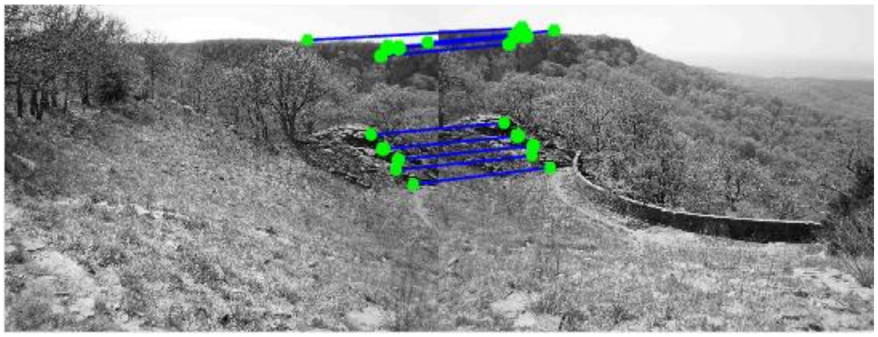


Figure 15: Considering only some of the best matchings

Now we consider the 10 best matchings using the linear model.

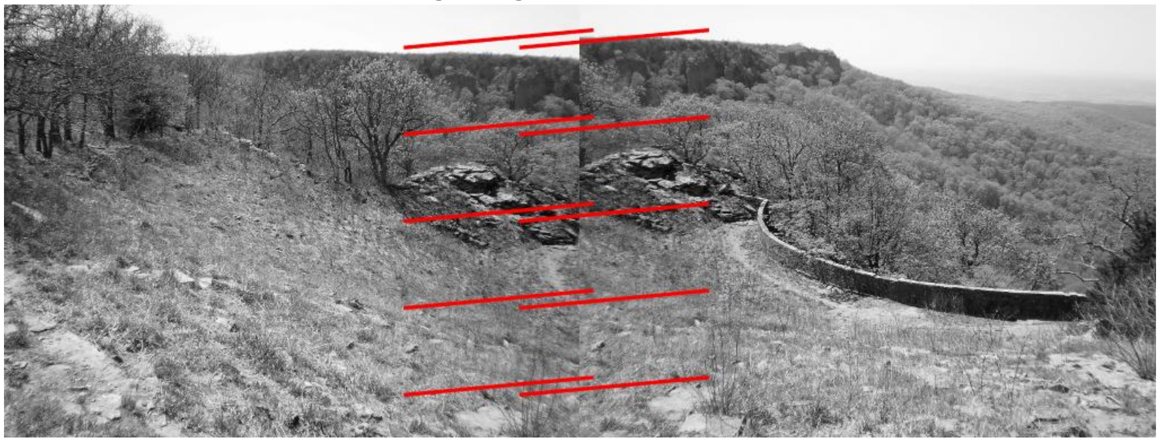


Figure 16: Linear model with 10 best matchings

Finally we use the affine model with also the 10 best matchings.

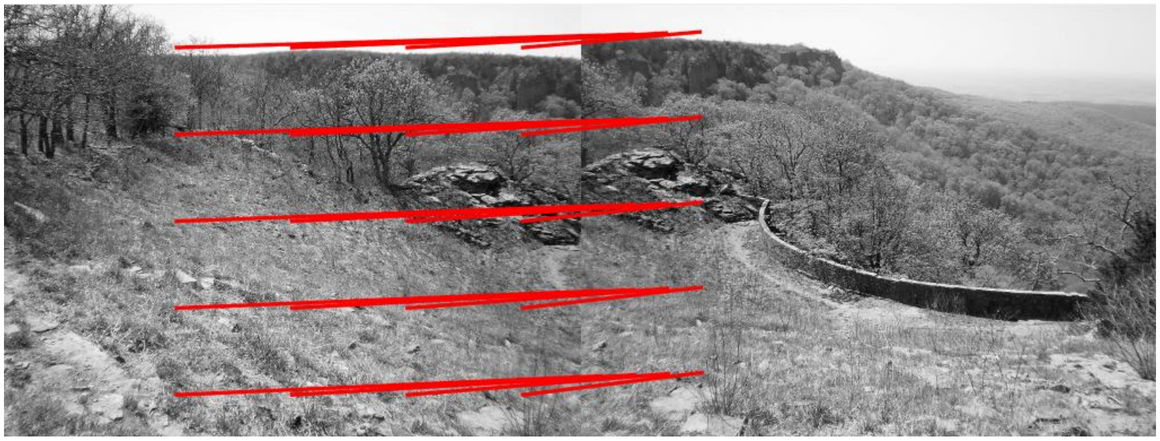


Figure 17: Affine model with 10 best matchings

This result is not as good as expected, the reason is the same as before, there is a high number of degrees of freedom. And any imperfect matching of match_sorted may influence negatively when computing the model.

In order to improve the result we will use the RANSAC method.

PANORAMA CREATION

The objective of this algorithm, is to automatically find the motion model between two images, and to form a final panoramic image.

The code associated with this task, is attached. The result of the algorithm is the following:

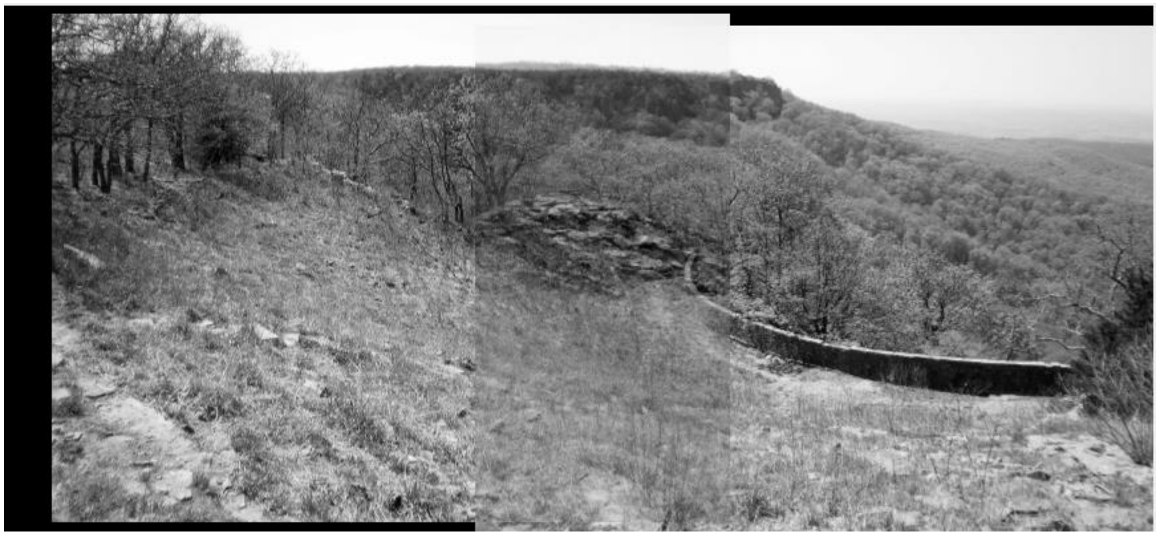


Figure 18: Final image after converting to panorama

The result is quite good, both images are merged in the center and the trees shows that the images are quite good converted to a panoramic image.