“What Makes a Patch Distinct?”: An Attempt

Jake Oliger
Indiana University
107 S Indiana Avenue
joliger@indiana.edu

Abstract

This project is an attempt to implement the algorithms described in the paper “What Makes a Patch Distinct?” by Ran Margolin, Ayellet Tal, and Lihi Zelnik-Manor. The goal is to take a given image and identify which parts of the image are distinct, i.e. different, important, or useful depending on a further task. Primary advantages of this approach are its speed and accuracy.

1. Introduction and background

Image distinctness is a difficult problem to solve well, and especially to solve well in a small amount of time. Previous approaches to this paper are described in more depth in Margolin et. al. [1], on which this paper is based, and so we will not waste time repeating what they have said. Instead, this paper will describe the problem in broad strokes and how we have interpreted the algorithm described in “What Makes a Patch Distinct?” [1] to approach the problem.

1.1. Distinctness and how it is approached

Image distinctness detection is the problem of figuring out which parts of an image are salient, or interesting, and which parts of the image are not. There are two main ways of approaching this problem: spectra (color) analysis, and pattern analysis.

Color analysis aims to find parts of the image that are colored most differently from the rest of the image. It is often approached as a segmentation problem, where a k-means clustering algorithm is fixed on the color spectra of an image to find salient color regions.

Pattern analysis attempts to find regions of the image that are patterned in a way unique compared to the rest of the image. This can be done by template matching of known patterns or by calculating RGB distance between “patches” in the image. Patches are similarly sized segments of the image that are used for comparing regions larger than just pixels, which are too small to contain any useful information.

The approach used in “What Makes a Patch Distinct” [1] is a combined approach of color and pattern analysis, with some interesting flavor added based on some intuitive ideas about how information in images is generally constructed. For starters, we begin not just with segmenting the image into patches, but with segmenting the image into so-called SLIC superpixels as described by Achanta et. al. in their paper “SLIC Superpixels” [2].

1.2. SLIC Superpixels

It is a ham-fisted approach to segment an image into NxN patches. The patches are arbitrary and don’t conform to any intuitive understanding of an image region. They are useful for computational purposes, but they discard potentially valuable semantic information in the image. Since semantic information is useful in describing distinctness, we wish to retain it. Enter SLIC superpixels.

The concept of a superpixel is a segmented region of the image that doesn’t necessarily conform to a grid pattern, but which conforms to boundaries within the image. So, intuitively, pixels within a superpixel ought to be relatively similar to one another as shown in Fig 1.

Figure 1 A segmented image of a toy shepherd

A SLIC superpixel is a superpixel generated using the SLIC algorithm as proposed by Achanta et. al. [2]. SLIC stands for *simple linear iterative clustering*, and performs this segmentation task based on CIELAB color space. For the purposes of this project, we use the SLIC implementation included in Scikit-Image.

2. Methods

Our method, however, will not work on superpixels alone. So, we will use the superpixels to determine the patches that contain the most variance, as these regions are most likely to contain intuitively distinct patches, and ignore the rest of the images. The Margolin et. al. paper suggests taking the 25% most variant superpixels. Using these superpixels, we move on to the pattern and color analysis.

2.1. Pattern analysis

From there, it is somewhat unclear exactly how they proceed in terms of patch selection. We implemented what can be thought of as a grid approach, where the image is broken down into a grid with squares of size NxN. The grid squares don't overlap each other, and squares that are split between two superpixels are ignored. Patches outside of our most salient superpixels are also ignored.

This is where Margolin et. al. pop off. Instead of calculating a simple Euclidean distance between one patch and all other patches, they pull two interesting tricks. First, they calculate the distance of all patches against just an average patch instead of all other patches, saving huge amounts of computation. Second, they calculate the distance in principal component analysis space instead of normal color space, thus preserving the internal statistics of each patch which leads to a better overall measure of distinctness between patches.

One final flavor addition to this patch distinctness measurement is combining the results from multiple resolutions. We were unsure if this meant multiple patch resolutions or multiple image resolutions, so we went ahead and implemented both. The distances are measured for the variety of patch sizes and image sizes and averaged to give us a more detailed map of salience. The result of this calculation, both before combination and after, in image form for the shepherd image is shown in Fig 2.

For our implementation, we created a new array representing the image and for each pixel in one of our patches, we placed the distance value for the patch as a whole. This lets us combine the salience from other metrics more easily. Importantly, we also normalize the distances so they are all between 0.0 and 1.0.

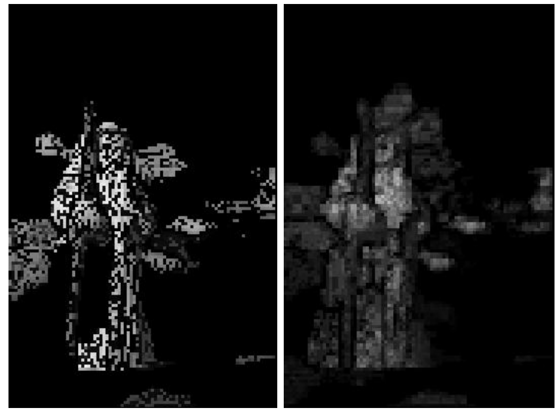

Figure 2 Left, the pattern distinctness distances calculated for the shepherd with a full image resolution and a patch size of 3. Right, the averaged distance calculations of patches of size 3, 6, and 9, and sizes 100, 50, and 25%.

2.2. Color analysis

For analyzing color distinctness, we don't need to use patches because the superpixels already contain semantically useful color segmentation. So, as described in Margolin et. al., we compare the average color of a superpixel with the average color of all other superpixels, and store this difference in an MxM matrix where M is the number of superpixels. Then the summation of each row (or column) of that matrix is equivalent to the distance of superpixel M from all other superpixels in RGB color space. The Margolin et. al. paper uses CIELAB color space, but for simplicity we ignored this conversion. The result of this calculation is shown in Fig 3.

The distance for each pixel then is set to the distance of the parent superpixel, again normalized to be a value in the inclusive range of 0.0 and 1.0.

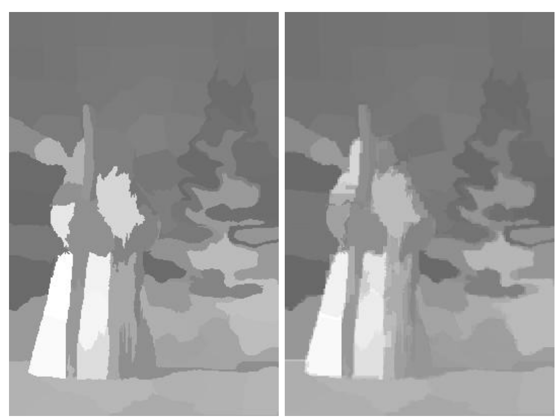

Figure 3 The color distance calculations for each segment in image form for the shepherd image. Left, for a single setting. Right, the averaged results of many settings as described in Figure 2.

2.3. Combining analyses

As described in Margolin et. al., to combine the pattern and color distance maps, we simply multiply the averaged maps for normalized pattern and color distances together elementwise. The resulting map is shown in Fig 4, alongside a color-modified version of the same image which simply masks the original image with the salience map to color it in. The more vivid regions correspond to more salience.

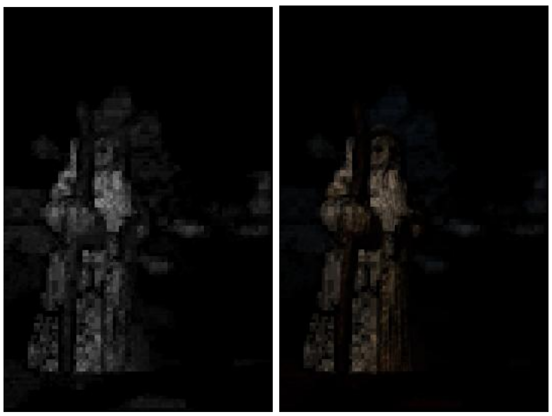

Figure 4 Our beloved shepherd, arriving as his destinated of determined salience. Left, raw salience as an image. Right, colorized using colors from original image scaled with salience intensity.

2.4. What we left out

Left out of this implementation for reasons of primarily time are the Gaussian filters meant to apply an intuitive understanding of salient smoothness and where salience generally is within an image. Inclusion of this in a future iteration of this project would lead to much improvement.

3. Results

The results for one image have been shown as we walked through the algorithm. The results for other images are shown in this section. We had a hard time finding the source images used in the original paper – the shepherd was ripped from the PDF – so instead we use our own personal images. The author dabbles in photography, so images were selected from his personal stash to exhibit a variety of potential salience, ranging from images that intuitively should be easy for the algorithm and images for which distinctness is at least intuitively illusive.

Due to the nature of these results and the lack of ground truths, our results are qualitative rather than quantitative. This is in part due to the nature of our dataset (we do not have one) but also due to the illusive nature of exactly what salience or distinctness are. They may be different for different applications, and so a generalized algorithm to determine salience is nigh impossible. This is just a starting point for further investigation and fine-tuning for specific problems, as is touched upon in the ensuing discussion.

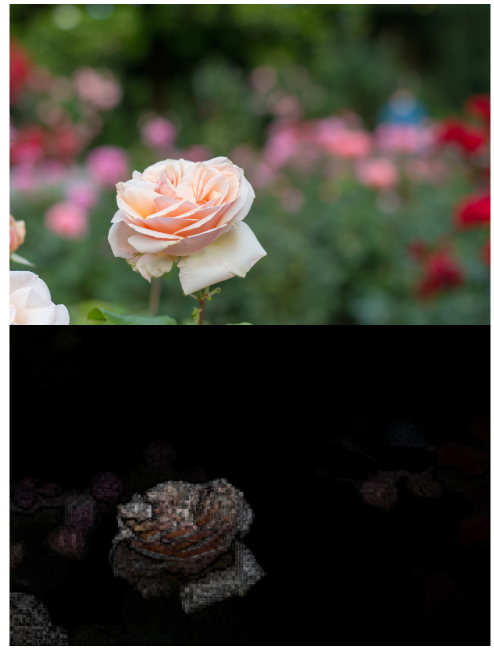

Figure 5 A peach rose. Above, the original image. Below, the color-masked salience detected by our implementation.

Figure 6 The author and his sister at Cannon Beach, OR, in front of the iconic Haystack Rock. Above, the original image. Below, the raw salience. The color map is difficult to see, so it is not shown.

Figure 7 A chicken, now deceased, grazing in a backyard during autumn. Above, the original image. Below, the color-masked salience.

The algorithm shows fairly good results on the above images, all things considered. The flower in Fig 5 is of course the best of them all, with a fairly obvious flower shining through in the color-masked salience image. The picture of the author and his sister in Fig 6 shows reasonable results as well. The shoreline and the torsos of the individuals are captured well, and are intuitively the most salient parts of the image. Interestingly, faces do not seem to be captured well by this algorithm in either Fig 6 or in Fig 4 with the shepherd.

Things go downhill a bit with the chicken in Fig 7. The chicken does appear, but so do many obviously not distinct leaves. This is probably mostly due to an issue with SLIC superpixel segmentation. Fig 8 shows how the segmentation played out.

Figure 8 The SLIC superpixel segmentation of the chicken from Fig 7.

As you can see, the segmentation has gone awry, showing reasonably segments for the chicken itself (at least its lower half) as well as for the leaves on the bottom half of the image, but the top half of the image is seemingly split into two large regions bordered in between by many, many small “super”-pixels. This lead the obviously quite variant lower super-duper-pixel to show up disproportionately in the final salience map.

4. Discussion and conclusion

We have shown how even a crude implementation of the work of Margolin et. al. leads to interesting results in salience. We have discussed at some length the accuracy of this approach, but thus far we have almost entirely neglected one of its other great advantages: its speed. For the shepherd image, even using a combination of many different parameters for determining salience, it runs in less than 15 seconds. This means it is easy to iterate on this algorithm, and it can even run in reasonable times for larger images such as the personal ones provided in this paper.

As mentioned in section 2.4, we did not include the very interesting Gaussian improvement mentioned in Margolin et. al. This is one blindingly obvious next step in improving our implementation. Additionally, it is clear from Fig 8 that some tweaking of the SLIC superpixel generation is required for more optimal results in naturally noisy images. It would also be great to obtain some of the datasets used by Margolin et. al. and compare our results to theirs.

It is maybe not obvious from reading this paper, but there were multiple points where we had to assume how a given calculation was made. In some cases, we may have been in error, and in other cases, we implemented multiple possibilities and combined them. This slowed the program down and may have led to worse results than Margolin et. al. were able to achieve. Still, the robustness of the method shone through. With some further refinements, results could be spectacular.

References

[1] R. Margolin, A. Tal and L. Zelnik-Manor, "What Makes a Patch Distinct?," *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1139-1146, doi: 10.1109/CVPR.2013.151.

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine S¨usstrunk, SLIC Superpixels, EPFL Technical Report 149300, June 2010.