Advanced Graphics Project

Chen Goldberg[*], Eran Cohen[†]

December 28, 2008

1 Introduction

In this project we deal with texture synthesis. We are given an image of a texture, and we would like to synthesize a new image in the same size or bigger than the original image. These textures are non-stochastic where texture elements can be separated 'easily'. We call these elements "textons". A texton is a fundamental micro-structure in an image.

In the first part of this project the goal is to have an algorithm that extract textons from a given image. After extracting the textons, the algorithm should cluster the textons to sets such that similar textons should be included in the same set. Our algorithm should decide how many sets should be. It is up to us to decide and implement how to cluster the textons and how they are shaped. The input is a color image of a texture. The output should be images of discriminative representatives of each set.

In the literature there are many works that deal with the fundamental components of an image or a texture. Some nomenclatures include texture particles, and some textons. In this work we refer to these image components as **semantic textons**. that is a texton we find should have some semantic value for a layman. Examples for such textons can be: an egg, a rock, bird or a buttercup. We also use semantic textons to differentiate it from the texton term used in paper such as [1] which refer to the more visual components of an image (e.g. lines, dots, curves). To avoid confusion we call these **visual textons**.

For the problem of texton extraction we offer two solution, each based on different hypotheses. We include both of them because we fail to reach an agreement as to which is better. The following section describes these two methods. The third section describes the software implementation of these methods, followed by a results section and a conclusion.

2 Methods

The basis on which we built our semantic texton extracting method is the clustering of visual textons as described in [1]: The input image is filtered using a filter bank made of gabor filters in numerous scales and orientations. The result is a feature space descriptor for each pixel, in which each feature reveals information about the surrounding of the original pixel. We then use

[*]ID: 039571161, E-mail: chengold@tau.ac.il
[†]ID: 038175436, E-mail: erancoh1@tau.ac.il

conventional dimension reduction methods such as PCA to lower the dimensionality of the feature space. We then add for each pixel its color information and again perform PCA.

We then perform K-means clustering on the new feature space and relate each pixel to one of the k clusters. Each of these pixel clusters stands for a visual texton which is called a texton channel, and together they combine a texton map of the original image.

2.1 Channel partitioning method

The intuition behind this factorization, is that each texton channel is made out of a group of pixels whose surrounding area behaves the same, thus making them appear as a non-stochastic texture. If we take this intuitive explanation seriously, then we might believe that a texton channel is actually a connected group of the semantic textons we sought finding. All we are left then is to find some way of breaking apart each texton channel to visually separated regions. We call this semantic texton extraction method "Channel partitioning" (a.k.a **Eran's Method**).

Given a texton map of the input image, we work on each texton channel separately. We partition each texton channel to visual elements by applying a standard binary edge detection on it. We then attempt to find connected components (i.e. pixel regions) in the texton channel which aren't separated by the edge map.

We then sort the connected components by their size in pixels, and select the k largest, where k is the number of textons we wish to obtain per class. The resulting connected set of pixels is considered a texton.

2.1.1 Discussion

In practice, the visual texton analysis step fails to capture whole texture areas which make a single semantic texton. The most obvious case is the one dealing with texton borders: Consider our texton to be a stone, then the stone's body is usually textured uniformly, thus mapped to a single texton channel. However, the borders of such a texton are mapped to different texton channels, which means that a method such as the "Channel partitioning" method would overlook the stone being a texton in its entire, and instead relate the border and filling to different semantic textons.

Furthermore, it is incorrect to assume that the stone's inner region will be mapped to a single texton channel. When the number of texton channels is set too high, even a relatively uniform texture pattern such as the stone's inner region will be separated into different visual textons. Thus, the "Channel partitioning" method is very sensitive to the number of visual textons. Moreover, A uniform mapping of an entire texton will only occur when each pixel inside the texton has roughly the same feature distribution. This happens when the entire texton behaves similarly to some of the filters. For example, an egg texton will be captured as a whole by this method, since an eggs behaves much like a gabor filter at a certain scale and orientation. Another example is color separation: with the lack of any significance from the filter bank feature, the only information put into use is the color values, and thus a segmentation according to colors is performed.

2.2 Patch selection method

Thus, it is unreasonable to rely on texton channels alone to encapsulate whole semantic textons. It is therefore a common practice to describe a textured area by the distribution of visual textons over it. Two regions or more are presumed to be of the same texture if the histogram of visual textons of the regions is similar (e.g. via a χ^2 test). Because texton of the same class are similar to one another, it makes sense to expect them to have similar histograms. Thus, the problem of extracting the textons is solved by clustering areas in the image which have similar histograms.

If we could take every possible continuous subset of pixels from the image and cluster them according to their χ^2 similarity, then we would hope to obtain a robust texton selection. Under this assumption we have the second texton extraction method: "Patch selection" (a.k.a **Chen's Method).**

Given a texton map of the input image, we sample rectangular windows in numerous scales uniformly over the image. For each window we compute the histogram of visual textons within it. We then perform a clustering of the histograms using K-means where k is the number of textons we wish to obtain.

It is important to avoid bias in choosing textons from a specific scale. We attempt to solve this by trying to obtain the same number of samples for each scale. Since large windows overlap greatly and small window don't vary much, we perform a preliminary clustering on each scale separately to assure discriminant representatives.

The final clustering gives us k classes of histograms, ordered in significance by their centroid distance. Each histogram is traced back to its window, and this window in its entire is a texton patch.

2.2.1 Discussion

The disadvantage of this method is that an exhaustive search over all windows is improbable, and the probability that a sampled window will capture a Texton in its entire is very unlikely. The greedy heuristic offered does only little to repair that. In addition, the use of window areas adds much noise to the sampled histograms.

3 Implementation

The implementation of these methods was done in Matlab version 2007a. We used libraries found on the Internet which are included in the distribution accompanying this document, along with some standard Matlab toolbars which weren't, such as the Image processing toolbox.

4 Results

We've included some of the better and worse results we obtained using our implementation (Figures 1, 2, 3, 4, 5 and 6). For a complete list of output images go to the examples/outputs directory.

5 Conclusion

The problem of texton extraction is a hard and ill-posed. The question "what is a texton?" was raised many times during the making of this part of project, and it was always answered unsatisfyingly. Each of the two methods shown here has it's own bag of problems. We hope that the second part of this project would redeem these methods by showing that a texture can be synthesized from these textons.

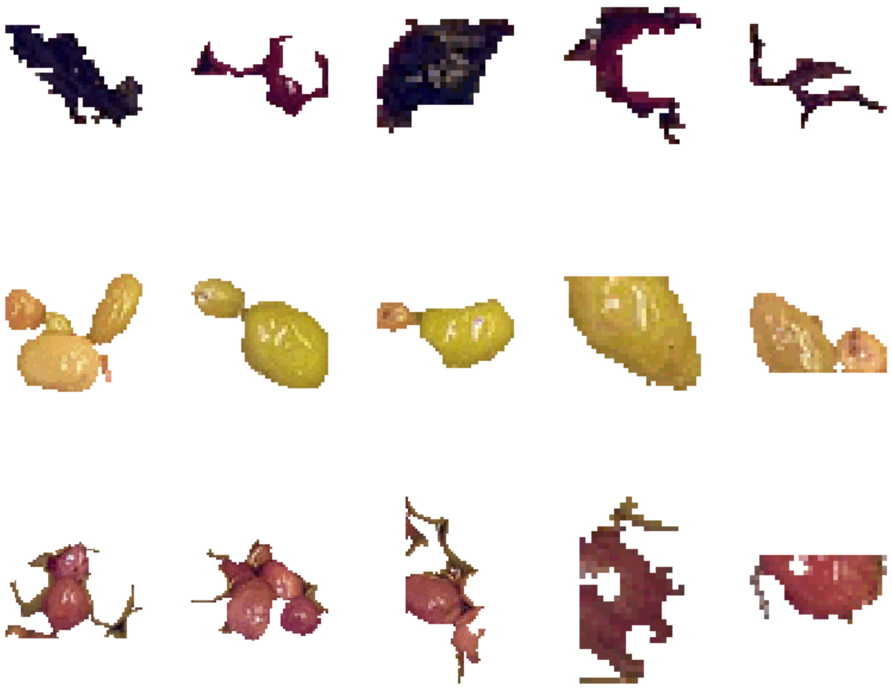

Figure 1: Textons extracted from the Olives image using Eran's method

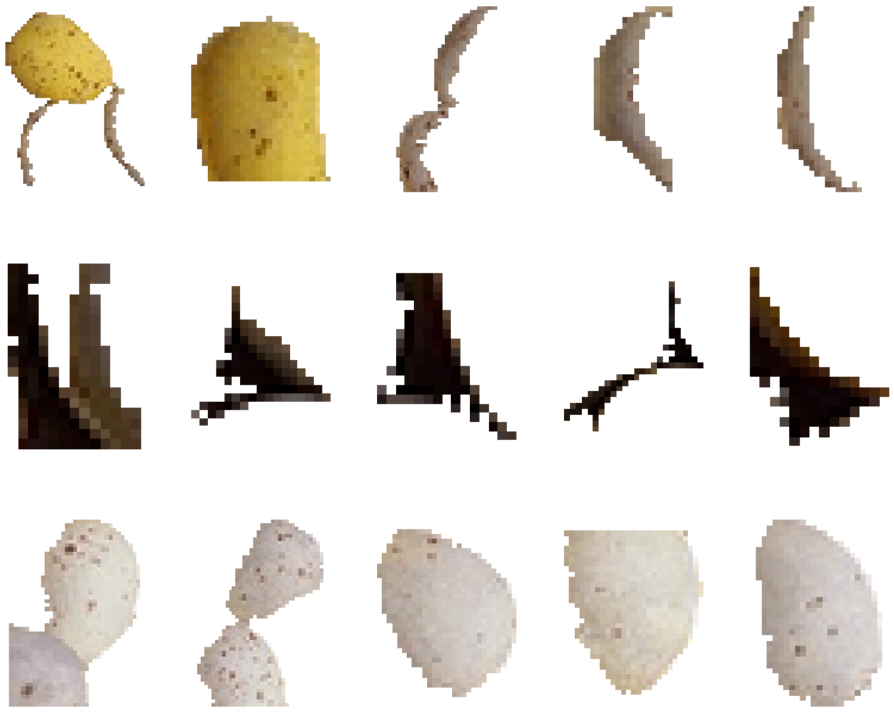

Figure 2: Textons extracted from the Eggs image using Eran's method

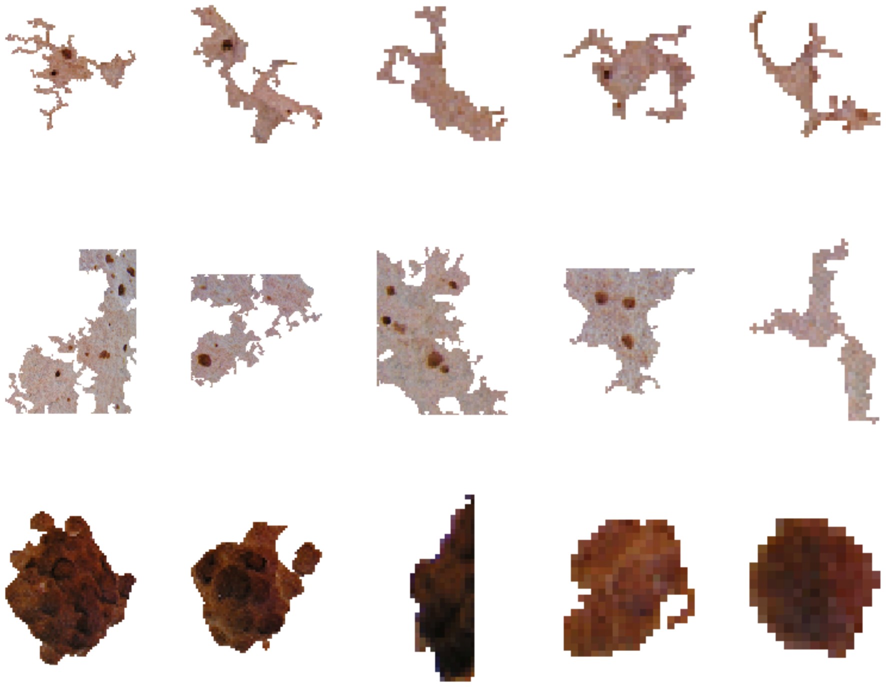

Figure 3: Textons extracted from the Rust image using Eran's method

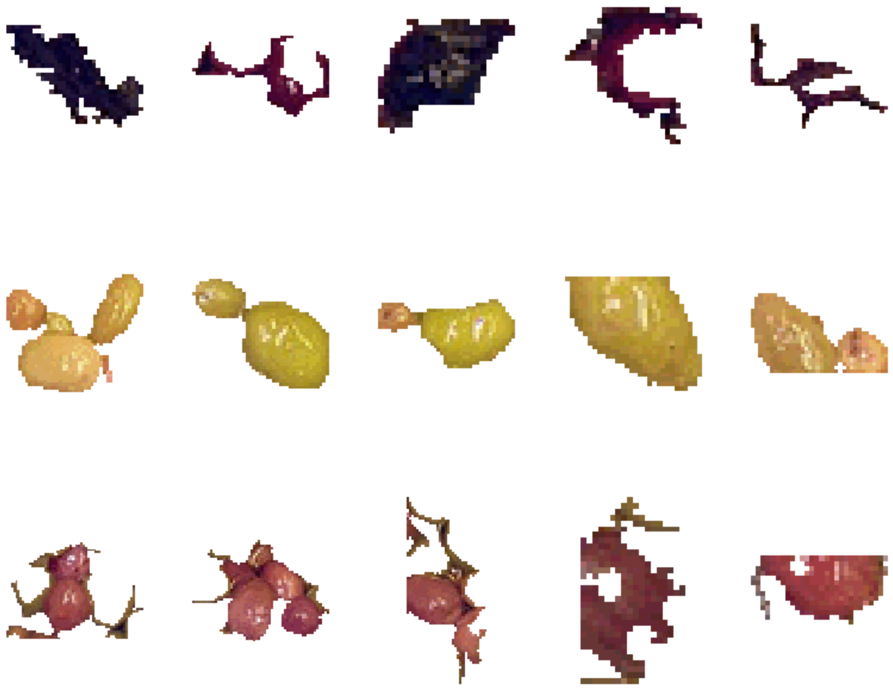

Figure 4: Textons extracted from the Olives image using Chen's method

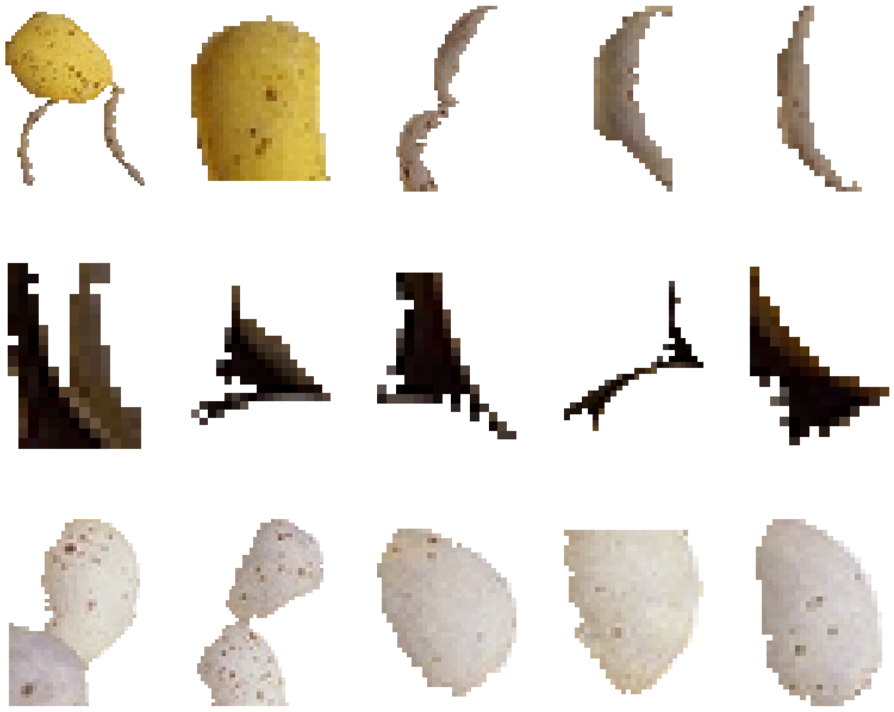

Figure 5: Textons extracted from the Eggs image using Chen's method

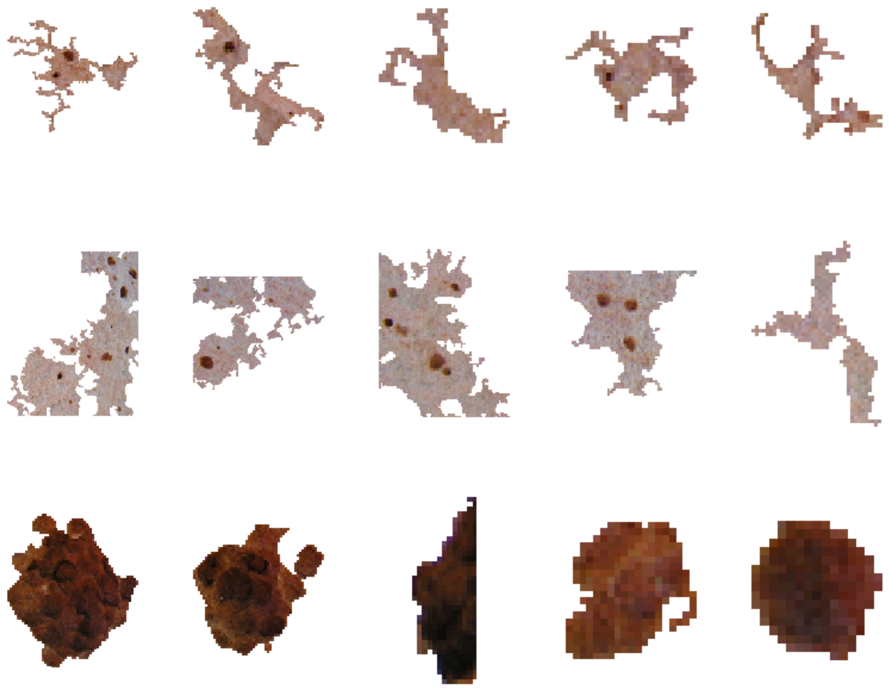

Figure 6: Textons extracted from the Rust image using Chen's method

We failed to decide on which method is better. While Eran's method produces cleaner more visually appealing results, Chen's method stays more loyal to the definition of textons while compromising on visual attractiveness and human intuition. Perhaps a two-method solution is the right way to go. That is, perhaps synthesizing a texture by combining the textons produced by both methods would provide a more robust solution for the ultimate problem at hand.

References

[1] Textons, Contours and Regions: Cue Integration in Image Segmentation / Jitendra Malik, Serge Belongie, Jianbo Shi and Thomas Leung IEE International Conference on Computer Vision 1999