CS7960: Project 1
Name: James Fishbaugh
Email: jfishbau@cs.utah.edu

Blob Detection by Scale-Space Filtering

Introduction

Our perception of the natural world is influenced by the scale at which the observation takes place. For example, consider the documentary film Powers of Ten, which begins with a one meter square overhead shot of a man. The camera begins to zoom out in powers of ten, to reveal a ten-meter square view, a one hundred meter square view, etc. The features we recognize in the images are related to the scale at which oberve them. We can clearly see the man at the finest scale, however as the camera zooms out, we begin to see the city, state, country, the Earth, and eventually, the entire universe.

We can naturally extend this idea of scale to a computer vision system where the goal is to detect relevant features in images. While small scale features may be detected at the voxel scale, larger features may go unnoticed. We desire a front end vision system that detects features at multiple scales. We model scale-space as convolution with Gaussian's of increasing σ. Figure 1 depicts the concept of scale-space, where the vertical dimension represents scale. Larger scales show more blurring, allowing for detection of larger scale image features.

Figure 1: *The concept of scale space. Image from http://cvr.yorku.ca/members/gradstudents/kosta/compvis/.*

In this project, we are interested in detecting blobs of various sizes. Figure 2 shows an example, where we want to detect the centers of sunflowers. The following sections discuss the details of our blob detection system and its implementation.

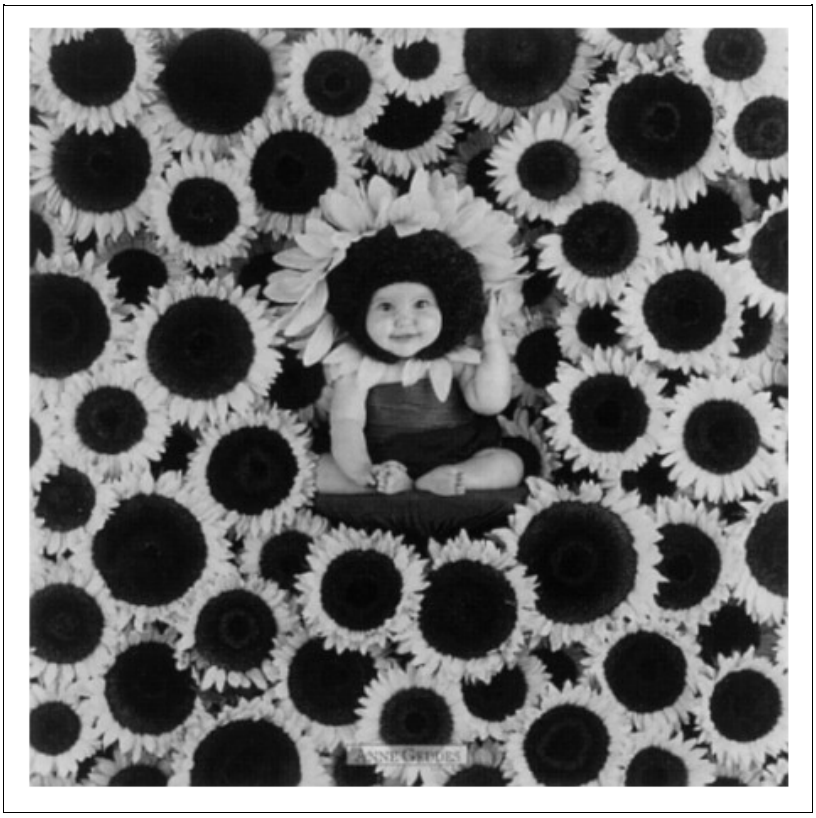

Figure 2: *An example image where we wish to detect blobs.*

Creation of Scale-Space

Scale-space can be created in many ways. We can model blurring as a diffusion process, where we solve a partial differntial equation on the image grid. We can achieve different scales by increasing the number of iterations we allow the diffusion to continue. Another method would be repeated convolution with a Gaussian of a given σ. Since the convolution of two Gaussians results in another Gaussian with the sum of σ_1 and σ_2, we can obtain different scales in this fashion. However, this is only practical for creating a linear scale-space.

The method used here to create a log scale-space is to create a Gaussian for each scale. We construct a linear sampling *t*, and compute σ by *exp(t)*. This results in an array of σ's with log sampling. For each σ, we compute the size of the discrete isotropic 2D Gaussian kernel by the heuristic *ceil(3σ)*2+1*. The image is convolved with each Gaussian, and the results are stored in an array. This was implemented in a method called create_scale_space. Figure 3 shows an example of a scale-space constructed from the image of sunflowers.

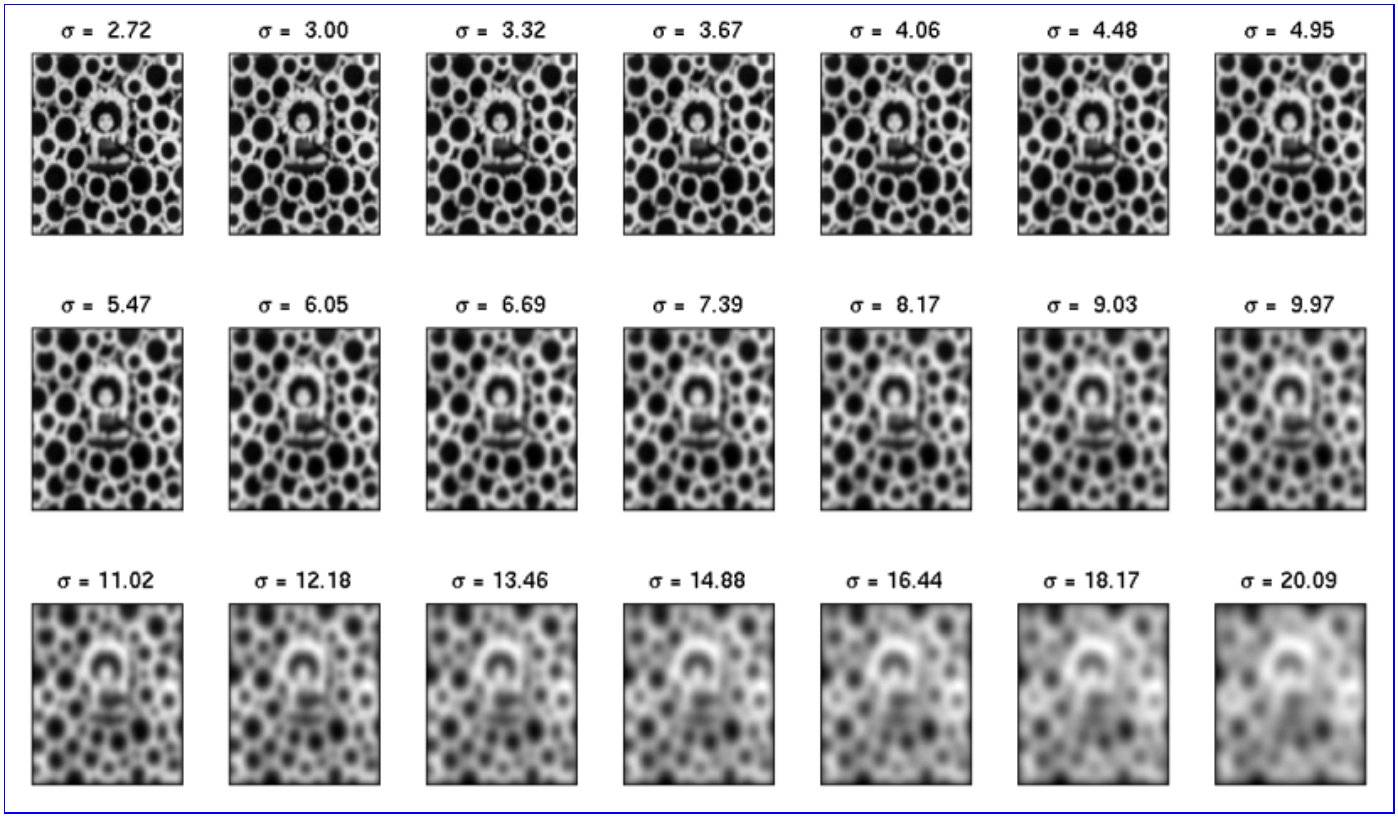

Figure 3: *Scale-space created with t from 1.0 to 3.0 incremented by 0.1.*

Laplacian Filtering

In order to detect blobs in our scale-space, we filter each image using a 2D discrete Lapalacian filter. This is the optimal filter for detecting symmetric blob-like shapes in an image that has been convolved with a Gaussian. Due to properties of convolution, we can instead take the Laplaican of a Gaussian, and convolve that with the images. Figure 4 shows the shape of the Laplacian of Gaussian filter.

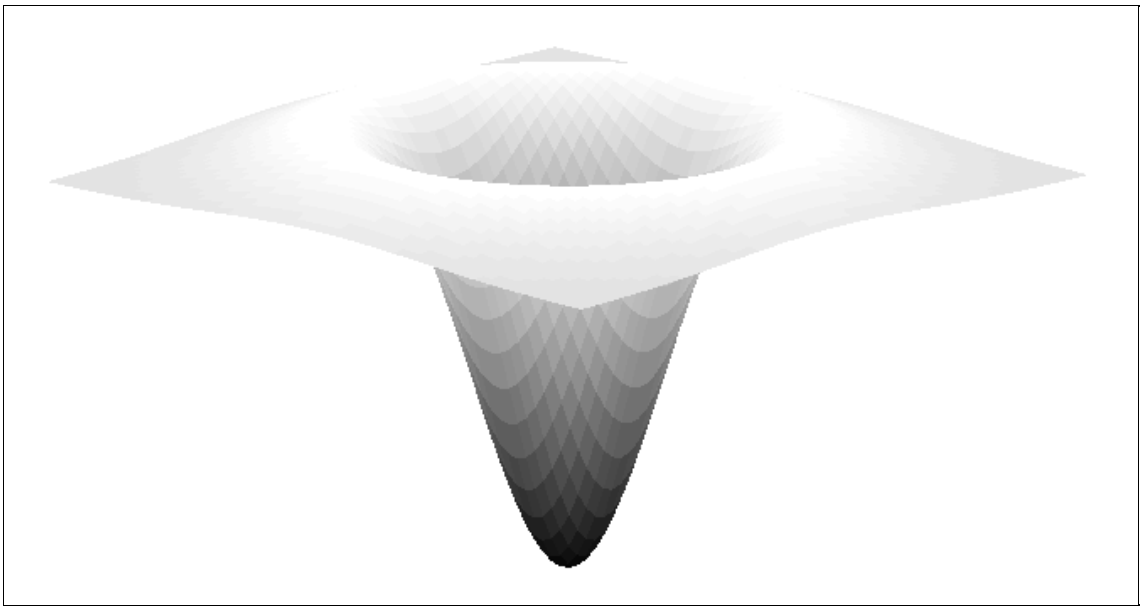

Figure 4: *Laplacian of Gaussian.*

As mentioned above, we can convolve the image with a Laplacian of Gaussian filter to obtain a scale-space where high responses denote the location of blobs of a certain size. Figure 5 shows the true representation of our scale-space. The next section discusses how we find the location of the blobs given a Laplacian of Gaussian scale-space representation.

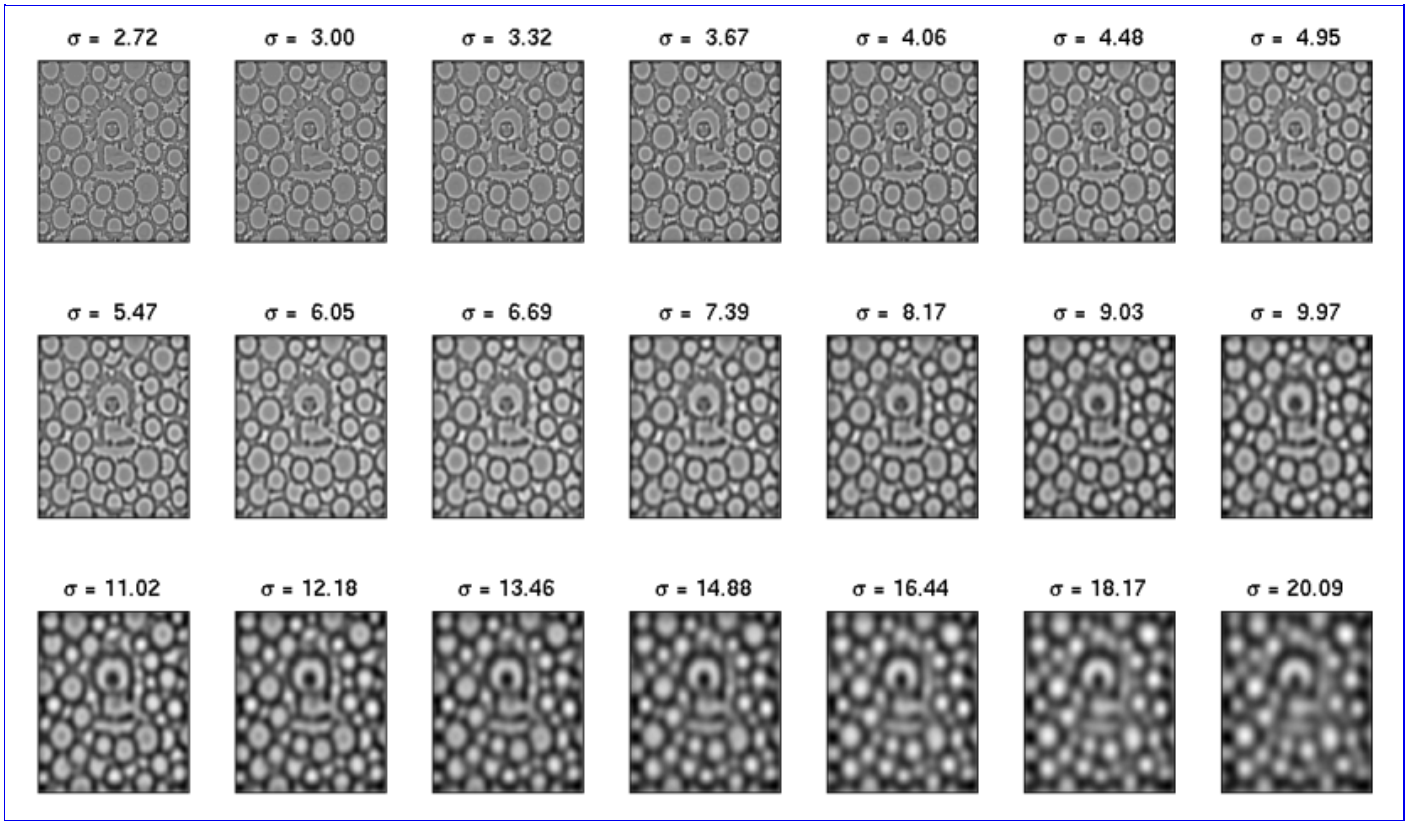

Figure 5: *Laplacian filtered scale-space created with t from 1.0 to 3.0 incremented by 0.1.*

Maximum Detection Strategy

In order to determine the location and size of blobs, we must find local maxima in the spatial domain as well as in the scale dimension. We define a threshold, where every pixel with gray value larger than the threshold is considered a local maxima. Furthermore, we restrict maxima to be larger than their 8 immediate neighbors. Using this scheme, we obtain maxima for every scale. As shown in figure 6, this leads to a lot of overlapping regions. We must also determine maxima in scale.

One possible way to alleivate this problem is to require a local maxima to show a higher response than its immediate neighbors in scale. The problem with this method is Gaussian blurring does not respect the location of features; it moves edges. This is clearly visible in figure 6, the circles correpsonding to the same blob are not perfectly concentric. Therefore, we must consider some neighborhood as we compare different scales. It is not immediately clear what size neighborhood is optimal. Too small a neighborhood and we are left with overlapping regions. Too large a neighborhood would force the selection of a single blob when multiple small blobs exist close to each other.

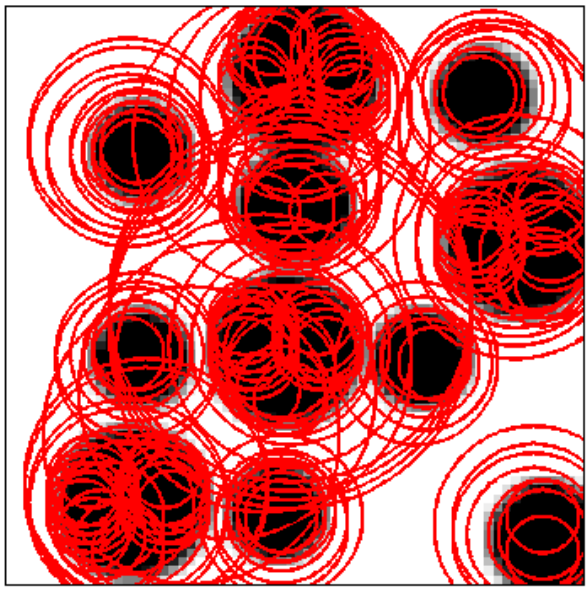

Figure 6: *Blobs detected before considering maxima in scale.*

We propose determining local maxima at each scale, allowing post processing to be done to determing the accurate maxima in scale. We model the blobs as circles with the center located at the local maxima with radius 1.5σ. Conceptually, we consider a list of circles of varying center location size. We iterate over all circles, computing the area of overlap between it and every other

circle. If the area of overlap is larger than a threshold (0.25 is reasonable), then we keep the circle generated by a higher response in scale and discard the other. When the process is finished, we are left with maxima in space and in scale. This pruning process was implemented in a method called prune_blobs, which accurately and efficiently determines maxima in scale.

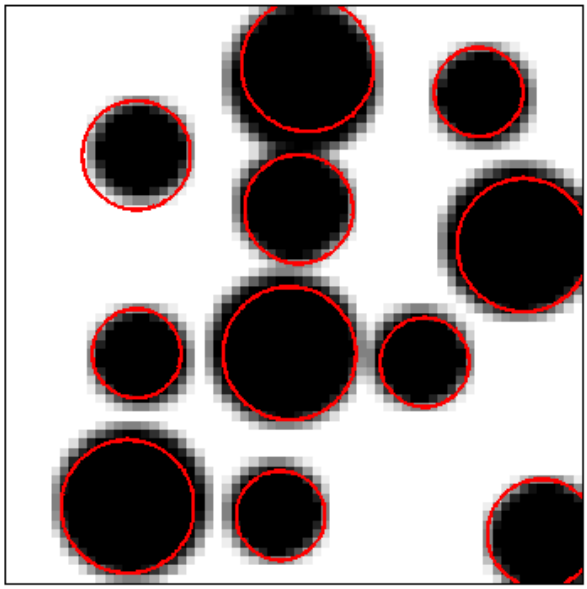

Figure 7: *Blobs detected after determining maxima in scale.*

Experiments

Figure 7 shows the results of our scale-space blob detector on two synthetic images and two images of sunflowers. The results are nearly perfect for the synthetic images. This is not entirely unexpected, as the blobs are perfectly circular, the edges are very strong, and there is no noise im the image. We do observe that the centers and sizes of the overlaid circles are slightly incorrect. This is likely due to the fact that Gaussian blurring causes localization errors and our radius of 1.5σ is only an approximation.

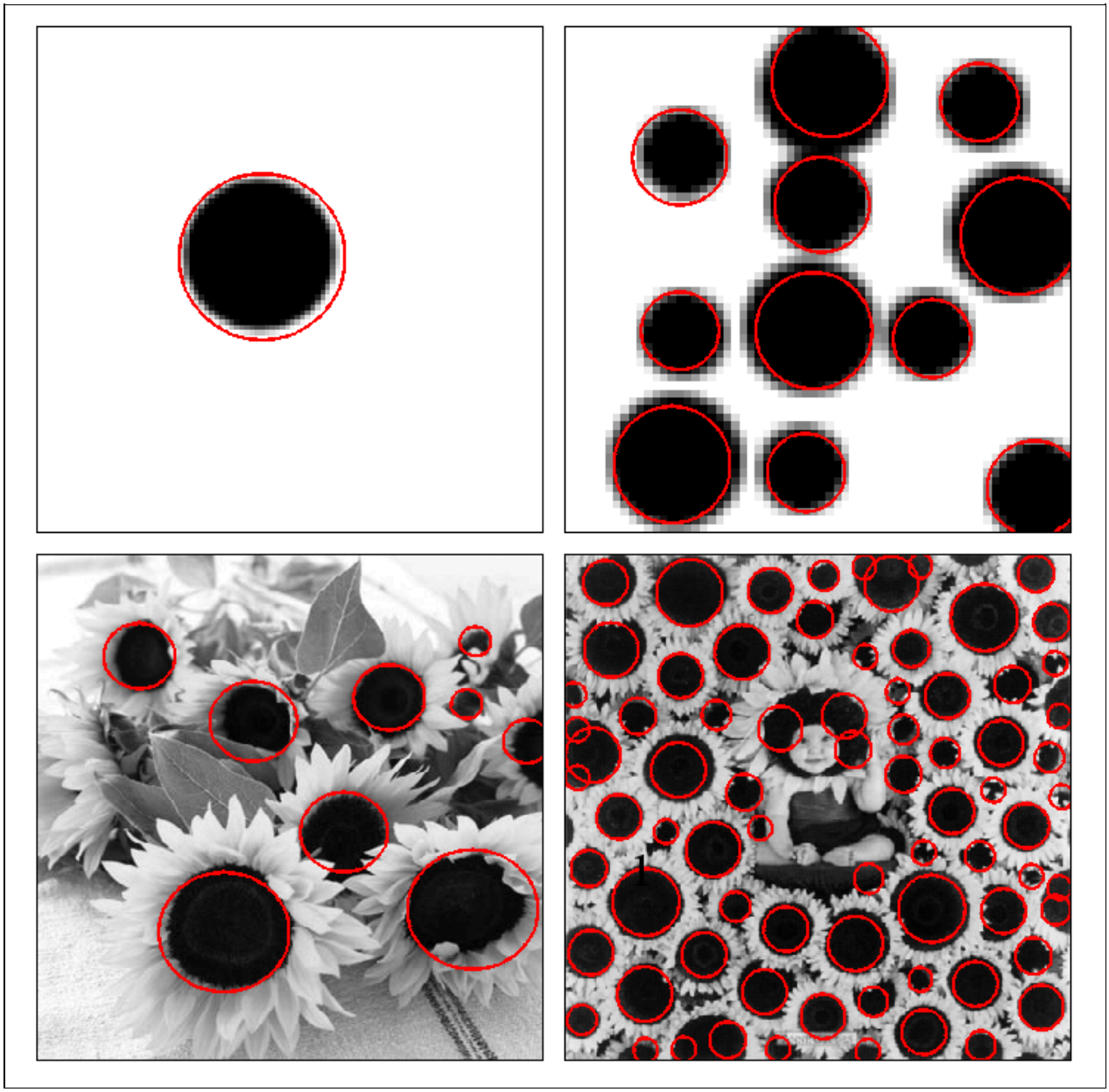

Figure 7: *Top row) Synthetic images. Bottom row) Sunflower images.*

The bottom row of figure 7 shows the blobs detected in two images of sunflowers. The results are actually quite good. The main problem is the detection of multiple small blobs when there is in fact only one large blob. In these cases, the large blob is not perfectly circular, and the filter is showing multiple high responses at a fine scale. This could likely be remedied by a finer sampling in scale or by refining the threshold value used for detecting maxima. However, the Laplacian of a Gaussian will produce a response for black regions even if they are not circular or blob like. An example of such a region is underneath the baby in the second sunflower image.

All code for this project can be viewed here.