

# Scale-Space Blob Detectors

Audio processing, Video processing and Computer Vision

Lab exercise 2

## Scale-space blob detectors

The goal of this lab exercise is to implement a scale-space blob detector based on the Laplacian of Gaussian (LoG) filter. You can follow this high-level script structure:

```
For s in scales
    Filter image with the corresponding LoG filter
    Square the filter response
    Threshold the square response and carry out spatial non-maximum
    suppression to reduce each resulting region to one pixel
Perform scale-space non-maximum suppression
Display the results
```

Let us work with the image “Sunflowers.jpg”.

In the following, there are some hints to help you.

Use `scipy.ndimage.filters.gaussian_laplace` to implement LoG filter

Explore the size of the sunflowers in the image and propose an appropriate scale-space.

- How would you choose the sigma parameter of the LoG filter, i.e., the scale of the filter?

In order to create the scale space, you have two options: either a) filtering the image with LoG filters of different scales; or b) keeping the same filter and down-sampling the image. Implement both options.

- Use `skimage.transform.resize` for down-sampling (and up-sampling, if needed).
- Which one is faster? Use `time.time()`.
- Do you need to still normalize the scale of the filter response when you down-sample the image instead of increasing the scale of the filter?

To implement non-maximum suppression, compute a 2D non-maximum suppression independently for each scale and then extend the procedure to the 3D scale-space.

- You may use a 3D array to represent your scale-space ( $M, N, N_s$ ), where  $M \times N$  is the image dimension and  $N_s$  is the number of scales.
- You may use the function `nms` (`filter_response`, `min_distance`, `threshold`), which is provided with the lab material to perform spatial 2D nonmaximal suppression.
- Explain how this function works.
- Build the 3D scale-space non-maximum suppression by yourself.

To display the results, you may use the function `plot_circles` (`image`, `cx`, `cy`, `rad`), also provided with the materials. Here you have an example of how the results may look like. In any case, your results may look different depending on your threshold, range of scales, and other implementation details.



## 2-page report

Include in your report the answers and explanations that follow:

- Which is faster, to use filters at different scales or to down-sample the image? why?

- Assuming that you are using one LoG filter and down-sampling the image, how would you choose the sigma parameter of the LoG filter, i.e., the scale of the filter?
- Do you still need to normalize the scale of the filter response when you down-sample the image instead of increasing the scale of the filter?
- Explain how the `nms` function works.
- Explain how you implemented the 3D scale-space non-maximum suppression.
- What did you find most difficult about this lab exercise?