EECS 4422/5323
Lab 2: Template Matching

Presented by: Abdullah Abuolaim

Outline

This lab will involve an advanced application of image filtering, including filter design and troubleshooting for variable input data.

- Problem Specification
- Template Creation
- Template Matching

Problem Specification

Imagine you are working on a personal shopping robot. Your job is to enable the robot to recognize target items on store shelves.

- For this specific lab, your job is to locate boxes of Cheerios cereal in a dataset of pictures of grocery store shelves containing cereal
- You have at your disposal two exemplar images, *cheerios1.jpg* and *cheerios2.jpg*. This is your training data; you can use these two images however you like to create your image filters.
- You have five images, *cereal1.jpg* to *cereal5.jpg* of test images; your goal is to create a binary mask for each image with as many pixels corresponding to a plain box of Cheerios set to 1 and as few pixels as possible not corresponding to a Cheerios box set to 0.

Learning Outcomes

- Gain experience with the design and application of image filters
- Understand how to derive a filter from a template image
- Understand some of the challenges which arise from template matching

Note that this is a deceptively simple lab which is actually rather challenging! You may not be able to get results which are very good; understanding why your approach fails is also a very important skill to have, so if you are not getting the results you want try to think about why.

Template Creation

You have two pictures like the one shown on the right to work with; you may make as many kernels as you like using these images.

To create a filter kernel for template matching, it is useful to isolate specific sub-portions of the image. You can do this graphically in your favourite image processing application by cropping to your target and saving the crop as a new image (note: lab computers have GIMP available).

Template Creation Tips

- It is helpful to target visually distinctive portions of the image. Logos, specific text, or unusual colours or patterns are all good targets.
- Think about how you want to normalize values in your template.
- Do you need a large template kernel (which will be slow to process), or can you think of any techniques from class which might apply to handling different template scales?

An example crop to base a template on.

Template Matching

Once you've built your template kernels, you can apply them to the test images using the OpenCV function `cv2.filter2D`, an example of which can be found in the Filtering demo provided from the class notes.

Note: Do you want to use convolution or correlation? Check the function API to make sure you are using the one you want.

To create a binary mask, the most common approach is to check a threshold of activation against the responses to your filter kernels (i.e. `mask = filtered_img > threshold`). Once you have your output in a binary format, you can combine output from multiple kernels using logical operators (eg. AND or OR). You may find morphology operators useful, too, in refining your output masks.