Computer Vision Systems Programming VO

A Recap of Image Processing

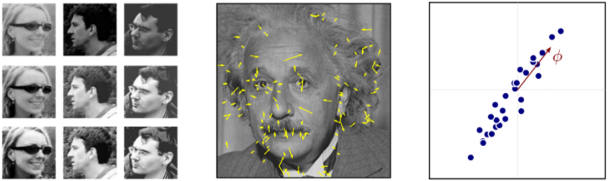
Christopher Pramerdorfer

Computer Vision Lab, Vienna University of Technology

Topics

A brief recap of Image Processing (IP)

- Assuming you are already familiar with IP
- Focus on methods that are widely used in practice



Images from Prince 2012

Relation of IP and CV

IP encompasses operations that

- Take images as input
- Produce images or representations (e.g. descriptors)

We regard IP as preprocessing for CV

- IP has great influence on CV performance

Contrast Normalization

Reduce variation due to contrast and intensity changes

We cover two techniques

- Whitening
- Histogram equalization

Images from Prince 2012

Transform pixel values so that

- Their mean is zero
- Their variance is one

Images from Prince 2012

Contrast Normalization

Whitening – Matlab Implementation

```
img = single(rgb2gray(imread('image.png'))); % load
m = mean(img(:)); % compute mean
s = std(img(:)); % compute standard deviation
whitened = (img - m) / s; % normalize
```

Contrast Normalization
Histogram equalization

Transform pixel values so that distribution is "flat"

- Cumulative histogram linear over value range

Images from Prince 2012

Contrast Normalization
Histogram Equalization – C++ Implementation

```
// cv = OpenCV namespace
cv::Mat img, equalized; // storage
img = cv::imread("image.png", CV_LOAD_IMAGE_GRAYSCALE); // load
cv::equalizeHist(img, equalized); // normalize
```

Noise Reduction and Change Detection

Reduce image noise

Locate intensity changes

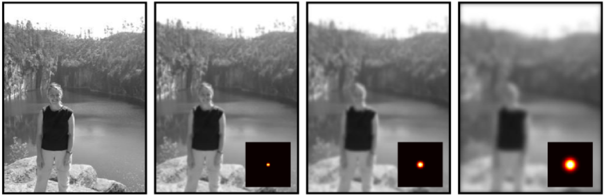
Accomplished via **linear filtering** (but there are other means)

- Pixel values linear combination of neighbor values
- Computed via convolution (or correlation)

$$f'(x,y) = \sum_{i,j} f(x-i, y-j)h(i,j)$$

Use a 2D Gaussian as kernel h:

$$h(i,j) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{i^2+j^2}{2\sigma^2}\right)$$



Images from Prince 2012

```matlab
img = rgb2gray(imread('image.png')); % load
h = fspecial('gaussian', [3 3], 0.5); % create Gaussian kernel
filtered = imfilter(img, h); % filter
```

Use a Laplacian of Gaussian (LoG) filter as kernel h

- Gaussian for noise reduction
- Laplacian approximates $\nabla^2 = f_{xx} + f_{yy}$

LoG filters respond to intensity changes

- Regardless of direction
- At a frequency defined by σ of Gaussian

Substrate for SIFT interest points

Change Detection via LoG Filtering

Images from Prince 2012

Use a Gabor filter as kernel h, which consists of

- A Gaussian for noise reduction
- A Sinusoid for change detection

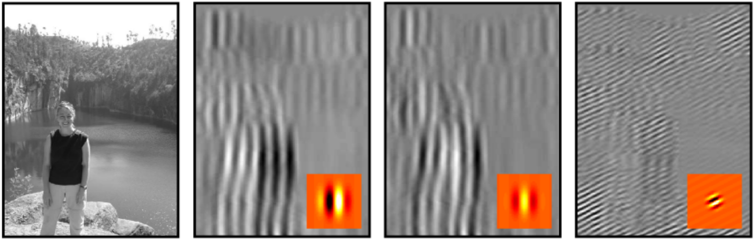
Gabor filters respond to intensity changes at a

- Phase and orientation defined by the Sinusoid
- Frequency defined by the Gaussian and Sinusoid

Substrate for object recognition and scene understanding

Noise Reduction and Change Detection

Change Detection via Gabor Filtering



Images from Prince 2012

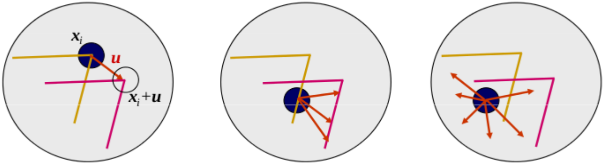
```
cv::Mat img, gabor; // storage
img = cv::imread("image.png", CV_LOAD_IMAGE_GRAYSCALE); // load
h = cv::getGaborKernel(...); // create Gabor kernel
cv::filter2D(img, gabor, CV_32F, h); // filter
```

Interest Point Detection

Interest points are image locations that

- Can be detected reliably in multiple images of the same object
- Which means they are **invariant** to image transformations

This excludes everything but “corners” (aperture problem)

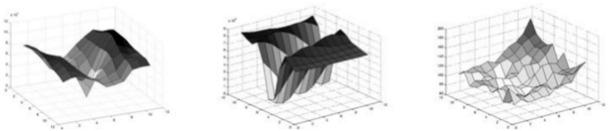


Images from Szeliski 2010

Corners characterized by intensity change in multiple directions

Harris corner detector exploits this by

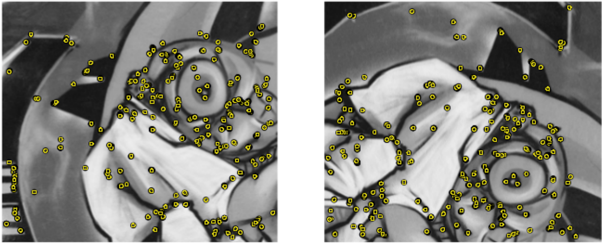
- Checking gradient distribution in local neighborhood
- Corner: gradient distribution has two large eigenvalues



Images from Szeliski 2010

Harris interest points

- Are invariant to translation and rotation
- Stable under varying lighting conditions



Images from Tuytelaars and Mikolajczyk 2008

```matlab
img = rgb2gray(imread('image.png')); % load
corners = corner(img, 'Harris', maxNum); % detect corners
```

Bibliography

Prince, S.J.D. (2012). **Computer Vision: Models Learning and Inference**. Cambridge University Press.

Szeliski, Richard (2010). **Computer vision: algorithms and applications**. Springer.

Tuytelaars, Tinne and Krystian Mikolajczyk (2008). “Local invariant feature detectors: a survey”. In: **Foundations and Trends in Computer Graphics and Vision**.