Subpixel Algorithms

CS4243 Computer Vision and Pattern Recognition

Leow Wee Kheng

Department of Computer Science
School of Computing
National University of Singapore

Motivation

- Digital images are discretized into pixels.
- Each pixel correspond to an integer-valued location.
- Integer-valued locations are not accurate enough for many applications, such as
    - tracking
    - camera calibration
    - image registration and mosaicking
    - 3D reconstruction
- To achieve better accuracy, need floating-point-valued locations, i.e., subpixel localization.

General Idea:

- Develop a model of the feature to be localized.
- Apply conventional algorithm on input image to detect feature up to pixel accuracy.
- Iteratively match model with input image to localize detected feature with subpixel accuracy.

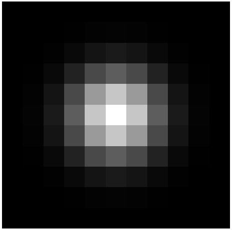
Notes:

- Most subpixel algorithms require a good estimate of the location of the feature.
- Otherwise, the algorithms may be attracted to the noise instead of desired features.

Point Localization

Here, we illustrate the general approach using a point as an example. How does a point look in an image?

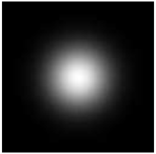
(a) A point.



(b) The enlarged image of a point.

- A point usually occupies more than one pixel.
- A point does not have sharp edges.
  The edges are smooth or blurred.

An appropriate model of a point is 2D Gaussian.



2D (unnormalized) Gaussian

$$g(x, y) = \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{1}$$

So, a point can be modeled by the 2D function M as follows:

$$M(x, y; A, B, \sigma, u, v) = A + B \exp\left(-\frac{(x-u)^2 + (y-v)^2}{2\sigma^2}\right) \qquad (2)$$

- M: intensity
- (x, y): any location in the image.
- A: intensity of background (dark region).
- B: peak intensity of point (brightest region).
- (u, v): peak location, i.e., center of point.
- σ: amount of spread of the Gaussian.

In short-hand notation: $M(\mathbf{x}, \boldsymbol{\theta})$

- $\mathbf{x} = (x, y)^T$: variable image location
- $\boldsymbol{\theta} = (A, B, \sigma, u, v)^T$: parameters of the point model.

If the model M matches a point in image I perfectly, then

$$M(\mathbf{x}, \boldsymbol{\theta}) = I(\mathbf{x}) \tag{3}$$

for all locations $\mathbf{x}$ within the model M.

- (u, v) gives the location of the point.
- Since u, v can be take on floating-point values,
  they indicate a subpixel location.

How to obtain a good match?

Compute error of match $E(\boldsymbol{\theta})$:

$$E(\boldsymbol{\theta}) = \sum_{\mathbf{x} \in W} [M(\mathbf{x}, \boldsymbol{\theta}) - I(\mathbf{x})]^2 \tag{4}$$

where W is the extent of M (like a small window or template).

Next, apply appropriate algorithm to find the $\boldsymbol{\theta}$ that minimizes the error $E(\boldsymbol{\theta})$.

The subpixel location is the (u, v) of the optimal $\boldsymbol{\theta}$.

Method 1: Direct Solution

- Do the usual thing: $\partial E / \partial \boldsymbol{\theta} = 0$.
- Then, rearrange the terms to try to obtain a set of equations that can be solved.

Method 2: Apply Optimization Algorithm

Some possible algorithms:

- Gradient descent.
  - Compute $\partial E/\partial \boldsymbol{\theta}$.
  - Then, change $\boldsymbol{\theta}$ iteratively until it converges:

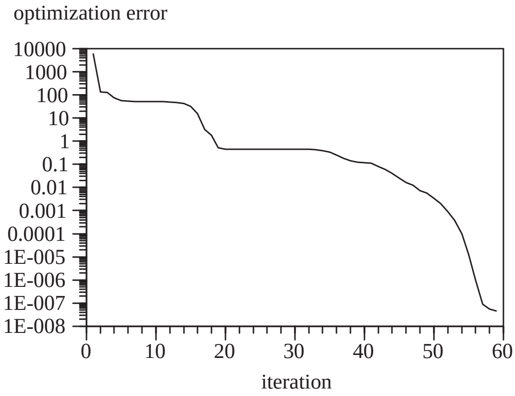
$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) - \eta \frac{\partial E}{\partial \boldsymbol{\theta}} \tag{5}$$

    where η is a constant update rate.
- Powell's direction set method.
  - Does not require user to provide gradient function $\partial E/\partial \boldsymbol{\theta}$.
  - Can estimate gradient by itself.
- Conjugate gradient method.
  - Polak-Ribiere method requires user to provide gradient function.
  - In general, requires user to provide the Hessian (2nd derivatives).

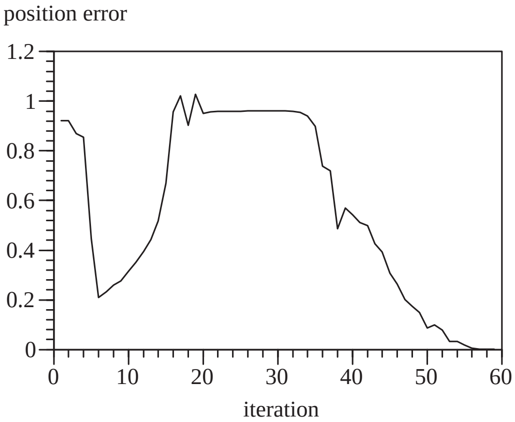
Example: Use gradient descent method to localize a point.

Actual parameter values: $(A, B, u, v, \sigma) = (10, 170, 4.4, 3.7, 1.8)$
Initial estimate $\boldsymbol{\theta}(0)$: $(A, B, u, v, \sigma) = (0, 100, 5, 3, 1)$

Optimization error $E(\boldsymbol{\theta})$ over iteration t:



Position error over iteration t:

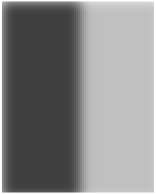


- Remember to get an initial good estimate of the parameters $\boldsymbol{\theta}(0)$ using other standard feature detection algorithms.
- Otherwise, optimization algorithm may be trapped in a local minimum, which may correspond to a wrong result.

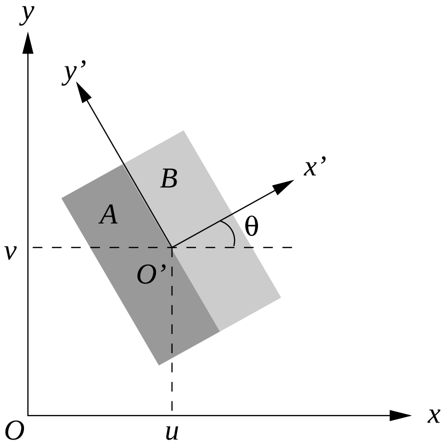
Edge Localization

Edge localization can be performed in a similar manner.

An edge is defined by a change of intensity:



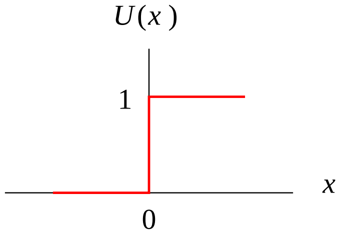
Derivation of Edge Model



- (O, x, y) is the global coordinate system of the image.
- (O', x', y') is the local coordinate system in which the edge is defined.

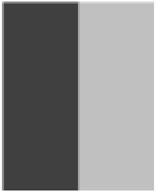
A unit step edge is defined as

$$U(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$



An ideal 2-D step edge S located at O' in the coordinate system (O', x', y') along the y'-axis is given by

$$S(x', y') = U(x')\,. \tag{7}$$

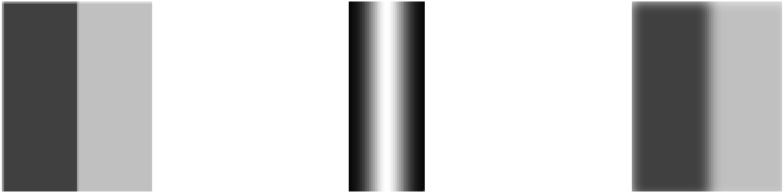


A 2-D blurred edge F can be modeled by convolving the 2-D step edge S with a 1-D Gaussian G across the edge:

$$F(x', y', \sigma) = \int G(w; \sigma)\, S(x' - w, y')\, dw \tag{8}$$

where

$$G(w; \sigma) = \exp\left(-\frac{w^2}{2\sigma^2}\right) \tag{9}$$



(a) sharp edge (b) 1-D Gaussian across edge (c) blurred edge

O' is located at (u, v) of the global coordinate system.

So, transform edge from local system to global system:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} (x-u)\cos\theta + (y-v)\sin\theta \\ -(x-u)\sin\theta + (y-v)\cos\theta \end{bmatrix} \tag{10}$$

Let the gray level on the darker side be A and the gray level on the brighter side be B. Then, the final edge model M is:

$$M(x, y, \boldsymbol{\theta}) = A + BF(x', y', \sigma) \tag{11}$$

where $\boldsymbol{\theta} = (u, v, \theta, \sigma, A, B)^T$ is the parameter vector.

Now, can compute the error of match $E(\boldsymbol{\theta})$ as

$$E(\boldsymbol{\theta}) = \sum_{\mathbf{x} \in W} \left(M(\mathbf{x}, \boldsymbol{\theta}) - I(\mathbf{x}) \right)^2 \tag{12}$$

where W is the extent of M.

Next, apply appropriate algorithm to find the $\boldsymbol{\theta}$ that minimizes the error $E(\boldsymbol{\theta})$.

The subpixel location is the (u, v) of the optimal $\boldsymbol{\theta}$.

The orientation of the edge is perpendicular to θ of the optimal $\boldsymbol{\theta}$.

Further Reading

- Subpixel corner localization algorithm [DB93, DG90].
- Various optimization algorithms [PTVF96].

Reference I

R. Deriche and T. Blaszka.
Recovering and characterizing image features using an efficient model based approach.
In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 530–535, 1993.
http://citeseer.nj.nec.com/deriche94recovering.html.

R. Deriche and G. Giraudon.
Accurate corner detection: An analytical study.
In *Proceedings of 3rd International Conference on Computer Vision*, pages 66–70, 1990.

Reference II

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery.
Numerical Recipes in C.
Cambridge University Press, 2nd edition, 1996.