Proposal v1.0

Reconfigurable Image Processing Library

Simone Casale Brunet

Ecole Polytechnique Fédérale de Lausanne
Multimedia Group Laboratory
SCI-STI-MM

March 25, 2011

Chapter 2

Introduction

A gentle introduction to this library

Chapter 3

Library Description

In this chapter is described the main structure of the CAL Image Processing library, called libImage. The main structure is the follow:

1. Spatial Domain Processing
2. Frequency Domain Processing
3. Image Restoration
4. Geometric Transformation
5. Image Registration
6. Color Image Processing
7. Image Compression
8. Morphological Image Processing
9. Image Segmentation
10. Core Functions

3.1 Spatial Domain Processing

The term spatial refer to the image plane itself. Methods in this category are based on direct manipulation of pixels in an image and are denoted by the expression:

$$g(x, y) = T[f(x, y)] \tag{3.1}$$

where $f(x, y)$ is the input image, $g(x, y)$ the output processed image and T is the operator of f, defined over a specified neighborhood about point (x, y).

3.1.1 Linear Spatial Filtering

The mechanism of linear spatial filtering consist simply in moving the center of a filter mask (or filter window) from point to point in an image f. At each point (x,y), the response of the filter at that point is the sum of products of the filter coefficients and the corresponding neighborhood pixels in the area spanned by the filter mask.

3.1.2 Non Linear Spatial Filtering

The mechanism of non linear spatial filtering is the same as discussed for the linear one. However, whereas linear spatial filtering is based on computing the sum of products (witch is a linear operation), non linear spatial filtering is based on nonlinear operations involving the pixel in the neighborhood.

A special case for the non linear spacial filtering is the intensity or gray-level transformation function. In this case the T transformation has a neighborhood of size $1x1$ (a single pixel): the value of g at (x, y) depends only on the intensity of f at that point.

3.2 Frequency Domain Processing

Methods in this category are based on filtering carried out in the frequency domain via the Fourier Transform and are denoted by the expression:

$$G(u, v) = T[F(u, v)] \tag{3.2}$$

where $F(x, y)$ is the $2D$ Fourier Transformation of the input image $f(x, y)$, $G(x, y)$ the output processed image in the frequency domain and T is the operator of F, defined over a specified neighborhood about point (x, y).

3.2.1 Linear Frequency Filtering

As for the linear spatial filtering, The mechanism of linear frequency filtering consist simply in moving the center of a filter mask (or filter window) from point to point in the frequency transformation F of the image f.

3.2.2 Non Linear Frequency Filtering

The mechanism of non linear frequency filtering is the same as the non linear spatial filtering, soft that all the considerations are made from the frequency transformation F of the image f.

Blurring: Averaging / Lowpass Filtering Sharpening: Differencing / Highpass Filtering Gaussian Highpass / Lowpass Filter

3.3 Image Restoration

The objective of restoration is to improve a given image in some predefined sense. Restoration attempts to reconstruct or recover an image that has been degraded by using a priori knowledge of the degradation phenomenon. Thus, restoration techniques are oriented toward modeling the degradation and applying the inverse process in order to recover the original image.

3.3.1 Noise Modeling

The ability to simulate the behavior and effects of noise is central to image restoration. To basic type of noise model can be used: noise in the spacial domain and noise in the frequency domain.

Spatial Noise Modeling

Spacial noise model is described by the noise probability density function.

Frequency Noise Modeling

Frequency noise model is described by various Fourier properties of the noise.

3.3.2 Filtering Restoration

Spatial Filtering Restoration

The image degradation process is modeled as a degradation function that with an additive noise term operate on an input image f(x,y) to produce a degraded image image g(x,y):

$$g(x,y) = H[f(x,y)] + n(x,y) \tag{3.3}$$

Frequency Filtering Restoration

Supposing that H is a linear, spacial invariant process, it is possible to model the degradation function in the frequency domain as:

$$G(u,v) = H(u,v)F(u,v) + N(u,v) \tag{3.4}$$

where the terms in capital letters are the Fourier transform of the corresponding terms in the convolution equation for the equation presented in the Spatial Filtering Restoration.

Restoration in the presence of noise only-spatial filtering adaptive spatial filters periodic noise reduction frequency domain filtering wiener filtering constrained latest square (regularized) filtering lucky-richardson algorithm

3.4 Geometric Transformation

Geometric transformations modify the spatial relationship between pixel in an image and are expressed as a coordinates transformation as:

$$(x,y) = T(w,z) \tag{3.5}$$

One of the most commonly used form of spatial transformation is the affine transform:

$$[xy1] = [wz1]T \tag{3.6}$$

this transformation can scale, rotate, translate or shear a set of point, depending of the value choosen for the elements of T.

3.4.1 Rotations

ROT 90 scaling rotation shear translation

3.5 Image Registration

Image registration is a process that take two images of the same scene and align them so they can be merged for visualization or for quantitative comparison.

Geometric transformations, listed before, are used frequently to perform image registration.

3.6 Color Image Processing

An RGB color image is MxNx3 array of color pixels, where each color pixel is triplet corresponding to the red, green and blue components of an RGB image at a specific spatial location.
RGB images are stored as uint8 array and their range values is [0,255]
Colors are generally represented using RGB values. However, there are other color spaces whose use in some applications may be more convenient and/or appropriate. These include NTSC, YCbCr, HSV, CMY, CMYK and his color spaces.

For **color images** the transformation model is in the form:

$$s_i = T_i(r_i)i = 1, 2, ..., n \tag{3.7}$$

where ri and si are the color components of the input and output images, n is the dimension of the color space of ri and Ti are referred to as full-color transformation (or mapping) function.

For **monochrome images** the transformation model is in the form

$$s_i = T_i(r)i = 1, 2, ..., n \tag{3.8}$$

where r denotes gray-level values, si and Ti are as above, and n is the number of color components in si.

rgb2Y Y2rgb other color spaces? RGB2Y

3.7 Image Compression

Image compression address the problem of reducing the amount of data required to represent digital image. Main important standards are supported (i.e. JPEG and JPEG2000).

3.8 Morphological ImageProcessing

Mathematical morphology tools are used for extracting image components that are useful in the representation and description of region shape such as boundaries, skeletons and the convex hull.

Dilatation erosion bridge fill closure remove diag

3.9 Image Segmentation

Segmentation subdivides an image into its constituent regions of objects. The level to witch the subdivision is carried depends on the problem being solved. That is, segmentation should stop when the object of interest in an application have been isolated.

Segmentation both for gray-level image and color image is supported.

Point detection, line detection

3.9.1 Edge detection

sobel

3.9.2 Interest region

SquareRoi

3.9.3 PredefinedRegion

CenterRegion

3.10 Core Functions

3.10.1 Math

clip

OpenImageY

DisplayImageY

Chapter 4

Standards

In this chapter are described the nomenclature and typification standard that each actor must follow in order to be included in the Reconfigurable Image Processing (RIP) library

4.1 Actor Nomenclature

All the actors must follow this 2 naming rules:

1. camel case (CamelCase)
2. the name must start with a capital letter

Examples:

- OpenImage is a valid name
- Sobel is a valid name
- 2RGB is not a valid name
- compressInformation is not a valid name
- Image_Y is not a valid name

4.2 Actor Port Structure

Chapter 5

Actor Dependencies