ECE5780 Lab 3
3D Edge Detection

A. P. Reeves

Edge Detection Stages

1. Filter (Prefilter smoothing (LPF)) remove noise

2. Enhancement: Gradient Estimation 1'st derivative,

3. Detection: select an appropriate threshold

4. Post processing
   - Non-maximum Suppression
   - Thresholding with hysteresis

Gradient Edge Operators 2D

- Estimate the gradient at each pixel location

$$\frac{\partial f(x,y)}{\partial x} \qquad \frac{\partial f(x,y)}{\partial y}$$

- Edge evidence = Magnitude of gradient =

$$\left(\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2\right)^{1/2}$$

Gradient Edge Operators 3D

- Estimate the gradient at each pixel location

$$\frac{\partial f(x,y,z)}{\partial x} \qquad \frac{\partial f(x,y,z)}{\partial y} \qquad \frac{\partial f(x,y,z)}{\partial z}$$

- Edge evidence = Magnitude of gradient =

$$\left(\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2\right)^{1/2}$$

Gradient Estimations

- Use a linear or nonlinear combination of orthogonal gradient functions to estimate the gradient (magnitude and direction) at each pixel

- Digital edge operators

(1) 2 x 2

$$\Delta 1 = \frac{\partial f(x,y)}{\partial x} = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}$$

$$\Delta 2 = \frac{\partial f(x,y)}{\partial y} = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix}$$

Edge Estimation

- To apply an operator convolve the mask with the image

$$\Delta 1 = f(x,y) - f(x+1,y)$$

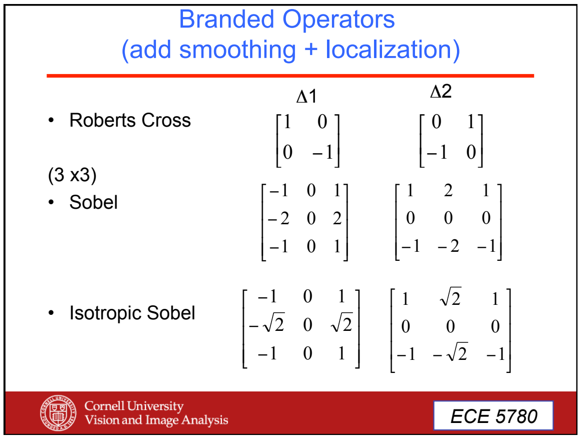
- Magnitude:

$$s(x,y) = \left(\Delta 1^2 + \Delta 2^2\right)^{1/2}$$

- Direction:

$$\phi(x,y) = \tan^{-1}\left(\frac{\Delta 2}{\Delta 1}\right)$$

- Frequent Approximation: $s'(x,y) = |\Delta 1| + |\Delta 2|$

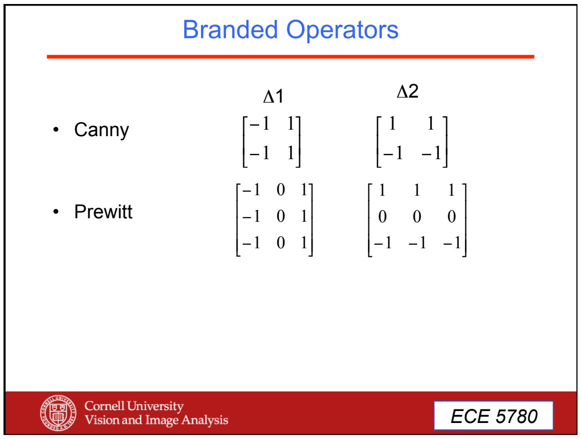


Branded Operators (add smoothing + localization)

- Roberts Cross $\begin{array}{cc} \Delta 1 & \Delta 2 \\ \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \end{array}$

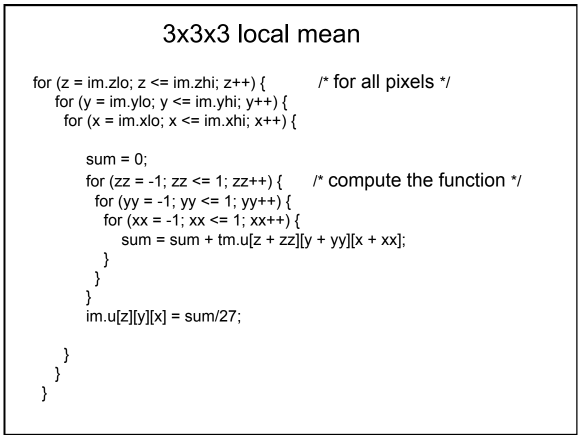
(3 x3)

- Sobel $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$
- Isotropic Sobel $\begin{bmatrix} -1 & 0 & 1 \\ -\sqrt{2} & 0 & \sqrt{2} \\ -1 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{bmatrix}$



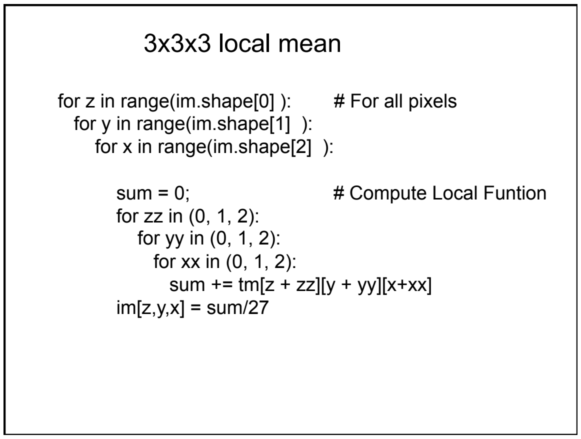
Branded Operators

- Canny $\begin{array}{cc} \Delta 1 & \Delta 2 \\ \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \end{array}$
- Prewitt $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$



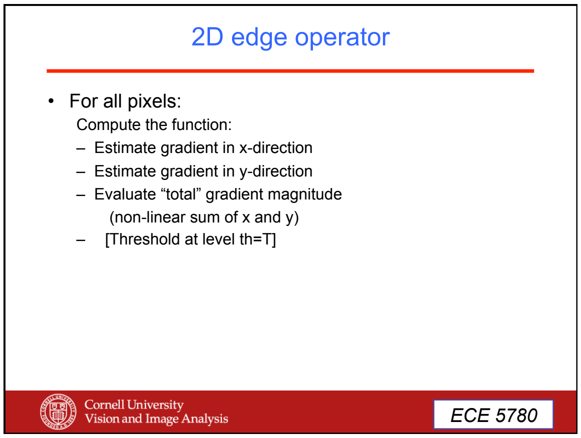
3x3x3 local mean

```
for (z = im.zlo; z <= im.zhi; z++) {          /* for all pixels */
    for (y = im.ylo; y <= im.yhi; y++) {
      for (x = im.xlo; x <= im.xhi; x++) {

          sum = 0;
          for (zz = -1; zz <= 1; zz++) {      /* compute the function */
            for (yy = -1; yy <= 1; yy++) {
              for (xx = -1; xx <= 1; xx++) {
                  sum = sum + tm.u[z + zz][y + yy][x + xx];
              }
            }
          }
          im.u[z][y][x] = sum/27;

      }
    }
  }
```



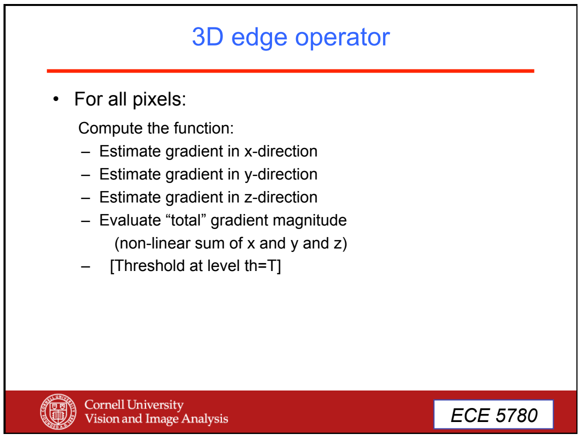
3x3x3 local mean

```
for z in range(im.shape[0] ):       # For all pixels
   for y in range(im.shape[1]  ):
      for x in range(im.shape[2]  ):

         sum = 0;                     # Compute Local Funtion
         for zz in (0, 1, 2):
            for yy in (0, 1, 2):
               for xx in (0, 1, 2):
                  sum += tm[z + zz][y + yy][x+xx]
         im[z,y,x] = sum/27
```



2D edge operator

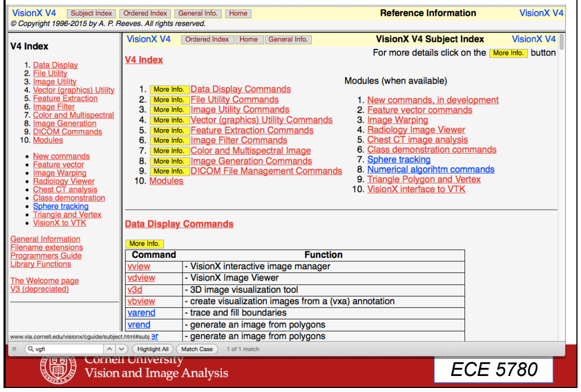
- For all pixels:
  Compute the function:
  – Estimate gradient in x-direction
  – Estimate gradient in y-direction
  – Evaluate "total" gradient magnitude
    (non-linear sum of x and y)
  – [Threshold at level th=T]



3D edge operator

- For all pixels:
  Compute the function:
  – Estimate gradient in x-direction
  – Estimate gradient in y-direction
  – Estimate gradient in z-direction
  – Evaluate "total" gradient magnitude
    (non-linear sum of x and y and z)
  – [Threshold at level th=T]

VisionX V4 Command Index



© Copyright 1996-2015 by A. P. Reeves. All rights reserved.

V4 Index

1. Data Display
2. File Utility
3. Image Utility
4. Vector (graphics) Utility
5. Feature Extraction
6. Image Filter
7. Color and Multispectral
8. Image Generation
9. DICOM Commands
10. Modules

- New commands
- Feature vector
- Image Warping
- Radiology Viewer
- Chest CT analysis
- Class demonstration
- Sphere tracking
- Triangle and Vertex
- VisionX to VTK

General Information
Filename extensions
Programmers Guide
Library Functions

The Welcome page
V3 (depreciated)

VisionX V4 Subject Index

For more details click on the More Info. button

V4 Index

1. More Info. Data Display Commands
2. More Info. File Utility Commands
3. More Info. Image Utility Commands
4. More Info. Vector (graphics) Utility Commands
5. More Info. Feature Extraction Commands
6. More Info. Image Filter Commands
7. More Info. Color and Multispectral Image
8. More Info. Image Generation Commands
9. More Info. DICOM File Management Commands
10. Modules

Modules (when available)

1. New commands, in development
2. Feature vector commands
3. Image Warping
4. Radiology Image Viewer
5. Chest CT image analysis
6. Class demonstration commands
7. Sphere tracking
8. Numerical algorihtm commands
9. Triangle Polygon and Vertex
10. VisionX interface to VTK

Data Display Commands

More Info.

Command	Function
vview	- VisionX interactive image manager
vdview	- VisionX Image Viewer
v3d	- 3D image visualization tool
vbview	- create visualization images from a (vxa) annotation
varend	- trace and fill boundaries
vrend	- generate an image from polygons
er	- generate an image from polygons

www.via.cornell.edu/visionx/cguide/subject.html#subj

2D Edge Detection: VisionX V4

- Edge detection (see subject topic image filters OR search for key "edge"
- **vedge** [if=<infile>] [of=<outfile>] [th=<value>] [a=<window>]
- vderiche - Deriche edge operator
- vsobel - Sobel edge function
- vrcross - Roberts cross edge operator
- vedgex - edge operator cleanup by non-maximum suppression, and thresholding with hysteresis

3D Edge Detection: VisionX V4

1. Filter: remove noise

- vmean - mean filter
- vgfilt - Gaussian filter
- vmedian - image median filter
- vsf - very general convolution filter

2. Enhancement: gradient magnitude

- v3grad - calculates the 1D, 2D or 3D gradient of an image **(output is in floating-point format)**

3. Detection: threshold

- vpix - point operations on the pixels of an image

Misc.

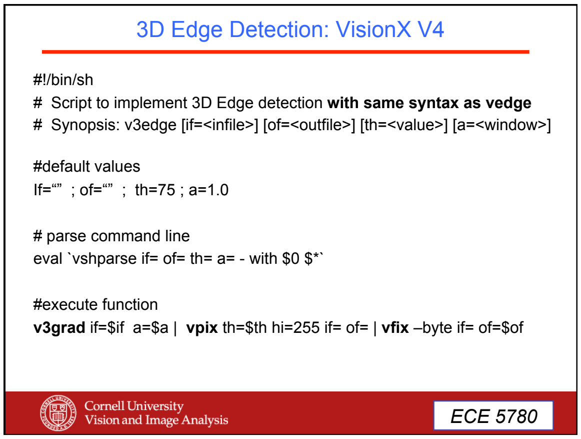
- vfix - change the image format of a VisionX image file (2D)
- vifx - Scale image data and convert to byte or float format (3D)

3D Edge Detection: VisionX V4

```
#!/bin/sh
# simple sh script to implement 3D Edge detection
#  Synopsis:    v3edge <input-file> <output-file> <threshold> <window>
#                                   $1             $2             $3            $4


vgfilt if=$1 xs=$4 ys=$4 zs=$4 of= | v3grad if= of= \
           | vpix th=$3 hi=255 if= of= | vfix –byte if= of=$2

#### Alternative implementation using built in filter of v3grad
#
#    v3grad if=$1  a=$4 |  vpix th=$3 hi=255 if= of= | vfix –byte if= of=$2
```

3D Edge Detection: VisionX V4

```
#!/bin/sh
# simple sh script to implement 3D Edge detection
#  Synopsis:    v3edge <input-file> <output-file> <threshold> <window>
#                                   $1             $2             $3            $4

###### version using temporary files instead of pipes

vgfilt if=$1 xs=$4 ys=$4 zs=$4 of=tmp1
v3grad if=tmp1 of=tmp2
vpix th=$3 hi=255 if=tmp2  of=tmp3
vfix –byte if=tmp3 of=$2

rm tmp1 tmp2 tmp3
```



3D Edge Detection: VisionX V4

```
#!/bin/sh
#  Script to implement 3D Edge detection with same syntax as vedge
#  Synopsis: v3edge [if=<infile>] [of=<outfile>] [th=<value>] [a=<window>]

#default values
If=“”  ; of=“”  ;  th=75 ; a=1.0

# parse command line
eval `vshparse if= of= th= a= - with $0 $*`

#execute function
v3grad if=$if  a=$a |  vpix th=$th hi=255 if= of= | vfix –byte if= of=$of
```

Cornell University
Vision and Image Analysis

ECE 5780