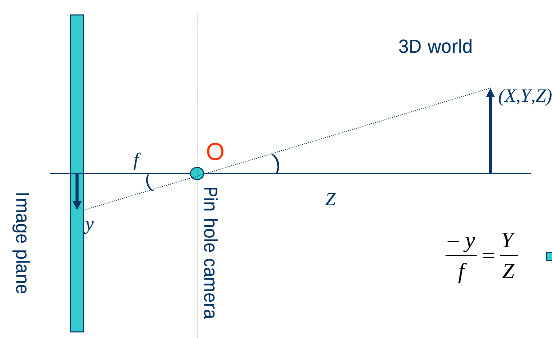
CAP 5415 Computer Vision
Fall 2005

Dr. Alper Yilmaz
Univ. of Central Florida
www.cs.ucf.edu/courses/cap5415/fall2005
Office: CSB 250

Perspective Projection



$$\frac{-y}{f} = \frac{Y}{Z} \quad \Rightarrow \quad y = -f\frac{Y}{Z}$$

$$x = -f\frac{X}{Z}$$

Camera Parameters

- Image coordinates (x_{image}, y_{image})
- Image center (o_x, o_y)
- Camera coordinates (x_{camera}, y_{camera})
- Real world coordinates (X, Y, Z)
- Focal length f
- Effective size of pixel in millimeter (k_x, k_y)

Camera to image

$$x_{image} = k_x x_{camera} + o_x$$

$$y_{image} = k_y y_{camera} + o_y$$

$$\begin{bmatrix} x_{image} \\ y_{image} \\ 1 \end{bmatrix} = \begin{bmatrix} k_x & 0 & o_x \\ 0 & k_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{camera} \\ y_{camera} \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} U_{image} \\ V_{image} \\ S \end{bmatrix} = \begin{bmatrix} k_x & 0 & o_x \\ 0 & k_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U_{camera} \\ V_{camera} \\ S \end{bmatrix}$$

$U_{camera} = -fX$, $V_{camera} = -fY$, $S = Z$

Real world to camera

- World and camera coordinates
- t_x, t_y, t_z and $r_{1,1} \dots r_{3,3}$ are extrinsic camera parameters

$$\mathbf{X}_{camera} = R\mathbf{X}_{world} + T$$

$$\begin{bmatrix} X_{camera} \\ Y_{camera} \\ Z_{camera} \\ 1 \end{bmatrix} = \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_x \\ r_{2,1} & r_{2,2} & r_{2,3} & t_y \\ r_{3,1} & r_{3,2} & r_{3,3} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{world} \\ Y_{world} \\ Z_{world} \\ 1 \end{bmatrix}$$

Real world to image

$$\overbrace{\begin{bmatrix} U_{image} \\ V_{image} \\ S \end{bmatrix}}^{\text{homogenous image coordinates}} = \overbrace{\begin{bmatrix} -fk_x & 0 & 0 & o_x \\ 0 & -fk_y & 0 & o_y \\ 0 & 0 & 1 & 0 \end{bmatrix}}^{\text{intrinsic camera parameters}} \overbrace{\begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_x \\ r_{2,1} & r_{2,2} & r_{2,3} & t_y \\ r_{3,1} & r_{3,2} & r_{3,3} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}^{\text{extrinsic camera parameters}} \overbrace{\begin{bmatrix} X_{world} \\ Y_{world} \\ Z_{world} \\ 1 \end{bmatrix}}^{\text{homogenous world coordinates}}$$

$$x_{image} - o_x = -f_x \frac{r_{1,1}X_{world} + r_{1,2}Y_{world} + r_{1,3}Z_{world} + t_x}{r_{3,1}X_{world} + r_{3,2}Y_{world} + r_{3,3}Z_{world} + t_z} \quad \text{(A)}$$

$$y_{image} - o_y = -f_y \frac{r_{2,1}X_{world} + r_{2,2}Y_{world} + r_{2,3}Z_{world} + T_y}{r_{3,1}X_{world} + r_{3,2}Y_{world} + r_{3,3}Z_{world} + T_z} \quad \text{(B)}$$

Estimation of Camera Parameters

- Given corresponding world and image points
- Divide (A) to (B), rearrange result

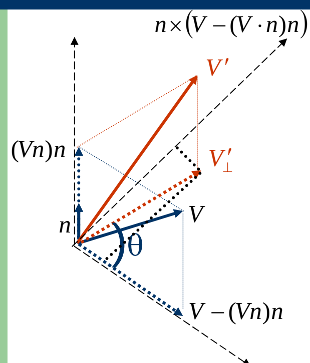
$$x_i X_i v_1 + x_i Y_i v_2 + x_i Z_i v_3 + x_i v_4 - y_i X_i v_5 - y_i Y_i v_6 - y_i Z_i v_7 - y_i v_8 = 0 \quad \textbf{(C)}$$

$$v_1 = r_{21} \quad v_2 = r_{22} \quad v_3 = r_{23} \quad v_4 = t_y \quad v_5 = \alpha r_{11} \quad v_6 = \alpha r_{12} \quad v_7 = \alpha r_{13} \quad v_8 = \alpha t_x$$

- Rearrange into matrix and solve using SVD
- Estimate scale factor ➔ r_{2i} and t_y are there!!
- Compute α similar to scale factor
- Compute r_{3i} from r_{1i} and r_{2i}.
- Estimate f_x, f_y and t_y.
- Finally compute o_x and o_y from other knowns

Alper Yilmaz, Fall 2005 UCF

Rotation around arbitrary axis



$$V = (V \cdot n)n + (V - (V \cdot n)n)$$

$$n \times V = n \times (V - (V.n)n)$$

$$n \times (n \times V) = (V.n)n - V$$

$$V'_\perp = \cos\theta (V - (V \cdot n)n) + \sin\theta (n \times V)$$

$$V' = V'_\perp + (V \cdot n)n$$

$$V' = -\cos\theta (n \times (n \times V)) + \sin\theta (n \times V) + n \times (n \times V) + V$$

Alper Yilmaz, Fall 2005 UCF

Derivative

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x) = f_x$$

Discrete Derivative Finite Difference

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$ Backward difference

$$\frac{df}{dx} = f(x) - f(x+1) = f'(x)$$ Forward difference

$$\frac{df}{dx} = f(x+1) - f(x-1) = f'(x)$$ Central difference

Derivatives in 2 Dimensions

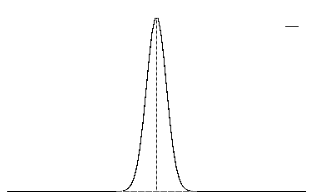
Given function $f(x, y)$

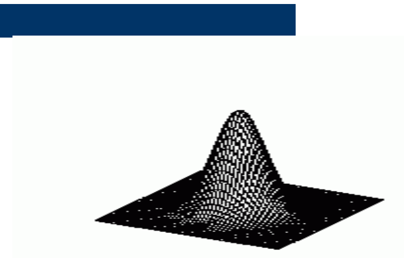
Gradient vector $\nabla f(x, y) = \begin{bmatrix} \frac{\partial f(x, y)}{\partial x} \\ \frac{\partial f(x, y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix}$

Gradient magnitude $|\nabla f(x, y)| = \sqrt{f_x^2 + f_y^2}$

Gradient direction $\theta = \tan^{-1} \frac{f_x}{f_y}$

Gaussian Filter





$$g(x) = e^{\frac{-x^2}{2\sigma^2}}$$

$$g(x, y) = e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

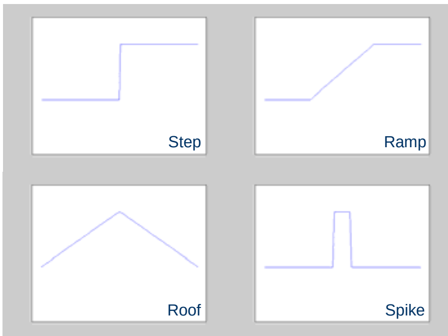
$$g(x) = [.011 \quad .13 \quad .6 \quad 1 \quad .6 \quad .13 \quad .011]$$

Properties of Gaussian

- Separable filter (2D Gaussian can be separated into 2 1D Gaussians)
- Smooth function, it has infinite number of derivatives
- Fourier Transform of Gaussian is Gaussian.
- Convolution of a Gaussian with itself is a Gaussian.
- There are cells in eye that perform Gaussian filtering.

What is an Edge?

- Discontinuity of intensities in the image
- Edge models
  - Step
  - Roof
  - Ramp
  - Spike



Recap (Edge Detection)

- Prewitt and Sobel edge detectors
  - Compute derivatives
    - In x and y directions
  - Find gradient magnitude
  - Threshold gradient magnitude
- Difference between Prewitt and Sobel is the derivative filters

Prewitt Edge Detector

Prewitt's edges in x direction

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \rightarrow I_x$$

Prewitt's edges in y direction

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \rightarrow I_y$$

Sobel Edge Detector

Sobel's edges in x direction

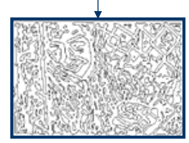
$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \rightarrow I_x$$

Sobel's edges in y direction

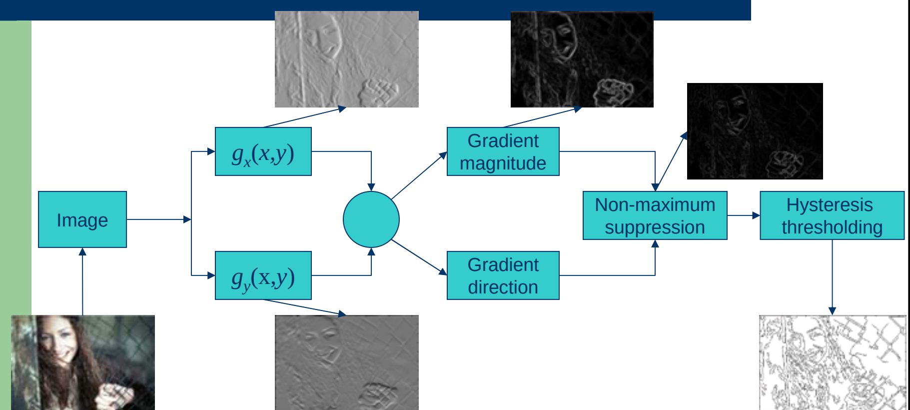
$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \rightarrow I_y$$

Marr-Hildreth Edge Detector

Image → $\Delta^2 g(x)$ → ($I * \Delta^2 g$) → Find zero-crossings → compute slope → Threshold

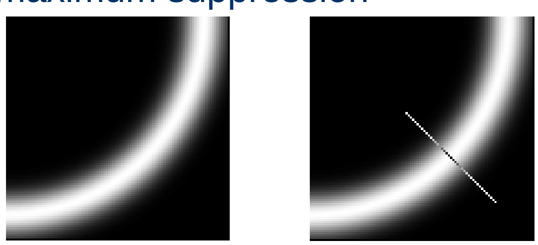


Canny Edge Detector



Canny Edge Detector
Fourth Step

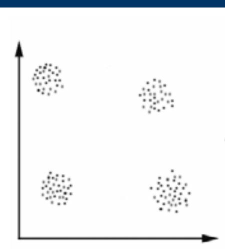
- Non maximum suppression



Canny Edge Detector
Hysteresis Thresholding

- If the gradient at a pixel is
  - above "**High**", declare it an '**edge pixel**'
  - below "**Low**", declare it a "**non-edge-pixel**"
  - **between** "low" and "high"
    - Consider its neighbors iteratively then declare it an "edge pixel" if it is **connected** to an 'edge pixel' **directly** or via pixels **between** "low" and "high".

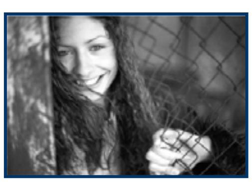
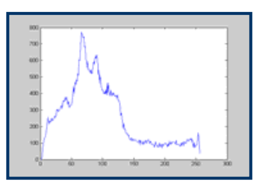
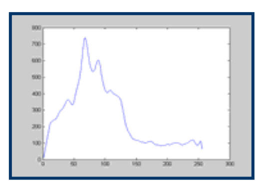
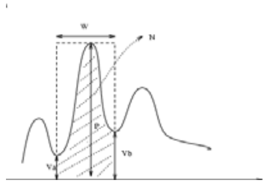
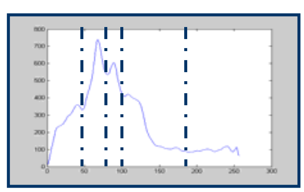
Clustering (Segmentation)



Similarity Constraints

- Same gray levels.
- Distance less than threshold
- Distance to mean less than threshold
- Small standard deviation within region

Histogram and segmentation

Connected Components

- Recursive algorithm
  - One pass
- Sequential algorithm
  - Two pass
    - Labeling
    - Merging

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Steps in Seed Segmentation

1. Compute and smooth image histogram.
2. Detect good peaks and set thresholds
3. Apply connected component algorithm.
4. Merge small regions, split big regions

Region Growing Approaches

- Region splitting and merging
  - Region similarity function
- Phagocyte algorithm (boundary melting)
  - Compute intensity similarity between pixels of regions
- Likelihood ratio test
  - Two hypothesis: $\exists$ one region, $\exists$ two regions

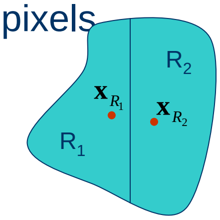
Steps in Region Segmentation

1. Apply seed segmentation
2. Apply one of the following to all regions
   1. Region splitting and merging
   2. Phagocyte algorithm
   3. Likelihood ratio test

Phagocyte Algorithm

- Boundary melting
  - Remove weak boundaries
  - Similar to region merging
- Boundary weakness is based on color similarity between each pair of pixels

$$S(\mathbf{x}_{R_1}, \mathbf{x}_{R_2}) = \left| I(\mathbf{x}_{R_1}) - I(\mathbf{x}_{R_2}) \right|$$



Alper Yilmaz, Fall 2005 UCF

Likelihood Ratio Test

$$P(H_1) = \left(\frac{1}{\sqrt{2\pi}\sigma_{\text{one region}}} \right)^{m_1+m_2} e^{-\frac{m_1+m_2}{2}},$$

$$P(H_2) = \left(\frac{1}{\sqrt{2\pi}\sigma_A} \right)^{m_1} e^{-\frac{m_1}{2}} \left(\frac{1}{\sqrt{2\pi}\sigma_B} \right)^{m_2} e^{-\frac{m_2}{2}}$$

$$LH = \frac{P(H_2)}{P(H_1)} = \frac{(\sigma_0)^{m_1+m_2}}{(\sigma_A)^{m_1}(\sigma_B)^{m_2}}$$

Merge regions if $LH<T$.

Alper Yilmaz, Fall 2005 UCF

Difference Between Segmentation and Edge Detection

- Closed boundary
  - Edges are usually open
  - Segmentation provides closed boundaries
- Local or global
  - Edges are computed in the locality
  - Segmentation is global
- Increasing feature vector dimensionality
  - Does not drastically improve edge detection
  - Improves segmentation (motion, texture information etc.)
- Boundary position
  - Localized in edge detection
  - Usually not localized (recent advancements use locality as well)
    - Especially contour based segmentation

Region Properties

- Area
- Centroid
- Moments
- Perimeter
- Compactness
- Orientation

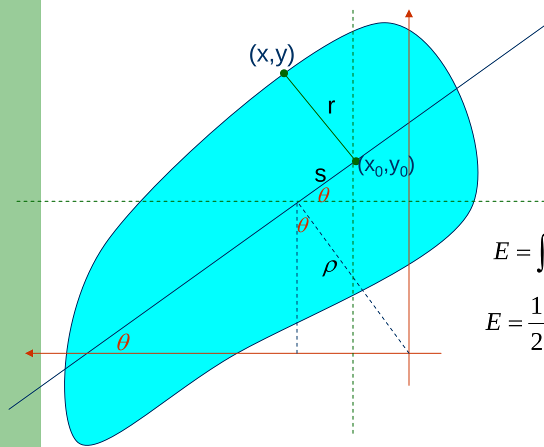
$$A = \sum_{x=0}^{m}\sum_{y=0}^{n} B(x,y)$$

$$\bar{x} = \frac{\sum_{x=0}^{m}\sum_{y=0}^{n} xB(x,y)}{A} \qquad \bar{y} = \frac{\sum_{x=0}^{m}\sum_{y=0}^{n} yB(x,y)}{A}$$

$$\mu_{pq} = \int\int (x-\bar{x})^p (y-\bar{y})^q B(x,y)\ d(x-\bar{x})d(y-\bar{y})$$

$$C = \frac{Perimeter^2}{4\pi A}$$

Recap
Region Orientation



$$E = \iint r^2 B(x,y)\,dxdy$$

$$E = \iint (x\sin\theta - y\cos\theta + \rho)^2 B(x,y)\,dxdy$$

$$E = \frac{1}{2}(a+c) - \frac{1}{2}(a-c)\cos 2\theta - \frac{1}{2} b \sin 2\theta$$

Some Rules

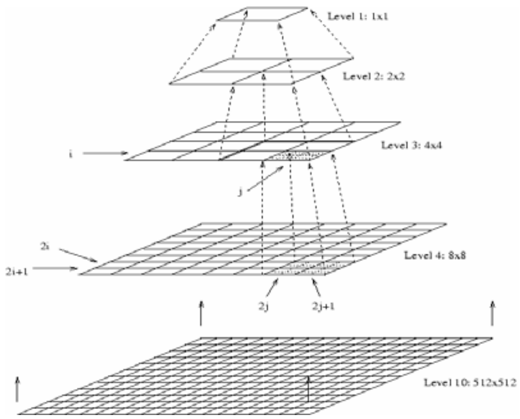
$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad |\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

$$\frac{\partial}{\partial x}\left(e^{f(x,y)}\right) = \frac{\partial f(x,y)}{\partial x} e^{f(x,y)} = f_x e^{f(x,y)}$$

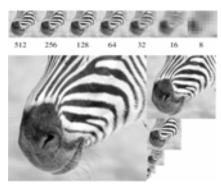
$$\frac{\partial}{\partial x}\left(I\left(f(x,y), g(x,y)\right)\right) = \frac{\partial I}{\partial f}\frac{\partial f}{\partial x} + \frac{\partial I}{\partial g}\frac{\partial g}{\partial x}$$

$$\frac{\partial}{\partial y}\left(I\left(f(x,y), g(x,y)\right)\right) = \frac{\partial I}{\partial f}\frac{\partial f}{\partial y} + \frac{\partial I}{\partial g}\frac{\partial g}{\partial y}$$

Pyramid Representation



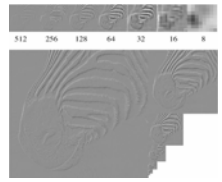
Gaussian Pyramid



- Apply 1D Gaussian mask to alternate pixels along each row of image.
- Apply 1D Gaussian mask to alternate pixels along each column of resulting image from previous step.

Alper Yilmaz, Fall 2005 UCF

Laplacian Pyramid



Constructing LP

$$L_1 = g_1 - EXPAND[g_2]$$
$$L_2 = g_2 - EXPAND[g_3]$$
$$L_3 = g_3 - EXPAND[g_4]$$
$$L_4 = g_4$$

Reconstructing image from LP

$$g_4 = L_4$$
$$g_3 = EXPAND\ [g_4] + L_3$$
$$g_2 = EXPAND\ [g_3] + L_2$$
$$g_1 = EXPAND\ [g_2] + L_1$$

Alper Yilmaz, Fall 2005 UCF