Edge Detection via Objective functions

Gowtham Bellala
Kumar Sricharan

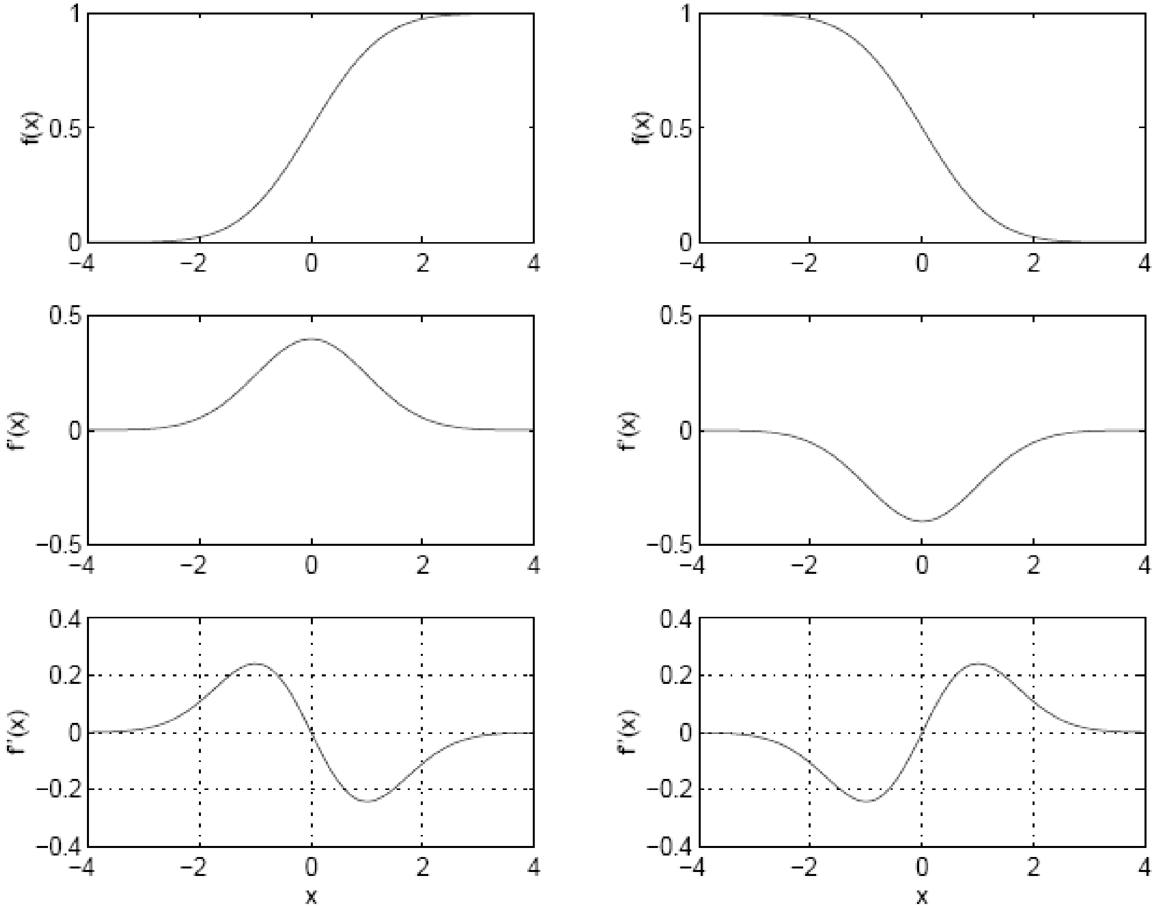
Edge Detection – a quick recap!

- Much of the "information" in an image is in the edges
- Edge detection is usually for the purpose of subsequent computer processing
- Examples : Image segmentation, object recognition, object detection
- General framework :

$$\underbrace{f(x,y)}_{\text{image}} \rightarrow \boxed{\text{edge detection system}} \rightarrow \underbrace{e(x,y)}_{\text{edge map}} = \begin{cases} 1, & (x,y) \in \text{edge} \\ 0, & \text{otherwise.} \end{cases}$$

$$\underbrace{f(x,y)}_{\text{image}} \rightarrow \boxed{\text{edge enhancer}} \rightarrow \boxed{\text{edge detector}} \rightarrow \boxed{\text{thinning}} \rightarrow \underbrace{e(x,y)}_{\text{edge map}}$$

Concept



- Approximate the derivative by finite differences

$$f'_a(x)|_{x=n\Delta_x} \approx \begin{cases} \dfrac{f[n]-f[n-1]}{\Delta_x} & \text{left difference} \\ \dfrac{f[n+1]-f[n]}{\Delta_x} & \text{right difference} \\ \dfrac{f[n+1]-f[n-1]}{2\Delta_x} & \text{central difference} \end{cases}$$

A Classical method - Sobel

- Sobel's method :

  Step 1: Utilizes masks S_x and S_y to obtain edge intensities E_x and E_y

  Step 2: Thresholds the edge map to detect edge points.

  Disadvantage : Edges detected are thicker than actual edges.

$$|\nabla f(x,y)| \approx \sqrt{(f_x[n,m])^2 + (f_y[n,m])^2}$$

$$f_x[n,m] \triangleq f[n,m] ** h_x[n,m] \text{ and } f_y[n,m] \triangleq f[n,m] ** h_y[n,m]$$

-1	0	1
-2	0	2
-1	0	1

S_x

1	2	1
0	0	0
-1	-2	-1

S_y

Canny

- Canny's Method :

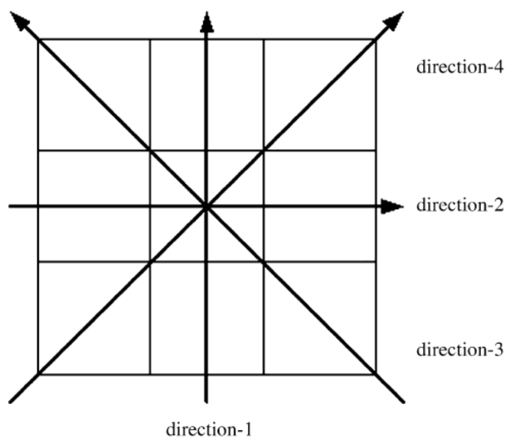
  Step 1: Apply a Gaussian filter to the image

  Step 2: Gradient of the smoothed image is computed in x and y directions

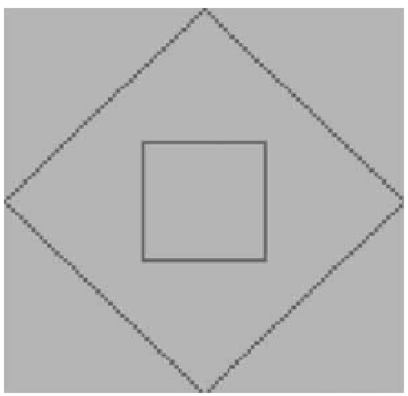
  Step 3: Edge and Direction maps are generated

  Step 4: Non Maxima Suppression (NMS) is performed

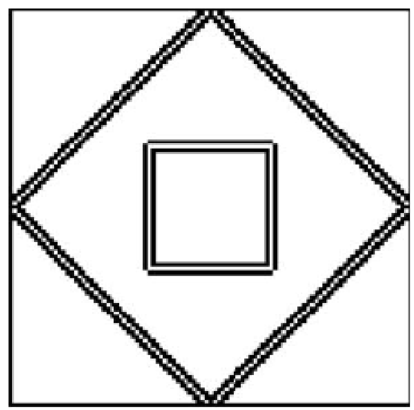
  Advantage over Sobel : Thinner edges



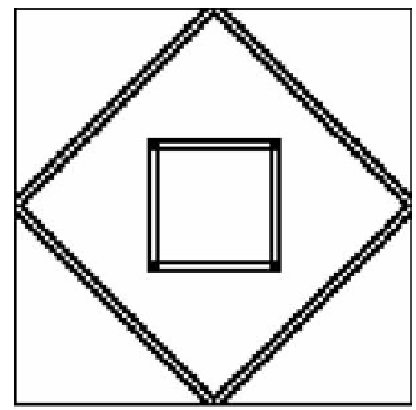
- However, both Canny and Sobel methods obtain zero intensity if a line of one – pixel width passes through the mask.
- Resulting in double edges as shown below



True Image



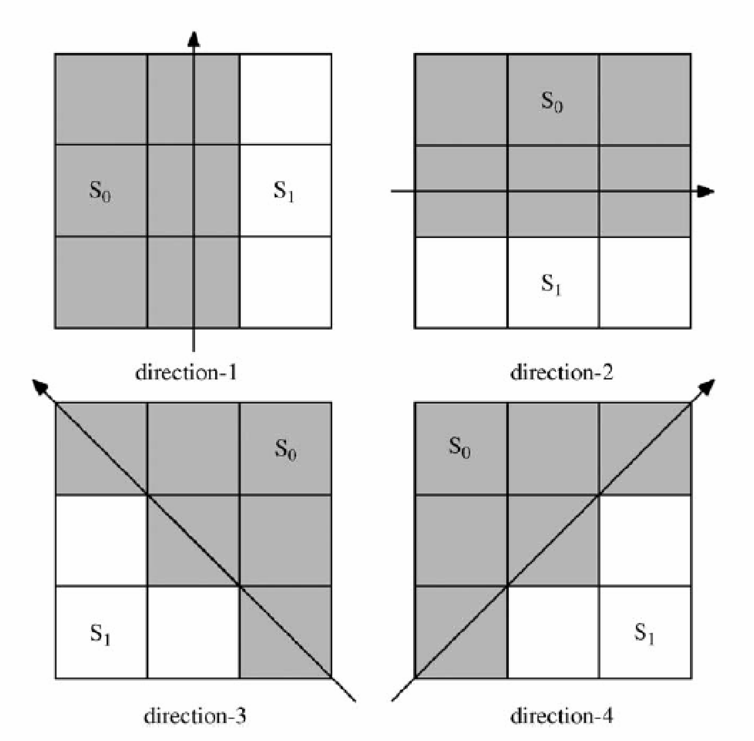
Sobel method

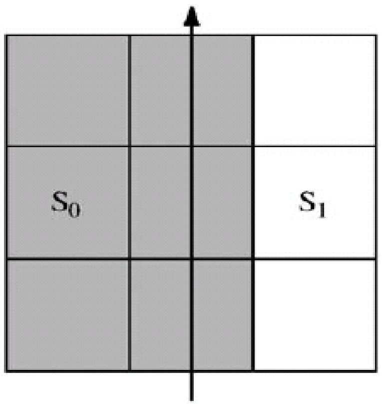


Canny method

Objective function method

- Algorithm follows an outline close to Canny
- Edge direction is decided by computing certain cost function





$$f_i = (L-1)\frac{N_f}{D_f},$$

$$N_f = \min\left(1, \frac{|m_0 - m_1|}{w_1}\right),$$

$$D_f = 1 + \frac{1}{15}\sum_{\substack{p_m, p_n \in S_0 \\ m>n \\ m \neq n}} \min\left(1, \frac{|p_m - p_n|}{w_2}\right) + \frac{1}{3}\sum_{\substack{p_m, p_n \in S_1 \\ m>n \\ m \neq n}} \min\left(1, \frac{|p_m - p_n|}{w_2}\right)$$

$$E(x, y) = \max(f_1, f_2, f_3, f_4),$$

$$D(x, y) = Arg(\max(f_1, f_2, f_3, f_4)).$$

- Non Maxima Suppression

(1) $D(x, y) = 1$ and $E(x, y) > E(x, y-1)$ and $E(x, y) > E(x, y+1)$.

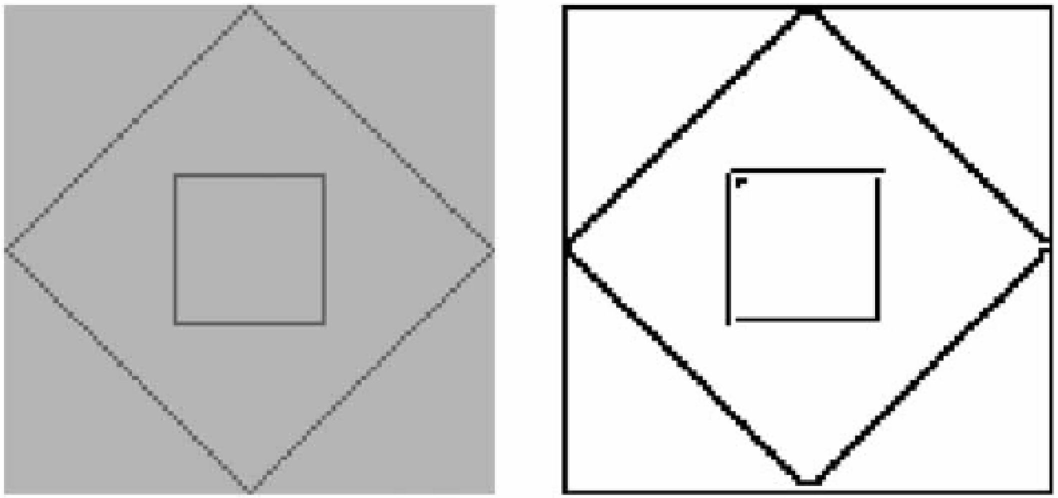
(2) $D(x, y) = 2$ and $E(x, y) > E(x-1, y)$ and $E(x, y) > E(x+1, y)$.

(3) $D(x, y) = 3$ and $E(x, y) > E(x-1, y+1)$ and $E(x, y) > E(x+1, y-1)$.

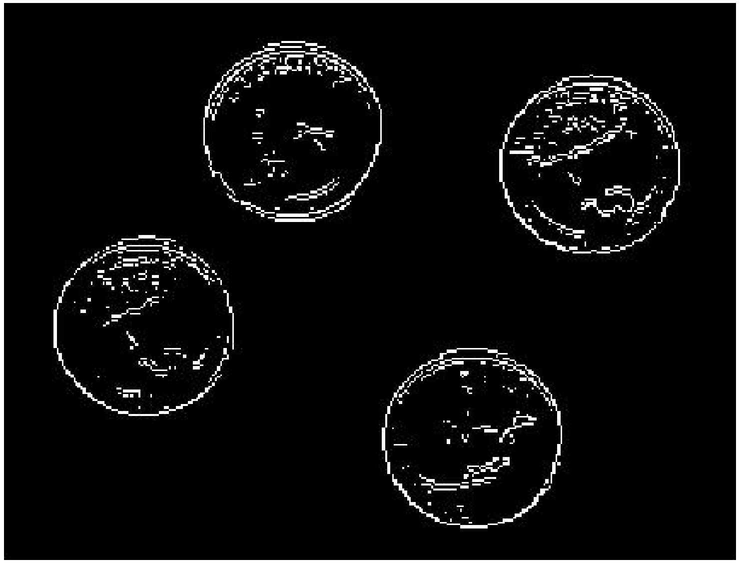
(4) $D(x, y) = 4$ and $E(x, y) > E(x-1, y-1)$ and $E(x, y) > E(x+1, y+1)$.

Pseudo Code

- Step 1 : Compute 4 objective function values
- Step 2 : Generate Edge map and Direction map
- Step 3 : Apply NMS to the edge and direction maps and extract the edge points in the image.



Original Image

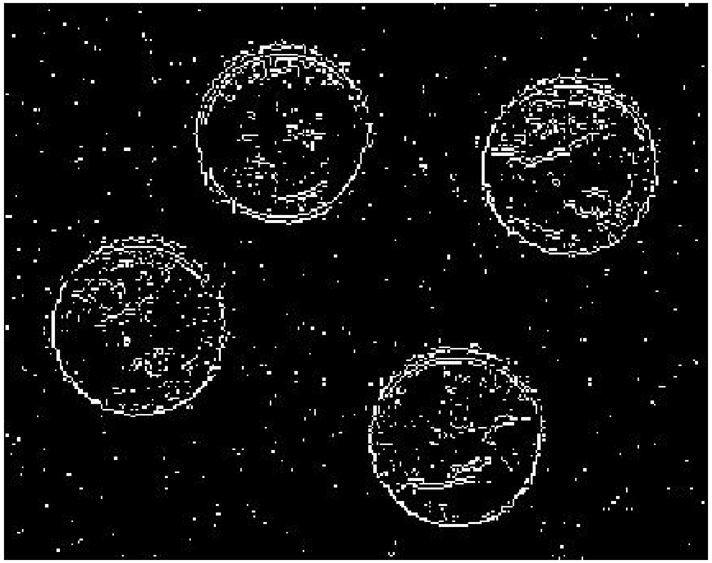


Edge Detected Image

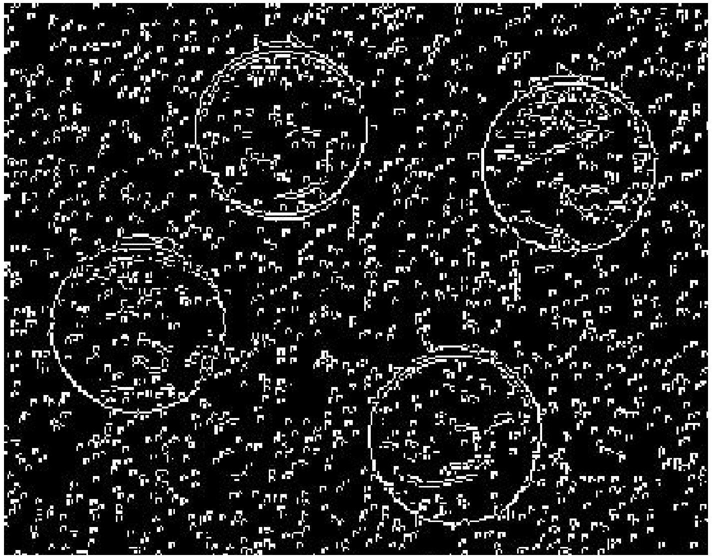
Drawbacks!

- w_0 and w_1 are constants determined from empirical results, not specific to an Image – leads to problems
- The algorithm outperforms Canny and Sobel edge detectors only in an ideal case.
- Fails to work efficiently in presence of Noise

Performance of the algorithm in noise



Gaussian Noise

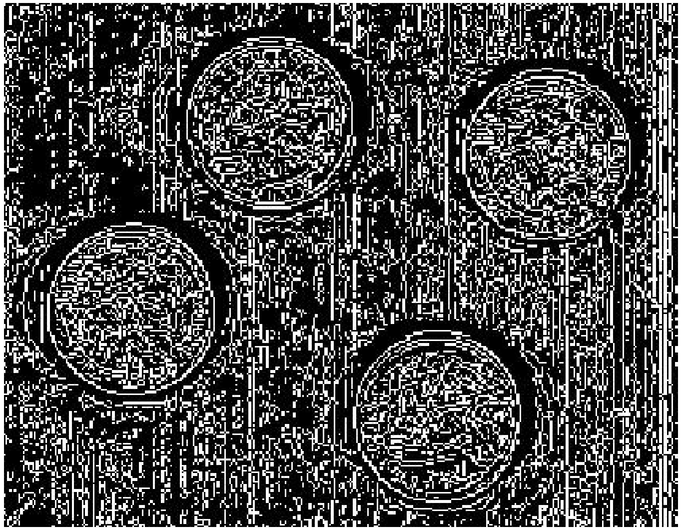


Salt and Pepper Noise

Proposed Improvements

A New cost function

- Original cost function depends on empirical estimates w0,w1
- Bad w0,w1 estimates will lead to poor edge detection
- We propose a more natural measure for inter-set and intra-set distances



Remodeling the cost function

- Cost function f = N_f/D_f

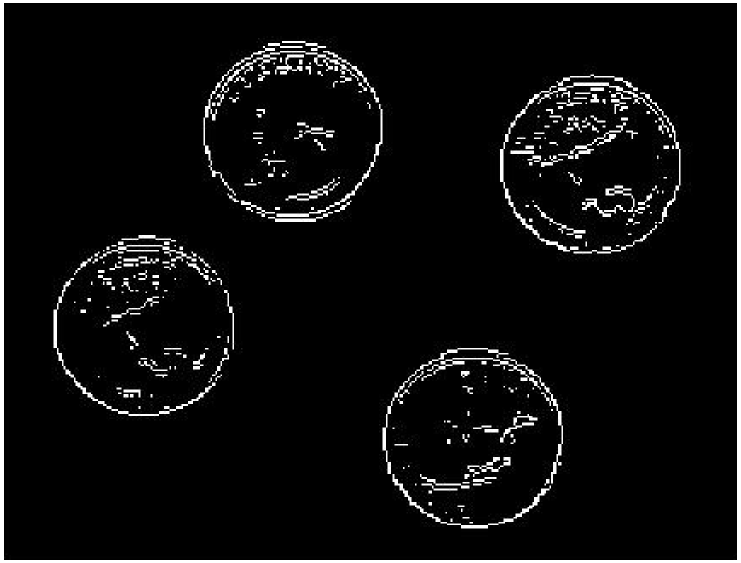
N_f = $$\sum_S N_S(\mu_S - \overline{x}_S)^2$$

D_f = $$1+\sum_S \sum_{i\in S}(x_i - \mu_S)^2$$

$$\mu_S = \frac{1}{N_S}\sum_{i\in S} x_i$$

$$\overline{x} = \frac{1}{N}\sum_i x_i$$

These distance measures have been borrowed from theory of LDA



Edge detected image using modified objective function

Objective function for edge detection in noise

- The drawback can be overcome by coming up with a clever choice of the objective function such that the existing framework can be used even in presence of noise

- We consider two different cases

  Case 1 : Image in presence of AWGN

  Case 2 : Image in presence of Salt and Pepper noise

Gaussian Noise

- How can I remove Gaussian noise?

  Averaging!
- But, we don't want to blur the edges

  Adaptive averaging – concept behind adaptive Wiener filtering

$$x^{1}(i,j)=\begin{cases}\mu(i,j) & \text{if } \left|\mu(i,j)-x^{0}(i,j)\right|<a\\ x^{0}(i,j) & \text{if } \left|\mu(i,j)-x^{0}(i,j)\right|>a\end{cases}$$

- Incorporate this into the objective function

Modified cost function for Gaussian Noise

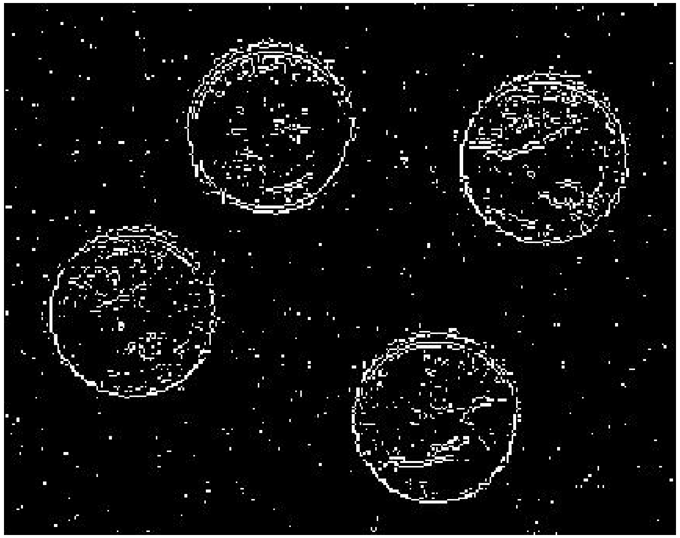
- Cost function f = N_f/D_f

N_f =

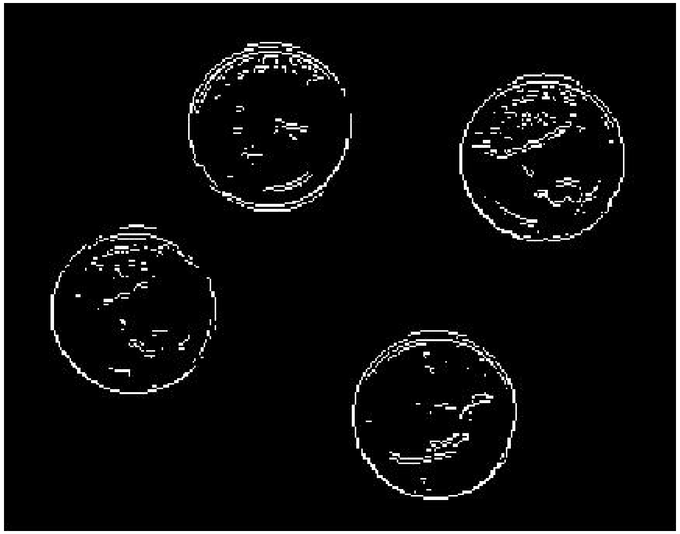
$$\begin{cases} \sum_S N_S(\mu_S - \overline{x}_S)^2 & \text{if } |\mu_S - \overline{x}_S| > a \\ 0 & \text{if } |\mu_S - \overline{x}_S| < a \end{cases}$$

Df =

$$\begin{cases} 1 + \sum_S \sum_{i \in S} (x_i - \mu_S)^2 & \text{if } |x_i - \mu_S| > a \\ 1 & \text{else} \end{cases}$$



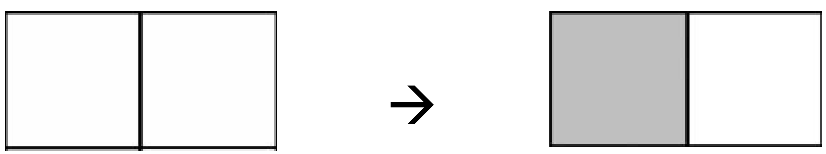
Original Algorithm



Proposed Algorithm

Salt and Pepper Noise

- Unlike Gaussian noise, Impulse/Salt and Pepper noise corrupts the image with very high values.
- Resulting in many false edges! - Why?



Normal edge detection = $f(i,j+1) – f(i,j)$

- Solution :

Proposed edge detection = $log(int(\alpha f(i,j+1))) – log(int(\alpha f(i,j)))$

where ‘int’ stands for greatest integer

Remodeling the cost function

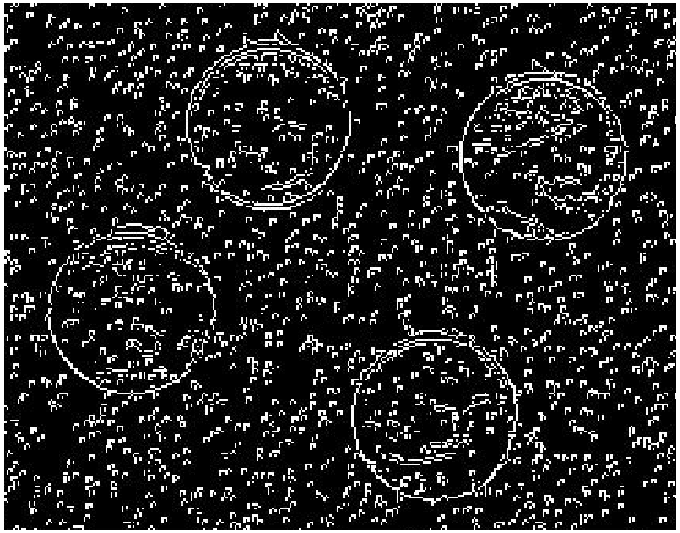
- Cost function $f = N_f/D_f$

where $N_f =$ $$\sum_S N_S(\log \mu_S - \log \bar{x}_S)^2$$

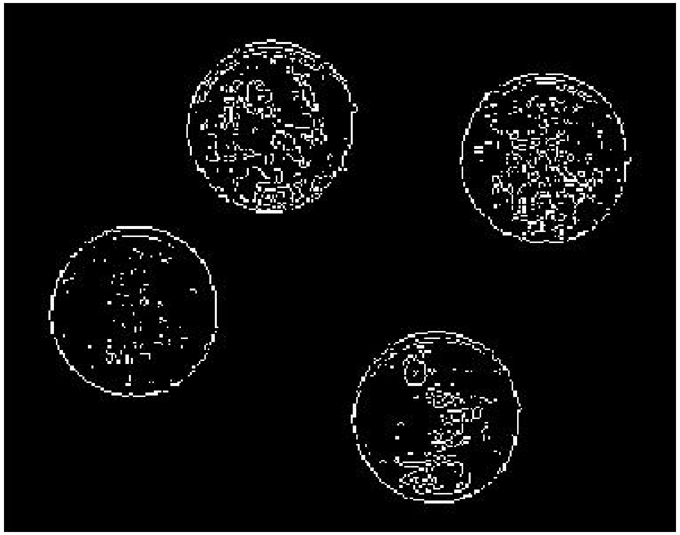
$D_f =$ $$1 + \sum_S \sum_{i \in S} (\log x_i - \log \mu_S)^2$$

$$\mu_S = \frac{1}{N_S} \sum_{i \in S} x_i$$

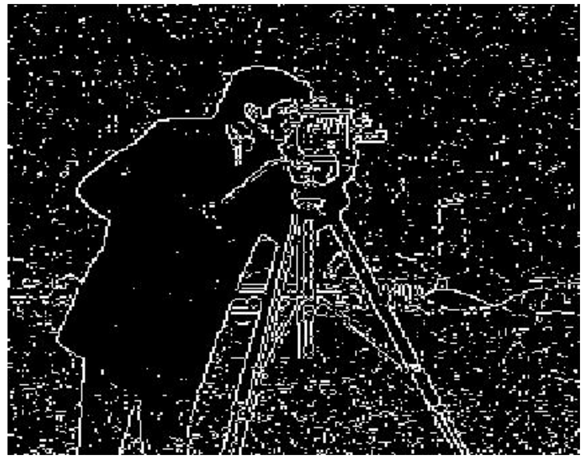
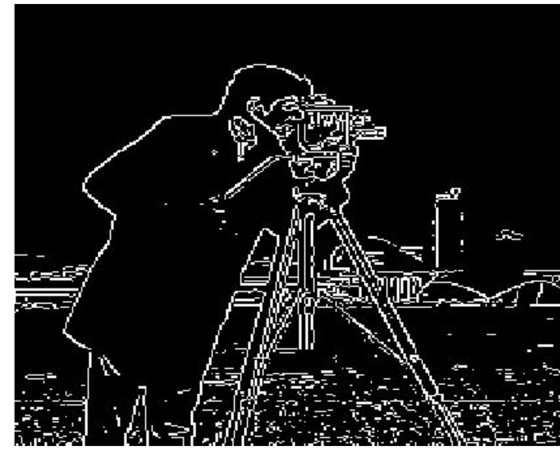
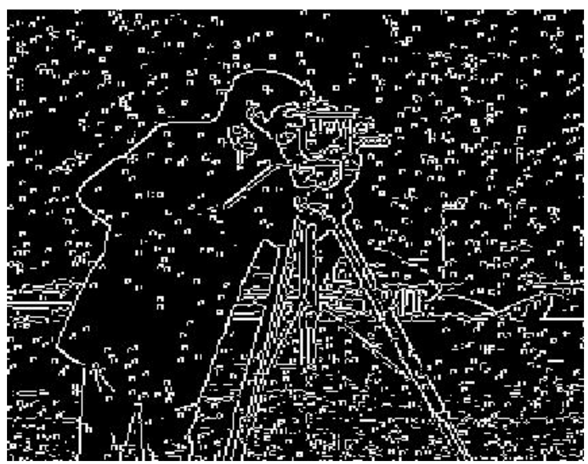
$$\bar{x} = \frac{1}{N} \sum_i x_i$$



Original Algorithm



Proposed Algorithm

	Original Algorithm	Proposed Algorithm
Gaussian Noise		
Salt and Pepper noise		

Conclusions

- Objective function based edge detection method works better than traditional edge detection methods.
- Not robust to noise
- Performance in noise can be improved by simple changes to objective function
- Versatile

Another application ?

- Edges in color images can be classified as object edges, shadow edges, reflectance edges, occlusion edges etc.
- There is a lot of recent work concerning classification of edges
- Our aim – use different objective functions to distinguish and classify different edges….

References

[1] ‘A novel edge detection method based on maximising objective functions’ , Kang, Wang, Journal of Pattern Recognition, 2006

[2] ‘A new edge detection method in Image Processing’, Zhang, Zhao, Su, Proc of ISCIT, 2005

[3] ‘ Color edge detection in presence of Gaussian noise using non linear pre filtering’, Russo, Lazzari, IEEE Transactions on Inst. and Measurements, 2005.

Thank You

Questions?