

**M. Tech. (Computer Science) Dissertation Series**

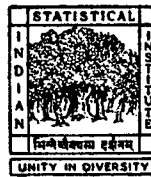
## **Corner Detection**

**a dissertation submitted in partial fulfilment of the  
requirements for the M. Tech. (Computer Science)  
degree of the Indian Statistical Institute**

**M.Suri Apparao**<sup>By</sup>  
**Roll No: CS0701**

**under the supervision of**

**Prof. Malay K Kundu**  
**Machine Intelligence Unit**  
**ISI, Kolkata**



**INDIAN STATISTICAL INSTITUTE**  
203, Barrackpore Trunk Road  
Calcutta-700 035

M. Tech. (Computer Science) Dissertation Report

## **Corner Detection**

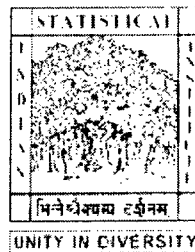
A dissertation submitted in partial fulfillment of the requirements for  
M. Tech.(Computer Science) degree of the Indian Statistical Institute

By

**M.Suri Apparao**  
**Roll No: CS0701**

Under the supervision of

**Prof. Malay K Kundu**  
**Machine Intelligence Unit**  
**ISI,Kolkata**

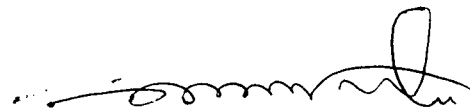


**Indian Statistical Institute**  
**203, B. T. Road**  
**Kolkata-700108**

# Indian Statistical Institute

## Certificate of Approval

This is to certify that the thesis entitled "Corner detection" by M.Suri Apparao towards partial fulfillment for the degree of M.Tech. in Computer Science at Indian Statistical Institute, Kolkata, embodies the work done under my supervision.



(Prof. Malay K Kundu)

MIU- Indian Statistical Institute  
Kolkata

Date:

Jayjay Kumar Kaha  
01/01/10

## ACKNOWLEDGEMENT

I take this opportunity to thank **Prof. Malay K Kundu** Machine Intelligence Unit, ISI-Kolkata for his valuable guidance, inspiration. Their pleasant and encouraging words have always kept my spirits up.

I would like to thank all of my colleagues, class mates, friends, and my family members for their support and motivation to complete this project.

Handwritten signature of M. Suri Apparao in black ink, including the date 11/11/10.

M.Suri Apparao

M.Tech (CS)

Date:

Indian Statistical Institute  
Kolkata

## **Contents**

## **Page No**

**1. Introduction**

**2. Motivation**

**3. Harris Corner Detector**

**4. Susan Corner Detector**

**5. Proposed Method**

**6. Feature Extraction**

**7. Multilayer Perceptron**

**8. Results**

**9. References**

# 1. Introduction

A corner is defined as the junction point of two or more straight line edges. Corners are special features in a image. They are of great use in computing the optical flow and structure from motion. The earliest corner detection methods involved first segmenting the image into regions and representing the object boundary as a chain code. Corners were identified where the direction changed rapidly. Later attempts were directed at coming up with a corner detector which operated directly on gray level images. These include the one developed by Zuniga and Haralick [SI, Kitchen and Rosenfeld and the one by Dreschler and Nagel. In these approaches corners are considered as the points where the rate of change of gradient direction is maximum.

Harris-Stephens is shown as the most successful detector. Additionally, another commonly cited corner detector, SUSAN, is compared with other detectors. Image gradient based corner detectors are very popular despite the fact that they are not robust to changes in corner orientation, corner angle and/or contrast. Moravec developed an interest point detector based on the auto correlation function. He measured the differences between the feature window and its shifted versions. The main idea is that for a defined neighborhood around a corner point, movement in any direction should yield a considerable intensity change. To determine the corner points, Forstner, in 1986, utilized the so called "auto-correlation matrix", which is in essence the outer product of image derivatives along x and y axes. Different functions of the trace and determinant of the same matrix are used by Harris, Stephens and Noble in 1988 and 1989, respectively.

In this work, we describe a Neural Network based corner detection technique that will use techniques w.r.to changes in corner orientation, angle and contrast.

## 2. Motivation

Susan corner detector is one of popular corner detector. But in this algorithm the need to adjust three different parameters (spatial smoothing, brightness threshold, number of iterations). That is these three parameters are user defined. It performs well even in the presence of noise. No image derivatives are used. Integrating effect and non-linear response give strong noise rejection. Susan corner detector gives better results if the values of the parameters are good. But there are no guidelines for selecting the good values for the parameters. But Harris algorithm is simple compare to Susan w.r.t user defined variables. In Harris algorithm two identical corners may not be selected for a given threshold if their orientations are different. Similar problem is observed for objects with identical orientations but different corner angles. The intensity difference (contrast) of the corner also affects the corner selection. A corner with small contrast is eliminated although it has the same corner angle and orientation with another one. We developed a neural network it take input is output of Harris detector and it gives results approaching to Susan corner detector results.

## 3. Harris corner detector

Among the gradient based corner detectors, many use the matrix constructed by the outer

$$C = \sum_{x,y \in R} w_{(x,y)} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

To obtain this matrix, firstly image gradient sums are evaluated over a neighborhood,  $R$ , of the point to be examined. Gaussian or similar weighting of gradients with  $w(x,y)$  within this feature

window improves the performance of the detector and is commonly employed. The cornerness measure of Harris-Stephens [3], another popular corner detector, is written in terms of the determinant and trace of the auto-correlation matrix  $C$ :

$$\begin{aligned}
 MHS &= \text{Det}(C) - k * \text{Trace}^2(C) \\
 &= \sum I_x^2 \cdot \sum I_y^2 - \left( \sum I_x I_y \right)^2 - k \left( \sum I_x^2 + \sum I_y^2 \right)^2
 \end{aligned}$$

where the constant  $k$ , a tunable parameter, is usually taken as 0.04 as suggested by Harris-Stephens.

### **Problems with Harris corner detector**

It turns out that for image gradient based detectors two identical corners may not be selected for a given threshold if their orientations are different. Similar problem is observed for objects with identical orientations but different corner angles. The intensity difference (contrast) of the corner also affects the corner selection. A corner with small contrast is eliminated although it has the same corner angle and orientation with another one. Such changes are likely to occur under affine transformations and/or varying illumination conditions in different feature matching scenarios. In our previous work, eigenvalue compensation for orientation and corner angle is applied to catch these missing corners. It was based on modifying the thresholds using empirical transformations.

As the corner gets wider or narrower, eigenvalues immediately decrease so these corners would be missed by the same threshold. So there is an effect of orientation angle for a corner angle of  $90^\circ$ . Eigenvalues are maximum at orientations of 0, 90, 180 and 270 degrees. As corner angle changes however, this pattern changes and mid-values ( $45^\circ$ ,  $135^\circ$ , etc.) also gain strength. If corner is narrower than  $20^\circ$  or wider than  $160^\circ$ , which resembles an edge or



a line, values in between ( $22.5^\circ$ ,  $67.5^\circ$ , etc.) give higher eigenvalues.

#### 4. SUSAN Detector

Proposed by Smith and Brady in 1995. It doesn't use any derivatives. SUSAN stands for Smallest "Univalue Segment Assimilating Nucleus"

It is based on the fact that each point within an image has associated with it a local area of comparable brightness

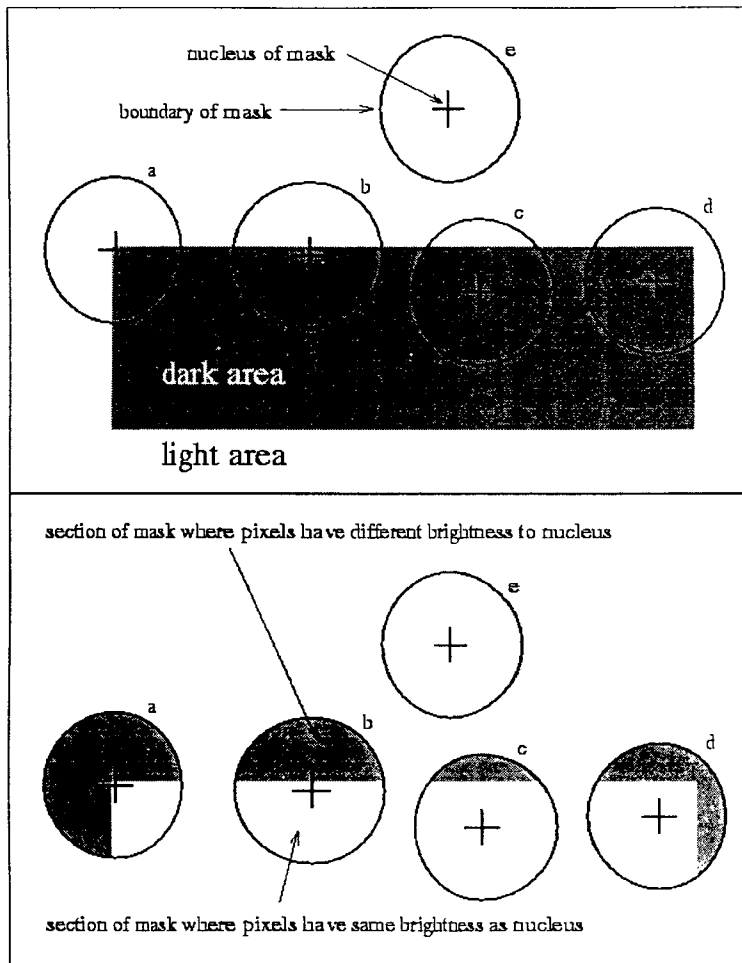
The Principle

=> It generates a circular mask around a given point in an image

=> It compares the intensity of neighboring pixels with that of the centre pixel (**nucleus** of the mask), the area with similar intensity to the nucleus is called **USAN area**

=> Repeat the procedure for each pixel within the image

- USAN area varies within the image depending on its location with respect to **special features** of the image
- The USAN area is maximum within the rectangular area but falls to a **minimum at an edge** and to an even smaller value corresponding to a **local minimum at a corner**
- This is the property upon which the corner finder algorithm is based, hence the name SUSAN



## The algorithm

1. Determine a circular mask, typically of 37 pixels around a nucleus for each point within the image
2. Calculate the difference in brightness between each pixel of the mask and that of its nucleus and

$$c(\vec{r}, \vec{r}_0) = \begin{cases} 1 & \text{if } |I(\vec{r}) - I(\vec{r}_0)| \leq t \\ 0 & \text{otherwise,} \end{cases}$$

3. Sum the number of pixels within the circular mask which have similar intensity levels to that of the nucleus

$$n(\vec{r}_0) = \sum_{\vec{r}} c(\vec{r}, \vec{r}_0)$$

4. Compare  $n$  with  $g$ , the *geometric threshold* which is set to half of the maximum value that  $n$  can be ( $n_{\max}/2$ )
5. At a perfect corner (where two straight edges intersect) the USAN area will always be less than half the size of the mask area, and will be a local minimum

$$R(\vec{r}_0) = \begin{cases} g - n(\vec{r}_0) & \text{if } n(\vec{r}_0) < g \\ 0 & \text{otherwise,} \end{cases}$$

## 5. Proposed Method

**Step1:** Finding corners of a given image using Harris corner detector.

**Step2:** Finding corners of a image using Susan corner detector.

**Step3:** Taking intersection of above two results as a valid corners

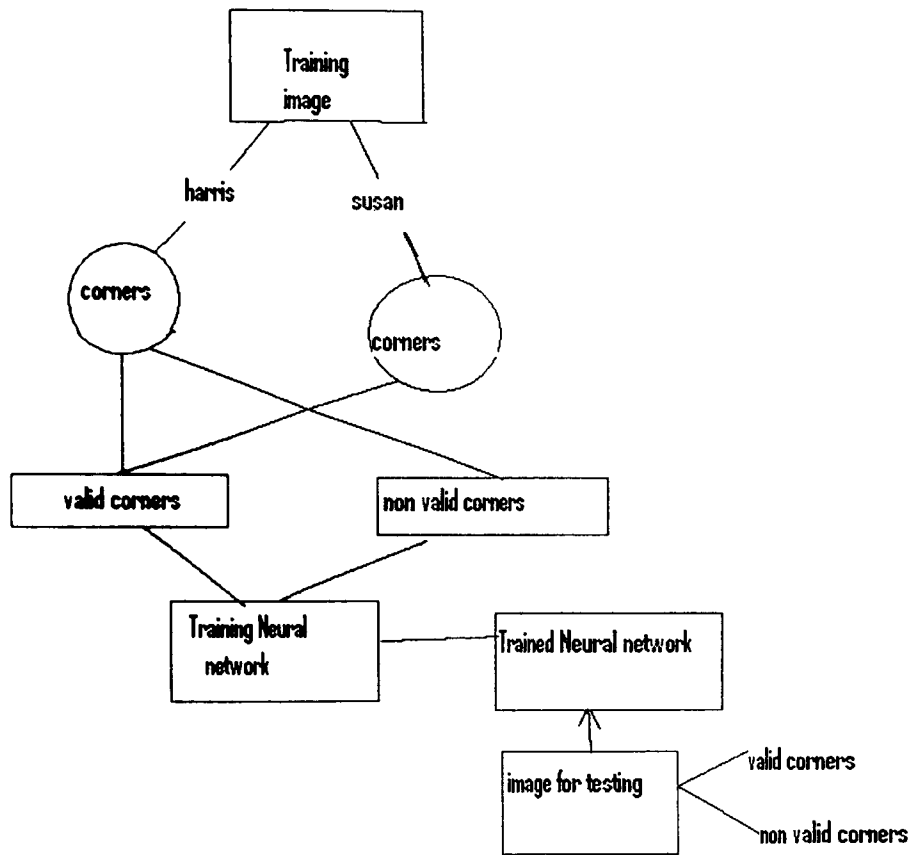
**Step4:** Taking the corners of Harris Corner other than valid corners as non-valid corners.

**Step4:** Feature extraction over these valid and non valid corners.

This Step is explained in the next section.

**Step5:** Train Multi Layer Perceptron with features set for both valid and non-valid corners.

**Step6:** Using this Neural network classifying corners of image getting from Harris corner detector.



FLOW CHART FOR PROPOSED METHOD

## **6. Feature Extraction:**

### **Features considered:**

- i) Corner angle
- ii) Corner orientation
- iii) Corner contrast
- iv) Support of a Corner
- v) edge pixel or not

### **Steps for finding the Features:**

**Step1:** Find edge image using canny edge detector

**Step2:** Finding edges of a corners

**Step3:** Finding corner angle

**Step4:** Finding corner orientation

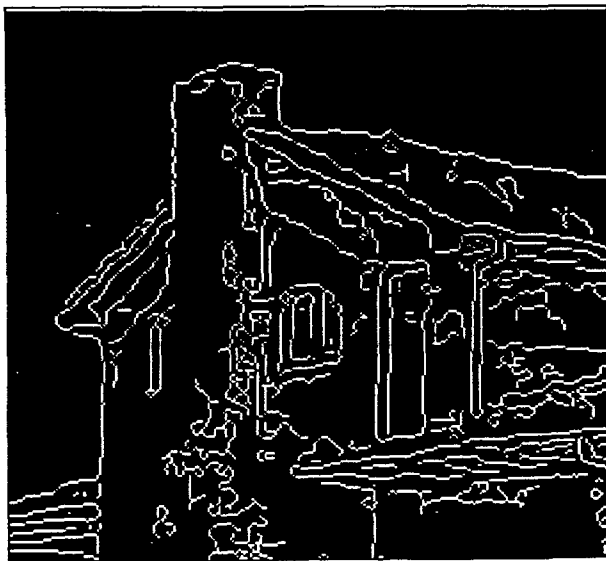
**Step5:** Finding support of a corner

**Step6:** Finding corner contrast

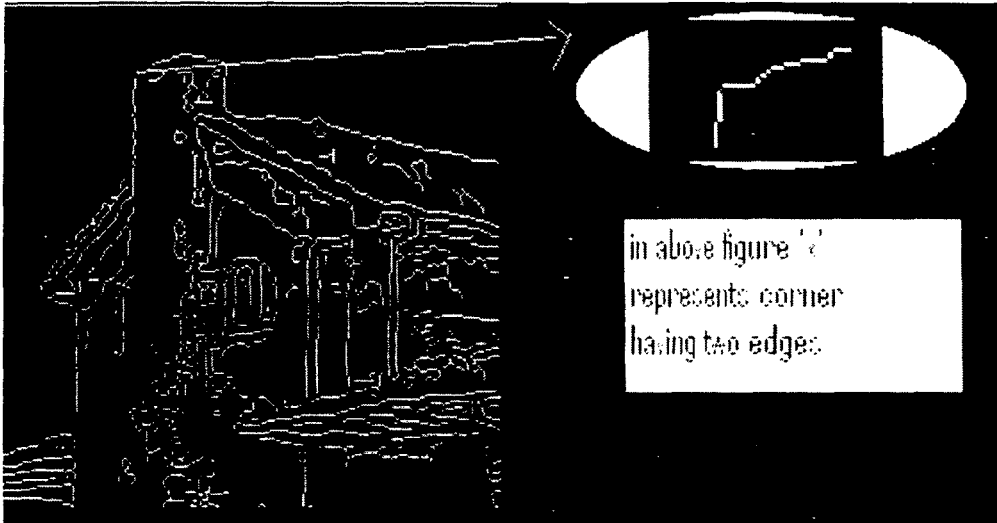
**Input image:**



**Step1: Finding edges using Canny edge detector**



## Step2: Finding Edges of a Corner



### Method:

Starting from the corner, scanning in two angles  $\theta_1$ ,  $\theta_2$  in the edge image (with in a window depend on image size), where two lines present whose lengths are at least equal to some threshold (say 5)

For Scanning in a given angle, we are having the function '**find\_line (corner, angle)**'.

This function gives the pixels in the given angle.

Angle of line with the positive X-axis in anti-clock wise direction which is bisecting the Corner angle between the two edge lines of the corner.

### Step 3: Finding Corner Angle

**Corner Angle:** Acute Angle between two edge lines of the corner.

Let us say  $\text{Edge1\_angle} < \text{Edge2\_angle}$ . (WLOG)

then

$\text{diff\_angle} = \text{Edge2\_angle} - \text{Edge1\_angle}$

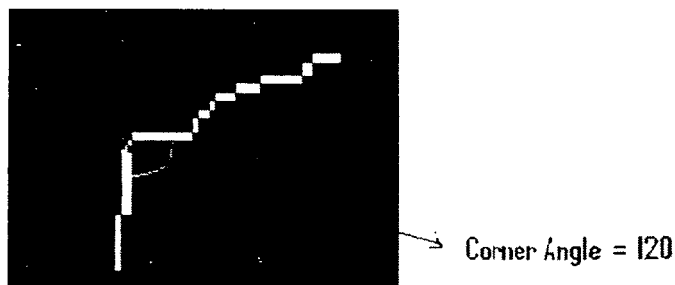
if ( $\text{diff\_angle} > 180$ )

$\text{Corner\_Angle} = 360 - \text{Edge2\_angle} + \text{Edge1\_angle}$

Else

$\text{Corner\_Angle} = \text{diff\_angle};$

**Example:**



For this corner and corresponding edges,

$\text{Edge1\_Angle} = 30;$

$\text{Edge2\_Angle} = 270;$

$\text{diff\_angle} = 270 - 30 = 240;$



here  $\text{diff\_angle} > 270$ , So  $\text{Corner\_Angle} = 360 - 270 + 30 = 120$ .

#### Step 4: Finding Corner Orientation

**Corner Orientation:** Angle of the line with the Positive x-axis in the anti-clock wise direction which is bisecting the corner angle of a corner.

For finding the Corner Orientation, we find the rotation angle from one edge to another edge in the anti-clock wise direction.

#### Pseudo Code:

Again assume that  $\text{Edge1\_Angle} < \text{Edge2\_Angle}$

After  $\text{Corner\_Angle}$  finding,

Finding the difference1 =  $\text{Edge2\_Angle} - \text{Edge1\_Angle}$

And difference2 =  $360 - \text{Edge2\_Angle} - \text{Edge1\_Angle}$

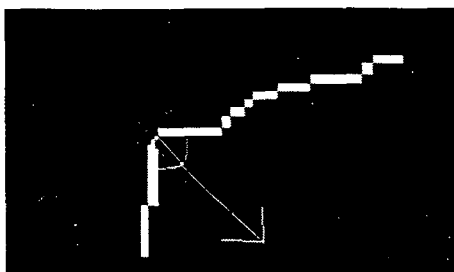
If ( $\text{difference1} < \text{difference2}$ )

$\text{Corner\_Orientation} = \text{Edge1\_Angle} + (\text{difference1} / 2)$

Else

$\text{Corner\_Orientation} = \text{Edge2\_Angle} + (\text{difference2} / 2)$

#### Example:



Line bisecting the angle is in the Corner Orientation.  
Above we have  $\text{Corner\_Angle} = 120$ ;

$$\text{difference1} = 270 - 30 = 240$$

$$\text{difference2} = 360 - 270 + 30 = 120$$

(Observe one of the difference is corner angle)

$$\text{Corner\_Orientation} = 270 + (120 / 2) = 330$$

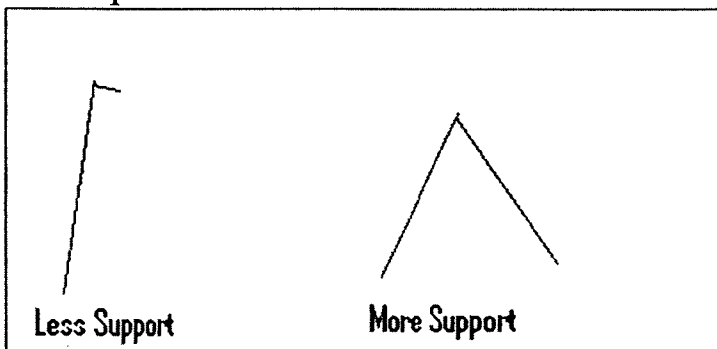
### **Step5: Finding support of a corner**

Let us say  $\text{Edge1\_Length} < \text{Edge2\_Length}$

$$\text{Support of a corner} = \text{Edge1\_Length} / \text{Edge2\_Length}$$

If any one of the edge have the length zero then support of a corner is also zero.

Example:

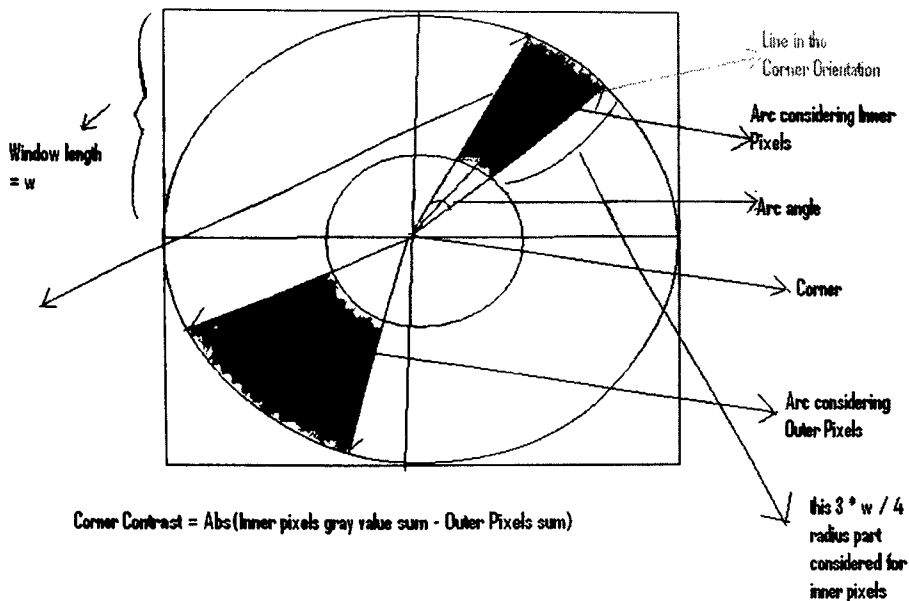


If Support is less than  $\text{support\_Threshold}$  (say  $0.8$ ), then it is not a corner. (considered length of a edge to be  $\geq 5$ ).

## Step6: Finding corner contrast

Window Length =  $w$

We define inner pixels and outer pixels of a corner as follows.



Consider the line in the orientation of the corner.

If corner angle  $< 60$  then Arc\_angle = 10 degrees

Otherwise Arc\_angle = 30 degrees

Consider the arc part with the above Arc\_angle in the Corner Orientation, whose pixels are at a distance  $\geq (w / 4)$  are called inner pixels.

And in the same manner consider the arc with the

Corner\_orientation + 180 for outer pixels but with angle of the Arc = 120 degrees for getting Outer Pixels.

After calculating the sum of inner pixels and sum of outer pixels, take the absolute difference between them as the Corner Contrast.

## 7.MLP

A multilayer perceptron is a feed forward artificial neural network model that maps sets of input data onto a set of appropriate output. It is widely used in pattern classification to classify any kind of patterns. It contains one input layer, an output layer and one or more hidden layers. Each layer contains some neurons and each neuron uses a nonlinear activation function. The two main activation functions used in current applications are tan hyperbolic and sigmoid functions, and are described by

$$\phi(v_i) = \tanh(v_i) \quad \text{and} \quad \phi(v_i) = (1 + e^{-v_i})^{-1}$$

where  $v$  is the weighted sum from the previous layer.

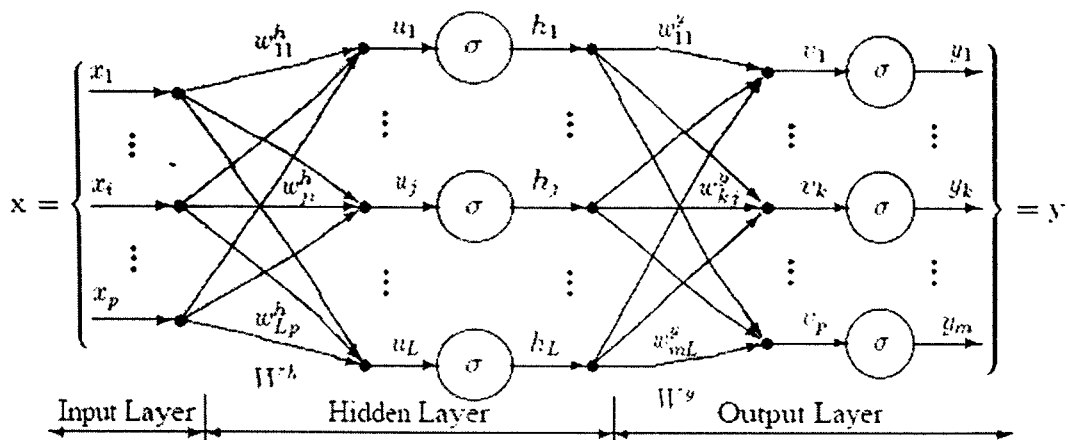
Nodes in the input layer transmit the input to all the nodes in the next layer (first hidden layer). Nodes in the other layers perform weighted sum of the inputs to that node and compute the activation function of that weighted sum and it also passes this function value to each node in the next layer. In the output layer these function values are the outputs of the network.

Let  $v_j^{h+1}$  be the total input received by neuron  $j$  in layer  $h+1$  then

$$v_j^{h+1} = \sum_i^h y_i^h w_{ji}^h - b_j^{h+1}$$

Where  $y_i^h$  is the output of the  $i$ th neuron in the preceding layer  $h$ ,  $w_{ji}^h$  is the weight of the connection from the  $i^{\text{th}}$  neuron in layer  $h$  to the  $j^{\text{th}}$  neuron in layer  $h+1$ , and  $b_j^{h+1}$  is the threshold of the  $j^{\text{th}}$  neuron in layer  $h+1$ .

Using the Back Propagation Algorithm the network will adjust its weights.



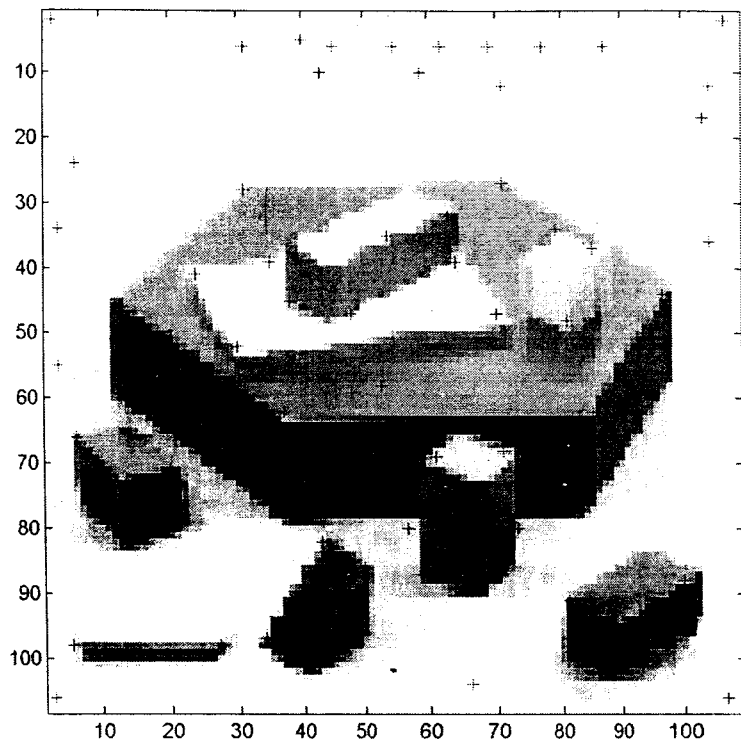
## 8.Results:

---

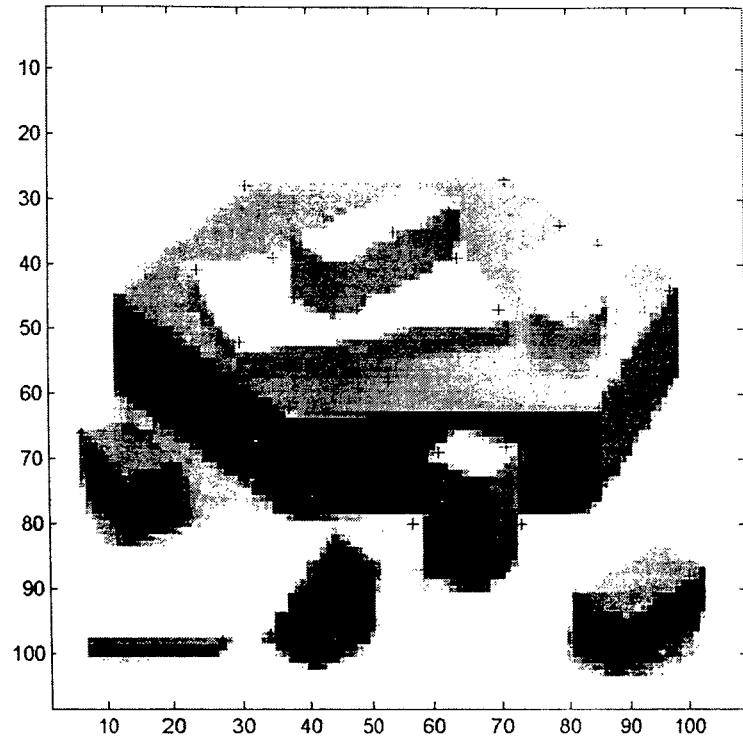


SUSAN CORNERS

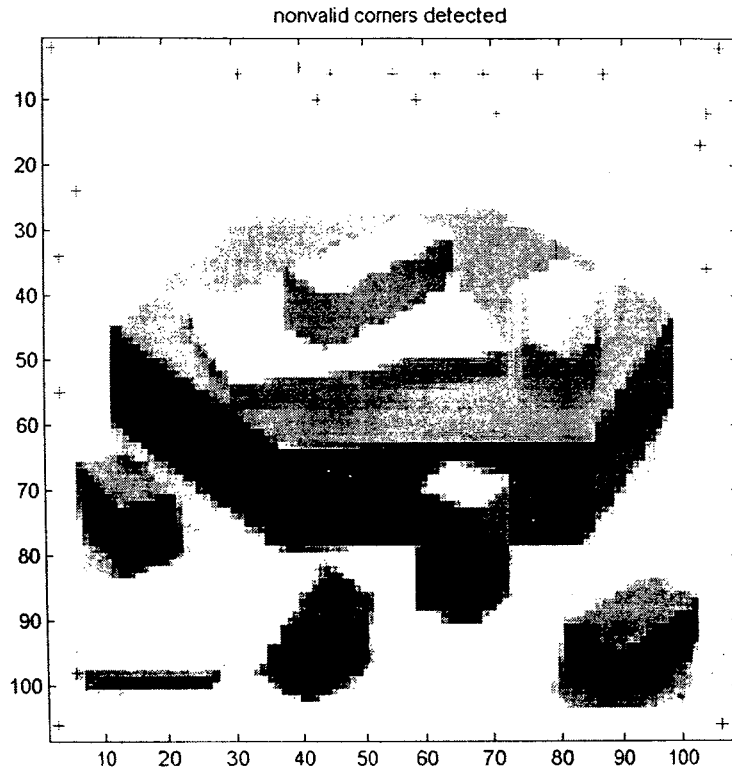
harris corners



valid corners detected







No of training patterns = 180 ;  
No of valid corners in training = 78 ;  
No of non valid corners in training = 102 ;  
No of testing patterns = 68 ;  
No of valid corners in testing data = 44;  
No of non valid corners in testing data = 22;

## Features for valid corners

<b>Edgepixel</b>	<b>support</b>	<b>angle</b>	<b>orientation</b>	<b>contrast</b>
1.000000	0.909091	172.500000	15.000000	24579.000000
0.000000	1.000000	60.000000	120.000000	1343.000000
1.000000	1.000000	287.500000	145.000000	21962.000000
1.000000	0.714286	345.000000	130.000000	10329.000000
1.000000	1.000000	240.000000	120.000000	7630.000000
0.000000	0.454545	262.500000	165.000000	17406.000000
1.000000	0.714286	382.500000	145.000000	10566.000000
0.000000	0.555556	80.000000	150.000000	15439.000000

## Features for non valid corners

<b>0.000000</b>	<b>0.600000</b>	<b>362.500000</b>	<b>15.000000</b>	<b>1410.000000</b>
<b>0.000000</b>	<b>0.800000</b>	<b>80.000000</b>	<b>160.000000</b>	<b>22740.000000</b>
<b>1.000000</b>	<b>1.000000</b>	<b>90.000000</b>	<b>160.000000</b>	<b>16349.000000</b>
intervals	harrisval	p.algoval	p.algononvalid	
0-15	61	6	55	
15-30	14	14	0	
30-45	10	10	0	
45-60	15	14	1	
60-75	11	10	1	
75-90	14	14	0	
90-105	7	6	1	
105-120	9	7	2	
120-135	5	5	0	
135-150	14	13	1	
150-165	6	6	0	
165-180	8	8	0	

## 9. Refereces

- [1] Yalin Bastanlar, Yasemin Yardimci ,”Corner Validation based on extracted corner properties,” *Computer Vision and Image Understanding* 112 (2008)243-261
- [2] C. Harris and M.J. Stephens.” A combined corner and edge detector”, In *Alvey Vision Conference*, pages 147–152, 1988.
- [3] S.M.Smith, M.Brady, SUSAN-a new approach to low level image processing, *International Journal of computer vision* 23 (1) (1997) 45-78.
- [4]Minakshi Banerjee ,Malay K.Kundu ,” Handling of impreciseness in gray level corner detection using fuzzy set theoretic approach”, *Applied Soft Computing*, Vol. 8, No. 4, pp. 1680-1691, 2008