✓**Halftoning/Dithering**:

- Reduces the number of gray levels by creating dot patterns or dither patterns to represent various gray levels
- Based on the idea of diffusing the quantization error across edges, where changes occur in the image
- Reduces effective spatial resolution
- Typically used for printing purposes

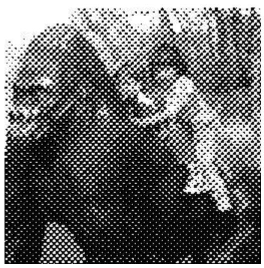
1

Halftoning and Dithering

Original image

Floyd-Steinberg error diffusion, 1 bpp

Bayer's ordered dither, 1 bpp

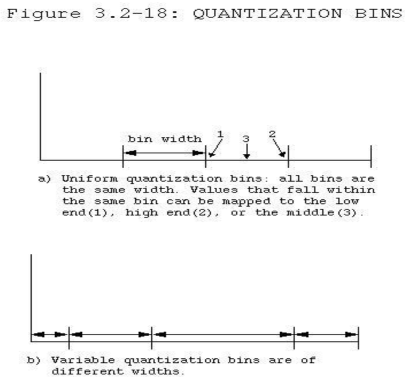


45-degree clustered-dot dither, 1 bpp

2

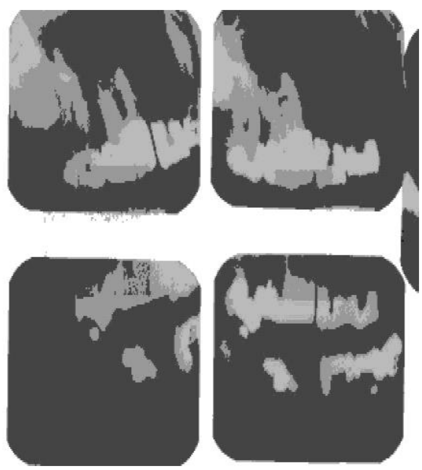
✓*Uniform bin width quantization*: The size of the bins for the quantization is equal

✓*Variable bin width quantization*: Unequal bin size, based on an application specific basis



Variable bin-width quantization

Original image



After variable bin-width quantization

✓**Spatial Quantization**:

- Reducing the image size by taking groups of spatially adjacent pixels and mapping them to one pixel
- May produce simple forms of geometric distortion
- Can be performed in following ways:
  - Averaging
  - Median
  - Decimation

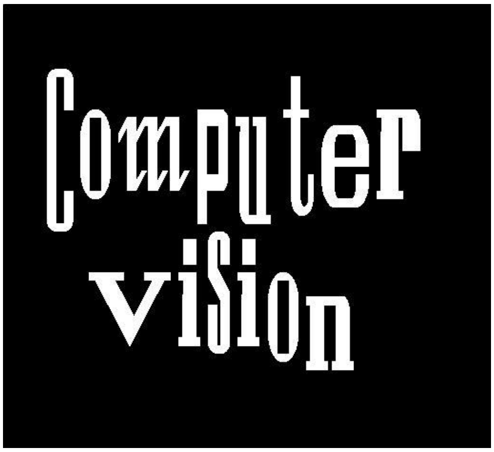
5

Spatial Quantization Methods

- *Averaging*: Groups of pixels are averaged and the group is replaced by the average
- *Median*: Pixel gray values are sorted in small neighborhood and the neighborhood is replaced by the middle value
- *Decimation*: Known as sub-sampling, reduces the image size by eliminating rows and columns

6

Spatial Reduction



Original 512*512 image

Averaging (64*128)

Median (64*128)

Decimation (64*128)

7

- *Anti-aliasing filtering*: Process of improving the image quality when applying the decimation technique, by preprocessing the image with averaging (mean) spatial filter

Result of spatial reduction of 512x512 to 128x128 via decimation

Result of spatial reduction of 512x512 to 128x128 via decimation, preprocessed by a 5x5 averaging filter

8

PART II

➢**Binary Image Analysis**

✓Binary images are useful in many computer vision applications which require simple object shape; such as positioning a robot to grasp an object, to check a manufactured object for defects, FAX, OCR

✓Most cameras provide us color or gray level images, thus we need to convert those images into binary images

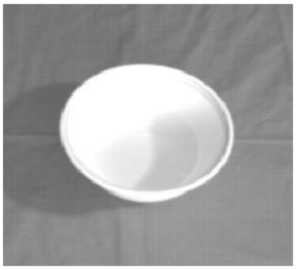
✓Next, we extract simple binary features and use them to classify binary objects

✓Thresholding via Histogram

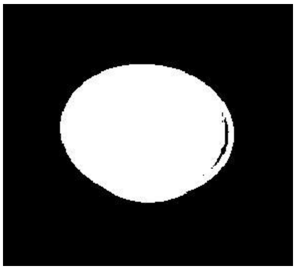
- Thresholding is required to create a binary image from a gray level image
- This is done by specifying a *threshold value* which will set all values above the specified gray level to ‘1’ and everything below the specified value to ‘0’
- Typically 255 is used for ‘1’ and 0 is used for the ‘0’ value

- In many applications the threshold value is determined experimentally and is highly dependent on lighting conditions and object to background contrast
- It is much easier to find a good threshold value with proper lighting, and good background to object contrast

Figure 3.3-1: Effects of Lighting and Object to Background Contrast on Thresholding



a) An image of a bowl with high object to background contrast and good lighting

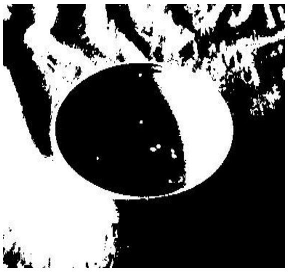


b) Result of thresholding image (a)

13

Figure 3.3-1: Effects of Lighting and Object to Background Contrast on Thresholding (contd)

c) An image of a bowl with poor object to background contrast and poor lighting



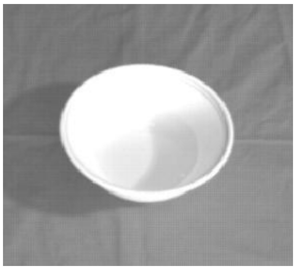
d) Result of thresholding image (c)

14

- The *histogram* is a plot of gray level versus the number of pixels in the image at each gray level
- Histogram of an image is examined to select the proper threshold value
- The peaks and valleys in the histogram are examined and a threshold is experimentally selected that will best separate the object from the background

15

Figure 3.3-2: Histograms



a) An image of a bowl with high object to background contrast and good lighting

b) The histogram of image (a), showing the threshold that separates object and background

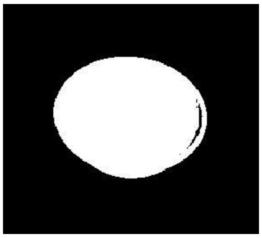


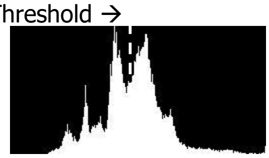
Image after threshold

16

Figure 3.3-2: Histograms (contd)

c) An image of a bowl with poor object to background contrast and poor lighting

Threshold →



d) The histogram of image (c), showing what appears to be a good threshold, but it does not successfully separate object and background

Image after threshold

17

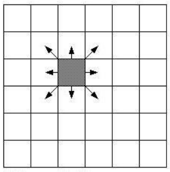
✓Connectivity and Labeling

- To handle images with more than one objects, we need to consider exactly how pixels are connected to make an object, and then find a method to label the objects separately
- However, the process of spatial digitization (sampling) can cause problems regarding connectivity of objects

18

- *Connectivity* refers to the way in which we define an object; after the threshold operation which pixels should be connected to form an object ?

- A pixel has eight possible neighbors:
  - Two horizontal neighbors,
  - Two vertical neighbors, and
  - Four diagonal neighbors

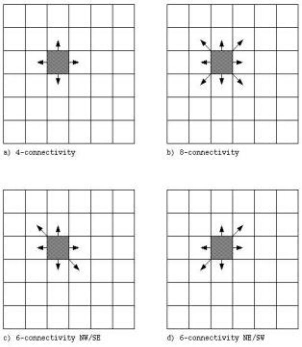


19

- Connectivity can be defined in three different ways:

  1. *Four-connectivity*
  2. *Eight-connectivity*
  3. *Six-connectivity*

Figure 3.3-3: CONNECTIVITY



- **Choosing the type of connectivity is application dependent, the key is to be consistent**

20

• *Connectivity dilemma*:

- This dilemma arises when we use four or eight connectivity for both object and background where closed curves do not separate the background (eight-connectivity), or we do not have a closed curve and the background is separated (four-connectivity)

21

- Consider the following binary image segment:

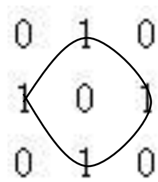
0	1	0
1	0	1
0	1	0

- Assuming four-connectivity there are 4 separate objects and 5 separate background objects. The dilemma is that if the objects are separated, shouldn't the background be connected?

22

- Assuming eight-connectivity we have one connected object, a closed curve, but the background is also connected

- This creates another dilemma because a closed curve should separate the background into distinct objects



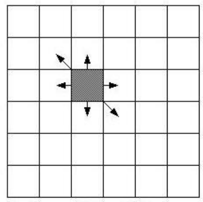
23

- These are methods to resolve the connectivity dilemma:

1. Use eight-connectivity for background and four-connectivity for the objects
2. Use four-connectivity for background and eight-connectivity for the objects
3. Use six-connectivity

24

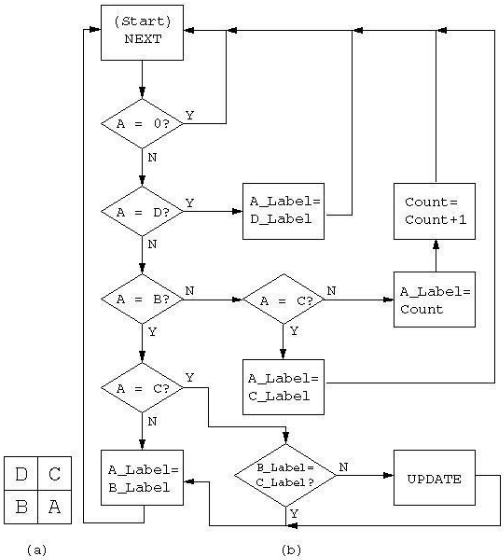
- The first two choices are acceptable for binary images, but get quite complicated when extended to gray level and color images
- Six-connectivity is a good compromise in most situations, as long as we are aware of the bias created by selection of one diagonal direction
- **We use the definition of six-connectivity with the NW and SE diagonal neighbors**

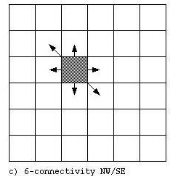


c) 6-connectivity NW/SE

- After selecting the connectivity, a *labeling algorithm* is required to differentiate between multiple objects within an image
- The labeling process requires us to scan the image and label connected objects with the same symbol
- Details of the labeling algorithm are connectivity dependent

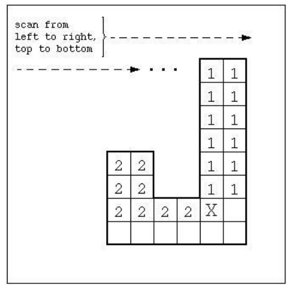
Figure 3.3-4: LABELING ALGORITHM





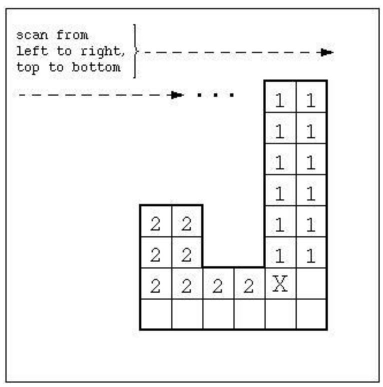
Assumed connectivity

Figure 3.3-5: MULTIPLE LABELS



The labeling algorithm requires an UPDATE function to keep track of objects with more than one label. Multiple labeling can occur during sequential scanning, as shown above on the 'J' shaped object. We label two different objects until we reach the pixel marked 'X', where we discover that objects 1 and 2 are connected.

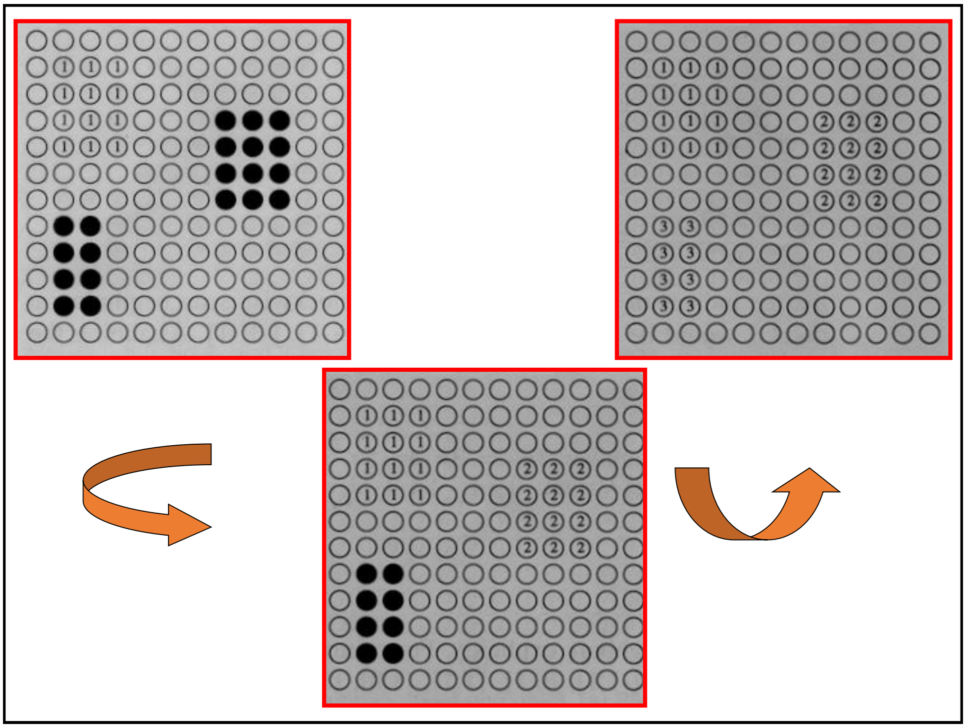
Figure 3.3-5: MULTIPLE LABELS

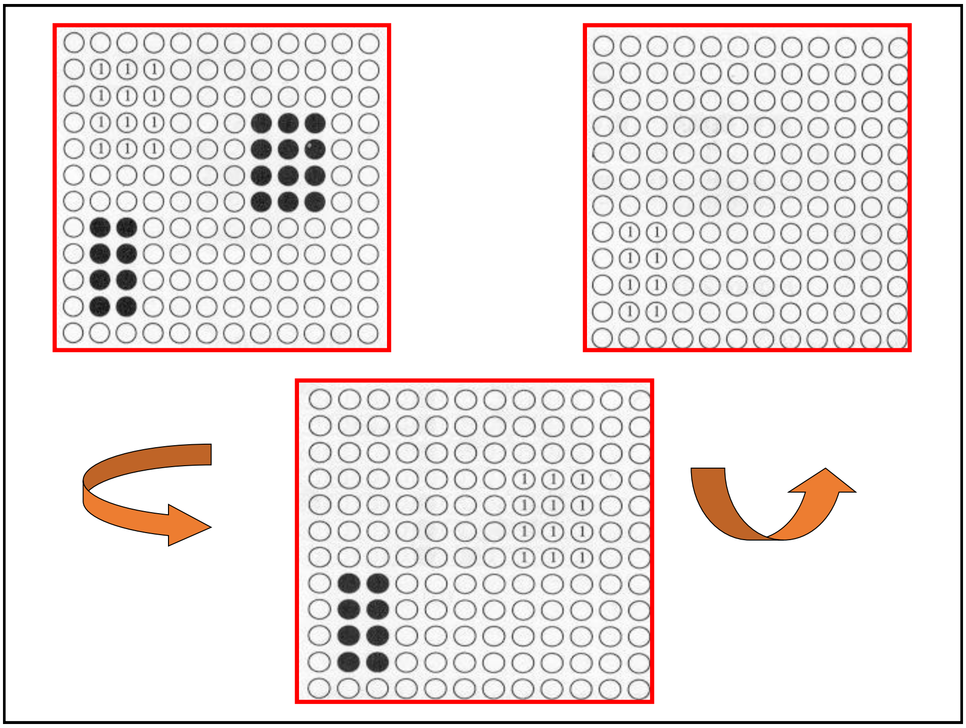


The labeling algorithm requires an UPDATE function to keep track of objects with more than one label. Multiple labeling can occur during sequential scanning, as shown above on the 'J' shaped object. We label two different objects until we reach the pixel marked 'X', where we discover that objects 1 and 2 are connected.

- By labeling the objects, an image filled with object numbers is created
- With this labeled image we can extract features specific to each object
- These features are then used to locate and classify the binary objects

29





✓**Basic Binary Object Features**

- Basic binary features extracted from labeled objects for classification

- The binary object features defined here include: ***area***, ***center of area***, ***axis of least second moment***, ***projections*** and ***Euler number***

- The first three contain information about where the object is, and the latter two something about the shape of the object

32

- ***Function $I_i(r,c)$*** is defined as follows:

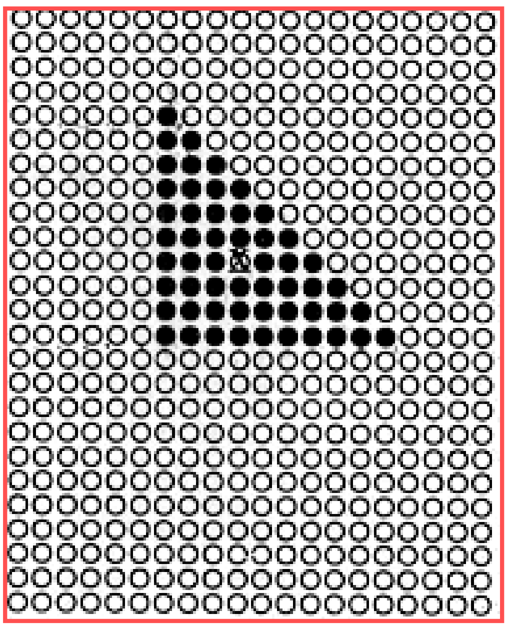
$$I_i(r,c)=\begin{cases} 1 \text{ if } I(r,c)= i^{th} \text{ object number} \\ 0 \text{ otherwise} \end{cases}$$

- *The* ***area*** **(*zero order moment*)** *of the i* th object is defined as:

$$A_i=\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I_i(r,c)$$

- The area A_i is measured in pixels, and <u>indicates the relative size of the object</u>

33



34

- The ***center of area*** (centroid) finds the midpoint along each row and column axis corresponding to the "middle" based on the spatial distribution of pixels within the object

It can be defined by the pair $(\bar{r}_i, \bar{c}_i)$:

$$\bar{r}_i = \frac{1}{A_i} \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} r\, I_i(r,c); \qquad \bar{c}_i = \frac{1}{A_i} \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} c\, I_i(r,c)$$

- It helps to locate an object in the two-dimensional image plane

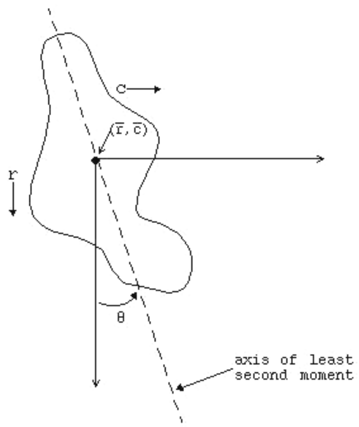
35

- The ***axis of least second moment***, provides information about the object's orientation
- This axis corresponds to the line about which it takes the least amount of energy to spin an object of like shape, or the axis of least inertia
- It is defined as:

$$\tan(2\theta_i) = 2\frac{\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} rc\, I_i(r,c)}{\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} r^2 I_i(r,c) - \sum_{r=0}^{N-1}\sum_{c=0}^{N-1} c^2 I_i(r,c)}$$

36

Figure 3.3-6: AXIS OF LEAST SECOND MOMENT



- The ***projections*** of a binary object are found by summing all the pixels along rows or columns

- If we sum the rows we have the *horizontal projection*, if we sum the columns we have the *vertical projection*

- The horizontal projection is defined as:

$$h_i(r) = \sum_{c=0}^{N-1} I_i(r,c)$$

- The vertical projection is defined as:

$$v_i(c) = \sum_{r=0}^{N-1} I_i(r,c)$$

- Projections provide shape information and are useful in applications like character recognition, where the objects of interest can be normalized with regard to size

Figure 3.3-7: PROJECTION

c ⟶ ; r ↓

									h(r)
0	0	0	0	1	0	0	0	0	1
0	0	0	0	1	1	0	0	0	2
0	0	1	1	1	1	1	0	0	5
0	1	1	1	1	1	1	0	0	6
0	0	0	0	0	1	1	0	0	2
0	0	0	0	0	0	0	0	0	0

v(c) ⟶ 0 1 2 2 4 4 3 0 0

To find the projections, we sum the number of 1s in the rows and columns.

- With the projection equations, the equations for the *center of area* can be defined as follows:

$$\bar{r}_i = \frac{1}{A_i} \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} r\, I_i(r,c) = \frac{1}{A_i} \sum_{r=0}^{N-1} r h_i(r)$$

$$\bar{c}_i = \frac{1}{A_i} \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} r\, I_i(r,c) = \frac{1}{A_i} \sum_{c=0}^{N-1} c v_i(c)$$

41

- A larger projection value along a given row or column will weight that particular row or column value more heavily in the equation

- This tends to move the center of area coordinate toward that particular row or column – note that all values are normalized by the object area

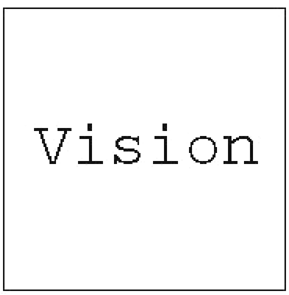
42

- The ***Euler number*** of an image is defined as the number of objects minus the number of holes

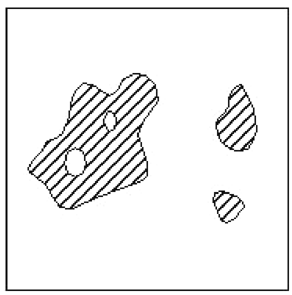
- For a single object, it relates to the number of closed curves the object contains

- It is often useful in tasks such as optical character recognition (OCR)

Figure 3.3-8: EULER NUMBER



a) This image has eight objects and one hole, so its Euler number is 8 – 1 = 7. The letter 'V' has Euler number of 1, 'i' = 2, 's' = 1, 'o' = 0, and 'n' = 1.



b) This image has three objects and two holes, so the Euler number is 3 – 2 = 1.

- The Euler number is also equal to the number of convexities minus the number of concavities, which are found by scanning the image for the following patterns:

$$\begin{array}{cc} CONVEXITIES & CONCAVITIES \\ \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \end{array}$$

45

- Each time one of these patterns is found the count is increased for the corresponding pattern

$$\begin{aligned} Euler\ number &= (Count\ of\ CONVEXITIES) - (Count\ of\ CONCAVITIES) \\ &= (Number\ of\ objects) - (Number\ of\ Holes) \end{aligned}$$

- The number of convexities and concavities can also be useful features for binary objects

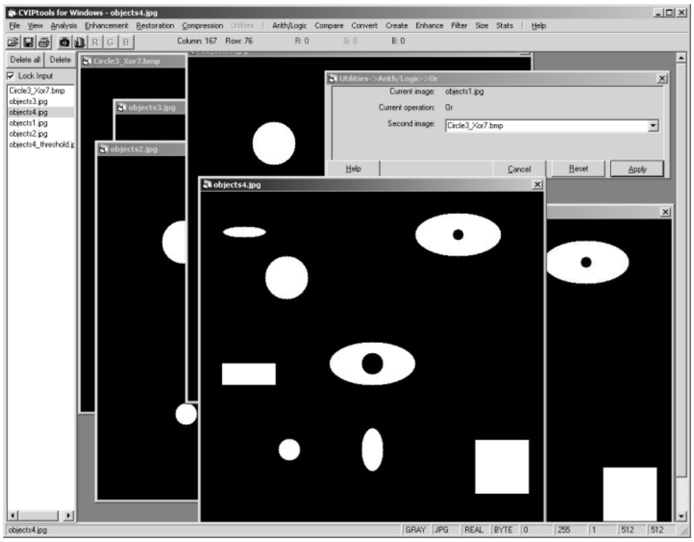
46

✓Binary Object Classification

- <u>The process of identifying binary objects through application of the image analysis process</u>
- CVIPtools is used to extract the objects and analyze the images
- The binary features are used for the classification

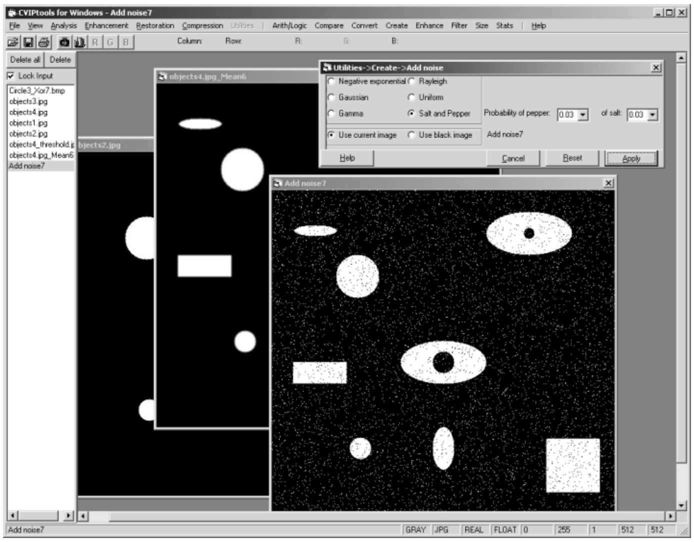
47

Figure 3.3-15: Composite image created by ORing individual object images together



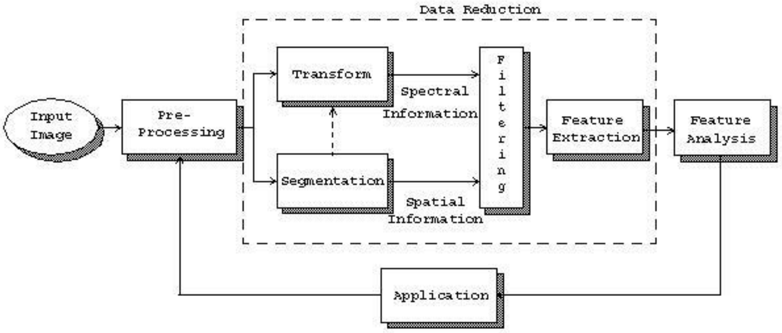
48

Figure 3.3-16: Adding salt and pepper noise to the blurred composite image



Apply the Image Analysis Process

Figure 3.1-3: IMAGE ANALYSIS

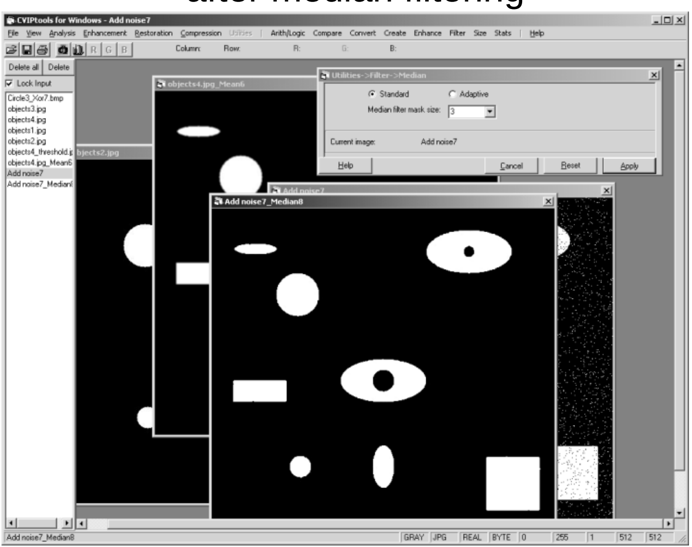


- **Analyze the images as follows:**

1. *Preprocessing*: noise removal with median
2. *Segmentation*: thresholding
3. *Filtering*: none required (?)
4. *Feature extraction*: area, center of area, axis of least 2nd moment, projections, Euler no.
5. *Feature analysis*: manually examine feature file
6. *Application feedback*: Can we identify objects successfully? If not, go to Step 1 and modify the algorithm based on our results

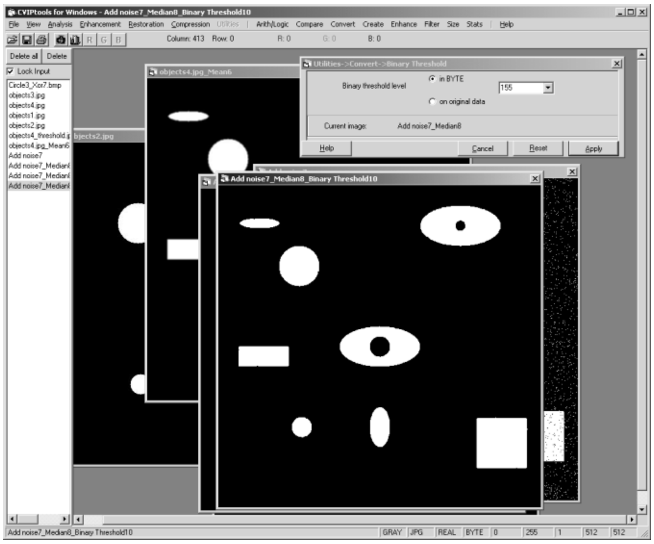
51

Figure 3.3-17: Blurry, noisy composite image after median filtering



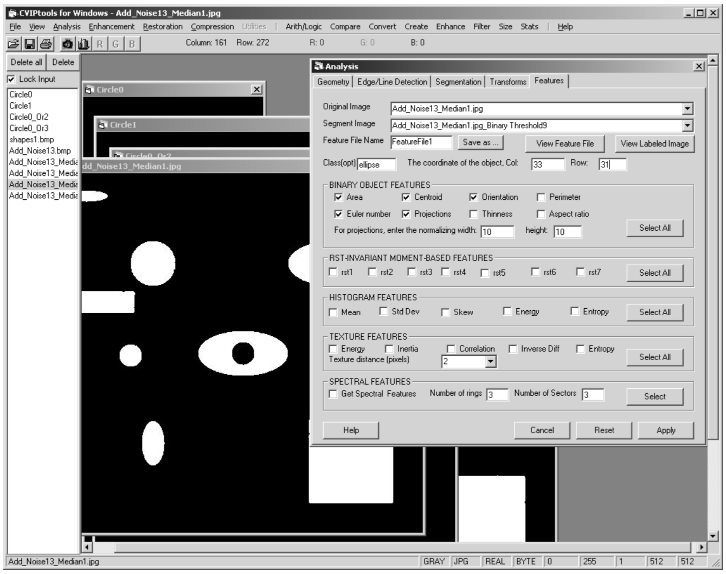
52

Figure 3.3-18: Image after thresholding, note the output shapes still have some minor distortion



53

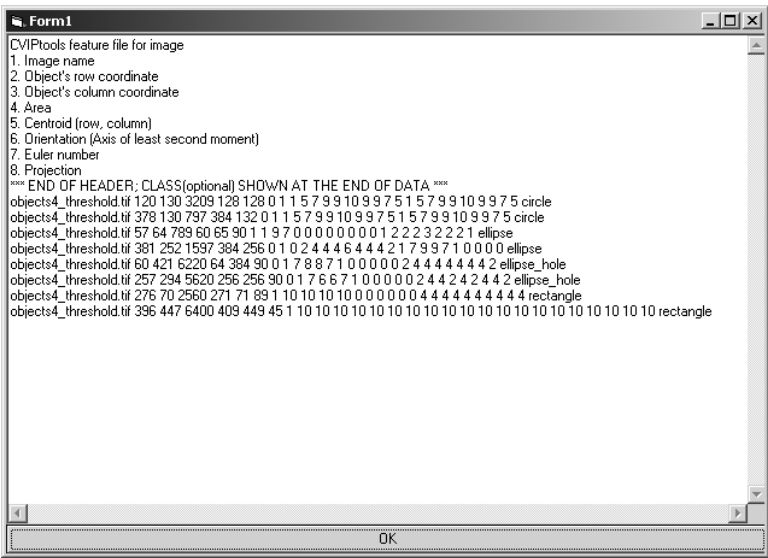
Figure 3.3-19: Feature tab with features selected



54

- Normalization (for projections) is done so that the number of projections does not get too large, and so that that the values will relate to object shape and not object size

Figure 3.3-20 CVIPtools Feature File



- The *feature file*, shown in Figure 3.3-20, consists of two main parts:

1. The *header*, which lists the details for each sample, and
2. The *data* for the extracted feature samples

57

- Every feature file includes the header with the image name, and the object's row and column coordinates, and a list of the selected features

- In the data for the extracted samples, each file is separated by a space, and each sample entry is separated by a new line

58

TABLE 3.1 Feature Data

Object type	*Orientation*	*Euler Number*	*Projections*
Circle	0 degrees	1	1 5 7 9 9 10 9 9 7 5 1 5 7 9 9 10 9 9 7 5
Circle	0 degrees	1	1 5 7 9 9 10 9 9 7 5 1 5 7 9 9 10 9 9 7 5
Ellipse	90 degrees	1	1 9 7 0 0 0 0 0 0 0 0 1 2 2 2 3 2 2 2 1
Ellipse	0 degrees	1	0 2 4 4 4 6 4 4 4 2 1 7 9 9 7 1 0 0 0 0
Ellipse_hole	90 degrees	0	1 7 8 8 7 1 0 0 0 0 0 2 4 4 4 4 4 4 4 2
Ellipse_hole	90 degrees	0	1 7 6 6 7 1 0 0 0 0 0 2 4 4 2 4 2 4 4 2
Rectangle	89 degrees	1	10 10 10 10 0 0 0 0 0 0 0 4 4 4 4 4 4 4 4 4 4
Rectangle	45 degrees	1	10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10

- The next step is to examine the data and look for features that will differentiate the classes

- It is observed that ***orientation*** along with the ***area*** and ***centroid***, would be useful to control a robot in finding and placing the objects

- Also, the ***Euler number*** feature will identify the class ellipse hole, since it is 0 for this class and 1 for all others

- The ***projections*** can be used to differentiate the circles, ellipses and rectangle

- In general, the ellipses have some zeros and increasing and decreasing projections, the circles have increasing and decreasing projections, and the rectangles have constant projections, possibly with some zeros

61

- Thus the algorithm is as follows:

```
If euler number = 0
    Then Object = ellipse_hole
Else (euler number = 1)
    If projections are increasing and decreasing
        If projections has zeros
            Then Object = ellipse
        Else (projections has no zeros)
            Then Object = circle
    Else (projections not increasing and decreasing)
        Then Object = rectangle
```

62

- The classification algorithm is developed by the use of a ***training set*** and ***test set***

  - ➢ ***Training set***: a set of sample images used to develop an algorithm
  - ➢ ***Test set***: a set of images used to see how well the algorithm actually works on a different set of images

- The idea is that the results will simulate the real application in practice, and will not be biased by the training process – it is easy to get 100% success on the training set!