REAL TIME PATTERN RECOGNITION USING MATROX IMAGING SYSTEM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology
In
Telematics and Signal Processing

By

Praveen N

Roll No: 207EC106

Department of Electronics and Communication Engineering

National Institute of Technology

Rourkela

2007-2009

REAL TIME PATTERN RECOGNITION USING MATROX IMAGING SYSTEM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology
In
Telematics and Signal Processing

By

Praveen N

Roll no: 207EC106

Under the Guidance of

Dr. S. MEHER
Asst Professor

Department of Electronics and Communication Engineering

National Institute of Technology

Rourkela

2007-2009

National Institute of Technology
Rourkela

CERTIFICATE

This is to certify that the thesis titled. "**Real-Time Pattern Recognition using Matrox Imaging System**" submitted by Mr. **Praveen N** in partial fulfillment of the requirements for the award of Master of Technology Degree in Electronics and Communication Engineering with specialization in "**Telematics and Signal Processing**" at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University/Institute for the award of any Degree or Diploma.

Date: 25th May, 2009

Dr. S. MEHER
Asst. Professor
Dept. of Electronics and Communication Engg.
National Institute of Technology
Rourkela-769 008

ACKNOWLEDGEMENTS

This project is by far the most significant accomplishment in my life and it would be impossible without people who supported me and believed in me.

I would like to extend my gratitude and my sincere thanks to my honorable, esteemed supervisor **Prof. S. Meher,** Department of Electronics and Communication Engineering. He is not only a great lecturer with deep vision but also and most importantly a kind person. I sincerely thank for his exemplary guidance and encouragement. His trust and support inspired me in the most important moments of making right decisions and I am glad to work with him.

I want to thank all my teachers **Prof. G.S. Rath**, **Prof. G.Panda, Prof. K. K. Mahapatra, Prof. S.K. Patra** and for providing a solid background for my studies and research thereafter. They have been great sources of inspiration to me and I thank them from the bottom of my heart.

I would like to thank all my friends and especially my classmates for all the thoughtful and mind stimulating discussions we had, which prompted us to think beyond the obvious. I've enjoyed their companionship so much during my stay at NIT, Rourkela.

I would like to thank all those who made my stay in Rourkela an unforgettable and rewarding experience.

Last but not least I would like to thank my parents, who taught me the value of hard work by their own example. They rendered me enormous support during the whole tenure of my stay in NIT Rourkela.

PRAVEEN N

CONTENTS

ABSTRACT

A primary goal of pattern recognition is to be able to classify data into a set of related elements. Many applications today take advantage of pattern recognition; among them are data mining, face recognition, web searching, robotics and a lot more. Pattern recognition concerning Artificial Intelligence has been in research and development for approximately 50 years. The ability to quickly locate one or more instances of a model in a grey scale image is of importance to industry. The recognition/ localization must be fast and accurate. Two algorithms are employed in this work to achieve fast recognition and localization. Both the algorithm relies on a pyramid representation of both the model image and the search image. Specifically these algorithms are suitable for implementation on a personal computer equipped with an image acquisition board and a camera. Finally, results are given for searches on real images with perspective distortion and the addition of Gaussian noise.

LIST OF FIGURE

Chapter 1

Introduction

1.1 Preview

Digital Image processing is a promising area of research in the field of electronics and communication engineering, consumer and entertainment electronics, control and instrumentation, biomedical instrumentation, remote sensing, robotics and computer vision and computer aided manufacturing (CAM). Fast pattern recognition is an invaluable component of many machine-vision algorithms used in industry today. The ability to quickly identify an object or a marking on an object can aid in finding a desired part among a group of parts, or to register a part so that work may be performed on it. In this work, a hardware independent application namely 'Matrox imaging' is used to perform the pattern recognition in the real time environment.

Matrox Imaging Library is a hardware-independent application designed to work interactively with images for image capture, storage, and processing applications. It is capable of running on computers running Windows 2000 or XP. Matrox Inspector provides two flexible pattern recognition techniques to solve machine vision problems such as alignment, measurement, and inspection of objects: NGC (Normalized Grayscale Correlation) [1] pattern matching, and Geometric Model Finder. These pattern recognition capabilities allows to find patterns (referred to as a model) in an image. The NGC pattern matching algorithm finds models using a pixel-to-pixel correlation. It can be as fast, or faster, than Model Finder at locating models, depending on our application. Matrox Inspector's Geometric Model Finder allows us to find patterns, or models, based on geometric features, or shapes. The algorithm finds models using edge-based geometric features instead of a pixel-to-pixel correlation.

:1.2 Literature Review

Finding targets in images involves two important steps: localization of candidates matches within the image, and each candidate is verified through a matching process. Most of the relevant literature in the field involves the task of matching, and as such assumes that the images being compared are of the same size. Matching algorithms can be broken down into two major categories, those based on correlation and those based on features. The second category includes features based on colour, gradient orientation, moments and shape descriptors. Matching algorithms often ignore issues related to scale and orientation, leaving those to be determined during the localization stage.

The concept of normalized correlation [1] was developed to combat the effect of different illumination levels on correlation techniques. Some of the earliest attempts at matching have been done using correlation methods [7]. A major disadvantage of correlation techniques is that, when combined with naive localization method s such as brute-force search, they are computationally very intensive, and as such tend to be slow. Another set of techniques uses appearance-based matching methods such as eigenfaces and related techniques [32]. "Eigenfaces for Recognition" seeks to implement a system capable of efficient, simple, and accurate face recognition in a constrained environment. Principal component analysis of the images of the faces is used in this method, for avoiding the question of which features are important for classification, and which are not.

For matching entire images [19,26,29,36] a variety of methods are available. These techniques typically generate indices to allow fast retrieval of similar images from a database, and they

often assume that images have been normalized in size. In [26] appearance-based matching is used to index into a database of images. This above method is largely based on eigen-image matching with a Karhunen-Loeve transformation to limit the size of the basis space required. In a method similar to work reported later in [25], shape matching based on eigen modes is also used to index into the database.

Another set of techniques involves matching moments [27], this technique is limited to the case where localization has already been performed, and the pattern to be matched has been segmented from the background before applying the moment operators. These techniques are often powerful when this can be done, as they can provide rotation and size invariant recognition [21,8], although the issue of localization is defined. In both [25,19] models are matched to target instances using deformable templates. In [19] a two-stage hierarchy is used to achieve the desired efficiency and accuracy: in the first stage, simple and easily computable shape features are used to quickly browse through the database to generate a moderate number of plausible retrievals when a query is presented; in the second stage, a deformable template matching process is used to screen candidates from the first stage to discard spurious matches.

The other class of approaches to matching is based on extracting features from the images under consideration. Two popular features are color [9,29,36] and edge orientation [29,17,19]. Color features involve matching color histograms over the region of interest, but the application is only restricted to color images. In [29] a technique for indexing images based on both color and edge-orientation

Histograms are given, with the intent of matching via nearest neighbor techniques. In [19] color histograms with moment invariants are used as an indexing feature to find top candidates for subsequent matching with deformable templates. . In [36] Fourier is combined with color histograms, moment-invariants and intensity projection onto x and y axes to provide a means

for indexing into a trademark database. Both [29, 19] use edge orientation histograms to provide indexing for matching. In a different vein, Lowe [17,20] uses magnitude and orientation information at image regions determined for computing local features based and have stable responses to a difference of-Gaussians filter, across multiple scales. The novel aspect of this work is that it matches features in the target to features in a search image based on a Hough transform.

Maximally-stable external regions (MSER) [10] is a technique that identifies regions based on their stability when thresholded and it uses a range of thresholds. These stable regions are considered features, and suitable descriptors can be used to describe their content. A lot of affine-invariant descriptors [22–24] uses standard Harris-Stephens corner detector [11] across multiple scales for finding a maximal response, and computing an elliptical region around the feature location whose scale and orientation can be used to created an affine-invariant descriptor for the feature. Some of these descriptors, like SIFT descriptors, are useful for comparing features across different images and across different scales and orientations. These techniques have proven useful in content-based image retrieval and indexing in video sequences [30], using the "bag of features" concept. All the techniques involving image gradients may run into problems for artificial images with high contrast and considerable regularity in edge orientations, for example corporate logos.

Next type of feature approach involves the use of probabilistic features [28,31]. In [28] uses estimates of joint density functions for the recovered features for encoding object appearance. While this system is fast), it is only intended for detection and not localization. Stauffer & Grimson [31] use a set of training images of the target to learn a joint distribution of pixel intensities in an image patch. Comparison of the joint density to that of a candidate image

patch takes about1 second on a 500 MHz Pentium, making the method too slow for fast object localization. The recent class of technique is centered on neural networks, but to date they are more suited to pattern classification than localization. Kulkarni [18] uses back propagation to describes a size/rotation invariant object recognition method to recognize feature vectors, but the vectors are based on moments and thus are good for matching but not localization. Work done by Wechsler & Zimmerman [34, 35] looks more promising, but it is complex to be fast on modest hardware. In [14] neural networks are used to learn optimal templates for nearest neighbor matching. In [36] classification are based on recovered feature vectors using a neural network.

Finally, the concept of using pyramid representations for image analysis is certainly not new, and is closely linked to the theory of scale-space [15,16]. In this work pyramid representations are used to overcome some of the computational problems associated with correlation methods. Bu rt [12] has constructed a pyramid-based attention model which uses Gaussian and Laplacian pyramids, and which is used for foveation, tracking and determining where to look next based on image saliency. This system can be used for pattern recognition by using a pat tern tree to represent salient features of objects at different resolutions. If enough features are matched, then the pattern has been found. Special hardware has been developed [33] to aid in the construction of image pyramids. In Lowe [17] the stability of features is determined using multiple image scales.

Ramapriyan uses a multi-level decision tree approach to finding templates in images. A set of target templates is partitioned in a manner that allows subsets of templates to be removed from consideration should initial computations at a particular image location contraindicates. If evidence supports further search, the candidates under consideration is narrowed until a best

match occurs. Since only a small number of matches are expected over the large number of possible image locations, search speed is greatly improved. In [13] Greenspan pro poses a decision tree method for determining 3-D object pose in images, again making use of hierarchical search in pattern matching.

1.3 The Problem Statement

The ability to quickly locate one or more instances of a model in a grey scale image is of importance to industry. The recognition/ localization must be fast and accurate. In this work, a hardware independent application namely 'Matrox imaging' is used to perform the pattern recognition in the real time environment. Matrox Inspector provides two flexible pattern recognition techniques: NGC (Normalized Gray scale Correlation) [1] pattern matching, and Geometric Model Finder.

Thus, the problem taken for this thesis work is "*Real-time Pattern recognition using Matrox imaging system".*

In this work we present an algorithm which incorporates normalized correlation into a pyramid image representation [1] structure to perform fast recognition. In second part of the work we are using geometrical feature based algorithm for the matching process.

1.4 Standard Test Images

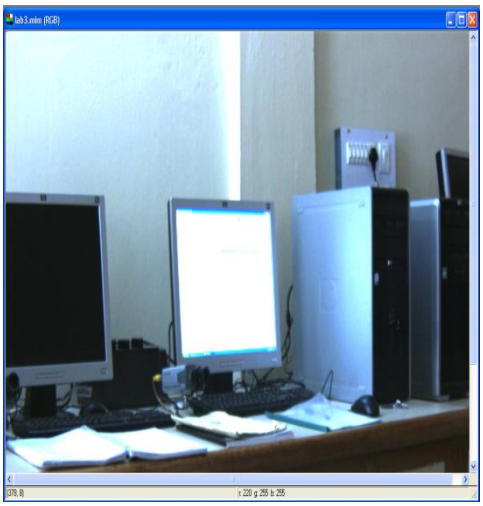
There are various standard test images, used extensively in literature, for this purpose. They are 'Lena', 'Baboon'. For real time processing, some real time images are also used in literature. Here original image has shown in fig. 1.1

1.5 Organization of the thesis

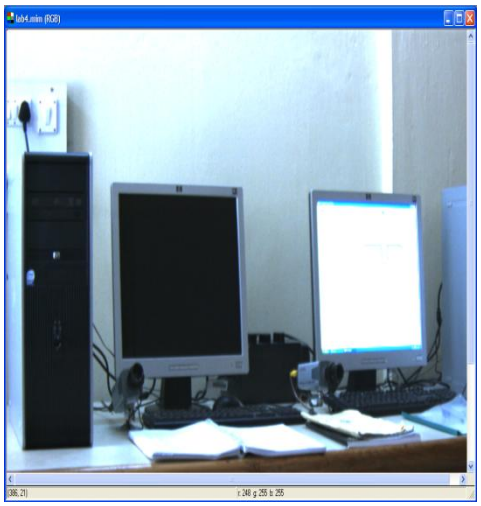
Following the introduction, the rest of image of the thesis is organized as follows. Chapter 2 gives basics of Matrox imaging application. Chapter 3 presents pattern matching application. Chapter 4 presents model finder application. Chapter 5 Comparisons and Discussion .Chapter 6 Conclusions and Scope Future work.

(a)

(b)



(c)



(d)

Fig. 1.1: Original test image of Lena, Baboon, real time images 1 & 2 in a, b, c, d, respectively.

Chapter 2

Matrox imaging library

Matrox Imaging Library is a hardware-independent application designed to work interactively with images for image capture, storage, and processing applications. It is capable of running on computers running Windows 2000 or XP. Matrox Inspector is an interactive Windows® application for image capture, processing, analysis, display and archiving. Based on the Matrox Imaging Library (MIL), it provides point-and-click access to an extensive set of functions for image processing, pattern recognition, blob analysis, edge extraction and analysis, measurement, character recognition, 1D and 2D code reading, calibration, and image compression. Matrox Inspector is a 32-bit Windows-based package, giving the full potential and ease of use of a graphical interface. As a companion product to Matrox Imaging library (MIL), it speeds up prototyping through live previews and access to all MIL modules.

Matrox Inspector is an interactive Windows® application for image capture, processing, analysis, display and archiving. Based on the Matrox Imaging Library (MIL), it provides point-and-click access to an extensive set of functions for image processing, pattern recognition, blob analysis, edge extraction and analysis, measurement, character recognition, 1D and 2D code reading, calibration, and image compression. MIL users employ Matrox Inspector as a companion tool to facilitate application development by having access to all MIL tools within a single interactive work environment. Matrox Inspector provides a familiar and easy-to-use interface with point-and-click access to imaging operations through pull-down menus, dialog boxes and toolbars as well as shortcuts. Options are conveniently presented to the user by frequency and likelihood of use. Live preview, apply and undo features allows the user to investigate the effect of operations without permanently changing the image. Results are presented as images, tables and/or graphs, and are immediately updated

with each new operation. Matrox Inspector facilitates colour imaging through various colour components display modes and simultaneous colour coded histograms and profile operations.

A workspace pane provides an at-a-glance overview of all documents and systems (digitizers) in use. The user can dynamically switch between systems and free systems for use with other MIL applications including the Matrox Intellicam frame grabber configuration utility. Matrox Inspector's interface can even be customized by adding, removing or editing menu items and toolbar buttons, and these changes can be shared with other users by saving them to file. Extensive on-line help for all operations is provided in the Microsoft® HTML help format, complete with keyword or full text searches.

2.1 Key Features

- easy-to-use interactive work environment for Microsoft® Windows® 2000 and Windows® XP
- directly acquire images from a variety of video sources using Matrox Imaging hardware
- load and save images including sequences in many file formats
- extensive set of accurate and robust image processing and analysis tools
- calibrate images to correct visual distortions and perform measurements in real-world units

- operations are performed live or on archived images including sequences
- track operation statistics (analysis results and speed)configure analysis tools
- control from and exchange data with other Microsoft® Windows® applications
- includes Matrox Intellicam camera configuration utility

2.2 Image Processing

Pictures, or images, are important sources of information for interpretation and analysis. These might be images of a building undergoing renovations, a planet's surface transmitted from a spacecraft, plant cells magnified with a microscope, or electronic circuitry. Human analysis of these images or objects has inherent difficulties: the visual inspection process is time-consuming and is subject to inconsistent interpretations and assessments. Computers, on the other hand, are ideal for performing some of these tasks.

For computers to process images, the images must be numerically represented. This process is known as *image digitization*. The digitization process divides an image into a two-dimensional grid of small units called *pixels* (picture elements). Each pixel has a value that corresponds to the intensity or color at that location in the image. Each pixel in the image is identified by its position in the grid and is referenced by its row and column number. Once images are represented digitally, computers can reliably automate the extraction of useful information using digital image processing. Digital image processing performs various types of image enhancements, distortion corrections, and measurements.

Matrox Inspector provides a comprehensive set of image processing operations. There are two main types of processing operations:

- Those that enhance or transform an image.
- Those that analyze an image (that is, generate a numeric or graphic report that relates specific image information).

2.2.1 Image enhancement and transformation

Matrox Inspector's image enhancement and transformation operations allow improving the quality of our image and/or transforming our image. These include:

- **Point-to-point operations.** These operations include brightness and contrast adjustment, constant thresholding, arithmetic, and image mapping operations. Point-to-point operations compute each pixel result as a function of the pixel value at a corresponding location in either one or two images.
- **Spatial filtering operations.** These operations are also known as convolution. They include operations that can enhance and smooth images, accentuate image edges, and remove 'noise' from an image. Most of these operations compute results based on an underlying neighborhood process: the weighted sum of a pixel value and its neighbors' values. You can use one of Matrox Inspector's pre-defined filters or create your own filter(s).
- **Morphological operations.** These operations include erosion, dilation, opening, and closing of images. These operations compute new values according to geometric relationships and matches to known patterns (structuring elements) in the source image. If necessary, you can create your own structuring element for one of Matrox Inspector's morphological operations.
- **Geometric operations.** These operations include scaling, rotating, translating and flipping of images, as well as polar-to-rectangular and distortion correction.

2.2.2 Image analysis

Matrox Inspector's image analysis operations summarize a frame of pixels into a set of values which relate specific image information. These include:

- **Basic statistical operations.** These operations extract statistical information from a specified region of an image, such as its distribution of pixel values or its profile.
- **Pattern recognition operations.** These operations allow determining how similar certain areas of the image are to a pattern. Patterns based on pixel values or on the presence of edges can be used.
- **Blob analysis operations.** These operations allow identifying and measure aggregated regions of pixels within an image (commonly known as blobs).
- **Edge Finder operations**. These operations allow identifying, selecting and measure edges (contours and crests).
- **Measurement Marker Locator operations.** These operations allow locating edges or pairs of edges (known as stripes) and measure their features.
- **Measurement operations.** These operations allow taking a variety of measurements between positional results, and take quick on-the-spot measurements, such as length, area, and angle.
- **Code operations.** These operations allow both reading and generating code symbols.
- **Character recognition operations.** These operations allow reading and/or verifying character strings in the target image using template-based or feature-based technology.

2.3 Different image processing operations available in Matrox

2.3.1 Denoising

Spatial filtering provides an effective method to reduce noise. Spatial filtering operations determine each pixel's value based on its neighborhood values. They allow images to be separated into high-frequency and low-frequency components. There are two main types of spatial filters which can remove high frequency components: low-pass filters and rank filters.

- Low-pass spatial filters

Low-pass spatial filters are effective in reducing Gaussian random noise (and high frequency systematic noise) when the noise frequency is not too close to the spatial frequency of significant image data. These filters replace each pixel with a weighted sum of each pixel's neighborhood. Note that as these filters remove noise, they have the side-effect of selectively smoothing the image and reducing edge information. Matrox Inspector provides several types of predefined low-pass Finite Impulse Response (FIR) filter operations, among them: **Average**, **Smooth**, and **Smooth 5x5**. The average filter places the same importance on all pixels in the neighborhood, whereas the smooth filters place more importance on the pixels that are closer to the center of the neighborhood. All the predefined FIR filters operate on a 3x3 neighborhood, except the **Smooth 5x5** filter, which operates on a 5x5 neighborhood.

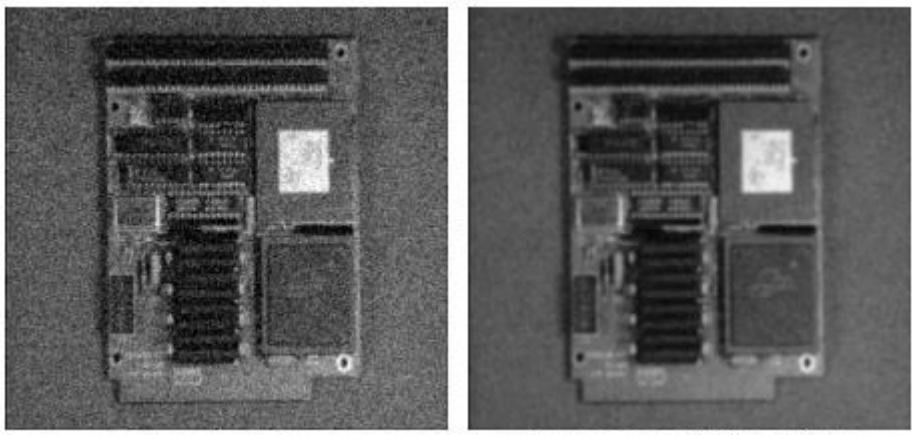


Fig 2.1 low pass spatial linear filter

- Rank filters

Rank filter operations are more suitable for removing salt-and-pepper type noise since they replace each pixel with a value from its neighborhood rather than with a weighted sum of its neighborhood. A median filter is a type of rank filter that replaces each pixel with the median of its neighborhood. Matrox Inspector provides a predefined median filter, which operates on each pixel's 3x3 neighborhood.

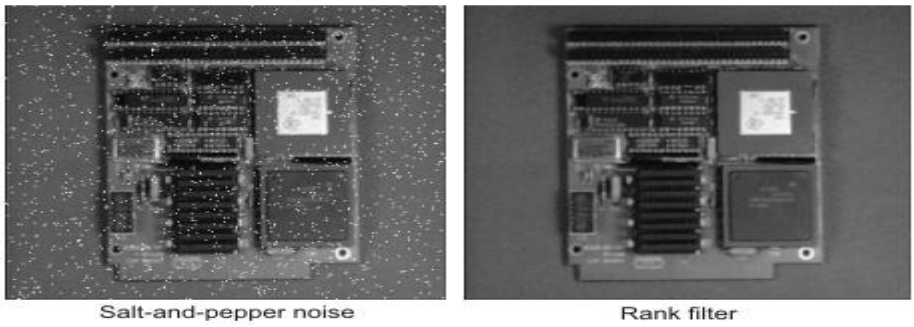


Fig 2.2 Rank filter

2.3.2 Thresholding images

Thresholding maps all pixels in a specified intensity range to a new constant value. Since a threshold image contains fewer pixel values than the original image, some operations can be performed more efficiently. Note that images with full grayscale levels are useful for some tasks, but have redundant information for other tasks.

- Binarizing

Thresholding can be used to binarize an image. Binarizing reduces an image to two grayscale values (for example, 0 and 255, as below).Alternatively, using the **Make ROI** option, the image will not be binarized; instead, an ROI will be created containing all pixels with values above the threshold value. Note that floating-point images are binarized, by default to the

maximum and minimum values in the image, not to the range of the buffer. Binary images can be used as a mask to identify blobs in a blob analysis application.

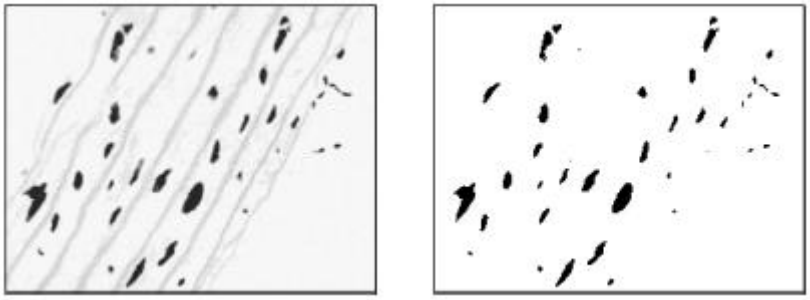


Fig 2.3 Binarizing of gray level image

- Two Level

In the above image, the objects of interest (the nuclei) are darker than everything else in the image, so a simple binarization will separate the objects from the background. In other cases, the objects might be both darker and lighter than the background. To identify the objects in these cases, force values above and below object values to a suitable background value and force object values to a suitable foreground value, as shown below. This can be done using a *two level* threshold.

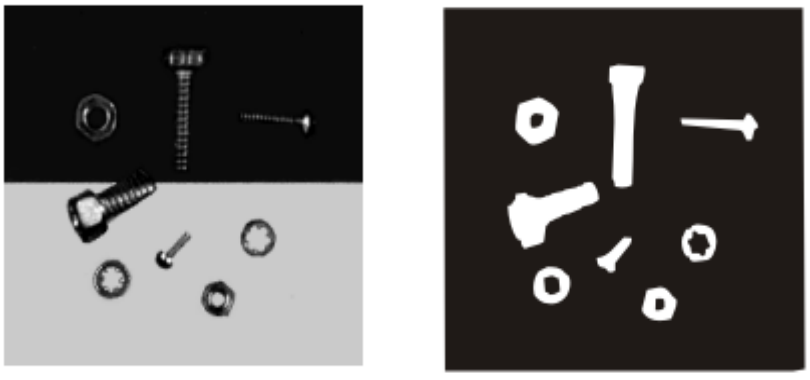


Fig 2.4 Image obtained after two level thresholds

- Partial Thresholds

In some situations, it is useful to set all pixels within a certain range of values to a uniform pixel value, without altering the values of the remaining pixels. This can be accomplished with a *low pass*, *high pass*, *band pass*, or *band reject* threshold. For example, a high pass threshold can set all values below a certain value to zero, creating a uniform background. A band reject threshold can be used to replace a particular value, or range of values, with another constant value. By using a band reject threshold on multiple ranges in the image, you can create a quantized imaged.

- Determining threshold value from a histogram

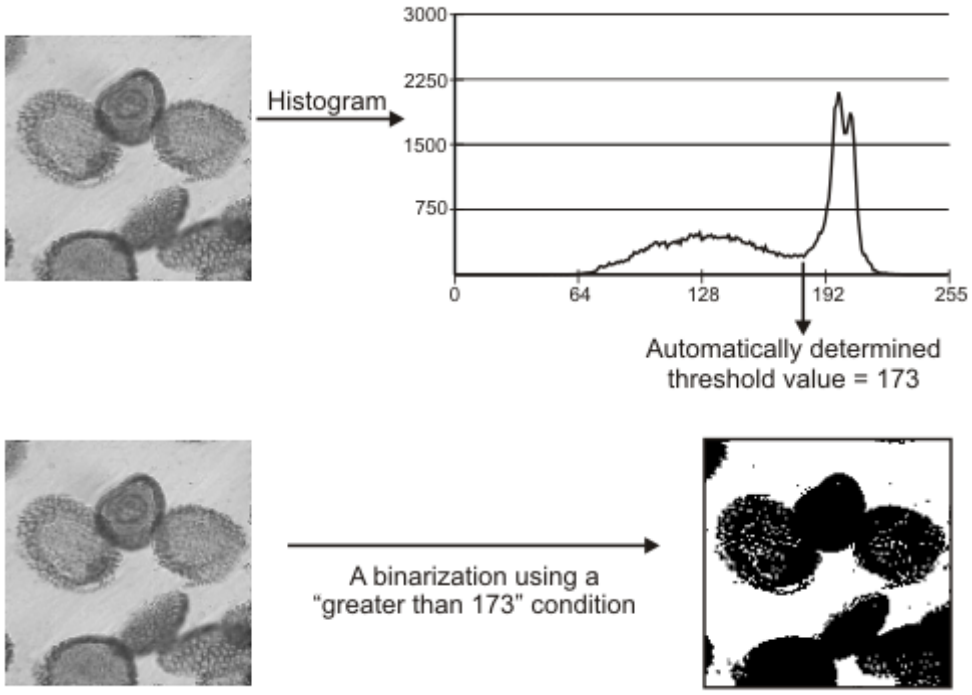


Fig 2.5 Determining threshold from histogram

Specifically, a histogram of the source image is internally generated; then the threshold value is set to the minimum value between the two most statistically important peaks in the histogram, on the assumption that these peaks represent the object and the background. If the histogram contains more than two peaks, then the threshold value will typically be set between the two principle peaks, though exceptions exist for pure black and full saturation (0,255 in an 8-bit unsigned image).

2.3.3 Accentuating edges

- Edges defined

Many applications highlight locations of an image that are marked by a rapid change in pixel values. This is done, for example, to accentuate the edges of the various objects and features in an image, or to highlight defects in a smooth object. In general, edges can be distinguished by a sharp change in intensity between adjacent pixels.

- Horizontal edges are created when vertically connected pixels have values that are different from those immediately above or below them.
- Vertical edges are created when horizontally connected pixels have values that are different from those immediately to the left or right of them.
- Oblique edges are created from a combination of horizontal and vertical components.

- Edge operations

There are two main types of edge processing operations:

- One that enhances edges to generate higher image contrast.
- One that extracts (detects) edges from the image.

Both these edge operations are types of convolutions (or spatial filtering operations) that replace each pixel with a weighted sum of its neighborhood. The weights applied to each pixel in a neighborhood determine the type of operation that is performed. For example, certain weights produce horizontal edge detection, whereas others produce vertical edge detection.

➢ Edge enhancers

Three edge enhancers are available:

- Sharpen1.
- Sharpen2.
- ReliefNW8U.

After an edge enhancement operation, the amplified edges accentuate all objects so that the eye sees an increase in detail. Note that this operation might not produce good results for further processing because when edges are enhanced, it also enhances noise pixels.

➢ Edge detection

Matrox Inspector provides a number of edge detection operations, each offering different advantages depending on the application.

- Horizontal and vertical edge detection.
- Laplacian edge detection1 and Laplacian edge detection2.
- Sobel1 and Sobel2.
- Sobel angle
- Prewitt.

2.3.4 Basic geometric transforms

Matrox Inspector provides a set of basic geometrical transform operations that can be used to adjust the shape and orientation of the images. Using these operations, the image can be resized, stretch or contract it in either direction, rotate it, shift it, copy it to a new location in another image, or create an image containing the same image multiple times.

- Scaling

Scaling allows correcting for aspect ratio distortions in an image, or reducing or magnifying the image to the appropriate size. The image can resize the along the horizontal and/or the vertical axis.

- Rotating images

When rotating an image, Matrox Inspector allows specifying the center of the rotation. The center of rotation is the position, within the image, around which the rotation takes place. The default center of rotation is the center of the image. The diagrams below display the effect of rotating an image by 60°, around two different centers of rotation. In these cases, the rotated images have been "clipped" to the original image size.

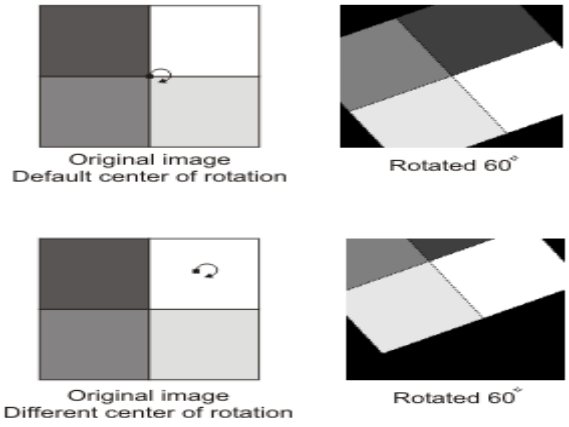


Fig 2.6 Rotated images

- Tiling

The source region can be positioned in the center or corners of the destination. In addition, when inserting a single copy of the source region, the destination image is filled with a specified background pixel value, and a single copy of the source region is positioned within it. Using the ***Single*** copy mode, a source region can copy into a new larger image, creating margins around the image. This can be useful for over scan purposes, especially when searching for edges and stripes at an angle. Edge and stripe searches cannot process areas that extend past the borders of the image, so creating this margin will allow search all the way into the corners. The margins also prevent their images from being cropped when rotating.

2.3.5 Creating custom filters

Spatial filtering operations include operations such as smoothing, Denoising, edge enhancement, and edge detection.

A spatial filtering operation (convolution) is a type of neighborhood operation that determines the new value for a pixel based on the weighted sum of the pixel and the pixel's neighboring values. There are two types of spatial filters: FIR filters operate on a finite neighborhood, while IIR filters take into account all values in an image. In Matrox Inspector, you can specify a predefined FIR or IIR spatial filter, or a custom FIR spatial filter.

- Finite Impulse Response (FIR) filters

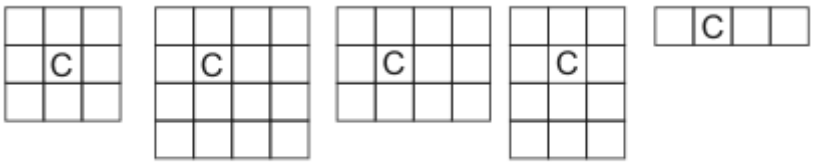
For FIR filters, the weights are often represented in a matrix (known as a *kernel*) and determine the operation type of the filter. For example, applying the following kernel results in a sharpening of an image:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Whereas, applying the following FIR filter smooth's an image (it also increases the intensity of the image by a factor of 16, so it will need to normalize the convolution result):

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

When using FIR filters, the dimension of the kernel determines the size of the neighborhood that is used in the operation. The result of the operation is stored in the destination image at the location corresponding to the kernel's center pixel. When the kernel has an even number of rows and/or columns, the center pixel is considered to be the top-left pixel of the central elements in the neighborhood.



Examples of neighborhoods and their center pixel.

A filter can contain up to nine kernels. A kernel can have up to fifty rows and fifty columns. The final pixel value produced by a kernel can be offset and divided by specified amounts; its absolute value can then be taken. After the results of each kernel are combined, a final offset and divide can be applied; the absolute value of this final result can then be taken. If saturation is enabled, final results that overflow or underflow the range of the destination image are set to the maximum or minimum value of the destination image, respectively. In summary, the order of operation is:

1. Matrox Inspector applies each kernel of the filter to the original image, using the offset, divide, and absolute value options defined for that kernel.
2. It then combines the results of each kernel. Matrox Inspector can either sum the results or take the maximum value of the results.
3. The final offset, divide, and absolute value options defined for the filter are then applied.
4. If enabled, results are saturated. If not enabled, overflows or underflows will wrap around.

- Infinite Impulse Response (IIR) filters

When using IIR filters, the weights are automatically determined by the type of filter, the mode of the filter, the type of operation to perform, and the degree of smoothness (strength of Denoising) applied by the filter. The type of filter determines the distribution of the neighborhoods' influence. MIL supports two types of IIR spatial filters: Canny-Deriche filter and Shen-Castan filter. For the Canny-Deriche filter, the neighborhoods' influence decreases much slower as the distance from the central pixel increases, compared to the Shen-Castan filter. There are two modes in which to perform the filter: kernel mode and recursive mode. When applying an IIR filter in recursive mode, the kernel size would be theoretically infinite if this mode actually used a kernel. When performing a custom IIR filter in kernel mode, the filtering is done using a kernel approximation (FIR) of the filter. Kernel mode has been provided to take advantage of your system if it is optimized for kernel mode; kernel mode is typically not as efficient as recursive mode unless the system is optimized for kernel mode.

Chapter 3

Pattern Matching

Pattern recognition in computer vision is the task of finding a given object in an image or video sequence. Humans recognize a multitude of objects in images with little effort, despite the fact that the image of the objects may vary somewhat in different viewpoints, in many different sizes / scale or even when they are translated or rotated. Objects can even be recognized when they are partially obstructed from view. This task is still a challenge for computer vision systems in general. In this section the "normalized gray scale correlation**"** algorithm[1] is used for pattern matching. It can be as fast, or faster, than Model Finder at locating models, depending on the application.

3.1 Steps to performing a search

The basic steps to search for a model in an image are:

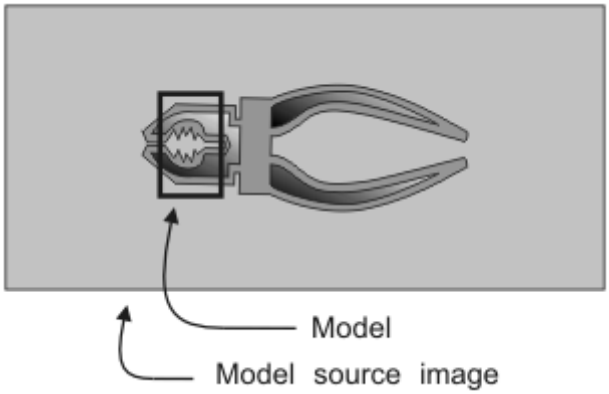
1) Define the search model. The model can be created manually or the Matrox Inspector automatically selects a unique model.

2) Fine-tune the model. This might involve masking regions of the model and/or redefining the model's hotspot (reference point).

3) Set the model's search constraints.

4) Search for the model in an image, or in a part of the image.

5) If required, save the model. Once the model is saved, it can load from disk for future searches. In such a case, steps 1 to 3 need not be performed unless the model requires adjustment.



To obtain the best search model, it should test several models created from images acquired over the full range of expected conditions for the target image (for example, lighting and angular displacement).

3.1.1 Defining a search model

The first step in performing a search is to define a search model. Keep in mind that a good model should not have very thin features as the identifying features that make it unique.

- The model can be create manually by selecting an area in an existing image, using the ROI drawing tool in the model's source image, followed by the *Analysis Pattern New Model* command.
- The Matrox Inspector automatically select a unique one (called an auto model) using the *Analysis Pattern New Auto model* command. In this case, Matrox Inspector searches, in an existing image, for the most-suitable unique area of a specified size, and creates a model from this area. Auto models are most useful in alignment applications.

The search model is always rectangular in shape. When using non-rectangular ROI drawing tools to define a search model, the model will be created using the ROI's bounding box. In this case, a mask might be necessary to exclude those pixels outside the non-rectangular ROI.

3.1.2 Fine-tuning the model

Once the search model defined, it might be need to fine-tune. This might involve masking certain regions of the model and/or redefining the model's hotspot in the *Dimension* tab of the *Pattern Matching* dialog box.

- Masking a model

Often, the search model contains regions that are not needed; Matrox Inspector ignores it when searching for the model in the target image. These regions might be noise pixels or simply regions that have nothing to do with what is searching for. In these cases, parts of the search model can be masked. Matrox Inspector then ignores the regions under the mask when searching for occurrences of the model. Note that the ROI has to be at least as large as the model.

- Redefining a model's hotspot

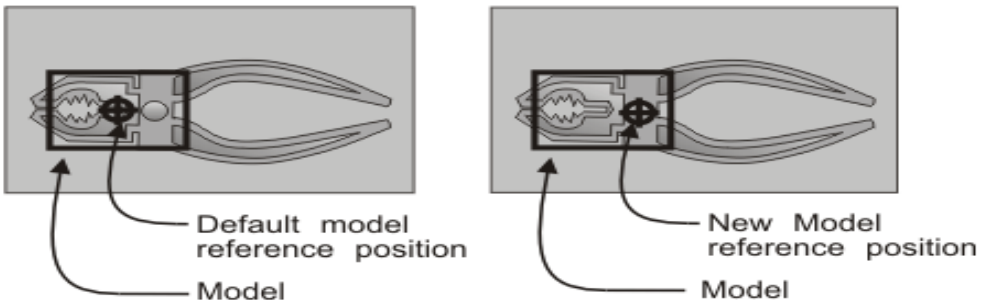


Fig 3.1 Redefining a model's hotspot

The returned coordinates of a search are the coordinates of the model's hotspot (reference position). These coordinates are relative to the origin of the image being searched (calibrated or non-calibrated). By default, a model's hotspot is at the model's geometric center.

If there is a particular spot from which the results are returned, model's hotspot can change relative to the spot, interactively by clicking on the mouse button in the *Dimension* tab of the *Pattern Matching* dialog box and using the cursor to place the hotspot, or by entering the required X and Y coordinates in the appropriate fields. For, example, if the model has a hole and the result is finding relative to this hole, the model's hotspot have to accordingly.

3.1.3 Setting a model's search constraints

Once a search model defined, it is assigned a set of default search constraints. These constraints can be changed using the various tabs of the *Pattern Matching* dialog box, in order to:

- Set the number of occurrences to find.
- Restrict the search to a specific region of the target image.
- Set the threshold for acceptance and certainty.
- Set the search speed.

In most applications, the above search constraints are all need to adjust. Matrox Inspector provides more advanced search constraints for cases where customizing the search strategy is needed, however this is not recommended unless absolutely necessary.

- Number of matches

With Matrox Inspector, how many matches to try to find can specify. If need is a one good match, set it to one (the default value) and avoid unnecessary searches for further matches. If a correlation has a match score above the certainty level, it is automatically considered an occurrence (default 80%), and the remaining occurrences will be the best of those above the acceptance level. Note that the maximum number of matches that can be found is 10000. If

we specify to find all matches, or a number that is greater than 10000, only the first 10000 matches will be displayed. Specifying a very high number of matches will slow down the display of results.

- Search region

When searching for a model in an image, the search region can be all, or only a part, of the image. The search region is actually the region in which to search for the model's hotspot; therefore the search region can be smaller than the model. If the model's hotspot is redefined, make sure that the search region covers this new reference position and takes into account the angular search range of the model. Search time is roughly proportional to the area of the search region. If speed is a consideration, make the search area as small as possible.

- The acceptance level

The level at which the correlation between the model and the pattern in the image is considered a match is called the acceptance level. The acceptance level can be set for the specified model in the *Search* tab of the *Pattern Matching* dialog box. If the correspondence between the image and the model (referred to as the *match score*) is less than this level, there is no match. The acceptance level affects the number of results returned, because only those match with a score greater than or equal to the acceptance level will be counted. A perfect match is 100%, a typical match is 80% or higher (depending on the image), and no match is 0%. If the images have considerable noise and/or distortion, it should have to set the acceptance level below the default value of 70%.

- Certainty level

The certainty level is the match score (usually higher than that of the acceptance level) above which the algorithm can assume that it has found a very good match and can stop searching the rest of the image for a better one. The certainty level is very important because it can greatly affect the speed of the search. Often, if the pattern wanted is unique in the image, so anything that reaches the acceptance level must be the match that wants; therefore, set the certainty and acceptance levels to the same value. A better value is about 80%, meaning that most of the time the search will stop as soon as it sees the real match, but in a degraded image (where nothing reaches the certainty level), it will take the extra time to look for the best match that reaches the acceptance level.

- The search speed

When searching for a model in an image, it is possible to increase or decrease the search speed. As increase the speed, the robustness of the search operation (the likelihood of finding a match) is decreased slightly, and match scores might be a little less accurate.

- Very High or High**.** As expected, the high speed settings allow the search to take all possible shortcuts, performing the search as fast as possible. The high speed setting is recommended when searching on a good quality image or when using a simple model, since accuracy might be slightly affected.
- Medium**.** This is the default setting and is recommended for medium quality images or more complex models. A search with this setting is capable of withstanding up to approximately 5 degrees of rotation for typical models.
- Low or Very Low**.** Use these settings only if your image quality is particularly bad and if we have encountered problems at higher speeds.

3.2 Rotation

By default, the angle of the search is 0 ^{o}. There are two ways to search for a model that can appear at different angles:

- **Rotated model search.** Search for rotated versions of the model.
- **Circular over scan model search.** Search for models taken from the same region in rotated images

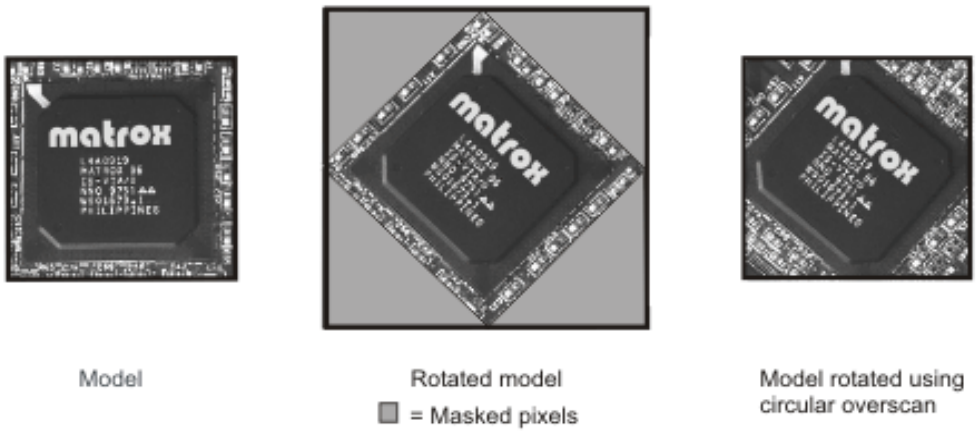


Fig 3.2 Rotated model search

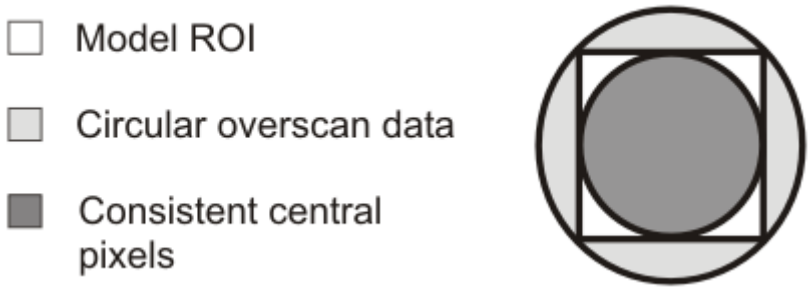
Matrox Inspector's implementation of a circular over scan search is faster and more accurate than that of a rotated model search when performing an angular search. A rotated model search should only be used when the pixels surrounding the ROI follow no predictable pattern. When we expect a small number of possible orientations of model occurrences in the target image (for example, models appearing at 90 ^{o} intervals), normalized gray scale correlation (NGC) used by pattern matching is very fast. When we don't know how much the model occurrences will vary in angle, or if we require results with very precise angle information, consider using *Geometric Model Finder* to perform the search.

- Rotated model search

To perform a rotated model search, follow the same procedure done when manually creating the model, using the *Analysis Pattern New Model* command. Make sure the *Analysis Pattern Use Circular Over scan* command is disabled. Enable the *Enable search with rotate* option found in the *Angle* tab of the *Pattern Matching* dialog box to set the search angle of the model, and the *Delta negative* and *Delta positive* of the angular range, if necessary. Angles are expressed in degrees and floating-point values are accepted.

- Circular over scan model search

To perform a circular over scan model search, follow the same procedure as done when performing a rotated model search, except enable the *Analysis Pattern Use Circular Over scan* command. When a circular over scan model is created, over scan data outside the ROI is also extracted from the source image. Specifically, Matrox Inspector extracts the region enclosed by a circle which circumscribes the model's ROI.



When a circular over scan model is preprocessed, Matrox Inspector will internally extract different orientations of the model from the over scanned model. It is recommended that the model's source ROI be as square as possible: the longer the rectangle, the smaller the number of consistent central pixels in every model. Therefore, this type of model should only be used when the model's distinct features lie in the center of the region, so that they are included in all models when rotated. Finally, the model must not be extracted from a region too close to the edge of the model's source image: an error will be generated if circular over scan data extends beyond the boundaries of the image.

3.3 The pattern matching algorithm

Normalized grayscale correlation (NGC) [1] is widely used in industry for pattern matching applications. The correlation operation can be seen as a form of convolution, where the pattern matching model is analogous to the convolution kernel. In fact, ordinary (un-normalized) correlation is exactly the same as a convolution:

$$\mathrm{r} = \sum_{i=1}^{i=N} IiMi$$

In other words, for each result, the N pixels of the model are multiplied by the N underlying image pixels, and these products are summed. Note, the model does not have to be rectangular because it can contain masked pixels that are completely ignored during the calculation. When the correlation function is evaluated at every pixel in the target image, the locations where the result is largest are those where the surrounding image is most similar to the model. The search algorithm then has to locate these peaks in the correlation result, and return their positions.

$$r = \frac{N\sum IM - \left(\sum I\right)\sum M}{\sqrt{\left[\left(N\sum I^2 - \left(\sum I\right)^2\right)\left(N\sum M^2 - \left(\sum M\right)^2\right)\right]}}$$

Unfortunately, with ordinary correlation, the result increases if the image gets brighter. In fact, the function reaches a maximum when the image is uniformly white, even though at this point it no longer looks like the model. The solution is to use a more complex, normalized version of the correlation function (the subscripts have been removed for clarity, but the summation is still over the N model pixels that are not "don't care"):

This expression, the result is unaffected by linear changes (constant gain and offset) in the image or model pixel values. The result reaches its maximum value of 1 where the image and model match exactly, gives 0 where the model and image are uncorrelated, and is negative where the similarity is less than might be expected by chance.

In this case, not interested in negative values, so results are clipped to 0. In addition, we use r^2 instead of r to avoid the slow square-root operation. Finally, the result is converted to a percentage, where 100% represents a perfect match. So, the match score returned is actually:

$$\text{Score} = \max(r, 0)^2 * 100$$

Note, some of the terms in the normalized correlation function depend only on the model, and hence can be evaluated once and for all when the model is defined. The only terms that need to be calculated during the search are:

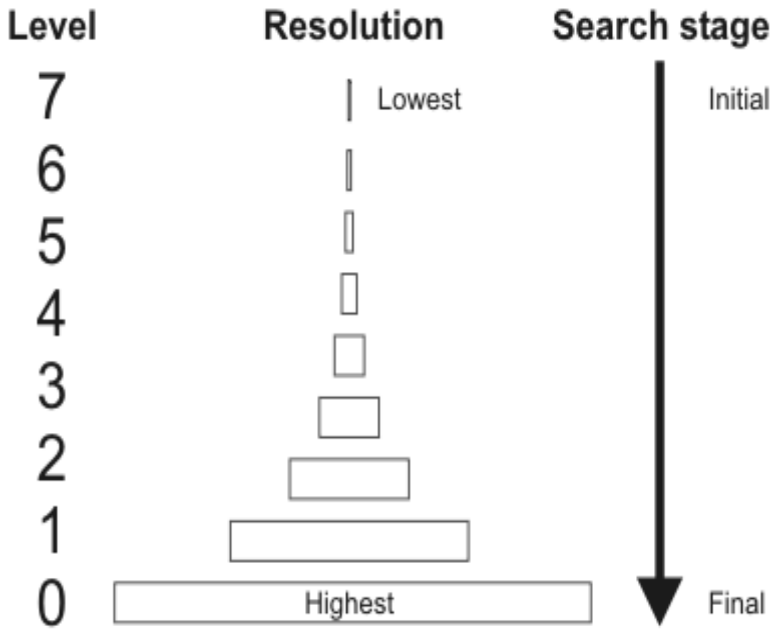
$$\sum I, \sum IM, \sum I^2$$

This amounts to two multiplications and three additions for each model pixel. On a typical PC, the multiplications alone account for most of the computation time.

3.4 Hierarchical Search

A reliable method of reducing the number of computations is to perform a so-called hierarchical search. Basically, a series of smaller, lower-resolution versions of both the image and the model are produced, and the search begins on a much-reduced scale. This series of sub-sampled images is sometimes called a resolution pyramid, because of the shape formed when the different resolution levels are stacked on top of each other.

Each level of the pyramid is half the size of the previous one, and is produced by applying a low-pass filter before sub-sampling. If the resolution of an image or model is 512x512 at level 0, then at level 1 it is 256x256, at level 2 it is 128x128 and so on. Therefore, when the level in the pyramid is higher, the lower the resolution of the image and model.



The search starts at low resolution to quickly find likely match candidates. It proceeds to higher and higher resolutions to refine the positional accuracy and make sure that the matches found at low resolution actually were occurrences of the model. Because the position is

already known from the previous level (to within a pixel or so), the correlation function need be evaluated only at a very small number of locations. Since each higher level in the pyramid reduces the number of computations by a factor of 16, it is usually desirable to start at as high a level as possible. However, the search algorithm must trade off the reduction in search time against the increased chance of not finding the pattern at very low resolution. Therefore, it chooses a starting level according to the size of the model and the characteristics of the pattern. For an 128x128 model and 512x512 image, it might start the search at level 4, which would mean using an 8x8 version of the model and a 32x32 version of the target image. If required, force a specific starting level or last level, using the *Search* levels section of the *Pattern Matching* dialog box's *advanced* tab.

3.5 Normalized Grey-Scale Correlation in a Pyramid Image

Fast pattern recognition has long been of interest to industry. The ability to quickly locate a desired part, or accurately determine its location for registration purposes is a valuable tool in manufacturing. One popular method is *normalized grey-scale correlation* (NGC) [1], in which a target image is correlated with the entire search region, and peak responses used to identify the location of the target. In the past specialized hardware was often required to do this with enough speed to make it viable, but recent a made it possible to perform searches on non-specialized hardware. This paper describes new techniques to allow for fast target locating using modest, non-specialized microcomputer hardware. More specifically, the algorithm is suited to implementation none a personal computer equipped with only an image acquisition board and a camera.

The algorithm uses a pyramid representation of both the model image and the search image, as well as an estimate of the gradient of the correlation surface. The algorithm begins by creating a pyramid representation of the model image. A worst-case analysis is then used to

determine how the following will effect on the correlation score of i) the number of pyramid levels combined with ii) the possible different registrations of the pyramid sampling grid with respect to the model, to determine the optimum number of pyramid levels to use. During the searching, a pyramid representation is built from the image being searched for instances of the model. The depth of both the model pyramid and target image are same. The top-level of the model pyramid is correlated with the top-level of the search-image pyramid to produce a correlation surface from which likely candidates for model instances can be chosen. The search process descends through the pyramid performing a steepest descent search based on the correlation-gradient at each level for each candidate. At the bottom level, either a model instance has been found or the candidate is rejected for failing to meet a minimum threshold for its correlation score.

3.5.1 NGC in a Pyramid Representation

This section describes the use of a pyramid image representation [1] to reduce the computational complexity associated with correlation search. Let I(x, y) represent an image Iw × Ih pixels in size. The top-left corner of the image is defined to be the origin. Each pixel is assumed to encode a grey value in the range 0 . . . 255, one byte per pixel, although color correlation is easily achieved by replacing pixel multiplications with inner-product operations performed on RGB values represented as vectors.

However, the cost of the computation increases considerably if normalized correlation[1] is used

$$C(x,y) = \frac{G(M(0,0), I(x,y))}{\sqrt{G(M(0,0), M(0,0))G(I(x,y), I(x,y))}}$$

Where

$$G\left(f_1(x_1, y_1), f_2(x_2, y_2)\right) = \sum_{i=0}^{M_w-1} \sum_{j=o}^{M_h-1} f_1(x_1 + i, y_1 + i) f_2(x_2 + i, y_2 + j)$$

An overview of the method is as follows: build pyramid representations of both the model and the target (search space), and perform correlation search at the top levels of the two pyramids. This can be done very quickly due to the reduced image sizes. The best match at the top level can be refined using a coarse-to-fine strategy in which the best estimate of location at level k in the pyramid is used as the starting point for the search at level k − 1. When the bottom of the pyramid is reached, the model has been found. Multiple instances can be found through repeating this procedure by choosing more than one match at the top level of the pyramid.

The method of building both the image pyramid and the model pyramid is described briefly. The structure of the pyramid is quite simple. An averaging-with-no-overlap scheme is used, and uses a sub-sampling rate of 2. This means that each level in the pyramid has dimensions equal to one-half of those for the level below, meaning that each level is one-quarter the size of the one immediately below. Each pixel in a given layer is the average of four pixels from the layer below. This type of pyramid can be built quickly, as each pixel in a new level only requires 3 ads and one shift to compute.

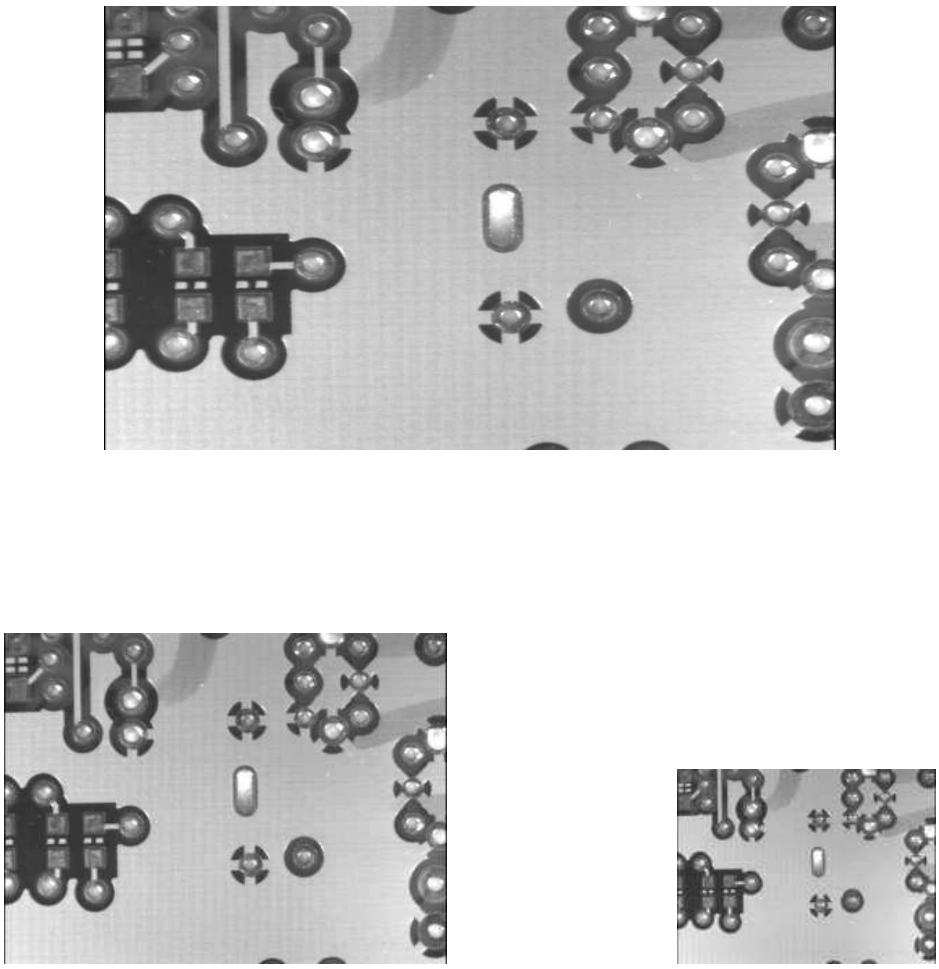


Fig. 3.3 The pyramid representation for a typical image is shown. The pyramid has three levels, with level 0 being the largest image at 320×240 (top), and level 2 being the smallest at 80 × 60 (bottom right). In the level 0 image, a search model is defined.

The number of levels is limited by $K_{max} \leq \log_2 \min(M_w, M_h)$. The advantages of building a pyramid become quickly obvious: the K^{th} level of a pyramid has $2^{2(K-1)}$ times fewer pixels than does the original image. Remember that the model pyramid at level K also has $2^{2(K-1)}$ times fewer pixels than at its level 0, so that the total cost of NGC at the top level of the pyramid is $2^{4(K-1)}$ times smaller than NGC on the original image. For a 4-level pyramid, this factor is 4096. Tthe localization of the pattern at the top level of the pyramid is not perfect,

but this can be used as an initial estimate for the next level of the pyramid. At each level of the pyramid a small number of correlations is used to refine the location estimate. Instead of performing an exhaustive local search for the new maximum at each level of the pyramid, it is possible to estimate the gradient of the correlation surface and use a steepest descent method to perform the search.

3.5.2 Derivation of Correlation Gradient

When the estimate of model location is refined at each level of the pyramid, it is possible to use an estimate of the gradient of the correlation surface to perform a steepest descent search. However, in applying steepest descent to a surface that we expect will have multiple maxima, it is important to identify regions expected to contain local maxima to allow the search to converge quickly. The use of the image pyramid does this: not only does it limit complexity of the top-level correlation, but it facilitates finding candidate regions and provides a means of quickly refining the search. In the derivation which follows the continuous case is developed: the transition to the discrete case is straight forward.

Given an image I(x, y) and a model M(x, y), with M being differentiable over all x and y, the correlation coefficient surface can be defined as

$$C(u\ v) = \frac{\iint I(x,y)(MW)(x-u,y-v)d_x d_y}{\left[\iint I^2(x,y)W(x-u,y-v)d_x d_y\right]^{\left(\frac{1}{2}\right)}}$$

The function W(x, y) is a windowing function which is used to force the contribution from M to zero at its boundaries:

$$W(x\ y) = 0:\ x<0,\ x>M_w,\ y<0,\ y>M_h$$

The notation (MW) (x, y) is shorthand for M(x, y) W(x, y). For simple correlation computation, W is normally chosen to be a box function. However, since we will want to differentiate Eq. 2, the windowing function should be chosen such that its derivative goes to zero at the borders of the window. It is assumed that ⌡ ⌡ $M^2(x, y)$ W(x, y) dx dy = 1, *i.e.* that M(x, y) is normalized with respect to the windowing function.

$$\nabla C(u,v) = \begin{bmatrix} \frac{\partial C}{\partial u} \\ \\ \frac{\partial C}{\partial v} \end{bmatrix} .$$

The gradient of the windowed correlation is

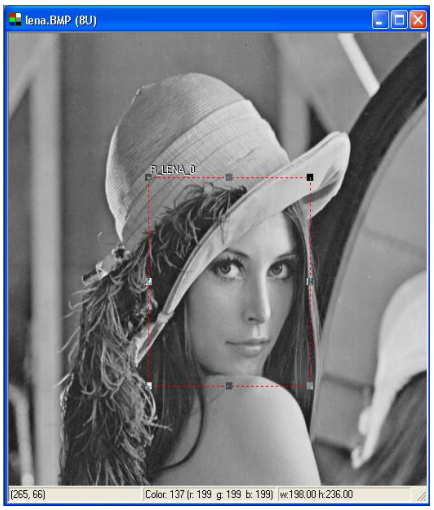
$$\nabla C^2(u,v) = 2\frac{\iint I(x,y)(MW)(x-u,y-v)d_x d_y}{\iint I^2(x,y)W(x-u,y-v)d_x d_y} \times \iint I(x,y)\nabla(MW)(x-u,y-v)d_x d_y$$
$$+\left[\frac{\iint I(x,y)(MW)(x-u,y-v)d_x d_y}{\iint I^2(x,y)W(x-u,y-v)d_x d_y}\right]^2 \times \iint I^2(x,y)\nabla W(x-u,y-v)d_x d_y$$

Since C2 (u, v) € [0, 1] we don't expect ▼C^2 (u, v) to be huge so long as the correlation surface is reasonably smooth, *i.e.* no discontinuities in I or M. As the amount of high-frequency content in M and/or I increases, we expect the correlation surface to become less smooth.

3.6 Simulation Results and Conclusion

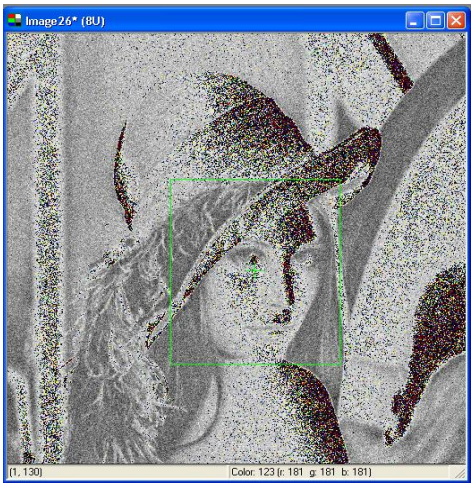
Results are also shown for real-world images and several monochrome images such as baboon, Lena where the target undergoes mild perspective distortion. In all cases the accept threshold is set to 0.75, representing a correlation score of 0.866 = √0.75. Except where

otherwise noted, the algorithm has been given the expected number of targets in advance. Results are given for search time *vs.* number and size of targets and false positive/negative results are also given.

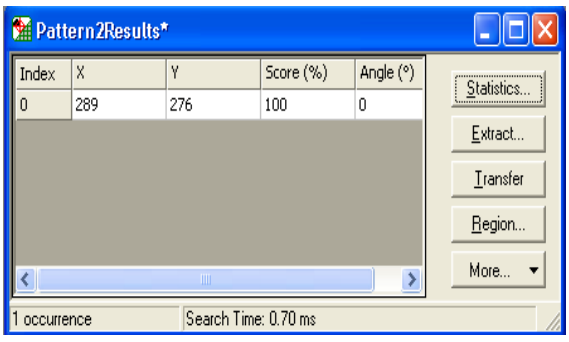


(a)

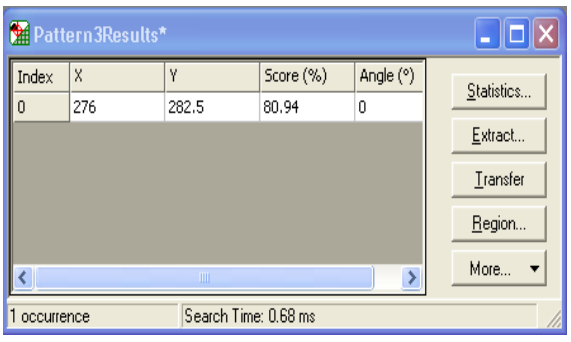
(b)



(c)

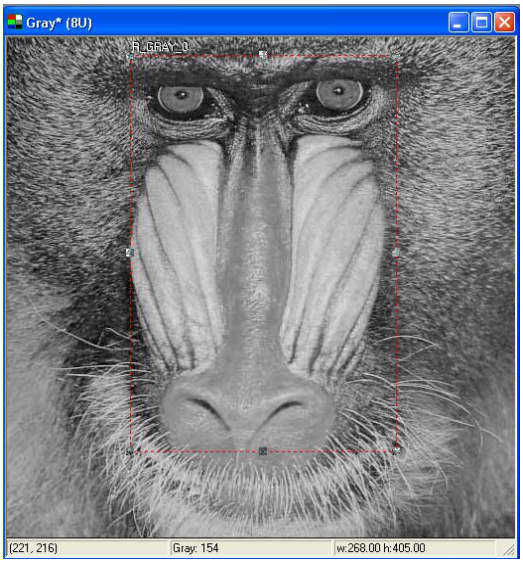


(d)

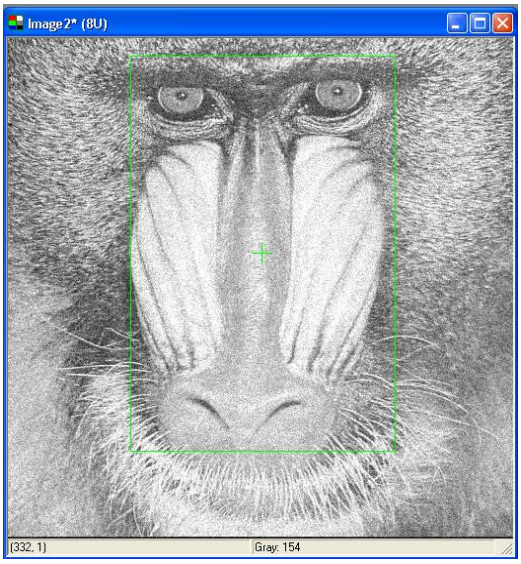


(e)

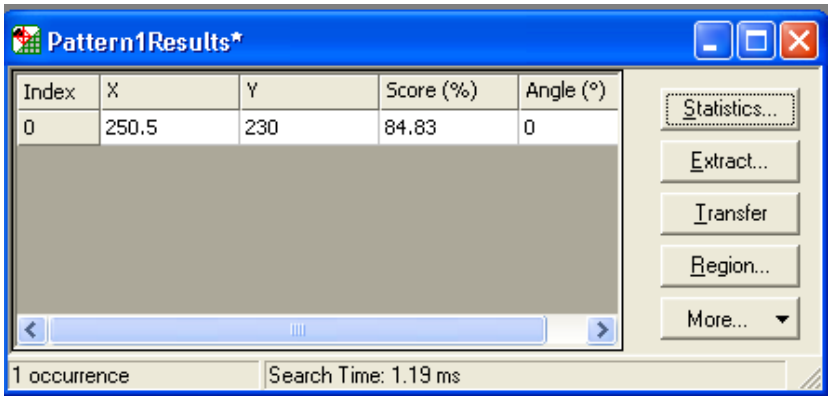
Fig 3.4.1: (a) model defined in Lena image, (b) model found in target, (c) model found in noisy target, (d) (e) Table showing score, hotspot, etc of result



(a)

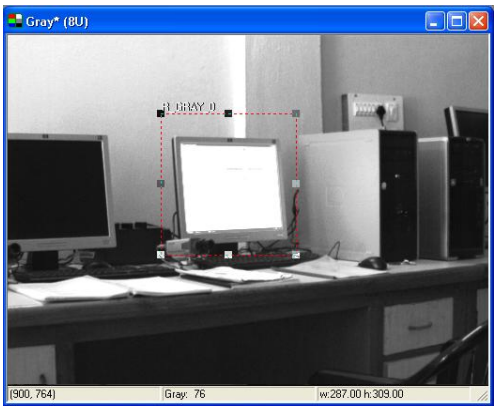


(b)

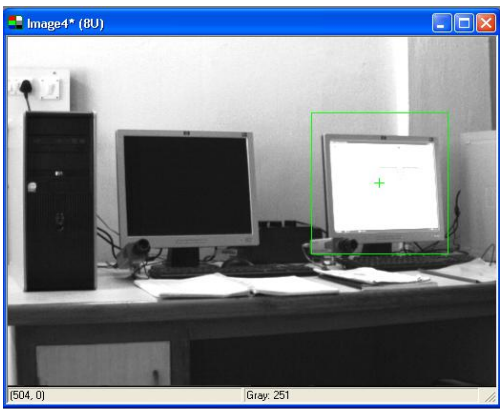


(c)

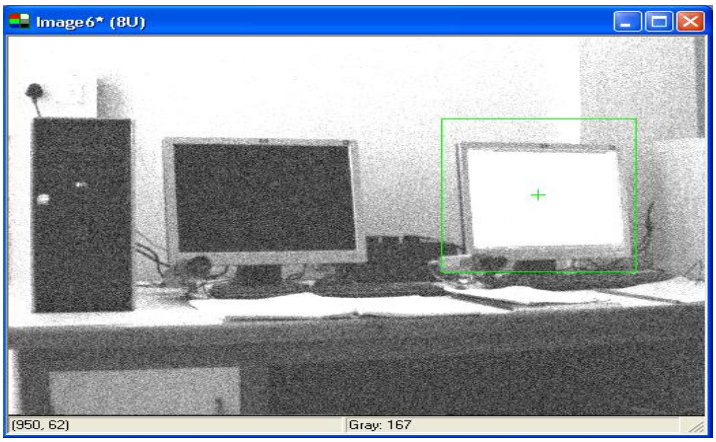
Fig 3.4.2: (a) model defined in Baboon image, (b) model found in noisy target, (c) Table showing score, hotspot, etc of result



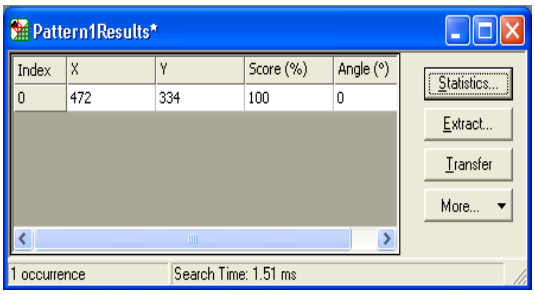
(a)



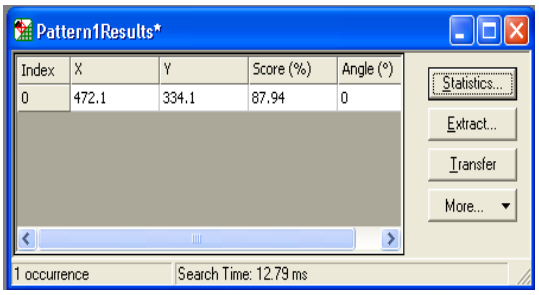
(b)



(c)



(d)



(e)

Fig 3.4.3: (a) model defined in real time image, (b) model found in target, (c) model found in noisy target, (d) (e) Table showing score, hotspot, etc of result

The algorithm[1] is also very fast even on modest hardware, making it attractive for machine vision applications in industry. Our method has good performance in finding targets even in the presence of small amounts of rotation and scale change. Further, we expect our method to better detect subtle differences in target instances as it does a pixel-by-pixel comparison. Further, searching for slightly re-scaled versions of the target across pyramid levels would allow our method[1] to find targets across a range of scales. Three main contributions are made: (1) incorporation of NGC search into a multi-scale image representation,(2) use of an estimate of the correlation gradient to perform steepest descent search at each level of the pyramid, and (3) a method for choosing the appropriate pyramid depth of the model using a worst-case analysis.

The result is a fast and robust method for localizing target instances within images. Further, since it is based on correlation search, the technique is simple and easily combined with PCA techniques for target matching. The algorithm is limited in that it does not attempt to deal with variable size or orientation of target instances, although it is shown to be robust to small variations in either parameter. The level of robustness is dependent on the actual pattern to be searched for, but this can be included in the worst-case analysis.

Chapter 4

Model Finder

Matrox Inspector Geometric Model Finder allows finding patterns, or models, based on geometric features. The algorithm finds models using edge-based geometric features instead of a pixel-to-pixel correlation. As such, Geometric Model Finder offers several advantages over correlation pattern matching, including greater tolerance of lighting variations (including specular reflection), model occlusion, as well as variations in scale and angle. Model Finder allows tailoring the search to fit the requirements of specific application. The search can be done for any number of different models simultaneously, through a range of angles and scale. Model Finder also provides complete support for calibration. Searches can be performed in the calibrated real-world such that, even without physically correcting the images, occurrences can be found even in the presence of complex distortions, and results returned in real-world units.

4.1 Basic concepts

The basic concepts and vocabulary conventions for Model Finder are:

Edges: Transitions in grayscale value over several adjacent pixels. Well-defined edges have sharp transitions in value. The smoother the image, the more gradual the change, and the weaker the edge

Active edges: Edges which are extracted from the model source image to compose the geometric model, and searched for in the target image.

Model: The pattern of active edges to find in the target image.

Occurrence: An instance of the model found in the target image.

Bounding-box: The boundary of the square or rectangular region which defines the height and width of the model or occurrence.

Model source image: The image from which to extract the model's active edges. In Matrox Inspector, a model can be defined from any 1-band, 8-bit unsigned image.

Model image: The image extracted from the source region in the model source image which is used as the model.

Model Finder context: The container for all models you want to find. The Model Finder context allows you to set global search settings for all the models contained within the context.

Mask: Used to define irrelevant, inconsistent, or featureless areas in the model, so that only the pertinent model details are used for the search.

Pre-processing: Extracts the active edges from the model images contained within the Model Finder context and sets internal search settings so that future searches will be optimized for speed and robustness.

4.2 Guidelines for choosing models

While finding models based on geometric features are a robust, reliable technique, there are a few pitfalls to be aware when defining the models, so that it is easy to choose the best possible model.

- Make sure your images have enough contrast

Contrast is necessary for identifying edges in the model source and target image with sub-pixel accuracy. For this reason, it is recommended that models that contain only slow gradations in grayscale values are avoided.

- Avoid poor geometric models

Poor geometric models suffer from a lack of clearly defined geometric characteristics, or from geometric characteristics that do not distinguish themselves sufficiently from other image features. These models can produce unreliable results.

Poor geometric model

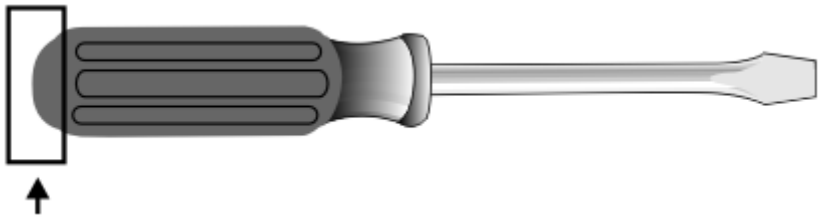
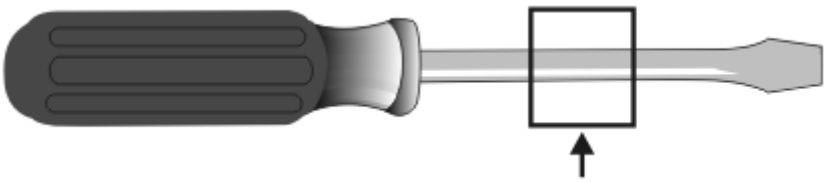


Fig 4.1 Simple curves lack distinguishing features and can produce false matches

- Be aware of ambiguous models

Certain types of geometric models provide non-unique, or ambiguous, results in terms of position, angle, or scale. Models that are ambiguous in position are usually composed of one or more sets of parallel lines only. Such models make it impossible to establish a unique position for them. A large number of matches can be found since the actual number of line segments in any particular line is theoretically limitless.

Model ambiguous in position



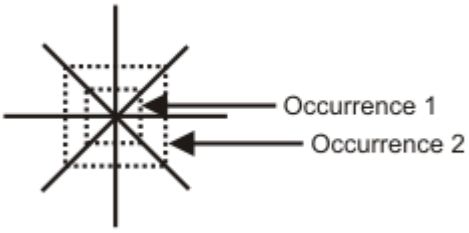
Model consisting of sets of parallel lines, without any

Fig 4.2 Distinguishing features are ambiguous in position.

Models that are ambiguous in scale are usually composed of only straight lines that pass through the same point; some spirals are also ambiguous in scale. Models that consist of small portions of objects should be tested to verify that they are not ambiguous in scale.

Some spirals can be

Ambiguous in scale



straight lines that pass only through the same points are ambiguous in scale

For example, a model of an isolated corner is ambiguous in terms of scale because it consists of only two straight lines that pass through the same point.

This is an ambiguous model for searching through a range of scale.

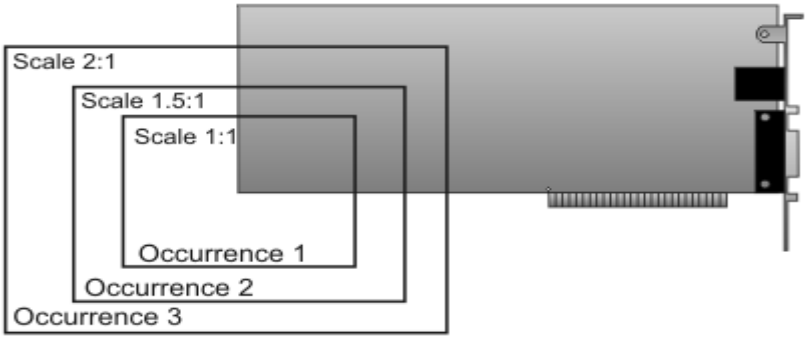
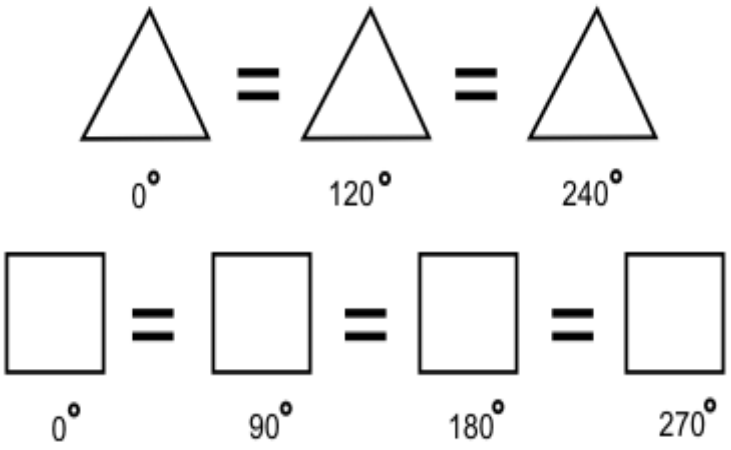


Fig 4.3 Distinguishing features are ambiguous in scale.

Symmetric models are often ambiguous in angle due to their similarity in features. For example, circles are completely ambiguous in terms of angle. Other simple symmetric models, such as squares and triangles, are ambiguous with respect to certain angles:



- Nearly ambiguous models

When the major part of a model contains ambiguous features, false matches can occur because the percentage of the occurrence's edges involved in the ambiguous features is great enough to be considered a match.

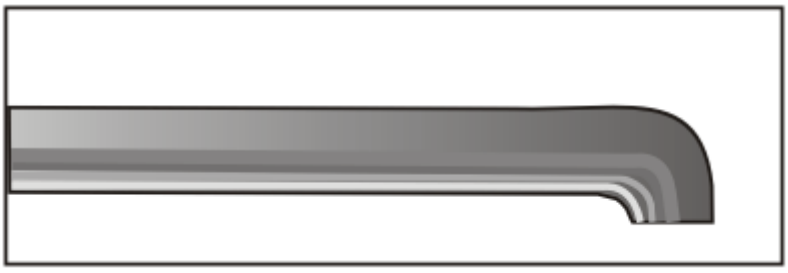


Fig 4.4 Nearly ambiguous models

To avoid this, make sure that the models have enough distinct features to be found among other target image features. This will ensure that only correct matches are returned as results. For example, the model below can produce false matches since the greater proportion of active edges in the model is composed of parallel straight lines rather than distinguishing curves.

4.3 Determining what is a match

A match occurs when the scores meet or exceed the acceptance value that is settled. The score and the target score are the primary factors in determining which occurrences are considered matches with the models in the Model Finder context.

The score is a measure of the active edges in the model found in the occurrence, weighted by the deviation in of these common edges. If a weight mask is used, edges are also weighted according to the weight mask.

The target score is a measure of edges found in the occurrence that are not present in the original model (that is, extra edges), weighted by the deviation in position of the common edges. Edges found in the occurrence that are not present in the model will reduce the target score. These scores are calculated as follows:

Score = Model coverage x (1 - Fit error weighting factor x Normalized fit error)
Target Score = Target coverage x (1 - Fit error weighting factor x Normalized fit error)

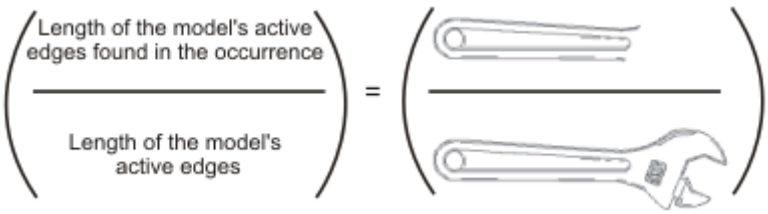
Note: the normalized fit error is the fit error converted to a number between 0.0-1.0

The model coverage, target coverage, and fit error components of the score and target score are explained below:

Model coverage. The model coverage is the percentage of the total length of the model's active edges found in the occurrence. 100% indicates that for every edge in the model, a corresponding edge was found in the occurrence.

Target coverage. The target coverage is the percentage of the total length of the model's active edges found in the occurrence, divided by the total length of edges present in the occurrence's bounding box. Thus, a target coverage score of 100% means that no extra edges were found. Lower scores indicate that features or edges found in the target (result occurrence) are not present in the model.

Model coverage(x100 for percentage)



Target coverage(x100 for percentage)

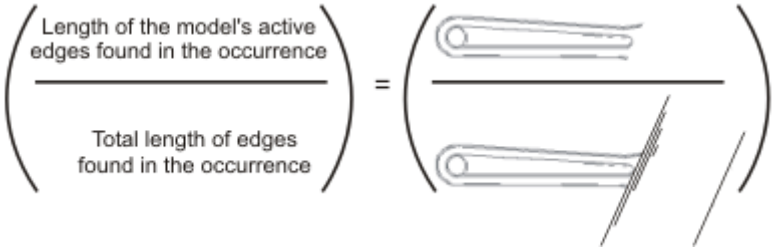
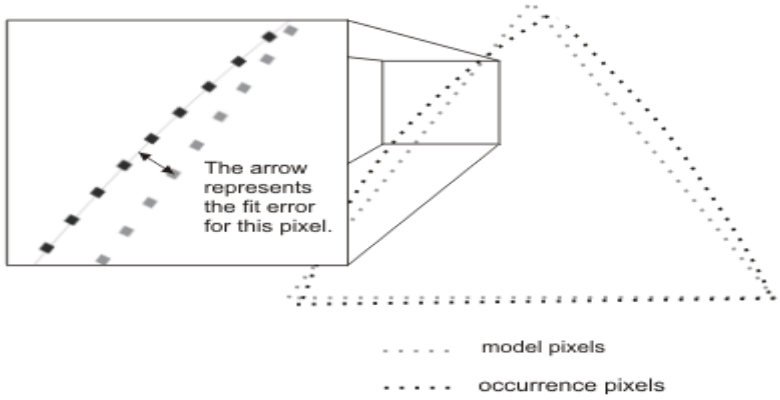


Fig 4.5 model and target coverage

Fit Error. The fit error is a measure of how well the edges of the occurrence correspond to those of the model. The fit error is calculated as the average quadratic distance (with sub-pixel accuracy) between the edges in the occurrence and the corresponding active edges in the model:

$$fit\ error = \frac{\sum_{\substack{all\ common \\ pixels}} [(error\ in\ X)^2 + (error\ in\ Y)^2]}{number\ of\ common\ pixels}$$

A perfect fit gives a fit error of 0.0. The fit error weighting factor (between 0.0 - 100.0) determines the importance to place on the fit error when calculating the score and target score. The fit error weighting factor is set using the *Model Advanced* tab of the *Model Finder* dialog box (default is 25.0).



4.4 Position, angle, and scale

The position, angle, and scale can be controlled when **Model Finder** searches for each model in a target. The position, angle, and scale can be restricted at which model occurrences can be found, to an expected (nominal) position, angle, or scale or to a given range.

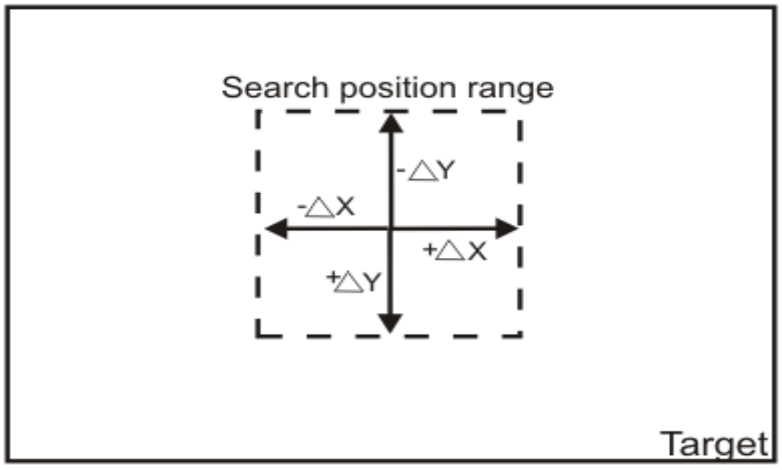
- Enabling calculations specific to searching within a range

Depending on whether searching for models within a range of positions, angles, and/or scales, *Model Finder* uses different search strategies to evaluate the edge-based features of the target candidates. Typically, to search for models within a range, calculations specific to the corresponding search strategy should be enabled for the context (*Search Position Range*, *Search Angle Range* and/or *Search Scale Range*). On the *Context General* tab, the ***Search*** *Position Range* and *Search Angle Range* search strategies are enabled by default, whereas the *Search Scale Range* is disabled by default.

4.4.1 Search position and search position range

Each model defined in a Model Finder context can be searched at a specific position, or within a position range. The position range limits the region in which the position of a model occurrence can be found; coordinates which fall outside of this region cannot be returned as results. Note that the position returned for an occurrence is determined by the model's reference axis origin; by default, this position is set to the center of the model, however it can be displaced if necessary.

The region defined by the default position range is the entire image plane (Whole World), meaning that all coordinates (even outside the target if applicable) can be returned as results. To specify a user-defined position range set the *Search Position Range* to *User defined* in the Model Search tab. Then, set the X/Y Position field to specify the nominal search position. Use *the X D.Neg, Y D.Neg, X D.Pos,* and *Y D.Pos* fields to set the position range relative to the nominal position.



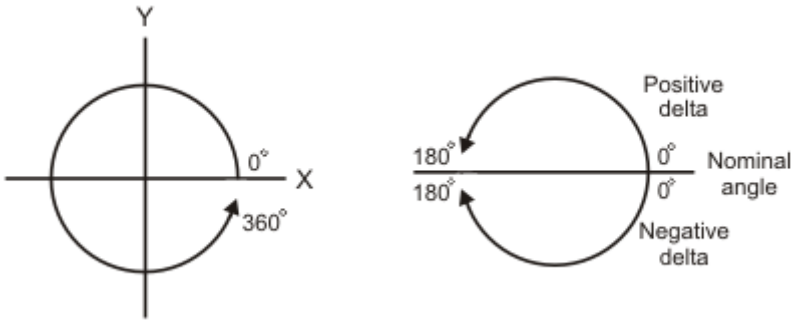
Depending on the number of details present in the target, using a small position range generally decreases the search time. Always set the search position range to the minimum required when speed is a consideration; the *X D.Neg*, *Y D.Neg*, *X D.Pos*, *Y D.Pos* fields in the *Search Position Range* area in the *Model Search* tab can be greater or equal to zero. If they are all set to zero, the occurrence must be at the position specified by the nominal position. Note that it is possible to specify a position range which defines a region partially, or totally, outside of the target; this might be necessary, depending on the reference axis origin of the model.

4.4.2 Angle and angular range

Each model defined in a Model Finder context can be specified at a specific angle, or within an angular range. For each model in the context, it is possible to specify the angle of the search, using the *Angle* field in the *General* tab. By default, the search angle is 0 ^{o}. It is possible to search within the full angular range of 360o from the nominal angle specified with the *Angle* field. Use the *Delta Positive* and *Delta Negative* control types to specify the angular range in the counter-clockwise and clockwise direction from the nominal angle, respectively; the default for both is 180o. The angular range limits the possible angles which can be returned as results for an occurrence. Note that the actual angle of the occurrence does not

affect search speed. If needed to search for a model at discrete angles only (for example, at intervals of 90 degrees), it is typically more efficient to define several models with different expected angles, than to search through the full angular range.

Angle convention in inspector Delta convention in inspector



4.4.3 Scale and scale range

The scale of the model establishes the size of the occurrence that expected to find in the target. If the expected occurrence is smaller or larger than that of the model, set the nominal scale of the occurrence for each individual model, using the *Scale* edit field. The supported scale factors are 0.5 to 2.0. When the scale of occurrences can vary around the specified nominal scale, specify a range of scales, using the ***Max*** (1.0 to 2.0) and ***Min*** (0.5 to 1.0) fields in the *Search Scale Range* area on the Model's *General* tab.

The minimum factor and the maximum factor together determine the scale range from the nominal scale (*Scale*).

The maximum and minimum factors are applied to the *Scale* setting as follows:

Maximum scale = (*Scale*) x (*Max.*)

Minimum scale = (*Scale*) x (*Min.*)

When calculations specific to scale-range search strategies are enabled, the scale range should be used to cover an expected variance in the scale; should not use the scale range to cover different expected scales at different positions. In this case, it is typically more efficient to define several models with different expected scales. This is because a large scale range could potentially slow down the operation; as well, unwanted occurrences could be found.

By default, calculations specific to scale-range search strategies are disabled. When disabled, must specify a good nominal scale for each model, which is within the model's scale range. Note that occurrences can still be found within the scale range specified for their model.

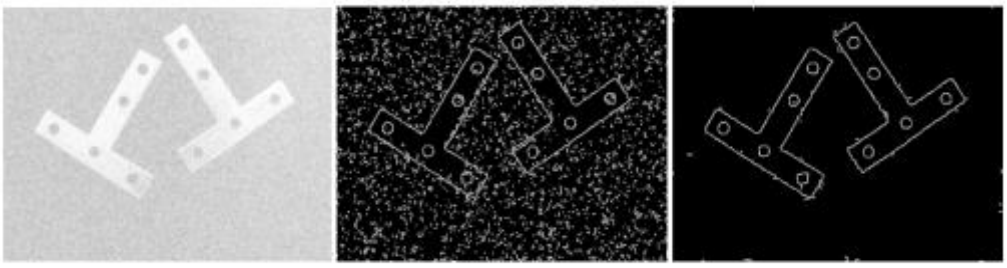
4.5 Context edge settings

Model Finder uses custom image processing algorithms to preprocess the image in order to simultaneously extract active edges and improve model source and target images by smoothing and reducing noise. The *Smoothness* and *Detail level* settings in the context's *Edge* tab of the *Model Finder* dialog box control these image processing algorithms, determining which active edges are extracted from the model source and target images.

4.5.1 Extracting edges

The edge extraction process involves a Denoising operation to even out rough edges and remove noise. The degree of smoothness (strength) of the Denoising operation used for all models in the context (using the *Edge* tab with the *Smoothness* field in the *Filter Control* area) can be controlled. The range of this control type varies from 0.0 to 100.0; a value of 100.0 results in a strong noise reduction effect, while a value of 0.0 has almost no noise reduction effect. The default setting is 50.0.

These settings only affect image type models and the target image.



Target image with Considerable noise

Edge map obtained with smoothness set to 50

Edge map obtained with smoothness set to 70

Note that using a very high smoothing level can result in a loss of important detail and a decrease in precision.

The detail level setting determines what is considered an edge. Edges are defined by the transition in grayscale value between adjacent pixels. This setting can be controlled for all models in the context, using the *Detail Level* field .The default setting (Medium) offers a robust detection of edges from images with contrast variation, noise, and non-uniform illumination. Nevertheless, in cases where objects of interest have a very low-contrast compared to high-contrast areas in the image, some low-contrast edges can be missed.

The following examples show the effect of *Detail level* setting.

Low contrast image

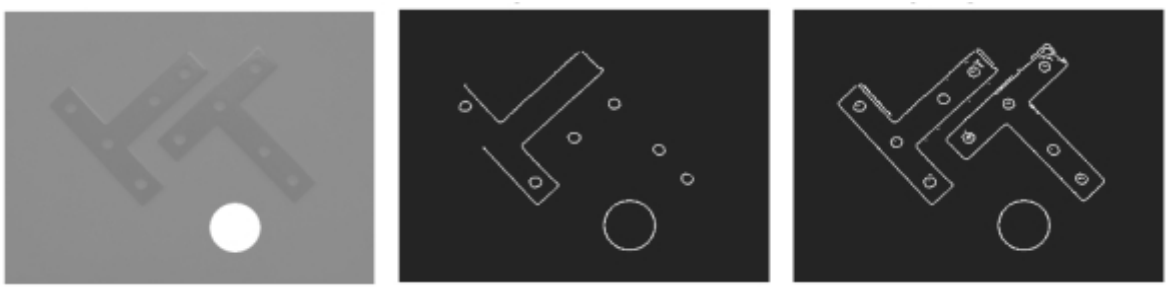
resulting edge map using Medium

Multi contrasted image

resulting edge map

resulting edge map

Using Medium using High



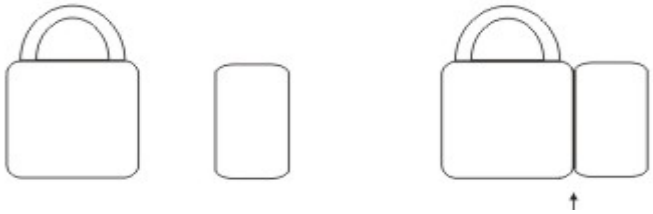
If the images contain low-contrast and high-contrast objects, a detail level setting of *High* should be used to ensure the detection of all important edges in the image. The *Very High* setting performs an exhaustive edge extraction, including very low contrast edges. However, it should be noted that this setting is very sensitive to noise. The Smoothness and *Detail level* settings are applied to all the model and target images for the specified Model Finder context. Note that model and target images are not directly modified; these settings merely extract the edge-based information from the images.

Generally, the defaults settings for *Smoothness* and Detail level are sufficient for the majority of images; when dealing with very noisy images it should be adjusted, extremely low or multi-contrasted images, or images with very thin, refined features. In such cases, it is recommended that try with different settings to achieve the necessary level of accuracy and speed required by your application. In the special case when special hardware is available to perform the convolution, it might be faster to use a kernel Finite Impulse Response (FIR) implementation of the smooth filter rather than the default recursive Infinite Impulse Response (IIR) implementation and the displayed default value for the filter mode will change automatically.

4.5.2 Shared edges

Edges that can be part of more than one occurrence are considered part of the occurrence with the greatest score. For example, in the illustration below, two occurrences of two simple models share a common edge. With shared edges enabled, these occurrences would have perfect scores.

Model A Model B occurrence 1 occurrence 2



Shared edges

However, with shared edges disabled (default), the shared edge would be considered part of occurrence, since it has the greater score; the score of occurrence 2 would be subsequently reduced by the loss of the shared edge in the score calculation.

4.6 Thresholding

Thresholding is the simplest method of image segmentation. From a grayscale image, thresholding can be used to create binary images. During the thresholding process, individual pixels in an image are marked as “object” pixels if their value is greater than some threshold value (assuming an object to be brighter than the background) and as “background” pixels otherwise. This convention is known as threshold above. Variants include threshold below, which is opposite of threshold above; threshold inside, where a pixel is labeled "object" if its value is between two thresholds; and threshold outside, which is the opposite of

threshold inside. Typically, an object pixel is given a value of "1" while a background pixel is given a value of "0." Finally, a binary image is created by coloring each pixel white or black, depending on a pixel's label.

Thresholding operations are either global or local and point- or region- dependent. Global thresholding algorithms choose one threshold for the entire image while local thresholding algorithms partition the image in to sub images and select a threshold for each sub image. Point dependent thresholding algorithms only analyze the gray level distribution of the image while region dependent algorithms also consider the location of the pixels.

Fig 4.6 .Example of a threshold effect used on an image

4.6.1 Adaptive thresholding

Thresholding is called adaptive thresholding [6] when a different threshold is used for different regions in the image. This may also be known as *local* or *dynamic* thresholding. The

key parameter in the thresholding process is the choice of the threshold value. Several different methods for choosing a threshold exist; users can manually choose a threshold value, or a thresholding algorithm can compute a value automatically, which is known as *automatic thresholding*. A simple method would be to choose the *mean* or *median* value, the rationale being that if the object pixels are brighter than the background, they should also be brighter than the average. In a noiseless image with uniform background and object values, the mean or median will work well as the threshold, however, this will generally not be the case. A more sophisticated approach might be to create a *histogram* of the image pixel intensities and use the valley point as the threshold. The histogram approach assumes that there is some average value for the background and object pixels, but that the actual pixel values have some variation around these average values. However, this may be computationally expensive, and image histograms may not have clearly defined valley points, often making the selection of an accurate threshold difficult. One method that is relatively simple, does not require much specific knowledge of the image, and is robust against image noise, is the following iterative method:

1. An initial threshold (T) is chosen; this can be done randomly or according to any other method desired.
2. The image is segmented into object and background pixels as described above, creating two sets:
    1. G_1 = {f(m,n):f(m,n)>T} (object pixels)
    2. G_2 = {f(m,n):f(m,n)$\leq$T} (background pixels) (note, f(m,n) is the value of the pixel located in the m^{th} column, n^{th} row)
3. The average of each set is computed.
    1. m_1 = average value of G_1
    2. m_2 = average value of G_2

4. A new threshold is created that is the average of m_1 and m_2
    1. T' = $(m_1 + m_2)/2$
5. Go back to step two, now using the new threshold computed in step four, keep repeating until the new threshold matches the one before it

This iterative algorithm is a special one-dimensional case of the k-means clustering algorithm, which has been proven to converge at a *local* minimum—meaning that a different initial threshold may give a different final result. Color images can also be threshold. One approach is to designate a separate threshold for each of the RGB components of the image and then combine them with an AND operation.

4.7 Sobel operator

The Sobel operator [5] is used in image processing, particularly within edge detection algorithms. Technically, it is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Sobel operator is either the corresponding gradient vector or the norm of this vector. The Sobel operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical direction and is therefore relatively inexpensive in terms of computations. On the other hand, the gradient approximation which it produces is relatively crude, in particular for high frequency variations in the image.

In simple terms, the operator calculates the gradient of the image intensity at each point, giving the direction of the largest possible increase from light to dark and the rate of change in that direction. The result therefore shows how "abruptly" or "smoothly" the image changes at that point and therefore how likely it is that that part of the image represents an *edge*, as well as how that edge is likely to be oriented. In practice, the magnitude (likelihood of an

edge) calculation is more reliable and easier to interpret than the direction calculation. Mathematically, the gradient of a two-variable function (here the image intensity function) is at each image point a 2D vector with the components given by the derivatives in the horizontal and vertical directions. At each image point, the gradient vector points in the direction of largest possible intensity increase, and the length of the gradient vector corresponds to the rate of change in that direction. This implies that the result of the Sobel operator at an image point which is in a region of constant image intensity is a zero vector and at a point on an edge is a vector which points across the edge, from darker to brighter values.

Mathematically, the operator uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives - one for horizontal changes, and one for vertical. If we define **A** as the source image, and $\mathbf{G}_x$ and $\mathbf{G}_y$ are two images which at each point contain the horizontal and vertical derivative approximations, the computations are as follows:

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * A$$

$$G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A$$

Where * here denotes the 2-dimensional convolution operation.

The *x*-coordinate is here defined as increasing in the "right"-direction, and the *y*-coordinate is defined as increasing in the "down"-direction. At each point in the image, the resulting gradient approximations can be combined to give the gradient magnitude, using:

$$G = \sqrt{{G_x}^2 + {G_y}^2}$$

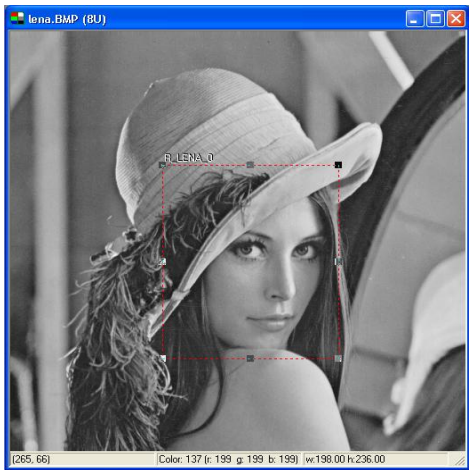
Using this information, we can also calculate the gradient's direction:

$$\theta = \arctan\left(\frac{Gy}{Gx}\right)$$

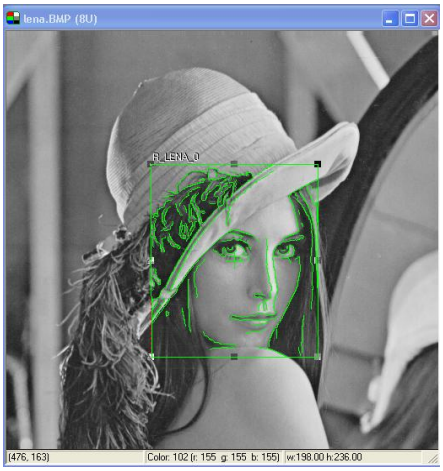
Since the intensity function of a digital image is only known at discrete points, derivatives of this function cannot be defined unless we assume that there is an underlying continuous intensity function which has been sampled at the image points. With some additional assumptions, the derivative of the continuous intensity function can be computed as a function on the sampled intensity function, i.e. the digital image. It turns out that the derivatives at any particular point are functions of the intensity values at virtually all image points. However, approximations of these derivative functions can be defined at lesser or larger degrees of accuracy. The Sobel operator represents a rather inaccurate approximation of the image gradient, but is still of sufficient quality to be of practical use in many applications. More precisely, it uses intensity values only in a 3×3 region around each image point to approximate the corresponding image gradient, and it uses only integer values for the coefficients which weight the image intensities to produce the gradient approximation.

4.8 Simulation Results and Conclusion

Results are also shown for real-world images and several monochrome images such as baboon, where the target undergoes mild perspective distortion. Results are given for search time *vs.* number and size of targets and false positive/negative results are also given.


lena.BMP (8U)
R_LENA_0
(265, 66)
Color: 137 (r: 199 g: 199 b: 199) w:198.00 h:236.00

(a)


lena.BMP (8U)
R_LENA_0
(476, 163)
Color: 102 (r: 155 g: 155 b: 155) w:198.00 h:236.00

(b)

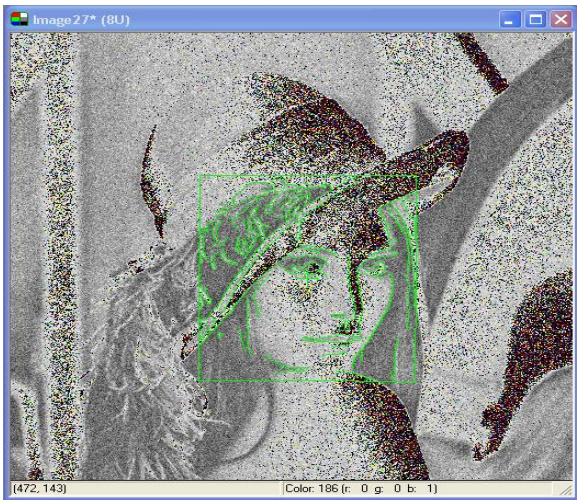
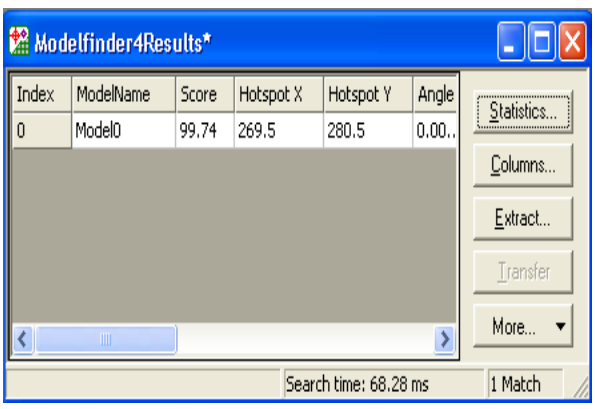
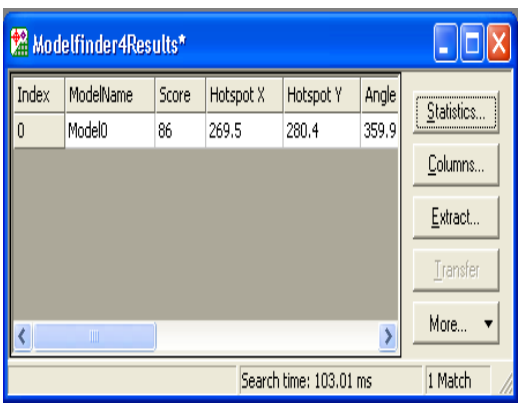


Image27* (8U)
(472, 143)
Color: 186 (r: 0 g: 0 b: 1)

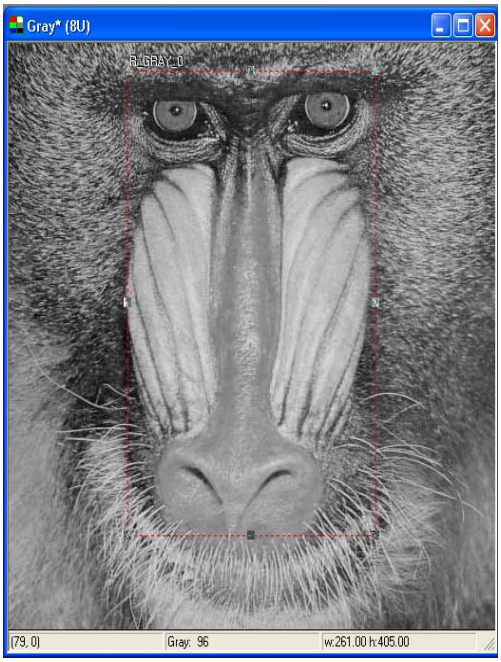
(c)

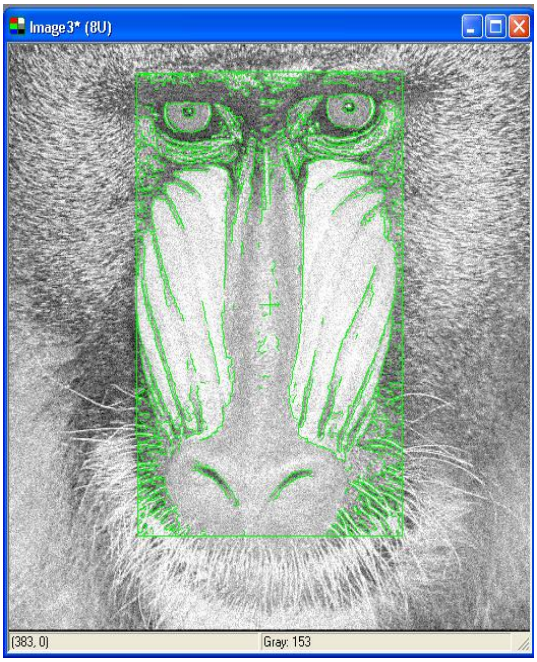

Modelfinder4Results*
Index
ModelName
Score
Hotspot X
Hotspot Y
Angle
0
Model0
99.74
269.5
280.5
0.00..
Statistics...
Columns...
Extract...
Transfer
More...
Search time: 68.28 ms
1 Match


Modelfinder4Results*
Index
ModelName
Score
Hotspot X
Hotspot Y
Angle
0
Model0
86
269.5
280.4
359.9
Statistics...
Columns...
Extract...
Transfer
More...
Search time: 103.01 ms
1 Match

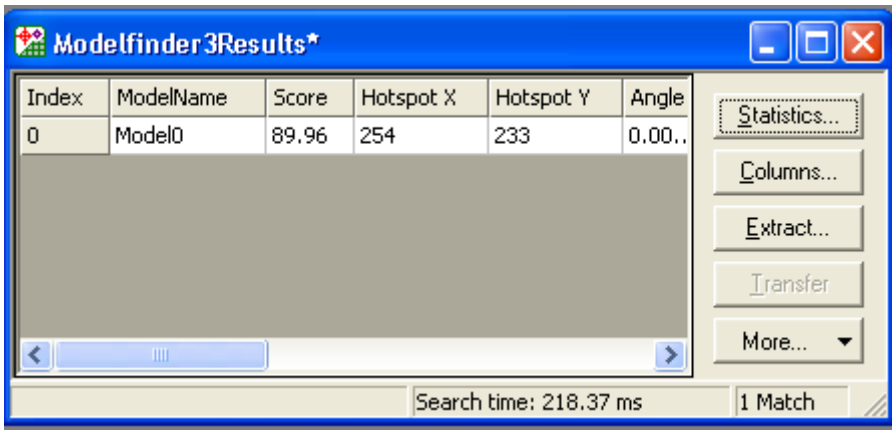
(d) **(e)**

Fig 4.7.1: (a) model defined in Lena image, (b) model found in target, (c) model found in noisy target, (d) (e) Table showing score, hotspot, etc of result



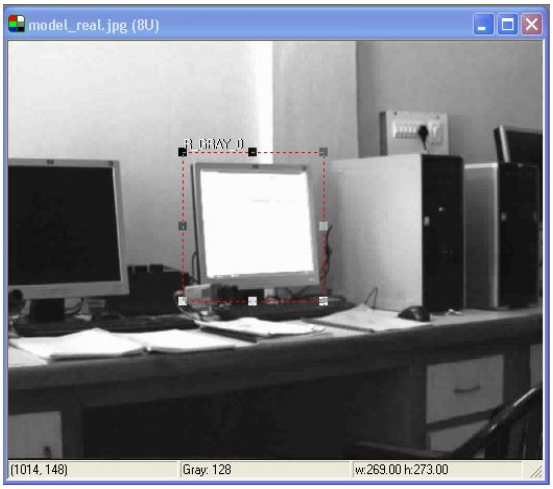


(a) **(b)**

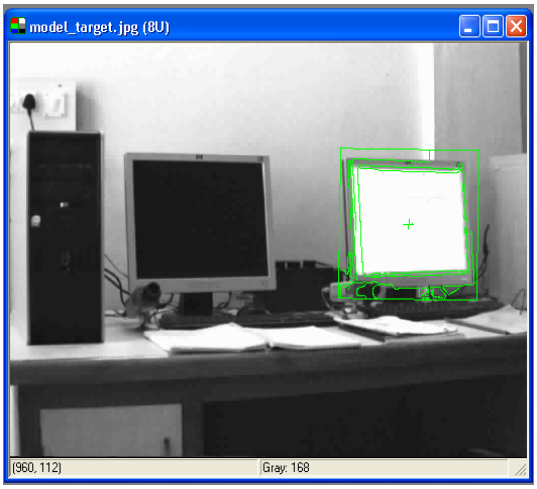


(c)

Fig 4.7.2: (a) model defined in Baboon image, (b) model found in noisy target, (c) Table showing score, hotspot, etc of result


model_real.jpg (8U)
(1014, 148)
Gray: 128
w:269.00 h:273.00

(a)


model_target.jpg (8U)
(960, 112)
Gray: 168

(b)

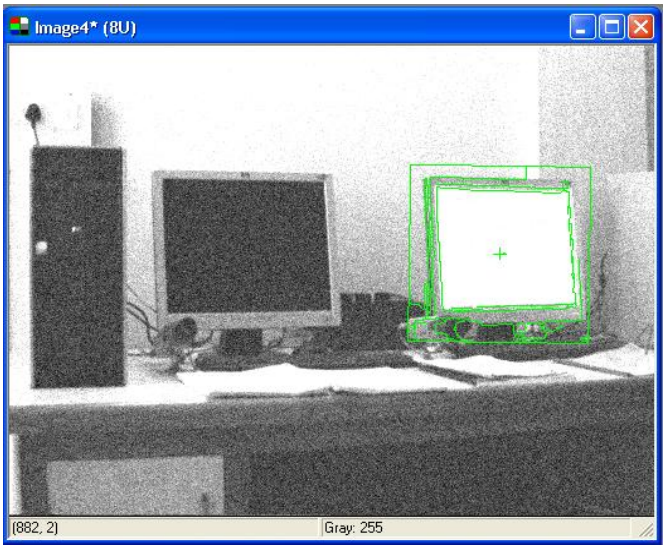
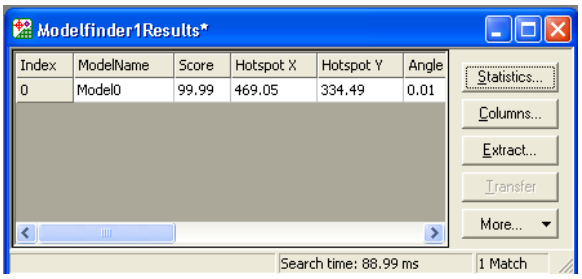
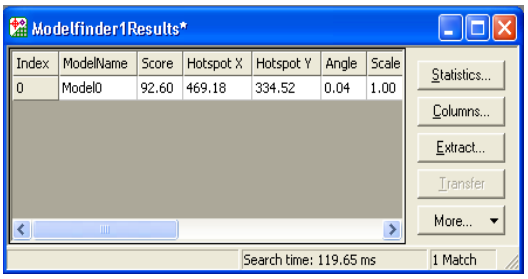


Image4* (8U)
(882, 2)
Gray: 255

(c)


Modelfinder1Results*
Index
ModelName
Score
Hotspot X
Hotspot Y
Angle
0
Model0
99.99
469.05
334.49
0.01
Statistics...
Columns...
Extract...
Transfer
More...
Search time: 88.99 ms
1 Match

(d)


Modelfinder1Results*
Index
ModelName
Score
Hotspot X
Hotspot Y
Angle
Scale
0
Model0
92.60
469.18
334.52
0.04
1.00
Statistics...
Columns...
Extract...
Transfer
More...
Search time: 119.65 ms
1 Match

(e)

Fig 4.7.3: (a) model defined in real time image, (b) model found in target, (c) model found in noisy target, (d) (e) Table showing score, hotspot, etc of result

Edge detection is an important issue in image processing. Most common and earliest edge detection algorithms are those based on gradient, such as Sobel operator and the Roberts operator. The characteristics of Sobel operator, regularity, simplicity and efficiency, makes it adequate for the implementation in application specific architectures such as Matrox imaging. Gradient-based algorithms such as the Sobel operator have a major drawback of being very sensitive to noise. The size of the kernel filter and coefficients are fixed and cannot be adapted to a given image.

Chapter 5

COMPARISION AND DISCUSSION

5.1 Comparative Study

In this work the performance of the normalized gray scale techniques has been compared with existing methods available in literature. This algorithm is very fast in detection and localization of patterns with simple and inexpensive hardware. Typical search times are less than 0.25 sec, and as low as 10-30 mS when the expected number of targets are known in advance. Only two methods, those of Lowe [18] and Schilele & Pentland [32] cite similar speeds. In comparing our algorithm to that of Lowe, we note that it is more complex to implement than our algorithm. The normalized algorithm is based on NGC, is robust over wide range of global illumination changes. Comparing to SIFT, it has rotation and scale invariance while our method does not. The feature based algorithm used in the second part has rotation and scale invariance.

5.2 Discussion

The normalized gray scale algorithm [1] is also very fast even on modest hardware, making it attractive for machine vision applications in industry. Our method has good performance in finding targets even in the presence of small amounts of rotation and scale change. The choice of accept threshold will have a strong effect on the result. As this threshold is reduced we would expect to find a large number of false positives. It must be remembered that lowering the accept threshold also leads to a large number of candidate matches at the top level of the pyramid, and this in turn will generally lengthen the search time. The feature based algorithm used in the second part has rotation and scale invariance.

Chapter 6

CONCLUSIONS AND SCOPE OF FUTURE WORK

6.1 Conclusion

This work describes a novel approach to pattern matching based on normalized correlation and Sobel based edge detection in a pyramid image representation. The result is a fast and robust method for localizing target instances within images. Further, since it is based on correlation search, the technique is simple and easily combined with PCA techniques for target matching. The normalized algorithm[1] is limited in that it does not attempt to deal with variable size or orientation of target instances, although it is shown to be robust to small variations in either parameter. But the feature based algorithm used in the second part has rotation and scale invariance.

6.2 Future Scope

In this thesis work, two pattern recognition algorithms, NGC and feature based algorithms are implemented. Future work includes further investigation of size, and even orientation, invariance in the search framework. While pattern recognition has been a field of research for approximately fifty years; there still remains a lot of future research. One important application of pattern recognition, which will be used towards security in the future, is person recognition, which is still in its infancy.

- Mathematical morphology is used for image processing and analysis, due to its basic concept of set theory, has an inherent advantage for image processing. It can perform tasks from simplest to the most demanding: noise reduction, edge detection, segmentation, texture and shape analysis, but also it can be applied to almost all applications dealing with digital image processing.
- Most fuzzy rule based image segmentation techniques to date have been primarily developed for gray level images. By combining with wavelet transform, fuzzy can be

used for multiscale feature detection. Much more researches are expected in the direction of fuzzy rule based feature extraction in future.

- Discrete Wavelet Transform is a very good tool for image processing. Image textures are classified based on wavelet transformation and singular value decomposition. This technique achieves higher recognition rates compared to the traditional sub band energy based approach.
- There is considerable interest in developing new parallel architectures based on neural network designs which can be applied to practical classification tasks.

REFERENCES:

[1] W. James Maclean and John K Tsotsos, Fast pattern recognition using Normalized Grey scale Correlation in a pyramid image representation, *In IEEE Conference on Machine Vision & applications*, February 16, 2007.

[2] J. P. Lewis, Fast normalized cross correlation, *In IEEE Conference on Vision Interface*, 15 October 2005.

[3] Swami Manickam, Scott D Roth and Thomas Bushman, Intelligent and Optimal normalized correlation for high speed pattern matching, Data cube Inc, Rosewood drive, MA01923, USA.

[4] Abbas M Al-Ghaili, Syamsiah Mashohor, Alyani Ismail and Abdul Rahman Ramli, A New Vertical Edge Detection Algorithm and its applications, *In* IEEE *Conference on Computer Engineering & Systems*, Nov 2008.

[5] Nick Kanopoloulos, Nagesh Vasanthavads and Robert L. Baker, Design of an image edge detection filter using the Sobel operator, *IEEE journal of solid state circuits, vol 23,no.2, APRIL1988.*

[6] Rishi R, Rakesh, Probal Chaudhuri and C. A. Murthy, Thresholding in Edge Detection: A Statistical Approach, *IEEE transactions on image processing, vol. 13, no. 7, July 2004.*

[7] Rafael C. Gonzalez and Paul Wintz, "*Digital Image Processing*", Addison-Wesley Publishing Company, Reading, Massachusetts, 2nd edition, 1987

[8] Ardeshir Goshtasby, Template matching in rotated images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(3):338–344, May 1985.

[9] Franc ois Ennesser and Gerard Medioni. Finding Waldo, or focus of attention using local colour information, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):805–809, August 1995.

[10] A. Baumberg, Reliable feature matching across widely separated views, In *Conference on Computer Vision & Pattern Recognition*, pages 774–781, 2000

[11] Chris Harris and Mike Stephens, A combined corner and edge detector, *In Proceedings Fourth Alvey Vision Conference*, pages 147–151, Manchester, United Kingdom, 1988.

[12] P. Burt, Attention mechanisms for vision in a dynamic world, *In Proceedings of the International Conference on Pattern Recognition*, pages 977–987, 1988.

[13] Michael A. Greenspan, Geometric probing of dense range data, *IEEE Transaction on Pattern Anal. Mach. Intell*, 24(4):495–508, 2002.

[14]Y. S. Huang, C. C. Chiang, J. W. Shieh, and Eric Grimson, Prototype optimization for nearest neighbour classification, *Pattern Recognition*, 35:1237–1245, 2002.

[15] Jean-Michel Jolion and Azriel Rosenfeld, *A pyramid framework for early vision*, Kluwer Academic Publishers, P.O. Box 17, 3300 AA Dordrecht, The Netherlands1994. ISBN: 0-7923-9402-X

[16] Tony Lindeberg, *Scale-space theory in computer vision*, Kluwer Academic Publishers P.O. Box 17, 3300 AA Dordrecht, The Netherlands, 1994. ISBN: 0-7923-9418-6.

[17] David G. Lowe, Object recognition from local scale-invariant features, *In Proceedings of the Seventh International Conference on Computer Vision*, pages 1150–1157, Kerkyra, Greece, 1999.

[18] Arun D. Kulkarni, *Artificial Neural Networks for Image Understanding*, Van Nostrand Reinhold, New York, 1994. ISBN 0-442-00921-6; LofC QA76.87.K84 1993.

[19] Anil K. Jain and Aditya Vailaya, Shape-based retrieval: A case study with trademark image databases, *Pattern Recognition*, 31(9):1369–1390, 1998.

[20] David G. Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*, 60:90–110, 2004.

[21] M. K. Hu, Visual pattern recognition by moment invariants, *IRE Trans. Information*

Theory, IT-8:179–187, 1962.

[22] Krystian Mikolajczyk and Cordelia Schmid, Indexing based on scale invariant interest points, *In IEEE International Conference on Computer Vision*, pages 525–531, 2001.

[23] Krystian Mikolajczyk and Cordelia Schmid, An affine invariant interest point detector, *In European Conference on Computer Vision*, volume 4, pages 128–142, 2002

[24] Krystian Mikolajczyk and Cordelia Schmid, Scale and affine invariant interest point detectors, *International Journal of Computer Vision*, 60(1):63–86, 2004.

[25] Chahab Nastar, Baback Moghaddam, and Alex Pentland, Flexible images: Matching and recognition using learned deformations, *Computer Vision & Image Understanding*65 (2):179–191, 1997.

[26] Alex Pentland, Rosalind W. Picard, and Stan Sclaroff, Photo book: Content based ma nipulation of image databases, *International Journal of Computer Vision*, 18(3):233–254, 1996.

[27] William K. Pratt, *Digital Image Processing*, John Wiley & Sons, Inc., New York, 2nd Edition, 1991.

[28] Bernt Schiele and Alex Pentland, Probabilistic object recognition and localization, *In Proceedings of the Seventh International Conference on Computer Vision*, pages 177–182, 1999.

[29] Stan Sclaroff, Marco La Cascia, and Saratendu Sethi, Unifying textual and visual cues for content-based image retrieval on the worldwide web, *Computer Vision & Image Understanding*, 75(1/2):86–98, 1999.

[30] Josef Sivic, Frederik Schaffalitzky, and Andrew Zisserman, Object level grouping for video shots, *In Proceedings of the European Conference on Computer Vision*, volume LNCS 3022, pages 85–98, 2004

[31] Chris Stauffer and Eric Grimson, Similarity templates for detection and recognition, *In*

IEEE Conference on Computer Vision & Pattern Recognition, Kauai, Hawaii, 2001.

[32] M. Turk and A. Pentland, Eigenfaces for recognition, *Journal of Cognitive Neuro science* , 3:71–86, 1991.

[33] G. vanderWal and P. Burt, A VLSI pyramid chip for multiresolution image analysis, *International Journal of Computer Vision*, 8:177–190, 1992.

[34] Harry Wechsler and George Lee Zimmerman, 2-D invariant object recognition using distributed associative memory, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):811–821, November 1988

[35] Harry Wechsler and George Lee Zimmerman, Distributed associative memory (dam) for bin-picking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*11 (8):814–822, November 1989

[36] J. K. Wu, C. P. Lam, B. M. Mehtre, Y. J. Gao, and A. Desai Narasimhalu, Content-based retrieval for trademark registration, *Multimedia Tools & Applications*, 3(3):245–2671996.

[37] Gustavo Carneiro and Allan D. Jepson, Multi-scale phase-based local features, *In pro ceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 736–743, Madison, WI, June 2003

[38] W. James MacLean and John K. Tsotsos, Fast pattern recognition using gradient descent search in an image pyramid, *In Proceedings of 15th Annual International Conference on Pattern Recognition*, volume 2, pages 877–881, Barcelona, Spain, September 2000.