MACHINE VISION FOR INDUSTRY

Dr. A. F. Ribeiro, Lic.Informática, MSc, PhD.
Departamento de Electrónica Industrial, Univ. do Minho, 4800 Guimarães, Portugal.
e-mail: fernando@dei.uminho.pt

ABSTRACT

Nowadays, industry is changing its way of working in order to get more competitive. Industry wants to get more and more automated in order to reduce production times, increase productivity, improve quality production, at a cheaper cost, to be less wasteful, with less need to have a skilled operative, to be more flexible meaning easy to implement changes and possible to leave the automated process working with little supervision around the clock.

Vision in Robotics helps controlling production in a relatively simple form, avoiding a skilled operative to spend his time 'watching' the machine doing his job. Moreover, the automatic inspection process does things faster and with improved quality than a human.

Robotics Vision and Image Processing tools are the most desired tools for quality control in industry. With the use of one (or more) cameras, and a computer controlling and analysing the extracted images, a software tool can solve a problem in a relatively easy way. An initial investment is needed to buy all the necessary Vision Hardware, but software can be built by using a few existing tools.

Instead of making a vision based program from scratch (re-inventing the wheel) to solve a specific problem, it is now possible to use existing image processing tools and build quickly and easily a software solution.

These tools work on grey-scale image processing level. These high-level vision software tools do not require that the developer program at the pixel level, which makes the technology accessible even to users with little machine vision experience. To reduce the amount of image to analyse, the user can work on 'regions of interest', reducing though the time and space to analyse/store the image. A description of the most important tools are described and its basic principle of functioning is explained. These tools can then be integrated and work together in order to make the full solution.

1. INTRODUCTION

Nowadays, the solution of a control system using Image Processing does not mean heavy and complex programming. Several software libraries exist which comprise tools that enable vision systems to solve several different industrial applications. No more need to programme at a bit level is necessary, allowing therefore faster development and more reliable algorithms. The main task is therefore to choose the best combination of image processing tools and organise them in the correct sequence to solve the application.

A typical example of an application is a system which locates an object within a noisy image and measures it area. First, the vision system has to capture the image containing the object and then use

the *Pattern Locating* tool to find the object. The region of interest is then filtered in order to reduce the background noise and finally another tool is used in order to measure the actual area of the object.

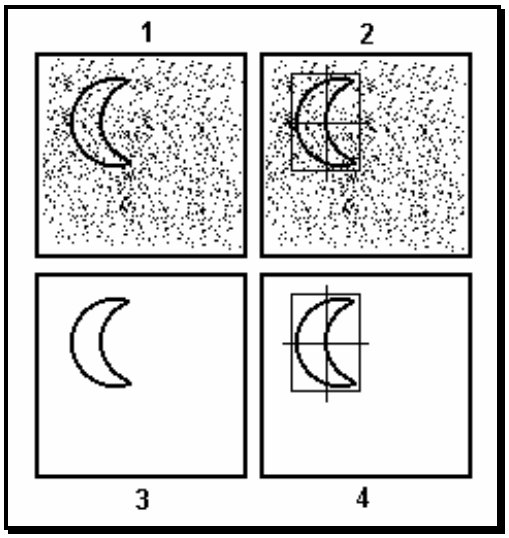

Figure 1 - Typical steps for object's area measurement

The final result can be viewed on a computer screen by an operator or can be output to any equipment in the manufacturing line so that it can automatically be controlled without any human intervention. The most important image processing tools will now be described.

2. PATTERN LOCATING

Many vision applications require positional information about a certain object within an image. *Pattern Locating* tools search a region within a given image. The purpose of this tool is to establish the locations of those areas which match a stored feature. The feature is normally defined through training using one or more sample training images. The result is a series of locations defining the origin of the feature when it is positioned at a matching area.

These tools return positional information with sub-pixel accuracy. They often solve the whole vision task and in other applications it represents an important step to the application.

Examples of applications which need this type of tool are:

- **Inspection** - The machine vision system can first locate the object of interest before other software tools analyse it for defects. By locating features on an object, the vision system can make position based inspections, like locating the corners of a bottle label to calculate correct label positioning.
- **Alignment** - To perform alignment tasks that can close the feedback loop for parts positioning equipment. The vision system measures the position of an object and feeds it to a part positioning device so that it can correct the part position by moving it to the desired location. Then, the manufacturing operation can be carried out like drilling or adhesive dispense.
- **Gauging** - Typically involve finding two or more points on an object and calculating the distance or angle between them. Therefore, the *pattern locating* capabilities form the basis of many gauging applications.
- **Guidance** - In robotics part picking a video camera mounted on the robotic arm captures an image of the part to be picked. This image is sent to the machine vision computer which locates the part, measures its position and transmits this data to the robotic arm.

A number of different methods for returning this information exist: the most important are the *search* and *contour finder*.

- ***Search*** - This tool works by scanning the input image for a match to a pre-trained grey-level "image" (or model) of the object of interest.
- ***Contour Finding*** - This tool locates objects by finding an outline within the input image that matches a pre-trained outline of the object of interest.

These two tools are complementary rather than alternate tools. For some applications the *Search* algorithm performs better while for some other it is the *Contour Finding*. In some cases both tools can be used to achieve more reliable results. Some characteristics will now be analysed.

In some applications the captured image can be affected by linear brightness changes (like when a lamp is replaced by a more powerful one) which means that the average grey-level will change uniformly across the entire image. The *search* still

performs well because it is normalised which means that it factors out uniform changes in brightness. *Contour Finder* also locates objects reliably because a change in brightness does not influence the contour of the object.

If the change in brightness is non-linear (like a shadow over part of the object), the *search* might not be the best choice since that shadow changes the signature of the object. However, the *Contour Finder* still performs well as far as the border remain visible.

Video (or electrical) noise can be troublesome under low light conditions (video signal needs to be amplified). *Search*, which finds objects based upon a detailed model is virtually unaffected while the *Contour Finder* performance might be affected.

Some images contain one or several objects that resemble the object of interest. *Search* can discriminate between similar objects based upon the multitude of pattern information contained in the pre-trained model. *Contour Finder* also can discriminate, although it does not have access to the pattern information.

2.1 Search

Consists of a vision software tool that locates objects by comparing a grey level model of the object of interest in the input image. It is based on a technique known as template matching, which means that the vision system scans the input image for a match to the object template that has been stored in memory. When a match is located it measures the object X,Y position and orientation. It also calculates a correlation of shape score which is a measure of how close the match is. It uses a full range of grey-scale values which makes it more reliable by providing more pattern data.

Another characteristic of *search* is that the equation used to compute correlation is normalised, which means that the algorithm can factor out variation in contrast and brightness.

Search requires that an operator first store a model of the pattern of interest in the vision system memory. Another technique is to capture several models and average them so that the final value might be more reliable.

Search can be configured to handle applications in which the objects can become rotated. This technique, called *search* with rotate, is necessary because the accuracy of X,Y locations returned is insensitive to rotation of the object with respect to the model. If the object is rotated too far the object is not found. There is though a limit in the angle allowed.

The result from this tools is three values: the X,Y position where the model matches better; the shape score which is a number between 0 (no match) and 1000 (perfect matching); and finally the contrast of the feature found in the image with respect to that of the model.

2.2 Contour Finder

It is a vision software tool that locates objects by finding their outlines or contours in the input image. It also requires that the user first train a model of the object to be located. Unlike *search*, it stores a model that represents the positions of all edge points in the object. It then compares the edge points of the model with the one of the image to locate the final location.

This tool is suited for applications in which the surface or texture of the object can vary from part to part. This tool also handles light changes that are not uniform across the camera field of view. Example is an object with a shadow cast on it. Objects with degraded surfaces, cracks or highlights caused by reflections are also found by this tool.

Contour Finder is based on the Hough Transform which is a method for locating patterns among groups of edge points. The edge pixels are detected and its magnitude and direction are calculated. This tool returns the X, Y position of the located object as well a confidence score. Artificial Intelligence is used to quickly scan on the positions that are most likely to yield a match between image and model.

The user can set a threshold so that only pixels above a certain value (brightness) will be considered. This way the user can focus on the edge point which are strong enough to warrant interest,

rather than wasting time analysing points which are not of any interest.

Like the *search* tool, *Contour Finder* can also be configured to tolerate rotation of objects up to a certain degree.

3. INSPECT

Inspect provides information about the size, position and connectivity of objects in an image. It is the basis of many inspection, guidance and sorting applications. It is useful for applications in which the objects vary greatly in size, shape or orientation. In these applications it is difficult to define a model for use by *search*. *Inspect* does not need pre-trained models. It measures the attributes of whatever appears within the designated window of interest. This tool is not very suitable for objects which contain the same level of grey as the background because it bases its decision, about what is part of the object or not, on the grey level.

A type of inspection application that deserves special mention is defect analysis. This type of application is combined with other tools. An example is a character defect inspection algorithm which first uses a *search* to find the object. Then the object can be subtracted from an ideal character image using image processing tools. The resulting difference image would contain only all the defects. Finally, the system would call the *Inspect* tool to count, measure and locate the defects and decide on its quality.

The information provided by *Inspect* can be used in order to sort or classify objects according to their size, shape and position.

This tool consists of two phases: deciding what constitutes an object and then measuring its attributes. The first phase is crucial because it sets limits on the accuracy and consistency of the second phase's results.

In the first part, called segmentation, the system decides if a pixel is an object pixel or a background pixel. A typical solution is setting a threshold value above which it is considered an object and below is a background (or vice-versa depending on the application). In some cases, a pixel might be part of an object and also part of the background (pixel in the border of the object). In this case, a sub-pixel technique has to be used and the pixel is weighted depending on its brightness value.

Applications in which several objects appear on the image may require a second step in determining what constitutes an object. Connective analysis determines whether object pixels are all joined together in one blob or whether there are more separate blobs. The user can decide whether to measure the entire scene or to measure individually each blob.

This tool provides the following measurements for each blob:

- **Area of the blob** - Weighted sum of all pixels in the blob.
- **Smallest enclosing rectangle** - A rectangle that has vertical and horizontal sides and just large enough to enclose the blob's pixels.
- **Angle** - Angle of its first principal axis. This axis is a line drawn through the blob so that as many blob pixels as possible are as close to the line as possible.
- **Principal moments** - The first and second moments of inertia indicate how close most pixels are to the first and second principal axes.
- **Centre of mass** - It is the point at which its first and second principal axes intersect.
- **Median** - It is the point at which the total weight of the pixels above the point equals the total weight below and the total weight of the pixels to the left of the point equals the total weight to the right.

4. CALIPER

The *Caliper* Tool is a vision tool which can be trained to find an edge pair within an image. If the edge pair is found within the image a result is produced, witch details the position, edges contrast, edges separation, edges polarity and score of the edge pair.

Caliper tool is modelled after the mechanical caliper. The user specifies the desired separation between the caliper "jaws" - or the distance between the edge pairs of interest. It then searches the image for the pair of parallel edges that match that description and precisely measures the actual distance between them.

It is mostly used in applications that require both accurate measurement and high speed. Examples of applications that use this tool are thickness components measurement and lead spacing in integrated circuit chips.

This tool is based on smart edge detection techniques which enables it to locate correct pairs of edges despite complex or confusing data in the image background.

First, a region of interest (window) is applied to the captured image. This means that the tool processes only those pixels inside that window (defined on the set-up step). Then, it projects the two dimensional window onto a one dimensional image. It is like if the window was collapsed into a single line. This accentuates the edges in the image and reduces the amount of data to be processed (making it faster).

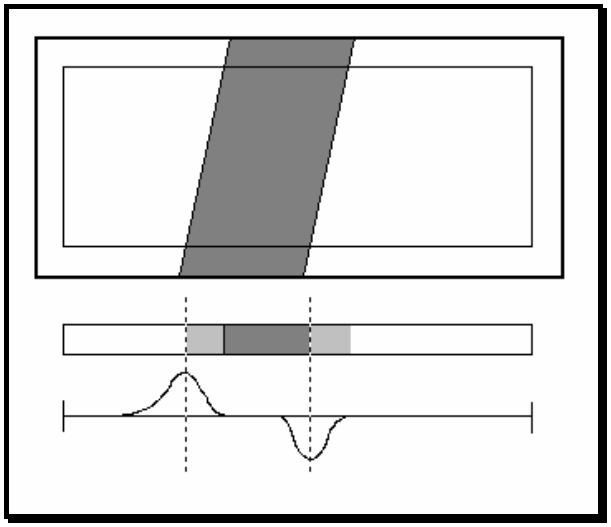

Figure 2 - Steps to the *Caliper* tool

Next, the *Caliper* tool applies an edge filter to the 1D window image to locate each edge. A zero value indicates no edge, while a higher or lower value indicates the presence of an edge. The sign of the value indicates the direction of the edge (a change from dark to light or light to dark).

The *Caliper* finds the edges of interest by scoring each located edge based on geometric constraints specified by the operator. Parameters include polarity (or direction) of the edge, position of the edge pair, distance between the edge pair and difference in grey value on either side of an edge.

5. GOLDEN TEMPLATE COMPARISON

The *Golden Template Comparison* (GTC) is a vision tool which can compare a trained template image with another similar image, and identify any differences between the two.

The result of a comparison is a binary image which contains the areas of the images which are significantly different. This image can be subsequently analysed using other vision tools such as the Blob Tool.

GTC provides reliable high speed detection and classification of defects within scenes that contain complex, grey-scale imagery. While simple images might be adequately inspected using blob analysis technique, GTC is designed to address applications in which the vision system must identify defects within complicated grey-scale patterns.

Applications best suited for this tool are those in which the object under test varies little in appearance from part to part and is not subject to significant changes in rotation or scale. A typical application is detection of ink dots, chips and cracks on semiconductor wafers and dies which are consistent in appearance due to its high repeatable manufacture processes.

The first step is for the developer to call the GTC training functions which are used to store in memory a template of the image. The operator shows several good samples of the object. A mean image is a statistical average of all the samples shown to the vision system. By using the average of several images reduces the variation between the stored images making the tool less susceptible to false defects. Standard deviation image is then created which defines those areas where there is little variation from part to part and those with great deal of variation. This allows different levels of sensitivity on the image.

The developer also sets parameters that define what constitutes a defect. Often, the captured images are not located at the same location as the template and therefore a pattern finding technique is needed to accurately register the captured image.

Then, the GTC tool subtracts the captured image from the template image and compare the result with the standard deviation to determine which data should be ignored and which data should be analysed as possible defects.

Finally, blob analysis techniques are used on the remaining data (the difference image) to classify the defects. GTC uses these techniques to measure the size, shape position of each group of pixels in the image. This data is then compared to the user defined criteria to determine whether the part should be accepted or rejected.

6. SCENE ANGLE FINDER

This tool measures the angular orientation of an object. It does so by determining the direction of dominant edges in an image. Importantly, it does not require pre-trained models. Thus, it is a valuable first step in any alignment and guidance applications. The developer can determine the object's orientation and then correct for angular deviations before running a model-based search to precisely measure X,Y position.

Scene Angle Finder works on any image that contains strong parallel and perpendicular edges. Some examples are semiconductor wafers, certain printed circuit boards and block letter printing.

This tool returns the scene's orientation with respect to horizontal. It also returns a quality score indicating how dominant the scene's linear patterns are. This score can be used to determine if there is enough linear information to produce reliable results.

7. POLAR CO-ORDINATE VISION

The *Polar Co-ordinate* vision tools enable the developer to access image information using polar co-ordinates. These tools can directly measure the angle of complex objects. They can also prepare polar images for further analysis by other vision tools. Such capability is very useful for applications in which the relevant information is arranged in circular form. Examples of such applications are inspecting the teeth of a gear, codes printed around an arc, etc.

Polar co-ordinate vision has two major modules. The first transforms cartesian images into polar images. In other words, it takes an image described in (x,y) co-ordinates and expresses the same information in (r,f) co-ordinates. All the other tools can then be used because they treat the pixel data just if it were expressed in (x,y). This effect can be visualised as cutting a ring and unwrapping it into a bar.

The second module measures the angle of a complex scene. It accomplishes this task by searching for a polar model within a polar image.

A polar model is made by applying the polar transformation to a ring-shaped subsection of the image. Such a model can be thought of as the "rotational signature" of the image. The system searches for this polar model within the polar image and finds the f position that provides the best model/image match. It then reports this f value as the object's angle.

8. CONCLUSIONS

Several important image processing tools were described in this paper. Nowadays, it is not necessary to program from scratch an algorithm in order to solve an industrial application. The use of these types of tools have been well tested in several applications. Therefore a vision system requires no more than choosing the correct tools for a specific application. This reduces very much the programming time, and increases the reliability of the package. Several other tools exist although only a few were described in this paper.

9. ACKNOWLEDGEMENTS

Mr. Ribeiro would like to thank all his family for the strong support given, specially his wife for understanding him on the most difficult moments. Cranfield University deserves a big thank you for

giving him the knowledge about Image Processing. Thank you to Integral Vision, Lda (Bedford, UK) and Sisqual (Porto, Portugal) for helping in the image processing aspects. Thanks to Universidade do Minho for giving the opportunity to present this paper.