Curvature Analysis for Recognition and Structural Verification of Industrial Objects

Q.M. Wu
M.G. Rodd
Department of Electrical & Electronic Engineering
University College of Swansea
SWANSEA SA2 8PP, United Kingdom

Abstract

In this paper, we develop several algorithms for verification of regular-shaped objects by breaking boundaries into fundamental primitives such as lines and circular arcs. We first discuss algorithms for detection of extreme points having high curvatures, and propose an algorithm for detecting knot-points formed by basic primitive segments. We then classify the existing flat industrial objects into four categories. For each category, we develop corresponding effective verification algorithms, according to the characteristics of the objects in each class.

Keywords: Curvature analysis, object inspection, knot-point detection.

1 Introduction

With the rapid progress in computer vision research, machine-vision inspection systems can be expected to play a more and more important role in industrial automation. To achieve this goal, many techniques and systems have been developed over the past decade. However, due to the inherent diversity of inspection tasks, it is difficult to develop generic techniques applicable to a wide range of applications. Very often, an *ad hoc* approach is adopted for each individual application task which is time- and labor-consuming.

In this paper, we present a set of generic algorithms for structural verification of *man-made, flat industrial parts.* Such shape inspection is clearly a common and critical issue in many inspection tasks — as diverse as the monitoring of the quality of punched sheet metal, the inspection of products (e.g., biscuits) in the food industry, or in the monitoring of mechanical parts (e.g., washers) in general manufacturing. Therefore it is important to develop a generic methodology for solving these inspection problems.

Most structural verification tasks can be performed based on the boundaries of objects. Thus the extraction of the desired boundaries of objects in an industrial environment is of great importance to the high-level verification processes. As has been discussed in [14], the successful extraction of these boundaries requires both effective algorithms and a priori knowledge. In this paper, we assume that the boundaries of objects are avilable and discuss issues of feature extraction and verification — all based on the object boundaries. The rest of the paper is organized into three sections. In Section 2, we discuss the boundary dominant point detection problem. Based on the discussion in Section 2, in Section 3 we develop a set of verification algorithms. Finally, some conclusions are drawn in Section 4.

2 Feature Extraction by Curvature Analysis

Feature extraction is a key step towards the successful recognition, and subsequent verification, of industrial objects. Since the objective of this paper is to develop effective vision algorithms for verifying the structure of industrial objects, it is crucial to analyse what critical features can be derived from object boundaries, and which of these are of value in the recognition and verification processes. Having determined which features can be important, we can then examine algorithms for the efficient and reliable detection of such features. This section starts by investigating the dominant features of real-world man-made objects, and then moves towards proposing techniques, based (where appropriate) on well-substantiated theory, and algorithms to handle the extraction of such features. The validity of such algorithms is analysed by application to practical situations.

2.1 Dominant Features

Features derived from object boundaries can include both global features and local features. The use of global shape features, however, (such as an object's perimeter or area) is limited for inspection purposes to particular environments, mainly because such properties do not normally reflect *local* variations in object shapes. In most real situations, however, it is the *local* variations which require close inspection.

It is evident in the literature [2, 9] that points having high curvatures and zero-crossing points make up an important set of features on object boundaries. However, despite much success achieved by the direct use of these important points in object recognition, there is evidence.

too, that the geometric primitive elements between these points are also important features. This point has been fully demonstrated in a paper by Fischler and Bolles [6].

The question which arises is: "If we recognize that the idea of using primitive elements is important in the development of machine-vision systems, what primitives should we use?" Studying the scenes around us in the real world, it is clear that whilst natural scenes seem to be described by many primitives (some of them complicated), most industrial objects can be described by relatively few simple geometric primitives — mainly because most industrial parts are designed via CAD systems in which only limited, well-defined primitive elements are used. Amongst the small number of primitives often used in CAD systems, lines and circular arcs are the most popular. We shall, therefore, concentrate in this paper on feature extraction of such well-structured objects.

The first key step in segmenting a boundary into primitives is to highlight the *joint-points* (which we refer to as *knot-points* in this paper). These are the points which denote changes in primitive segments. We then partition the planar object boundary into primitive segments — typically lines and circular arcs. For example, in inspecting a regular shape consisting of lines and circular arcs, if the knot-points between different primitive segments can be located, geometric features can then be easily transformed into primitives, and their spatial relations transformed into a simple symbolic string. The inspection task can then take place by matching the string with a model — maybe compiled directly from a CAD system.

It is, therefore, clear that the dominant features on object boundaries will include points with high curvatures, the knot-points between different geometrical primitives and, of course, the primitives themselves. It must be pointed out that most points on an object boundary having high curvatures are also knot-points. Since there are obvious changes in curvature at knot-points, they can be detected by analysing the changes in curvature on object boundaries.

However — to estimate the curvature of a digitized curve is not a trivial task, and hence, the detection of critical points on a boundary is not easy! In the remaining part of this section, we shall study these problems, with a particular focus on efficient and deterministic implementation. Based on the outcome of this study, we shall propose a flexible and robust way of segmenting curves, based on curvature analysis of the boundary. We shall show later how these techniques are useful in the development of efficient industrial inspection algorithms.

2.2 Curvature Estimation

For a continuous curve, curvature at any point of it can be accurately calculated, and appropriate information can be extracted by analysing changes in curvature. However, when we deal with digitized curves, it is not immediately clear how to define the *discrete* analog of curvature. Assuming that the boundary of an object is traced, say, based on *8-connectedness*, it is obvious that the change of slope takes discrete values which are multiples of 45^0, so that small changes in slope are impossible to detect. One solution to extracting "noise-free" curvatures from a contour is to "smooth" the curve. There are two categories of approaches which are correctly being employed for this purpose: the Gaussian-filter-based approach [12] and angle-detection schemes [7].

A). Gaussian-Filter Approach

One approach represents a planar curve by introducing a parameter l (a variable related to path length). Thus x and y coordinates can be represented as functions of l. To achieve the smoothing effect, each coordinate function is first smoothed before being used for computing the curvature. In order to compute the curvature at varying levels of detail, functions $x(l)$ and $y(l)$ are convolved with a one-dimensional Gaussian kernal $g(l,\sigma)$ of standard deviation σ.

Another typical approach in this category initially finds the bounding contour of a shape using an edge detector (such as those derived by Hueckel [8] and by Canny [3]), which generates the orientation of each edge point. As the boundary is tracked, the orientation at each point is estimated. The orientation function, denoted as $\phi(l)$, is thus established. The first Gaussian derivative filtered response $g'(l,\sigma) * \phi(l)$ and the second Gaussian derivative filtered response $g''(l,\sigma) * \phi(l)$ are then computed.

According to the definition mentioned earlier, the curvature at a point is the instantaneous change of orientation (slope) over the change of the arc length. Thus the change of the orientation over a short arc length between points P and P' is an approximation of the curvature at the point P. Therefore, the filtered response $g'(l,\sigma) * \phi(l)$ reflects the curvature at point $P(x_l, y_l)$. Compared to the approach that computes curvature using filtered coordinate functions, the physical meaning of this method is relatively clearer.

B). Angle-Detection Schemes

Angle-detection schemes provide another approach for estimating the curvature of a digitized curve. The fundamental principles of angle-detection schemes for curvature computation are derived from the curvature definition for continuous curves. We know that we cannot use the direction change between two successive link elements to compute the curvature directly, as was mentioned previously. However, if we use the angular difference between two successive vectors to calculate the curvature, we can get a *reasonably* accurate estimate of the curvature. A representative procedure for several angle-detection schemes is described in [7].

The curvature calculated using this procedure is customarily called *incremental curvature*. The incremental

curvature is a smoothed measure of curvature; the greater m (often referred to as a *smoothing factor or smoothing parameter*) is, the heavier the smoothing. The smoothing effect in the incremental curvature method is achieved by using a straight-line segment with a length of m boundary points to approximate the digital segment. This has solved the problem that the slope change between any two successive discrete boundary points is only the multiple of $\pi/4$ mentioned earlier, and leads to the estimate of the slope at a point P being closer to that computed by using a continuous curve to fit the digital one. It is clear that if a longer line segment is used in the approximation, the curvature estimate thus derived may not reflect the original fine features of the digital curve. On the other hand, if a small smoothing factor m is used, noise injected in digitizing the curve may be reflected in the estimate. Therefore, an appropriate smoothing factor has to be chosen. The "optimal" smoothing factor is dependent on the specific application. In practice, the factor appropriate to the class of curves to be processed can be chosen only by experimentation or by examining the class of curvature features that exist in the existing object categories. For a curve extracted from an image with a reasonably high resolution, The smoothing factor will range normally from a minimum of 5 to a maximum of 13 [15].

2.3 Extreme-Point Detection

There are several methods to locate peak curvature points and zero-crossings from curvature estimation. Assuming that the curvature estimate is computed using one of the approaches described in the preceding subsection, the location of curvature maxima (minima), or zero-crossings, simply involves the detection of peaks (or valleys) on a curvature plot. To locate these points accurately normally requires a two-stage processing procedure. In the first stage, a global curvature threshold T_c (for the complete curvature plot) is first of all used to select those boundary points with curvature values greater than T_c. For each corner on an object boundary, a point will be found which has a curvature value greater than T_c; often a small number of points, close to the corner point, have curvature values greater than the threshold. In the second stage, therefore, an input local threshold (for one peak only) is applied to the curvature estimates of the segment close to the peak, so as to eliminate those points whose curvature estimates are not local maxima in a sufficiently large segment of the curve.

From the above discussion, a few observations can be made:

- *Computational Aspect:* Both the algorithms in Section 2.2 (i.e., that which estimates the curvature by convolving the orientation function with a Gaussian kernel, and that which smoothes the coordinate functions) use Gaussian filtering to eliminate the noise injected through digitization and other sources. Due to the time-consuming convolution operation involved in Gaussian filtering, both algorithms are computationally expensive. In contrast, angle-detection schemes are conceptually, and computationally, simpler.

- *Level of Smoothing:* Both Gaussian-filtering approaches and angle-detection schemes estimate the curvature of a quantized curve by first undertaking a smoothing operation — the degree of smoothing depending on various parameters. In the Gaussian-filtering approach, the parameter is the standard deviation σ, and the change of σ from small to large will allow a coarse-to-fine detection scheme to be established. This will make it possible to find instances of curvature change primitives at one scale, even if they cannot be found at some other scales! In many cases, a useful parameter space (scale-space) can be established by plotting the zero-crossing points against the parameter l. In the angle-detection scheme, using different values of m will allow different degrees of smoothing. The larger m is, the heavier the smoothing. Therefore, in practical applications, an appropriate value of m must be chosen.

- *Threshold Selection:* The selection of the threshold value is important, since a high threshold could miss some critical points, but at the same time, would get rid of false points. A low threshold value, on the other hand, would give more-reliable critical-point detection, but would increase the number of false detections. Therefore the threshold selection requires intensive experimental verification.

- *Detection Accuracy:* The use of angle-detection schemes is accurate for detecting sharp corners because these methods explicitly measure the changes of slopes between two successive vectors, and hence points with high curvatures can be reliably detected. However, they do not produce satisfactory results when used to detect relatively smooth joints (which are normally difficult to perceive) because the linear fit destroys subtle curvature! The use of the Gaussian-filter-based approach enables the detection of these points since a peak (or valley), which corresponds to the position of the smooth joint, clearly shows in the second Gaussian derivative filtered response. This attribute will be discussed in greater detail later.

The review undertaken in this section has illustrated that each category of algorithms has some advantages and disadvantages. For industrial applications, however, better, more-generic, solutions should be sought.

2.4 Knot-Point Detection

In this subsection we describe a new algorithm which resolves the problems inherent in the two categories of algorithms previously discussed. This algorithm achieves this primarily by *combining* their advantages for detecting critical points! It is shown how this can lead to an approach which is reliable and computationally relatively inexpensive. Importantly, the proposed algorithm is capable of locating knot-points between basic primitives consisting of straight lines and/or circular arcs (i.e., regions of constant curvature) — hence efficiently segmenting contours into a number of desired primitives for use in a syntactic inspection system.

All the possible types of knot-points, which can exist between primitives of lines and circular arcs, can be classified into two categories, in terms of the complexity of detecting them in a digitized curve. In the first category all the knot-points are sharp corners; in the second category the knot-points are either smooth joins (smooth corners) or the knot-points formed by short segments. It is obvious that the knot-points in the first category are easy to detect, while those in the second category are extremely difficult to extract.

As was mentioned in the last section, angle-detection operators are good for locating sharp corners, particularly in terms of their computationally simple implementation, their reliability, and their accuracy in locating sharp corners. Gaussian-filter based approaches are computationally more expensive but are reliable in locating smooth joins, etc. The new knot-point detector is thus developed by making use of the advantages of both. It can be described as follows:

We start by using the angle-detection scheme to locate the sharp corners, and then use a set of criteria to distinguish if the segment between a pair of knots is a straight line, a circular arc, or a combination (i.e., a succession of primitive segments without knot points between them, which we call a "combination segment"). If it is a combination segment, a priori knowledge about the primitive type (i.e. a smooth join, a nick or a bump) of a segment can be obtained from the criteria that have been satisfied by the segment. Thus, the scale-space approach can be applied to this particular segment to detect the primitive, guided by the a priori knowledge (which is not available when using the scale-space approach alone).

We shall show that this scheme has advantages over several angle-detection schemes and the Gaussian-filter based approaches for detecting knot-points in the application classes referred to in this paper. The complete scheme consists of three major steps which are described below.

(i) Detecting Sharp Corners

The sharp corners can be detected by applying the angle-detection schemes. A corner is characterized by three incremental curvature regions — two for which the curvature δ_j^m fluctuates within relatively narrow limits due to digitization noise, separated by a third region consisting of a small number of nodes in which $\sum \delta_j^m$ equals a significant value. This characteristic enables us to detect sharp corners on the boundary of an object in a straightforward way. Extraction of sharp corner points simply involves (a) the estimation of discrete curvature, (b) global thresholding, and (c) local thresholding (or maybe local peak sharpening and thresholding), as mentioned in Section 2.3.

(ii) Testing Segment Types

Once the boundary has been segmented by sharp corners, various criteria can be applied to test the segment types. The detailed description of the criteria and their use for segment identification has been discussed in [16]. In many cases, just one of these criteria is enough to identify a circular arc segment. If a segment is accepted as neither a straight line nor a circular arc, it is regarded as a combination segment, which may include two or more primitive segments.

(iii) Detecting Knot-Points in Combination Segments

Two methods can be used for further processing combination segments. One is to split the segment at certain points [13], and the other is to apply a scale-space approach to these segments [11]. These types of segments almost all belong to the second category as defined in an earlier section, i.e., those with smooth joins which we refer to as the first class of combination segments, and those with short primitive segments which we refer to as the second class of combination segments.

First of all we discuss the detection of knot-points on the first class of combination segments. The common feature of this class of segments is that the change of slope is smooth, and the curvature discontinuity is therefore not easy to detect. However, there are indeed significant changes in curvature at these points. The detection of these discontinuities using an angle-detection scheme is, however, not reliable because of the "vague" features obtained to indicate the position of a point of discontinuity. As described in section 2.3, several methods can be successfully applied for this purpose. To highlight the process, we show the location of knot points on several combination segments of the first class. Figure 1 shows several segments with smooth joins formed by circular arcs and/or line primitives and their Gaussian-filtered responses. It can be seen that detection of the knot-points on these segments involves simply the location of peaks or valleys in the filtered responses of orientation functions, with second Gaussian derivatives.

In our experiments, we observed that the second method (estimating curvature using Gaussian-smoothed coordinate functions) performed better for locating the knot-points between two circular arcs with opposite signs for the flanking curvatures. The first method performed better in locating the knots on other smooth joins. Thus in some cir-

cumstances (e.g., when dealing with very complex shapes), one could use them, respectively, for the different types of smooth joins.

For the detection of knot-points on the second class of combination segments, again, one cannot rely on the angle-detection scheme which does not produce accurate estimation of curvature for a combination segment with short primitives. The scale-space approach is therefore employed. The detail of the approach is described in Asada and Brady's paper, "The Curvature Primal Sketch" [1]. To aid the interpretation of the "tree" established by "The Curvature Primal Sketch", we can (coarsely) identify the primitive types of a combination segment by making appropriate criterion tests before the scale-space approach is applied.

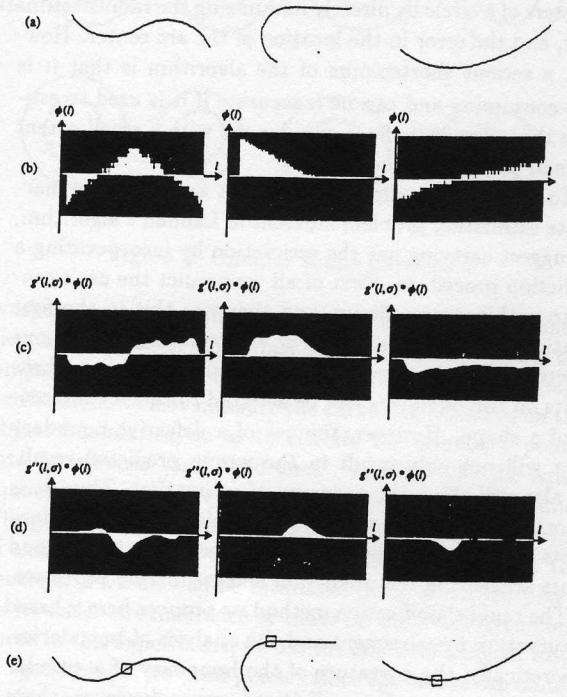

Figure 1: Knot-point detection on the first class of combination segments. (a) Segments with smooth joins. (b) The smooth joins in orientation space. (c) The first Gaussian-filtered response. (d) The second Gaussian-filtered response. (e) The knot-points detected.

3 Object-Structure Verification

Object inspection covers a broad area of topics, including the classification of objects, the verification of object structures and the detailed inspection of surfaces. The verification of object structures, which we study in this section, includes measuring the various dimensions of the parts, as well as checking to see whether there are any flaws, such as breakages or cracks on the outer side of a casting, or defects such as nicks, bumps, or cut corners on the boundaries of flat parts.

Different classes of parts normally have different shapes, but these shapes can generally be classified into 4 categories according to the complexity of the primitives which combine together to form an object's contour. The four categories are: (1) circular objects, (2) rectangular objects (including squares), (3) objects composed of lines and circular arcs, and (4) objects composed of complex primitives, by which we mean that the parameters of these primitives are difficult to estimate. In this section, we show that for each of those, efficient structural verification algorithms can be developed in terms of the common characteristics of objects within the same class. The concentration, however, is on the development of verification algorithms for objects in the first three classes (termed as *regular-shaped* objects), which are often encountered in many industrial inspection situations. The development of these algorithms will rely heavily on the techniques proposed in previous sections.

3.1 Rectangular Shapes

For generality, we assume that the position and orientation of objects to be inspected are not known a priori, but that the distance between the camera and the objects is known. The inspection thus involves verifying the size (length and width) of an object to determine if it is within the allowed tolerances, and locating defects such as nicks, bumps, and cut-corners on the boundary. The techniques proposed involve a sequence of four major steps.

- *Detection of local maxima on the curvature graph*: Since there are significant changes in edge orientation at the corners of an object, an analysis of the changes in curvature at boundary points provides useful information for locating the object's corners. The simple, but efficient, algorithm described in Section 2.2 is used here to calculate the curvature of a discrete curve. To find the major points of discontinuity, a threshold (its value is determined by experience, normally around $35^0 \sim 40^0$) is applied to the curvature graph, and those points with a curvature value greater than the overall threshold, and with a local maximum curvature value (since the curvature values of boundary points close to a corner point can also be greater than the overall threshold), are selected. As a result, only the real corner points, and any other points where there are possible defects causing high curvature values, are chosen and passed on to the next stage of processing.

- *Elimination of non-corner points*: To be able to estimate the size (length and width) of an object in the scene, we need to know which points found in the first step are the "real" corners, and which are not. This can be achieved by the following simple procedure: Assuming that $P_1, ..., P_i, ..., P_M$ are the points selected by step (1), whether or not point P_i is a

"real" corner depends on whether the angle between vectors P_iP_{i-1} and P_iP_{i+1} is equal, or close, to $\pi/2$.

- *Calculation of the dimensions of each object:* Having obtained the real corners of an object, its dimensions can then be determined by simply calculating the distance between each pair of adjacent corners. The average of the two longest distances is the object length, and the average of the two shortest is the object width. It should be noted that in circumstances where only three corners can be found, the object is simply rejected, since a cut corner exists.

- *Detection of ragged edges:* There are two ways of inspecting the boundary conditions. The first is to calculate the distance of each boundary point from the straight line formed by a pair of adjacent corners. The second is to calculate the distance between every boundary point and a straight line obtained by using the information on all boundary points between the two adjacent corners. It is clear that the former method is fast but not very accurate, and the latter is the converse. If an object fails either of these distance tests (i.e., a distance is greater than the allowed tolerance), it is rejected.

This procedure can be used in many practical applications. Figure 2 shows the boundaries of seven biscuits and the corner points, obtained after non-corner elimination procedure and marked with small black square areas. It can be seen that only three corners are found for the seventh object (counted from left to right) and therefore the object is directly rejected because of its cut corner. To

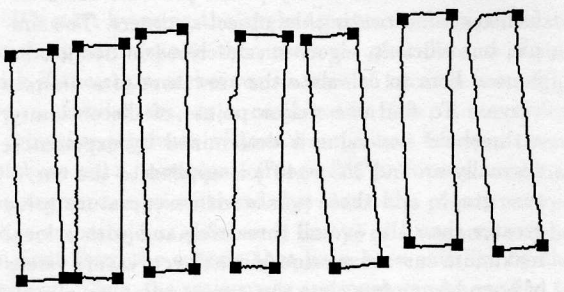

Figure 2: Inspection of rectangular objects.

verify the other objects, the size (length and width) of each object is estimated. As a result, all objects have passed the size test. To further inspect the objects, the condition of each object's boundary is checked. By calculating the deviation of boundary points from the straight line formed by two adjacent corners, objects 4 and 6 fail the test and are rejected. The rest of the objects successfully pass the shape-verification test.

3.2 Circular Shapes

The inspection of a circular object again requires the verification of the object's size (radius) and its boundary condition, as well as determining the center point. The method for inspecting circular shapes is slightly different from that for rectangular shapes. The key issue in inspecting a circular shape is, given a set of discrete points, to find the "real" center of the circular shape. Various methods, e.g., the Hough transform method, for accurately finding the center of a circular shape have been discussed by Cao and Deravi [4, 5]. In this subsection, we present a method developed with a view to achieving fast processing and reasonably high accuracy. The method developed is based on Landau's algorithm [10].

Landau's algorithm achieves good estimation of the parameters of a circle by directly minimizing the radius-estimation error, and the error in the location of the arc center. However, a serious shortcoming of the algorithm is that it is time-consuming and can be inaccurate if it is used to estimate the parameters for a circular arc with a small extent (a small portion of a circle).

To reduce the computational effort and solve the inaccurate estimation problem inherent in Landau's algorithm, we suggest carrying out the estimation by incorporating a prediction procedure. First of all we predict the arc's center to within coarse limits, and then use this in the first iteration of the estimation. For a circular shape without defects, such a prediction process is simple, since any three points on the boundary can be selected to predict the center of a shape. However, the use of a defective boundary point will not only result in the wrong predicted result, but also will affect the accuracy of estimation. Thus, it is important to *classify* roughly, and quickly, the boundary points into "good" ones and "bad" ones, so that only good points are used in the prediction and estimation processes.

The rapid classification method we propose here is based on curvature measurement and the analysis of boundaries. Theoretically, the curvature of the boundary of a circular shape is constant. For a digitized curve, however, there might be slight variations in curvature at different points of the boundary, which should, of course, be very small on a "good" boundary. Using this principle, possible defective sections on the boundary can be rapidly detected. Those sections with high curvature variations will subsequently be eliminated from the prediction and estimation processes. It should be noted that although this procedure for detecting high curvature-variation segments could be used directly (in some circumstances) to detect defective shapes, generally speaking it is not very reliable since the actual dimensions of a defect cannot be measured using this method.

Having now ensured we are using a good prediction, i.e., assuming that $\widehat{AB}$ is an arc which is a good section, an algorithm such as the one shown in [16] can be used to predict the center and radius of a circular shape. Thus, a point close to the real center of the circle can be found.

The estimation can proceed, based on this point, so as eventually to find the real center point. Having found the desired center of a circular shape, the accurate inspection of the circular shape is a relatively easy task. First of all, the distance between each boundary point and the center is calculated, and a histogram of these distances is established, plotted against the sequence of the boundary points. This histogram can be checked to see if the object being inspected is within the given tolerance!

The following describes the application of the above procedure to inspection of mechanical components. Figure 3 shows a number of washers. Inspection of these

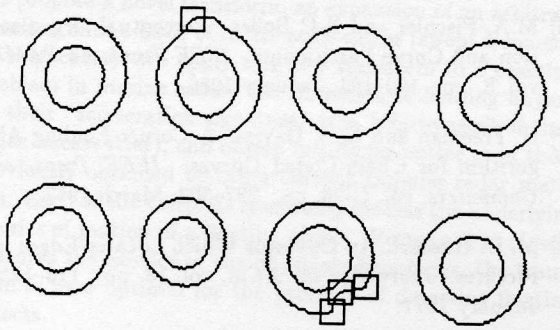

Figure 3: Circular shape inspection.

washers includes verification of their sizes and the condition of their outer and inner circumferences. Using the procedure developed in this subsection for circular shape verification, we first measure the curvature variation on the object boundaries, so as to classify the boundary points into "good ones" and "bad ones". In Figure 3 , the small square boxes show the areas where "bad pixels" are found. The good ones are then available for use in the computation of the centre of each washer. It should be noted that the centre of a washer can be estimated using either the pixels on its outer circumference or those on the inner circumference. In this experiment, we use the one with a higher ratio of "good pixels" to "bad pixels" since the centre calculated in this way would be more accurate.

The distance between each boundary point and the washer centre is calculated and shown in a histogram. To verify the condition of each washer, a threshold is applied to each histogram. The threshold is determined by the actual tolerance limit of the components. For example, assuming the tolerance limit is 2mm for the washers, a washer is rejected if its distance histogram has values 2mm above or below the standard histogram. As a result of this inspection, Washer 2 and 7 are failed because of flawed outer circumference, Washer 6 is failed because of its small size, and the rest are passed.

3.3 Objects Composed of Lines and Circular Arcs

The boundaries of a significant class of industrial parts consist of combinations of straight-line segments and circular arcs, mainly because such primitives are used in many CAD systems. The inspection of this class of objects can be approached in at least two different ways: the first is based on the measurement of the parameters of each primitive segment on the boundary, and the second compares a test contour with a template contour, by normalizing the test contour to the template contour. This latter procedure involves finding the normalization parameter values which minimize a chosen "distance" between the contours. Here, we present a technique for verification based on the first technique, a measurement of primitive segment parameters.

It is clear that to measure the parameters of each primitive segment, two subtasks are involved. The first is to segment the boundary, and the second is to find an algorithm which accurately estimates the parameters corresponding to each segment. The solutions to these individual problems have been presented respectively in Section 2.4, and Section 3.2. To demonstrate how these solutions are combined to solve the combined verification problem posed here, the method is described below in a practical example.

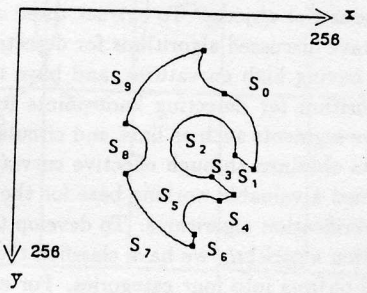

Figure 4: Inspection of objects composed of lines and circular arcs.

As shown in Figure 4, we adopted a test object (a mechanical part) from the literature and recreated it on a CAD system so that the designed parameters can be compared to those estimated using our algorithms. The aim of this experiment is to inspect the boundary by estimating the centers and radii of the circular arcs forming the boundary. The knot-points on the boundary are detected using the proposed knot-point detection algorithm as described in Section 2.4, so that the boundary is broken into primitive line segments and circular arc primitives. The parameters (the line length and the orientation for a line, and the center and radius for a circular arc) are then calculated for each segment. Table 2 shows the actual, the predicted and the estimated centers and radii of all the circular arcs on the boundary. It can be seen that centers and radii of the arcs estimated are equal, or very close to, the actual values. This shows that the parameter estimation algorithm described in Section 3.2 is accurate, and the verification technique proposed here is indeed valuable for applications in which the estimation of centers and radii of circular arcs is required.

Arcs	Actual			Predicted			Estimated (±0.1)		
	radius	center		radius	center		radius	center	
		x	y		x	y		x	y
arc 1	23	243	66	24	244	68	23	243	66
arc 2	50	225	136	50	224	132	50	225	136
arc 3	28	228	137	28	228	137	28	228	137
arc 4	18	233	197	18	233	197	18	233	197
arc 5	88	246	139	87	243	136	88	246	139
arc 6	23	157	144	23	157	144	23	157	144
arc 7	79	239	130	79	242	126	79	239	130

Table 1: **Comparison of the designed, predicted and estimated parameters.**

4 Conclusions

In this paper, we have developed several algorithms for verification of regular-shaped objects by breaking boundaries into fundamental primitives such as lines and circular arcs.

We first emphasized the the importance of critical points derived from object boundaries for the recognition and verification of industrial objects. To extract these dominant features, we have discussed algorithms for detection of extreme points having high curvatures, and have then proposed an algorithm for detecting knot-points formed by basic primitive segments such as lines and circular arcs.

The results obtained through effective curvature analysis have formed a valuable working base for the development of the verification algorithms. To develop the structural verification algorithm, we have classfied the existing flat industrial objects into four categories. For each category, we have developed corresponding effective verification algorithms, according to the characteristics of the objects in each class.

Acknowledgements

The authors wish to acknowledge the support of their employers, the University of Wales, for the work which has been discussed in this paper. They also wish to acknowledge the ACME Directorate of the Science and Engineering Research Council of the UK which now support the on-going implementation of the work.

References

[1] H. Asada and M. Brady. The Curvature Primal Sketch. *IEEE Trans. on PAMI*, vol. PAMI-8, January 1986.

[2] F. Attneave. Some Information Aspects of Visual Perception. *Psychological Review*, vol. 61, pp. 183–193, 1954.

[3] J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, November 1986.

[4] X. Cao and F. Deravi. Efficient Method for Multiple-Circle Detection. *Internal Report*, 1990.

[5] X. Cao, M.G. Rodd, F. Deravi, and Q.M. Wu. Detection of Multiple Circles Based on Adaptive Hough Transform. *Engineering Application of AI*, vol. 1, pp. 97–101, June 1988.

[6] M.A. Fischler and R.C Bolles. Perceptual Organisation and Curve Partitioning. *IEEE Trans. on PAMI*, vol. 8, pp. 100–104, January 1986.

[7] H. Freeman and L. S. Davies. A Corner-Finding Algorithm for Chain-Coded Curves. *IEEE Trans. on Computers*, vol. C-26, pp. 297–303, March 1977.

[8] M. H. Hueckel. An Operator Which Locates Edges in Pictures. *Journal of the ACM*, vol. 18, pp. 113–125, January 1971.

[9] L. Kitchen and A. Rosenfeld. Greylevel Corner Detection. *Pattern Recognition Letter*, vol. 1, p. 95.

[10] U.M. Landau. Estimation of a Circular Arc Centre and Its Radius. *Computer Vision, Graphics and Image Processing*, vol. 38, pp. 317–326, 1987.

[11] G. Medioni and Y. Yasumoto. Corner Detection and Curve Representation Using Cubic B-splines. *CVGIP*, vol. 39, pp. 267–278, 1987.

[12] F. Mokhtarian and A. Mackworth. Scale-Based Description and Recognition of Planar Curves and Two-Dimensional Shapes. *IEEE Trans. on PAMI*, vol. PAMI-8, no. 1, pp. 34–43.

[13] T. Pavlidis and S.L. Horowitz. Segmentation of Plane Curves. *IEEE Trans on Computers*, vol. C-23, August 1974.

[14] J. Powrie. *A Critical Evaluation of Low-Cost Boundary Extraction Schemes for Industrial Inspection Applications.* University of Wales, Department of Electrical and Electronic Engineering, 1990.

[15] C.H. Tek and R.T Chin. On the Detection of Dominant Points on Digital Curves. *IEEE Trans. on PAMI*, vol. 11, pp. 859–872, August 1989.

[16] Qing Ming Wu. *Knowledge-Based Algorithms and Architectures for Industrial Inspection.* PhD thesis, University of Wales, Dept. of Electrical and Electronic Engineering, University College of Swansea, 1990.