Fourth International Conference on knowledge-Based Intelligent Engineering Systems & Allied Technologies, 30th Aug-1st Sept 2000, Brighton,UK

Extracting Geometric Models from Medieval Moulding Profiles for Case-Based Reasoning

Gareth Bradshaw* and Carol O'Sullivan
Image Synthesis Group, Dept. of Computer Science, Trinity College, Dublin, IRELAND

Abstract

Cross-sectional profiles of medieval mouldings are often considered to be a vital form of data to Art Historians. Mouldings were often used to highlight salient areas of buildings, and they provide a wealth of information. To date little use of computer technology has been made in their acquisition, storage and analysis. This paper presents a representation for such planar curves, and an algorithm for its extraction from planar curves. The representation uses geometric primitives specifically line and arc segments, which allows the curves to be represented in a compact form, approximating the circular sections more effectively than using linear segments alone. These idealised models provide templates which provide a means for the identification of specific features, within profiles, which can then serve as characteristics for use in case-based reasoning.

1 Introduction

Medieval buildings are often considered to be one of the most beautiful types of architecture. One of their most notable features is the widespread use of carved stone to decorate salient structures, such as pillars, arches and the surrounds of windows and doors. There is a strong desire among *Art Historians* for preservation of these works. It is often not possible to prevent the deterioration of the stone work, due to environmental and financial difficulties. One collection of stones was actually buried for later retrieval. Further details on these mouldings may be found in [13, 14, 15, 16, 17, 23].

The vast majority of the mouldings found in buildings consist of a pattern which runs along the length of an arc, or radially around an axis, see Figure 1. Traditionally historians have described these mouldings by using a cross-sectional curve, often referred to as a "moulding profile". These profiles are often recorded on paper, having been measured using a variety of manual techniques, which are often unreliable, time consuming and require substantial expertise and care. Their storage requires vast amounts of paper to be filed.

Another important reason for keeping records of these works, is that is has long been believed that their structure contains information about their origins, as a particular mason, or school of masons, would favour certain features or sequences of features. Thus when trying to reason about the origins of a particular stone it is essential to be able to perform cross-referencing. Maintaining a manual indexing of a large collection of paper can be very time consuming and doesn't lend itself to comprehensive analysis. The representation of moulding profiles as digital curves reduces the storage requirements considerably and, allows large collections to be widely accessable.

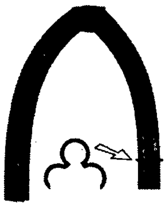
Reducing the size of the representation has important effects on the cost of performing comparisons and indexing, which are important factors in case-based reasoning [12, 20]. In the rest of this paper we describe a technique for extracting a compact representation from a digital planar curve. The representation utilises geometric primitives, lines and circles, to represent the geometry of the moulding. This not only provides a more compact representation, but also tries to represent the geometric constructs which were used in the original design. This can be justified by observing that the masons used relatively simple geometry, the arrangement of which evolved over time [13, 14, 15].

The rest of the paper is arranged as follows; in Section 2 we make reference to traditional and machine vision based techniques for collecting these profiles and introduce the form of data we consider. In Section 3 we introduce the algorithm, detailing the phases involved in approximating the models. In Section 4 we compare the algorithms performance to that of one which uses linear segments alone. The paper ends in Section 5 with conclusions and future work.

2 Profile Acquisition

As mentioned in Section 1 traditional methods for recording moulding profiles use manual techniques for the acquisition phase. These include shaping malleable wire across the surface of the stone (this is then transferred to paper); tracing the outline of the stone when the stone is ex situ; the measurement of the depth of the folds from a reference line; and using a template former which is a comb with mobile

*e-mail: Gareth.Bradshaw@cs.tcd.ie


(a) Arch

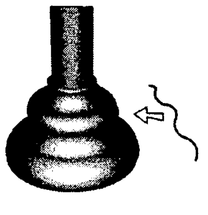

(b) Base

Figure 1: Rendered Examples of Medieval Mouldings and their Cross-Sections

teeth. All of these methods usually result in a drawing of the cross-section of the stone, which can then be digitised using a digitising tablet, or scanned and extracted using segmentation etc.

The techniques mentioned all rely on the expertise of the person doing the work, thus considerable care needs to be taken to ensure quality and consistency. As these methods all require contact with the surface they pose the risk of potentially damaging the surface. A better alternative is to use vision based sensors, such as those reviewed in [3] and [11] . These sensors usually produce a set of range measurements, in a point, line or area configuration. Most commonly a line based configuration is used as this provides the most cost effective balance between speed and complexity [3, 11].

Whichever technique is used for performing the measurements it is rarely the case that a moulding can be digitised in a single pass. This is due mainly to the limited working range of the scanner, and self-occlusion[1]. In order to be able to combine the sets of measurements (segments) into a single curve it is necessary to know their relative positions. This can be achieved by mounting the sensor on a kinematic arm (such as a Faro-Arm^{TM}), or attaching it to a magnetic position sensor (such as a Polhemus IsoTrak^{TM}). Much work has been done on merging sets of points and forming compact surface triangulations [1, 2, 5, 6, 8, 9, 10, 19, 24] . As we are only

[1]Self-occlusion is where the surface dips behind the body of the object and thus is invisible to the scanner.

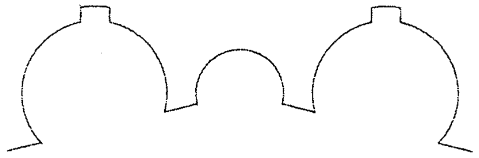


Figure 2: Planar curve representing the profile of a medieval moulding

interested in the cross-section of the moulding, the input to the algorithm is essentially a planar version of the surface triangulation. The curve is constructed as a set of points joined by linear segments, see Figure 2.

3 The Algorithm

The algorithm consists of three phases. The input is a planar curve consisting of connected points. The output is a (connected) set of geometric primitives. In this paper we limit this set to lines and circles, but the algorithm can be easily extended to include higher order primitives, such as ellipses. Such a representation will ease the tasks of storage, retrieval and feature extraction in a case-base. The phases of the algorithm can be summarised as follows :

- **Approximation** of the planar curve by a set of linear segments
- **Fitting** of geometric primitives to these linear segments
- **Connecting** the primitives into a connected chain representation

3.1 Approximation

As mentioned in Section 2 the input data is a set of connected points, which originate from either a scanned image or a device such as a laser scanner. As with any source of data there is inevitably a certain level of noise, and the data is usually densely spaced with no emphasis on dynamic levels of detail. Thus the initial phase of the algorithm aims to reduce the amount of data required for the representation. This is achieved by using a dynamic sampling strategy, concentrating effort on areas of high detail. The multiresolution representation not only reduces the amount of data, but also reduces the effects of noise.

The algorithm is based on the greedy insertions techniques used for surface tessellation and terrain rendering [4, 6, 8, 9, 10]. In surface tessellation algorithms the approximation is iteratively improved by the addition of a single vertex/point and the surface is represented by triangles connecting the vertices. In our algorithm, the input data is not assumed to be a reasonable approximation of the surface points. Thus instead of inserting a vertex we insert a break, which splits one of the linear segments in two. These linear segments approximate the surface by fitting *Least Squared Lines* [2]. The algorithm starts with an initial approximation, and uses a number of criteria to determine which line to split, where to split it and when to stop.

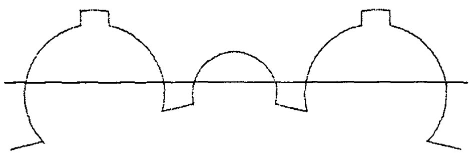
There are a number of alternatives for the choice of the initial approximation. The simplest is to use a single least squared line, see Figure 3(a). In [7] Weinshall uses a number of lines joining points of high curvature for creating approximations of planar curves for weak pattern matching. Schmid *et. al.* [22] also use curvature for segmenting boundary curves. This has the advantage of ensuring that corners are maintained.

At each iteration of the algorithm it is necessary to decide which segment needs to be split, the one which is considered to be the worst approximation seems like a logical choice. The measurement of how good an approximation each segments provides can be chosen from many alternatives e.g. *Worst Error, Average Error, Root Mean Square Error* etc. For our purposes we have chosen to use the *Worst Error*, which is defined in Section 3.3 Equation 3. It is also necessary to specify where to split the chosen segment. Many researchers introduce a vertex at the point which is furthest from the current approximation [4, 6, 8, 9, 10], but since we are fitting least squared lines this is not desirable, as no one point is considered to be correct. Therefore it is necessary to perform a test break at a number of points to find the best place to break the segment. A less expensive alternative is to simply break the line segment in the middle, see Figure 3(b), although this is may be sub-optimal.

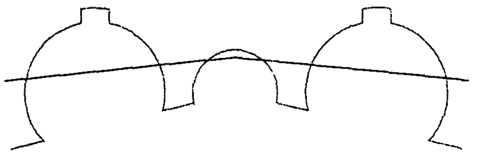
The algorithm terminates when the current approximation is within a threshold distance τ_s of the input data. As with the breaking criteria there is a choice of the actual measure of distance. Our algorithm stops when the *Worst Error* is less than the supplied threshold τ_s.

3.2 Fitting

At the beginning of this phase of the algorithm we have an unconnected set of lines, which fit the data in a least squared manner. In this phase we fit geometric primitives [3] to these lines. As we discussed in Section 1 it is believed that masons used such primitives to design the mouldings. Hence they should provide an accurate and compact representation of the data. The algorithm iteratively combines neighbouring segments into lines or circles. Using the same metric as in Section 3.1, the two segments which have



(a) Initial Approximation



(b) First Split

Figure 3: Greedy Splitting Algorithm with a single line for the initial approximation and using a midline split to improve the approximation

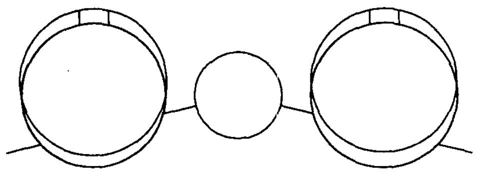


Figure 4: The model resulting from the fitting of lines and circles

the lowest resulting error are combined. This process is similar in operation to vertex removal or edge collapsing, used in surface triangulation and terrain rendering, and continues while the resulting error is less than a specified threshold τ_m.

As the arc formed by an extremely large circle approximates a straight line, it is possible for the algorithm to try to use a large circle where a line would be considered to be correct. In order to prevent this from occurring short arcs and large circles are forbidden. Also, as the circle is a higher order primitive than a line, a penalty ζ_c is associated with fitting a circle. Thus a circle is only ever used when the arc is long enough, the circle is of an acceptable size and $E_c * \zeta_c < E_l$, where E_c and E_l are the errors associated with fitting a circle and line respectively.

3.3 Connecting

The fitting phase of the algorithm yields a set of lines and circles which closely approximate those present in the input data. We have also conducted experiments using a modified version of *The Constrained Hough Transform* [18] to obtain the set of lines and

[2] *Least Squared Lines* are found using modified *Linear Regression*.

[3] The circles are formed using an iterative approximation algorithm to find its parameters (center and radius).

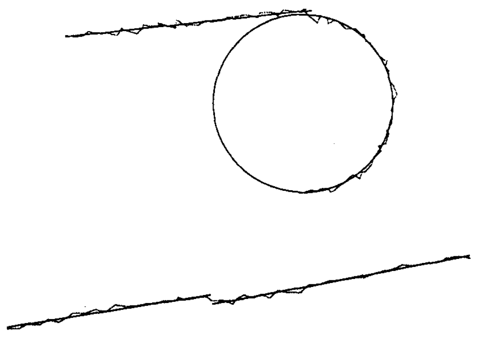
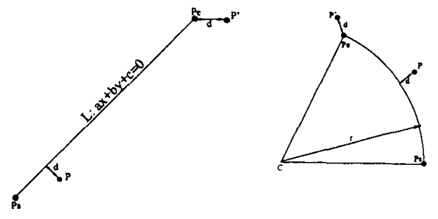


Figure 5: Examples of bad intersections between neighbouring fitted primitives (a) circle and line for which don't intersect, (b) two lines whose intersection lies far off to the right



(a) Line (b) Circle

Figure 6: Distance from Line Segment and Arc

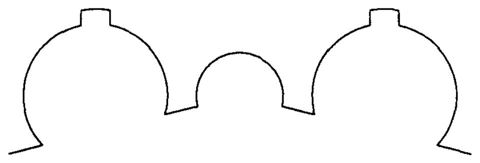


Figure 7: The connected set of segments resulting from the connecting algorithm

circles present in the data. The final stage of the algorithm aims to constrain the set of primatives such that they form a connected chain. This is necessary as the fitting stage doesn't make any attempt to consider the primitives relative to their neighbours. In this optional phase, a non-linear optimisation problem is formulated, which fits a connected series of lines and arcs to the data. The problem is formulated to find the optimal positions for the intersection (end) points of the segments, and an interior point for each arc. In order to formulate the optimisation problem, it is necessary for us to specify initial estimates for the variables (points) and an evaluator for the error in the approximation.

When supplying the initial estimate for the points, it is not always possible to use the intersection point of the neighbouring segments, as their least squared nature can prevent them from intersecting at a reasonable point, if they intersect at all, as illustrated in Figure 5. Thus the mid point between the ends of the segments is used.

The optimising algorithm needs to evaluate the residual error ε_i at each of the N input points, giving us a cost function:

$$E = \sum_{i=1}^{N} \varepsilon_i^2. \tag{1}$$

Thus, similar to the *Hausdorff Distance* we have defined the residual ε_i error of point P_i to be :

$$\varepsilon_i = \min_{j=1}^{M} D(P_i, S_j), \tag{2}$$

where segment S_j is the line/circle through the corresponding two/three points, from the set of M segments, and the distance measure D is defined, as follows :

$$D(P,S) = \begin{cases} D_c(P,S) & \text{if } S \text{ is a Circle Arc} \\ D_l(P,S) & \text{if } S \text{ is a Line Segment} \end{cases} \tag{3}$$

As the lines are finitely long segments, and we are using a subsection of the circle, it is necessary to define a special distance measure, as illustrated in Figure 6, which accounts for the shortest distance to the segment. This for a line we use the perpendicular when the point projects onto the segment :

$$D_l(P,L) = \begin{cases} \left\| \frac{L_a P_x + L_b P_y + L_c}{\sqrt{L_a^2 + L_b^2}} \right\| & \text{if } P \text{ projects onto } L \\ \min(\mid P - P_s \mid, \mid P - P_e \mid) & otherwise \end{cases} \tag{4}$$

and, similarly for the arc :

$$D_c(P,C) = \begin{cases} \| C_r - \mid P - C_c \mid \| & \text{if } P \text{ projects onto } C \\ \min(\mid P - P_s \mid, \mid P - P_e \mid) & otherwise \end{cases} \tag{5}$$

Figure 7 shows the model featured in Figure 4 having been optimised using the cost function (1).

4 Analysis

In order to evaluate our algorithm, we have compared it to one which uses only linear segments. We constructed a number of models which contain both arc and line segments. From these a set of points was generated, introducing noise by randomly perturbing the points, using a *Linear Congruential* random number generator [21]. We are interested in how well the error is distributed throughout the model, and the number of primitives taken to construct it.

Figure 8(a) [4] shows the RMS error between the fitted models and a set of perfect points taken from the original model. The RMS error for N points is evaluated as :

$$E = \sqrt{\frac{\sum_{i=1}^{N} \varepsilon_i^2}{N}}, \qquad (6)$$

where ε_i is as defined in Section 3.3 Equation 2. The ε_i^2 term is used so as to favour a number of small errors over a single large one.

From the graph it is easy to see that our algorithm is better at distributing the error than one using lines alone. This is to be expected as the models contained circular sections which are poorly represented by linear segments. The performance of our algorithm does degrade when the applied error approaches the merging threshold τ_m, as the algorithm starts to diverge from the original model and tends towards a closer approximation of the input data. However, it is still no worse than using lines alone.

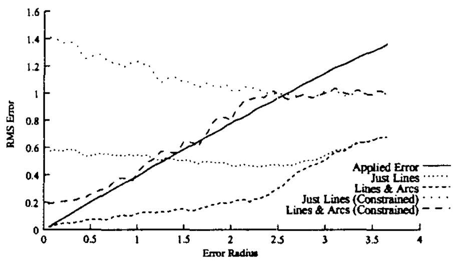
Figure 8(b) shows the number of primitives necessary to approximate the input curve to the desired level. Our algorithm requires fewer primitives than one which uses lines alone, this is due to circular areas being poorly approximated by lines. Also apparent is an increase in the number of primitives used when the applied noise exceeds τ_m and the fitted model diverges from the original.

5 Conclusions and Future Work

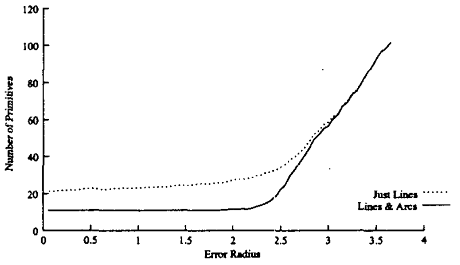
An algorithm for the approximation of planar curves, using not only linear segments but also arc segments, was presented. The algorithm performs best when the merging threshold τ_m is greater than the level of noise in the curve, and at worst performs as least as well as using linear segments alone. The number of primitives is also reduced, being at worst on par with using linear segments alone.

A distance measure for evaluating the goodness of fit was utilised, which also provides a similarity measure for extracting features, based on these models, from moulding profile curves. These features will form an integral part of a case-based system for storing and retrieving similar profiles based on their presence.

[4] For the graphs in Figure 8 $\tau_s = 0.25$ and $\tau_m = 2.5$. Results are averages over 15 applications of the algorithms with 1000 points for fitting and 5000 for RMS calculations.



(a) RMS Error



(b) Number of Primitives Fitted

Figure 8: The number of primitives fitted and the RMS error for our algorithm and a purely line based one

The algorithm presented used only circles and lines. However it is easily extendible to use other primitives, such as ellipses, and could be extended to approximate non-planar curves.

6 Acknowledgements

We would like to thank Rachel Moss and Prof. Roger Stalley of The Department of The History of Art, Trinity College, Dublin for their input and support. This research is funded by Enterprise Ireland Basic Research Grant SC/98/604.

References

[1] A.P. Ashbrook et al. Segmentation of range data into rigid subsets using surface patches. Technical report, Dept. of Artificial Intelligence, The University of Edinburgh.

[2] Chandrajit L. Bajaj, Fausto Bernardini, and Gouliang Xu. Automatic reconstruction of surfaces and scalar fields from 3d scans. *Computer Graphics Proceedings*, 1995.

[3] Gareth Bradshaw. 3d sufrace geometry mesaurement technique. Technical report,

Image Synthesis Group, Dept. of Computer Science, Trinity College Dublin, October 1999. Available online from www.cs.tcd.ie/publications.

[4] H.L. de Cougny and M.S. Shephard. Surface meshing using vertex insertion. *5th International Meshing Roundtable*, pages 243–256.

[5] David W. Eggert, Andrew W. Fitzgibbon, and Robert B. Fisher. Simultaneous registration of multiple range views for use in reverse engineering of cad models. Technical report, Dept. of Computer Science, University of New Haven et. al., 1996. Appeared at the 1996 International Conference on Pattern Recognition.

[6] Michael Garland and Paul S. Heckbert. Fast polygon approximation of terrain and height fields. Technical report, School of Computer Science, Carngie Mellon University, 1995.

[7] Yoram Gdalyahu and Daphna Weinshall. Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes. Technical report, Institute of Computer Science, The Hebrew University.

[8] Paul S. Heckbert and Michael Garland. Survey of polygonal surface simplification algorithms. Technical report, School of Computer Science, Carngie Mellon University, 1997.

[9] Hugues Hoppe. *Surface Reconstruction from Unorganised Points*. PhD thesis, 1994.

[10] Hugues Hoppe. Progressive meshes. *SIGGRAPH 1996 Proceedings*, pages 99–108, 1996.

[11] Reinhard Klette. Image analysis and object surfaces. Technical report, The University of Auckland, Augusy 1997. CS-TR-152/CITR-TR-9.

[12] Janet Kolodner. *Case-Based Reasoning*. Morgan Kaufmann Publishers Inc., 1993.

[13] H.G. Leask. *Irish Churches and Monastic Buildings, vols 1-3*. Dundalgan Press, 1955-60.

[14] R.K. Morris. A development of later gothic mouldings in england c.1250-1400; part 1. *Architectural History*, 21:18–57, 1978.

[15] R.K. Morris. A development of later gothic mouldings in england c.1250-1400; part 2. *Architectural History*, 22:1–48, 1979.

[16] R.K. Morris. An english glossary of medieval moldings:with an introduction to mouldings c.1040-1240. *Architectural History*, 35:1–17, 1992.

[17] R.K. Morris. *Mouldings*. Grove Dictionary of Art. 1996.

[18] Clark F. Olson. Constrained hough transforms for curve detection. *Computer Vision and Image Understanding*, 73;3:329–345, 1999.

[19] Steven J. Owen. A survey of unstructured mesh generation technology. Technical report, Dept. of Civil and Environmental Engineering, Carngie Mellon University.

[20] Enric Plaza and Agnar Aamodt. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AICom - Artificaial Intelligence Communications*, 7;1:39–59, 1994.

[21] William H. Press et al. *Numerical Recipes in C, Second Edition*. Cambridge University Press, 1988-1995.

[22] G. Schmid, L.Altamirano Robles, and W.Eckstein. Automatic segmentation of boundaries in line segments and cirular arcs. Technical report, Institut für Informatik, Technische Universität München.

[23] R. Stalley. *The Cistercian Monasteries of Ireland*. Yale University Press, 1987.

[24] Greg Turk and Marc Levoy. Zipped polygon meshes from range images. Technical report, Computer Science Dept., Stanford University.