3-D Curve Matching Using Splines

Eyal Kishon and *Trevor Hastie*
AT&T Bell Laboratories Murray Hill, NJ 07974
Haim Wolfson
Robotics Research Laboratory New York University, and
Computer Science Department Tel Aviv University Tel Aviv 69 978, Israel

Abstract
A machine vision algorithm to find the longest common subcurve of two 3-D curves is presented. The curves are represented by splines fitted through sequences of sample points extracted from dense range data. The approximated 3-D curves are transformed into 1-D numerical strings of rotation and translation invariant *shape signatures*, based on a multi-resolution representation of the curvature and torsion values of the space curves. The *shape signature* strings are matched using an efficient hashing technique that finds longest matching substrings. The results of the string matching stage are later verified by a robust, least-squares, 3-D curve matching technique, which also recovers the Euclidean transformation between the curves being matched. This algorithm is of average complexity $O(n)$ where n is the number of the sample points on the two curves. The algorithm has applications in assembly and object recognition tasks. Results of assembly experiments are included.

1 Introduction

Curves in 3-space carry a large amount of information about the scenes they appear in, and can successfully characterize objects drawn from large sets of candidates. The curves can either be 'painted curves' (i.e. curves that correspond to changes of reflectivity without any rapid local depth changes), curves of intersection between two surfaces (either convex or concave), curves of occlusion (i.e. object boundaries), or curves of maximum curvature ('ridges'). The data required to characterize objects by this method reduces to a small group of curves extracted from the object, and is is extremely compact as compared to the full 2-D surface specification. Matching algorithms based on 3-D curves are simpler and more efficient than surface matching algorithms, because the points that represent a 3-D curve are naturally ordered as a sequence.

Schwartz and Sharir ([2]) developed a matching algorithm which finds the position and orientation of an observed curve best matching (in the least squares sense) a previously stored model curve. This algorithm was proven to be robust for practical applications, and was extended to support cases in which several subcurves had to be simultaneously matched against the same curve (see [1] for extended bibliography).

The algorithm by Schwartz and Sharir requires the observed curve to be a proper sub-segment of the stored model curve. Such prior segmentation is not always available in composite scenes of overlapping objects. This problem is particularly evident for assembly tasks, where two objects match along a common boundary without any obvious start point and end point for the common subcurve (see figure 1). Thus we were motivated to develop a general curve matching algorithm that solves the following problem:
Given two curves, find the longest matching subcurve which appears in both curves.

We reduce the 3-D curve matching task into a 1-D string matching problem, find the proper matching substrings, transform the problem back to the 3-D domain, and match the appropriate subcurves. Specifically, the 3-D curves are transformed into sequences of *local*, *rotationally*, and *translationally invariant* shape signatures. We then apply a hashing technique to find long matching substrings. Finally, we apply the least squares algorithm to select the best candidate. As a by-product we obtain the Euclidean transformation aligning both curves along their longest matching subpart.

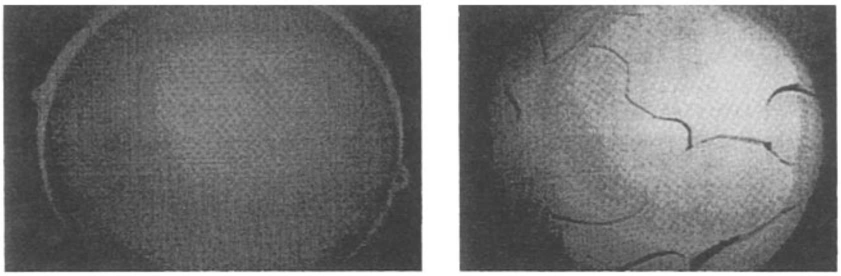
Figure 1: (left) A plastic ball (right) The 'broken' ball assembled from the separate pieces

2 Shape Signatures

In this section we describe the shape signatures which are used in our matching algorithm. A signature should uniquely characterize a relatively short segment of the curve. Hence we require signatures to be:

i) local (calculated at each point),

ii) translationally and rotationally invariant,

iii) stable, so that small changes in the curve induce small effects on the signature.

It is well known from Differential Geometry that smooth space curves can be uniquely reconstructed within a rigid motion (i.e rotation and translation) using three geometric invariants: arc length s, curvature $\kappa(s)$ and torsion $\tau(s)$ as a function of s.

Since curvature and torsion are essentially second and third order derivatives respectively, it is essential to smooth the data a small amount before computing the signatures. Regression splines are a convenient class of smoothers for this task, since they permit one to easily compute derivatives. We chose a quintic spline representation (order 6) since we want the torsion to be continuous, which in turn will make the signature based on the torsion more stable.

3 The Matching Algorithm

This section describes the *shape signature* matching algorithm applied in our experiments.

All the curves in the data-base are preprocessed as follows. The algorithm accepts as its input the curvature-torsion signature strings computed at different levels of resolution. For each signature we record the *curve number* and the *sample point number* at which this signature was generated. The data is stored in a hash-table.

In the matching stage, the observed curve is sampled and signatures (at different levels of details) are computed at the sampling points. For each signature we check the appropriate entry in the corresponding hash-table, and for every (*model curve, sample point*) pair, appearing in the hash table, we add a vote for this model curve and the relative shift between the model curve and the observed curve.

At the end of this process we determine which *(model curve, shift)* pairs received the most votes, and determine the approximate start and end points of the corresponding signature substrings in the observed and model curve.

Given the starting point and endpoint of a signature subsequence, we identify the actual subcurve to which they correspond, and apply the robust least-squares matching algorithm to the corresponding subcurves.

For a detailed description of the matching algorithm see [1]. The algorithm is of average complexity of $O(n)$, and improves the result of Schwartz and Sharir which is of complexity $O(nlogn)$.

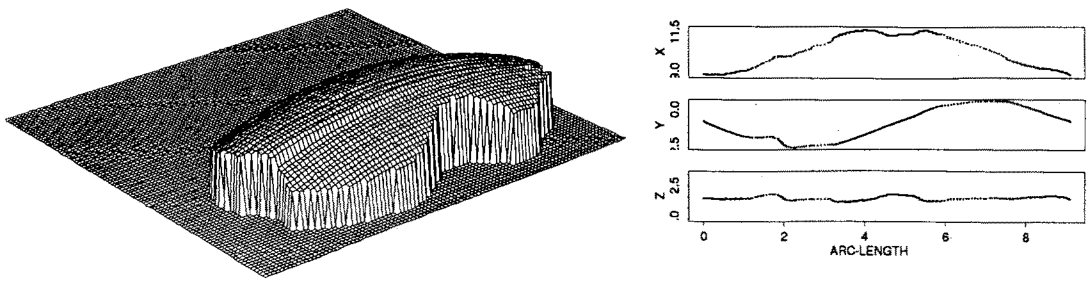

Figure 2: (left) Range image (right) boundary curve X,Y,Z

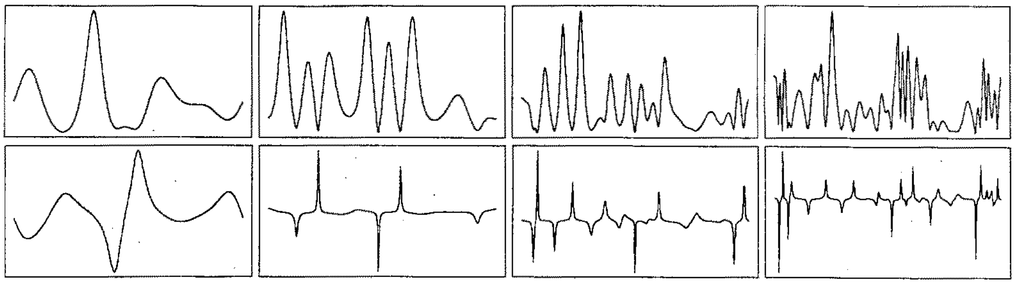

Figure 3: (top) Curvature (bottom) torsion computed at four different resolutions

4 Experimental Results

Figure 2 shows a perspective view of the range image of one of the pieces, and the boundary curve extracted from that piece. Figure 3 shows the curvature and torsion derived from the curve for the 4 different levels of smoothing. Figure 4 show the results of matching individual pieces of the ball and finding the best subcurve match.

References

[1] E. Kishon and H.J. Wolfson. 3-D Curve Matching. In *Proceedings of the AAAI workshop on Spatial Reasoning and Multi-Sensor Fusion*, pages 250–261, St. Charles, Ill., 1987.

[2] J. T. Schwartz and M. Sharir. Identification ot Partially Obscured Objects in Two or Three Dimensions by Matching of Noisy Characteristic Curves. *The International Journal of Robotics Research*, 6(2):29–44, 1987.

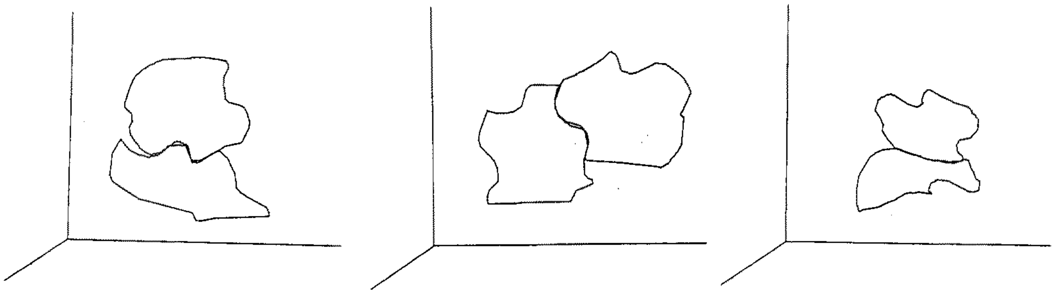

Figure 4: The results of matching four different pieces