SLAC Computation Research Group
Stanford, California

CGTM No. 161
October 1974

EXPERIMENTS WITH A GENERAL DOTTED CURVE FINDER

Vito DiGesu and Charles Zahn

ABSTRACT

This paper reports preliminary results of a program to recognize dotted curves from a set of points in two-dimensional space. Part of the program is similar to an earlier program [1] which was used successfully to recognize particle tracks in streamer chamber photographs. The improved version of the program tolerates a substantial amount of noise and will recognize a wide variety of curves whose directions vary smoothly. Sample problems are generated by a Monte Carlo procedure which adds noise of three kinds in specified amounts. Several pictures demonstrate the quality of recognition relative to the noise level. The generated curves and noise present difficulties typically encountered in digitized data from particle physics photographs, but the recognition method is not restricted to this form of data.

General Dotted Curve Recognition

A program capable of organizing a set of points in the plane into a number of smooth curves will be called a dotted curve recognizer. We assume that the criterion of success is that the resulting curves are consistent with the impression a human receives (see Figure 1) upon viewing a graphic display of the dots. The approach developed here is, therefore, a conscious attempt to simulate the effects of human perceptual organization, especially the two Gestalt principles of "proximity" and "good continuation". As in previous work [1], the recognition is divided into two parts, first the detection of short sequences of closely spaced points forming very oblong (nearly linear) shapes, then the linking of these short "segments" together into longer curves with slowly varying directionality.

The dots are organized into segments by a program employing the minimum spanning tree as the low-level perceptual organizer. Paths are extracted from the minimum spanning tree and are recursively subdivided into nearly linear segments. At this lowest level, the sequences of points are determined by the proximity of points as measured by the usual Euclidean distance in the plane. The next level of organization, although not employing the minimum spanning tree, nevertheless uses a measure of "closeness" between segments which is designed to approximate the good continuation principle described by Gestalt psychologists. This closeness measure reflects the direction of segments as well as their relative proximity in the plane. The quality of curve recognition depends, critically, on the closeness between segments, and the function used in this work will be described in more detail later.

From Dots to Segments

The program which transforms a set of planar points into a set of segments begins by constructing the minimum spanning tree (MST) of the points.*

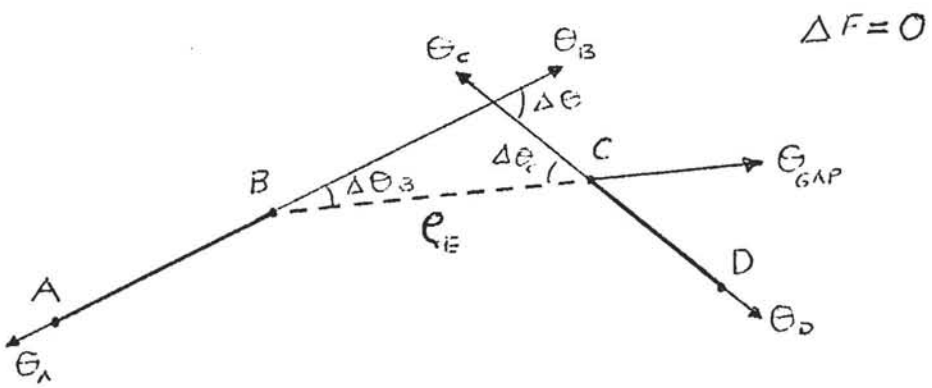
*Readers not familiar with graph-theoretical terminology may consult [2].

It then deletes all hairs (edges connecting nodes of degree 1 to nodes of degree > 2) and all nodes of degree > 3. The remaining nodes of degree 3 have their longest edge deleted and the tree then consists of nodes of degree ≤ 2, that is, a set of disjoint paths.

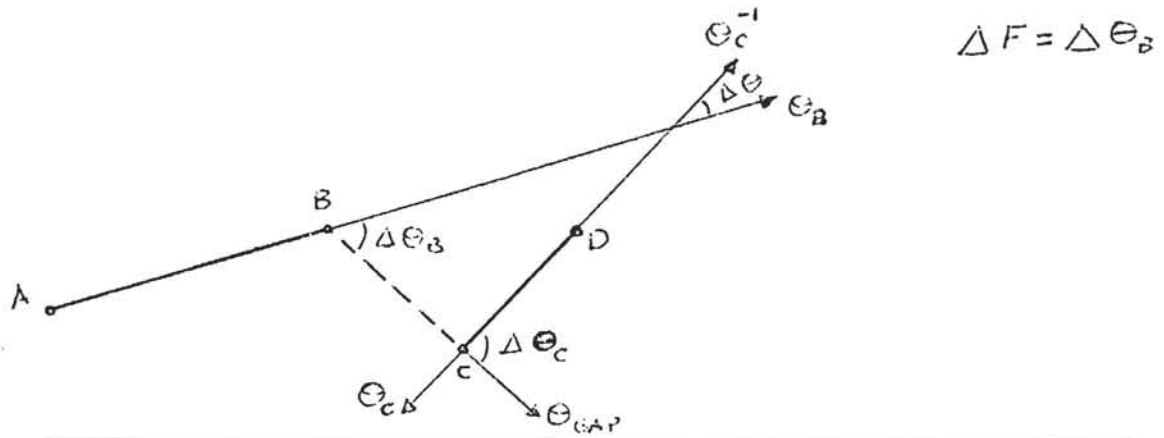
Each of these paths is broken down recursively until the remaining portions of path represent almost linear sequences of points. The method known as "iterative endpoint fit" was used to recursively break non-linear paths into adequately linear subpaths. Each subpath is considered to be a line segment in the plane, determined by the two end-nodes of the subpath. The reader is referred to [1] for further details.

From Segments to Curves

Each segment is considered as a pair of endpoints and each endpoint is recorded as a coordinate position (X,Y) and an angular direction θ, as depicted below. The angle θ is always tangent to the line segment and is directed away from the segment. Between each pair of segments we define a measure of spatial proximity ρ_E and a measure of directional compatibility ρ_θ. The spatial measure ρ_E is simply the distance between the closest endpoints, one from each segment (see below). The first ingredient of the angular measure ρ_θ is $\Delta\theta$, which represents the difference in angular direction between θ_B and θ_C^{-1} where α^{-1} is the angular direction opposite to direction α and B,C are the closest endpoints determining the distance ρ_E.



The second ingredient of ρ_θ is ΔF which measures the extent to which the polygonal curve (A,B,C,D) has an inflection. If θ_{GAP} is the angular direction of the line (gap) BC, then let $\Delta\theta_B$ be the angular difference $|\theta_{GAP} - \theta_B|$ and let $\Delta\theta_C$ be similarly equal to $|\theta_C^{-1} - \theta_{GAP}|$. An inflection occurs precisely if the two angular differences $\Delta\theta_B$ and $\Delta\theta_C$ are bends of opposite sense and this is equivalent to the inequality $\Delta\theta \neq \Delta\theta_B + \Delta\theta_C$. If $\Delta\theta = \Delta\theta_B + \Delta\theta_C$, then we let $\Delta F = 0$; otherwise $\Delta F = \min\{\Delta\theta_B, \Delta\theta_C\}$.



As a measure of angular compatibility ρ_θ we use the following function of $\Delta\theta$ and ΔF

$$\rho_\theta = \tan\left(\frac{\pi}{2} \cdot \min\left\{\frac{\Delta\theta}{\Delta\theta_{MAX}} + \left(\frac{\Delta F}{\Delta F_{MAX}}\right)^2, 1\right\}\right)$$

The values $\Delta\theta_{MAX}$ and ΔF_{MAX} are parameters which act as thresholds for $\Delta\theta$ and ΔF respectively in the sense that $\rho_\theta \to \infty$ as $\Delta\theta \to \Delta\theta_{MAX}$ or $\Delta F \to \Delta F_{MAX}$. In this work we have used $\Delta\theta_{MAX} \simeq \pi/3$ and $\Delta F_{MAX} \simeq \pi/9$.

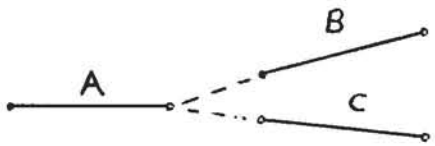
The two measures ρ_E and ρ_θ are used to form a combined closeness measure ρ defined by

$$\rho = \sqrt{\rho_E^2 + (K_\theta \cdot \rho_\theta)^2}$$

where K_θ is a scale factor, required because ρ_E is a distance in the plane, but ρ_θ is the tangent of an angle. For K_θ, we have used a distance equal to one-fourth of the breadth of the picture area. This virtually equates an

angle of 45° with one-fourth the picture breadth because tan $45^{\circ} = 1$.

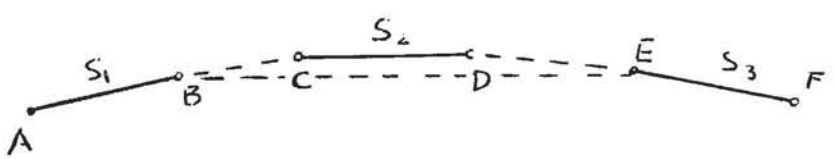
The above closeness measure ρ is a symmetric function of the two segments and, furthermore, $\rho = 0$ if and only if the two segments share a common endpoint and have identical direction. The function ρ violates the triangle inequality in a very strong way as can be seen from a simple example.



Segment A is very close to both B and C but B and C are not close at all; nor would we want ρ to define B and C as close.

The closeness measure ρ between segments is used to establish a set of links between certain pairs of segment endpoints, as follows. Whenever $\rho(S_1,S_2) < \rho_{MAX}$ for two segments S_1 and S_2 and E_1,E_2 are the endpoints of S_1 and S_2, which are closest, then we establish a two-way link between E_1 and E_2 and say that each endpoint is a neighbor of the other. The threshold ρ_{MAX} has been set to the same value as K_θ, the scale factor. Whenever $\rho_{MAX} = K_\theta$, it is not difficult to see that ρ exceeds ρ_{MAX} if $\Delta\theta > \Delta\theta_{MAX}/2$ or $\Delta F > \Delta F_{MAX}/2$. The effective thresholds for $\Delta\theta$ and ΔF are then exactly one-half what they would be if $\rho_{MAX} = \infty$. When a segment endpoint has more than one neighbor we select the nearest one (as measured by ρ) to be the preferred neighbor.

After each endpoint has been linked to its neighbors, we mark certain redundant links as shown below.



If segment S_1 is linked to S_2 via the pair of segment endpoints B,C, and S_2 is linked to S_3 via D,E, then a link B,E from S_1 to S_2 will be marked if such exists. The unmarked neighbors of an endpoint are called "branch points".

The next step is to generate a set of possible curves using the branch point links established above. An arbitrary segment is selected and we grow the curve first from one endpoint, then from the other. The curve is grown iteratively by choosing the closest branch point for the initial endpoint. In this context, closest means smallest ρ value. If the closest branch point is endpoint A of segment S_2, then we simply continue the curve growth at the other endpoint (opposite A) of segment S_2. This process is continued until some endpoint has no branch points or a segment is encountered which is already part of the curve being grown. When a curve has been grown at both ends of the initial segment, it is placed in a list of possible curves and a new curve is grown from an unused segment. For each curve, we measure the "size" (i.e., number of segments) and the "density" as given by the inverse of the average length of the gaps between segments. The curve growing may result in two or more possible curves which share some common segments.

To select the best curves from among the possible curves, we first delete all curves with few (≤ 2) segments. Then we compare all pairs of curves to detect common segments. If two curves have one or more segments in common, we determine the ratio of the number of non-shared segments between the two curves. If this ratio or its inverse is > 3, then the shorter curve is deleted. Otherwise, we check the relative densities of the two curves, deleting the sparser curve if it's density is significantly lower than the denser curve. The remaining curves represent the output of the dotted curve recognizer.

Arc Segments

The segments generated from dots and used to make curves were described

above as line segments in the plane. The restriction to very straight sequences of points, each approximating a line, has the disadvantage that information is lost at areas on the curve which have a small radius of curvature. In an attempt to remedy this problem, we have modified the programs so that the segments may be linear or short arcs of circles. Each short sequence of points not adequately linear is tested for approximate circularity and may thereby become an arc segment, rather than being subdivided until its parts are so small they are rejected.

This generalization of segments was easy to incorporate into the curve finder because all subsequent use of segments considers them as pairs of endpoints, each with a direction out of the segment. Arc segments fulfill these requirements; they differ only in that the endpoint directions are not parallel as they are for linear segments.

The effect of this modification has been to help the curve finder continue around some fairly sharp bends without breaking the curve. The gain in quality of recognition has been only modest, but it is an improvement.

Data Generation and Results

The program which generates sample problems begins by generating points along an analytic curve, with interpoint spacing determined from random Poisson variables. Each point is then perturbed by the addition of a random vector drawn from a two-dimensional Gaussian (normal) distribution, with small variance σ centered at the origin. The ratio between σ and the average interpoint spacing along the curve seems to be a relevant measure of the smoothness of the dotted curve. After the dotted curve or curves have been generated and perturbed, then we add uniformly distributed background noise in a quantity specified as a percentage of the total number of points on curves. Finally, bursts of highly concentrated noise are added by choosing approximately 10 points from a uniform distribution on a small circle and placing them in the picture area at random.

Figures 1 and 2 display two pictures generated using the random dotted curve generation program. Figure 1 represents rather small perturbations of the curve points and background noise of only 10%, but there are 10 bursts whose destructive affect can be seen very clearly from the display of segments. Figure 2 represents more perturbation of the curve points and more background noise (20%), with only 5 bursts. With two curves, the complexity of the picture is somewhat greater. The roughness of the segments reflects the increased perturbation. The shorter non-closed curve has a rougher appearance because the perturbation variance σ is larger in relationship to average interpoint distance than for the longer closed curve. Both curves have the same σ but the closed curve has a larger interpoint distance.

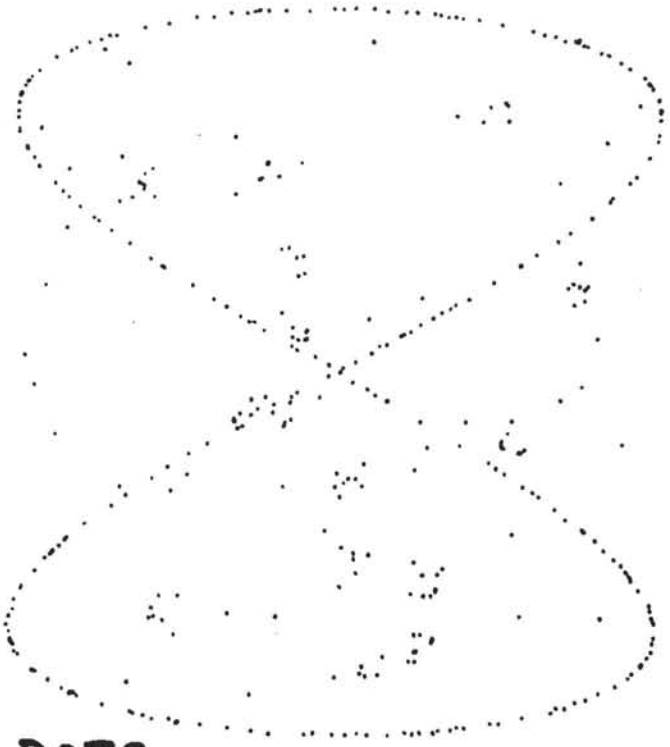
Large scale experiments have not yet been performed to test the curve recognition but Figures 1 and 2 are typical of the results we have seen so far. The first figure is an easy job for our program, whereas the second presents some difficulties. There are four breaks in the curves of Figure 2 which represent mistakes. Three of these could be recovered at the next higher level of organization. The two short breaks were caused by sharply bending curves and it is probably safer to postpone the decision in these two cases until more is known about the curvature on both sides of the break. The long gap at the center of the picture was considered too long to jump at this level of organization, but can be bridged easily and safely now that we see how linear the curve is on both sides of the gap. It is especially gratifying that the program was not confused by two bad segments near the center of the picture, caused by rather severe noise.

The curve recognition program requires approximately four seconds to run on an IBM 370/168 for a picture containing 500 points. The entire program is written in PL/1 and no attempt has been made to optimize. It could probably be made to run in less than one second without a great deal of effort.

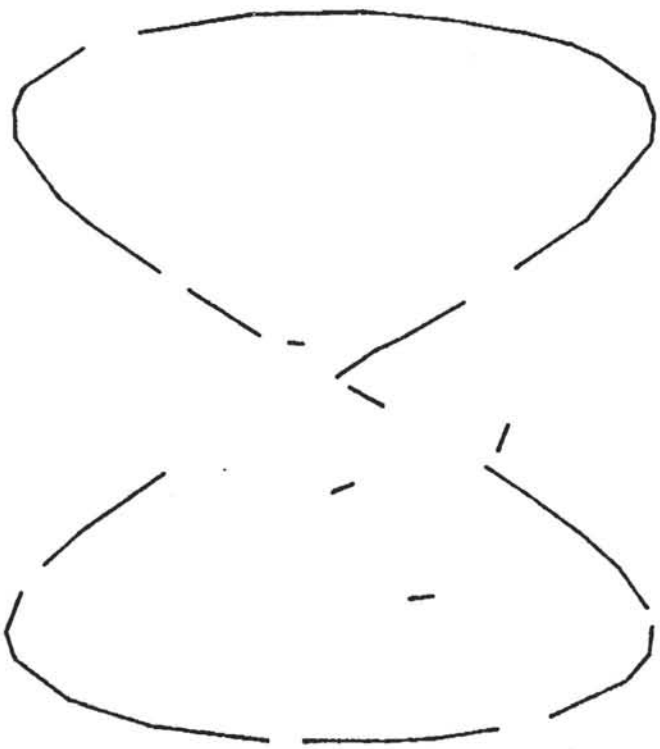
References

[1] C. T. Zahn, "Using the Minimum Spanning Tree to Recognize Dotted and Dashed Curves", Proceedings of an International Computing Symposium 1973, A. Günther et al (eds.), North Holland Publishing Co., 1974.

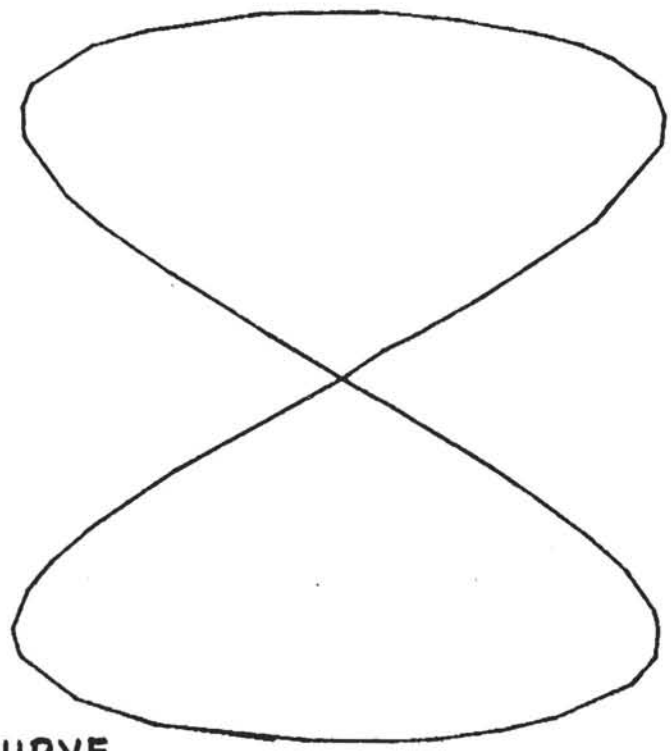
[2] C. T. Zahn, "Graph-theoretical methods for detecting and describing Gestalt clusters", IEEE Trans. Comp. C-20, No. 1, 68 (1971).



DOTS

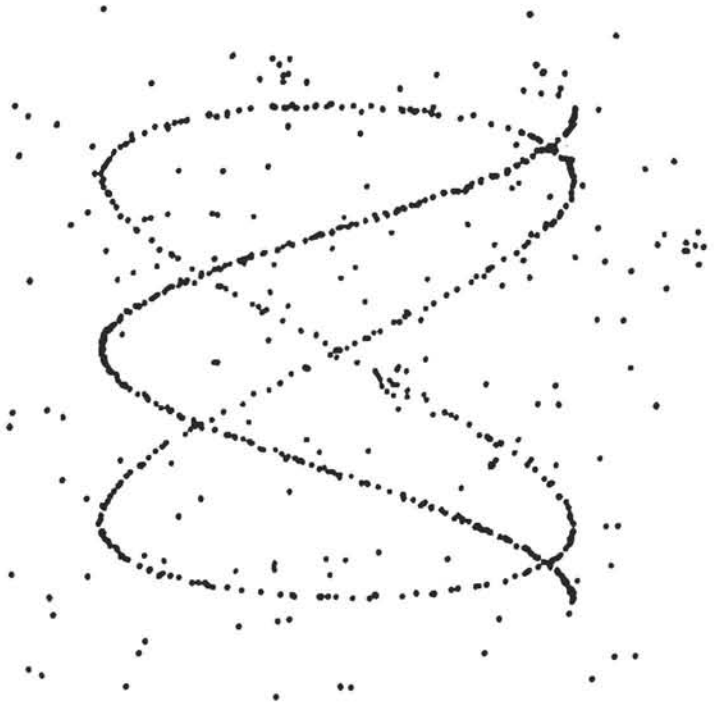


SEGMENTS

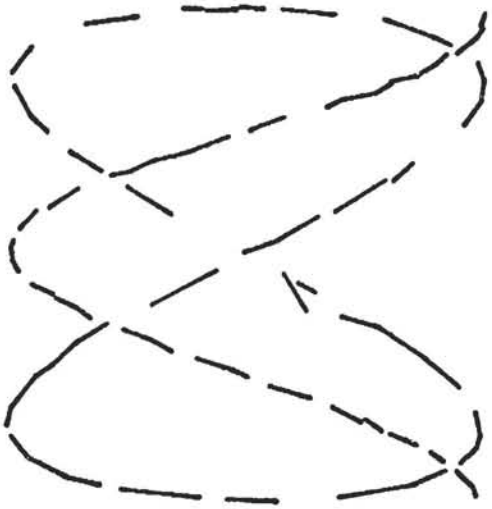


CURVE

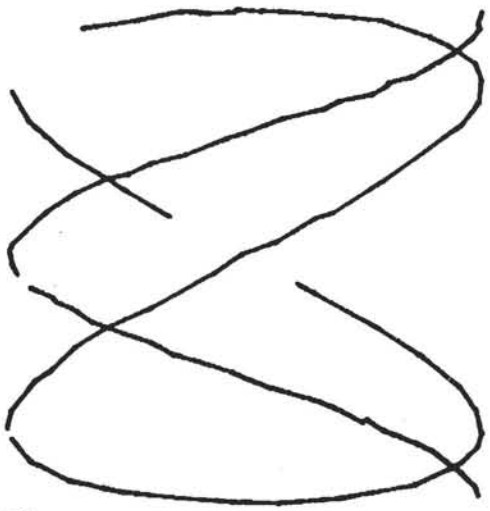
FIGURE 1



DOTS



SEGMENTS



CURVES

FIGURE 2