SCANtechnology: Feature Recognition for Automated Sculpture Reproduction

GRETCHEN GREENE, ROBERT GREENE, STEEN MARKVORSEN, HANS BRUUN NIELSEN, PETER RØGEN, ROBERT SINCLAIR, AND MICHAEL UNGSTRUP

1 Introduction

This report deals with a problem in the milling process of stone sculpture reproduction. Data for the milling process are supplied through a laser scanning of the original sculpture as measurement of a height function $z = h(x, y)$. 'Scan-points' are situated $0.2-2$mm apart on 'scan-lines' with intermediate distance $0.2-5$mm. The precision of the x-, y-, and z-measurements is within ± 0.1mm. These data define a surface which can be reproduced in stone using a CAD/CAM-system, that calculates tool paths for the milling machine.

The milling process can be described as follows. After an initial rough saw and hammer step to remove most of the excess stone, five or six tools are used in order of decreasing size. A tool is run back and forth across the stone (see Figure 1), at each point penetrating to the depth where it first hits the desired sculpture surface. The smallest tool requires many passes to cover the entire block of marble, accounting for up to 80% of the 10 hours of milling time. Increased efficiency at this stage could significantly reduce total production costs.

One question is whether we can isolate areas where the smallest tool is needed. A second question is whether, within those areas, there is a better path for the tool to follow than the back and forth pattern of horizontal lines. Better may mean more efficient space coverage, but it may also be a question of aesthetics. A sculpture with evenly spaced horizontal tool marks doesn't look good (see Figure 1) and the features may stand out more clearly if the tool marks in some sense paralleled the features and followed the natural contours. To answer these questions, we looked first to identify the small tool regions, and then sought to reduce each region to a curve or curves to get a basic tool path.

What are the small tool regions? A ridge in the sculpture, like the bridge of a nose, can be cut out by fairly large tools. There is plenty of room for maneuvering. Even

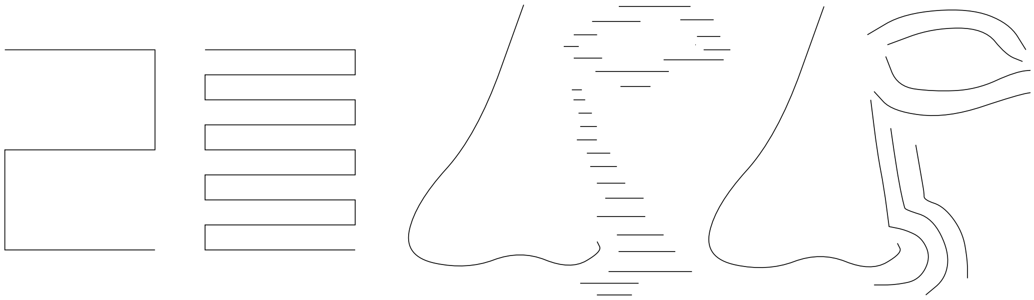

Figure 1: On the left are shown the tool paths currently used for the large and small tools respectively. On the right hand side, a sketch of the result obtained by moving the tool back and forth in horizontal lines, as compared to moving the tool along the features of the surface.

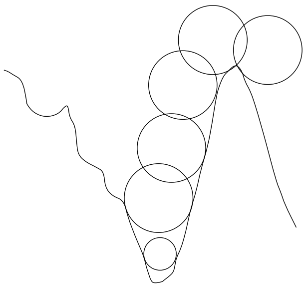

Figure 2: The figure illustrates how only small tools will fit at the bottom of valleys in the surface. The smaller tools all have spherical ends with diameters down to 3mm.

a steeply sloping side is not a problem until we hit the bottom of a valley. It is only here in the valley where a large tool cannot fit (see Figure 2). Since the smallest tools all have spherical ends with diameters down to 3mm, it seemed natural to ask: *How large a sphere would fit on the surface at each point?*. We answer this question in the following section.

2 Maximal Principal Curvatures

We model the sculpture surface as a graph (Monge patch) surface S defined by a height function h over a (usually rectangular) domain Ω in the plane $\mathbb{R}^2$:

$$S = \left\{(x, y, z) \in \Omega \subset \mathbb{R}^2 \;\middle|\; z = h(x, y)\right\}.$$

At each point p of the surface S the normal line l orthogonal to the surface at p defines a bundle of planes, the normal planes of S at p, see Figure 3.

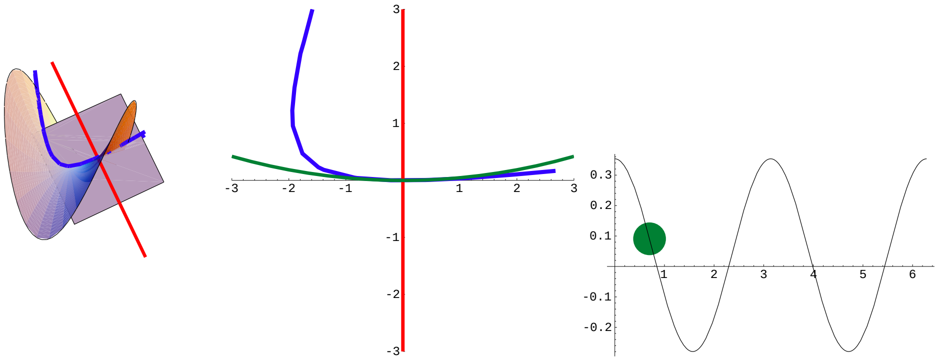

Figure 3: This graph surface (on the left hand side) is defined by $h(x, y) = \frac{1}{2}(x^2 - \frac{1}{2}y^2)$. The surface is cut by a normal plane. The planar intersection curve (in the middle) is approximated by a circle. On the right hand side, the curvature (the signed reciprocal of the radius of the approximating circle) of the intersection curve is graphed versus the turning angle of the normal plane when the normal plane is rotated around the normal line.

The intersection of a given fixed normal plane with the surface contains a (planar) curve through p. This curve has a signed curvature at p: The sign is positive if it curves upward in the direction of the $z-$axis and negative if it curves downward. The numerical value is the reciprocal value of the radius of the unique circle which is the best approximation to the curve at p.

When the plane is rotated around the normal line the corresponding curvature of the planar intersection curve will oscillate between the two so called *principal curvatures* $\lambda_1(p)$ and $\lambda_2(p)$ of the surface at p, see Figure 3.

We are mainly interested in the function

$$\max_P = \max\{\lambda_1(p), \lambda_2(p)\}$$

which measures the largest curvature of the normal intersection curves at p (as "seen by the milling tool" from above). I.e., $1/\max(p)$ is the largest possible radius of the milling drill needed to drill away the stone material right above the surface S at the point p.

The analytical expression for $\max_P(p)$ as a function of the location parameters (x, y) is readily found from the classical formulas for the Gauss curvature K and the mean curvature H of the surface at each point, see e.g. [1]:

$$K = \lambda_1\lambda_2 = \frac{h_{xx}h_{yy} - {h_{xy}}^2}{\left(1 + {h_x}^2 + {h_y}^2\right)^2}$$

and

$$2H = \lambda_1 + \lambda_2 = \frac{\left(1 + {h_x}^2\right)h_{yy} - 2h_x h_y h_{xy} + \left(1 + {h_y}^2\right)h_{xx}}{\left(1 + {h_x}^2 + {h_y}^2\right)^{\frac{3}{2}}},$$

where h_x, h_{xy} etc. denote partial derivatives of the height function. This gives:

$$\begin{aligned} \max_P(x, y) &= H + \sqrt{H^2 - K} \\ \min_P(x, y) &= H - \sqrt{H^2 - K}. \end{aligned}$$

We are then able to plot the contour levels in the (x, y) plane of the function $\max_P(x, y)$. Thereby we localize the regions where each given drilling head (of a given radius) can be used most effectively, i.e. where large drilling heads can be used to cut away as much material as possible and at the same time get close to the surface and where small drilling heads are needed to trace out the finest details of the sculpture.

In particular we can now point out those regions in the (x, y) - plane where $\max_P(x, y)$ is larger than or equal to the curvature of any given spherical drilling head. The system of these regions - which at places may degenerate to curves or even to points - we will henceforth denote the *ρ-Feature Regions* of the surface. If we let ρ denote the radius of the spherical drilling head (e.g. $\rho \geq 1.5$mm), then for each ρ

$$\mathrm{FR}(\rho) = \left\{ (x, y) \in \Omega \;\middle|\; \max_P(x, y) \geq \frac{1}{\rho} \right\}.$$

If the radius of the second finest drilling head is 3mm - say, then all features with radius of curvature larger than 3mm have been milled out leaving the rest to the finest tool. Hence, FR(3mm) is the set of "finest" Feature Regions, i.e. the regions that have to be drilled lastly with the finest drilling tool available.

To each connected component of a given Feature Region FR(ρ) we may now associate a center line of that component. Our candidate for this center curve construction is described below. The set of center lines for FR(ρ) is called the *ρ-Feature Lines* of the surface and is denoted FL(ρ).

The concrete drilling of the stone with a drilling head of radius ρ should then begin by drilling along the set of curves FL(ρ) and then work through all of FR(ρ) by traversing consecutively in each component of FR(ρ) the neighbouring curves to the respective Feature Lines.

In the case of analytically well defined surfaces we get Feature Regions FR(ρ) that are clearly visible and recognizable from the original surfaces (for each choice of

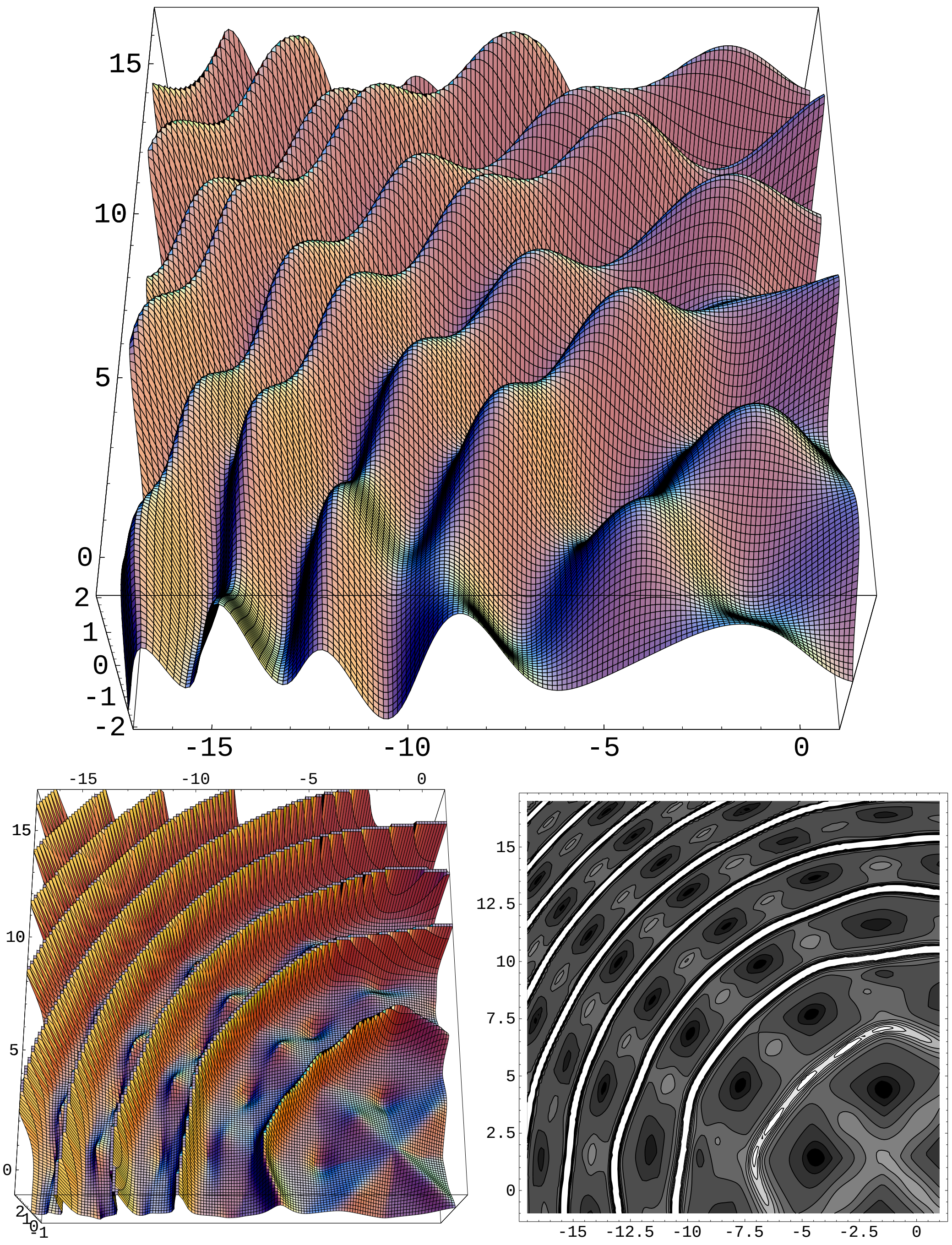

Figure 4: The Mountain Fold surface, shown on the top, is given by the height function: $h(x, y) = \sin(0.1(x^2 + y^2)) + \sin(x)\sin(y)$. At the bottom, the maximal principal curvature is shown as a height function over the domain of the surface to the left and to the right, as a contour plot, where white corresponds to high maximal principal curvature.

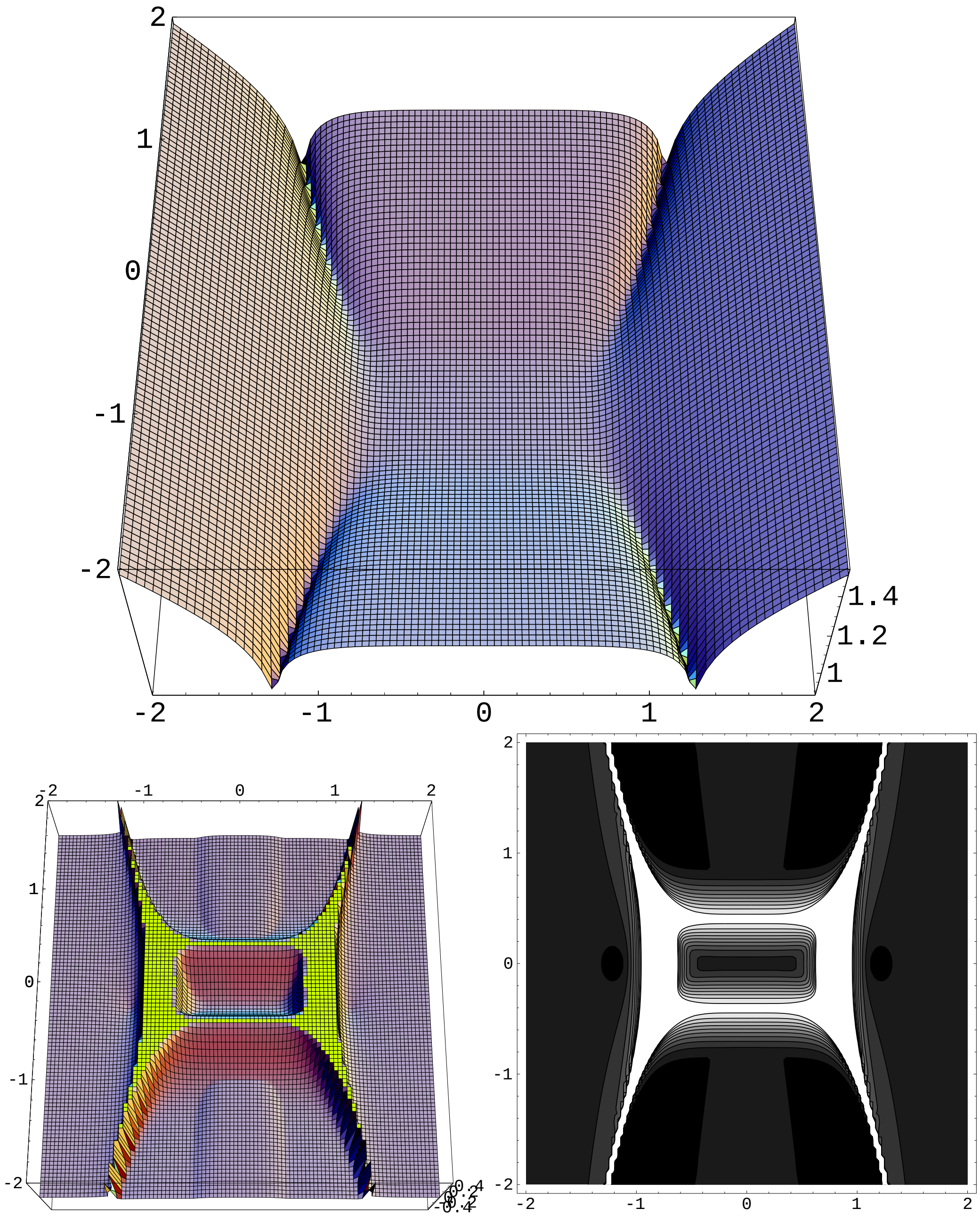

Figure 5: The Batman Face surface, shown on the top, is given by the height function: $h(x,\,y) = ((x^6 - y^2)^2 + 0.1)^{1/20}$. At the bottom, the maximal principal curvature is shown as a truncated height function over the domain of the surfaces to the left and to the right, as a contour plot, where white corresponds to high maximal principal curvature.

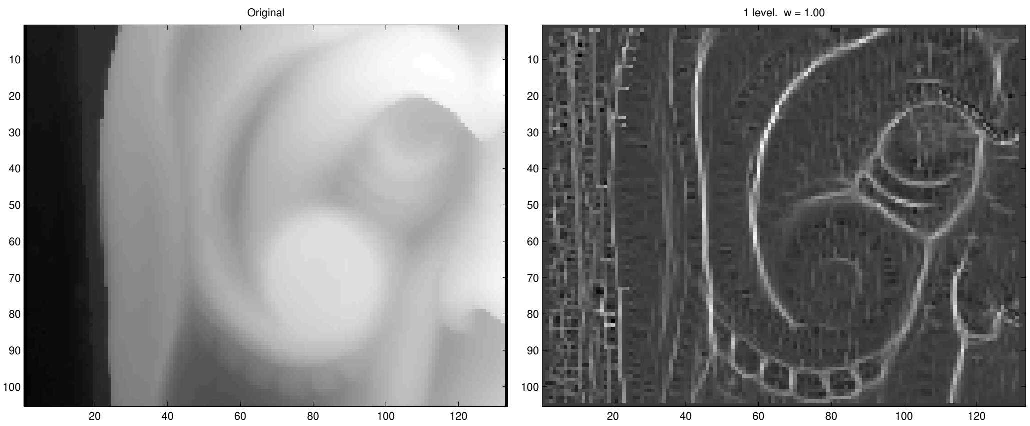

Figure 6: On the left hand side, the height of a piece of a sculpture, is shown in gray-scale over the lattice. The highest regions are white and the lowest regions are black. On the right hand side, the maximal principal curvature of the sculpture is shown in gray-scale over the lattice. High maximal principal curvature is white and the low maximal principal curvature is black.

cut off value ρ). See the two examples: Mountain Fold on Figure 4 and Batman Face on Figure 5. For each of these examples we show the surface, the graph of the function $\max_P(x, y)$ and the contour lines of the function $\max_P(x, y)$. The various Feature Regions are then the regions in the contour plot which have colours lighter than a given value.

In later sections we will therefore apply the same basic strategy to develop numerically robust Feature Lines from the discrete height function scan data. In this process we will concentrate on localizing the Feature Lines corresponding to the finest tools, since the geometry of these curves is naturally the most complicated to extract from the raw data.

3 Calculation of Maximal Principal Curvature on Sculptures

This section describes how to calculate $\max_P(x, y)$ given only a discrete height function on a regular lattice. The graph of such a function is shown in Figure 6 on the left.

As a first step the raw data is smoothed by replacing the height at each interior point by the average of the height at its eight neighbouring points, with weight one, and the height at the point itself, with weight four. This is done to reduce noise coming from the finite precision in measurements (cf. Section 1).

To calculate the principal curvatures the first and second partial derivatives of the height function are needed (see Section 2). To get these partial derivatives we associate to each interior lattice point an "osculating" second degree polynomial. A local approximation by a second degree polynomial, involves 6 parameters. These parameters are found by weighted least squares fit to the heights in the lattice point and its 8 immediate neighbours or an extended set including also the next 16 neighbours. Experiments indicate that the choice of weights and the number of included neighbours has very little influence." The second degree polynomial defines the partial derivatives needed to evaluate the formula for the maximal principal curvature given in Section 2. Finally, the resulting maximal principal curvature lattice function is smoothed as above. The result is shown in Figure 6 on the right. Note, how the white regions on the left hand side of this figure capture all the valley-features in the picture on the right. To find the ridge-features of the picture one has to calculate the minimal principal curvature.

To check the robustness of the calculation of the maximal principal curvature of the sculpture variation of the chosen weights, inclusion of the next nearest neighbours, and repeated or omitted smoothing have been tried. These alternatives give only slight changes of the calculated maximal principal curvatures. Hence, this calculation must be characterized as robust.

The “finest tool set” FR(3mm) (see Section 2), i.e., the set where the second finest tool can not get down in the clefts is shown on the left hand side of Figure 8.

4 Finding Center Lines

The idea that a region in the plane that resembles a strip has a well-defined center line is a basic geometric intuition. When we drive on a two-lane road, we accept without hesitation that it is a plausible thing for there to be a line down the middle of the road, dividing the road into two halves. But making this sort of idea precise and computing effectively exactly where the center line is turns out to be a surprisingly subtle matter. And a considerable literature has grown up around it. See e.g. the references in [2].

For regions bounded by two curves which are in some sense approximately parallel, for example, a road, it is natural to define the center line to be the locus of points that are equidistant from the two boundary curves (and in the region). In the case of a strip bounded by two parallel lines, this gives the obvious intuitive answer for the center line of the strip. And it produces visually plausible answers in more general but similar cases.

A point on the center line in this sense has the property that there are two distinct shortest connections to the boundary, one to one of the boundary curves and one to the other. This suggests that one might define the center curve of a general region to be the (closure of) the set of interior points for which there are at least two shortest connections to the boundary of the region. Alternatively in the case of smooth boundary one might consider the locus of points with at least two equilength perpendiculars to the boundary (whether or not these were shortest connections). Of course the set of points with at least two shortest connections to the boundary need not be a curve in the conventional sense. For a circle, it would be the center of the circle alone, for example. And for other regions the center line thus defined may have bifurcations. But even so the definition agrees well with visual intuition. And at least the set defined has no interior so that it is plausibly a candidate for being a curve in some sense.

5 An Approximate Method for Finding Center Lines

In the present project, we are concerned primarily with strip-like regions that are bounded by two curves that are close together compared to their extent, except at the "capped off" ends of the regions. In effect we want to look at roads with an end and a beginning. For such regions, the locus of point equidistant from the "sides" of the strip is an appropriate idea of the center line. The ends being a small part of the region can be largely ignored for our purposes.

The description of the center line points as those equidistant from the boundary curves is conceptually simple but is not particularly efficient computationally when the boundary curves are given in terms of a fairly dense but discrete set of points. The difficulty arises from the fact that in effect one would need to check equidistance for every interior point of some fine lattice and for each such point the check would involve finding the distance to all the boundary points or at least to all those that were reasonably near-by. Too many variables to be varied!!

5.1 A step by step algorithm

A more efficient, if somewhat approximate, method can be based on the following observations: First, for a parallel sided straightline boundaries strip (as shown on Figure 7), a center line point can be found by choosing an arbitrary point (“initial point” on the figure) of the interior, drawing a line (dotted on the figure) through it (not parallel to the boundary), and taking the midpoint of the intersection of that

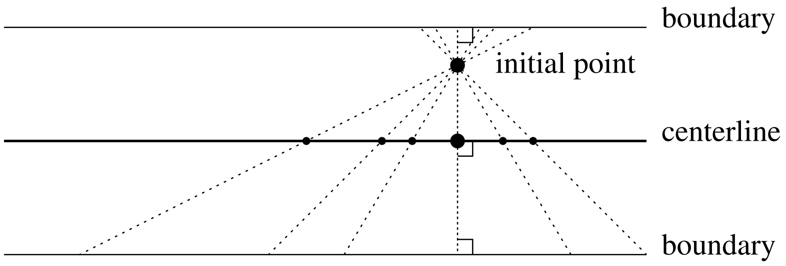

Figure 7: A parallel sided straightline boundaries strip with center line.

line with the strip. Secondly if the line through the interior point has the shortest possible intersection with the strip (among all lines through the given point), then the center line is perpendicular to that line. This is the tight-dotted situation on Figure 7.

For more general strip-like regions, these observations no longer apply precisely. But they remain good approximations. Moreover, it is plausible, in the second part, to check not all lines through the given point but some discrete, finite set of them and to take the shortest among those.

Following this general line of thought one can arrive at the following algorithm:

Step 0: Choose a point of the interior.

Step 1: For line vertical, horizontal, and at 45 degrees above and below the horizontal, find the endpoints of the intersection of the line with the region (or, more precisely, the endpoints of the connected component of the intersection that contains the chosen point).

Step 2: Choose the shortest of the intersections segments of Step 1 and find its midpoint.

Step 3: Move perpendicularly to the chosen intersection segment from its midpoint by some prechosen distance e. (The appropriate choice of e depends on the geometric smoothness of the boundary. Irregular boundaries would suggest the use of small e while smoother more nearly straight boundaries would allow larger choices of e).

Step 4: Return to Step 1 with the point found in Step 3 and proceed as before (the step 3 point replaces the initializing step 0 point).

The step 2 midpoints are the points desired as the center line. The Step 3 points are "helper " points, not necessarily center line point candidates, although for small e values they will be near-by.

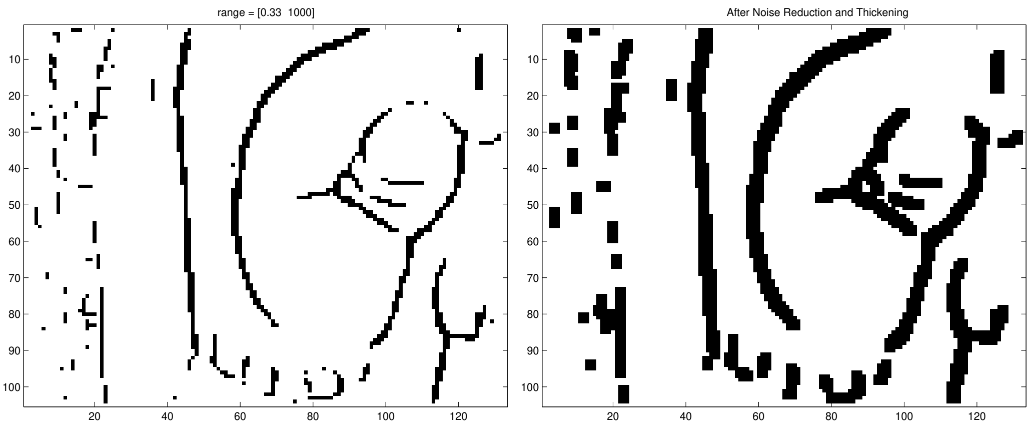

Figure 8: Left: The finest tool set. Right: A noise reduced and thickened version.

This algorithm is an approximate one. The center line candidate points of Step 2 are not necessarily actually center line points in the equidistant sense defined earlier. And the effectiveness of the approximation is not easy to analyse in generality. But experimentation (in Section 5.2) in numerous actual instances gave approximate center lines that were in fact satisfactorily close to the precise center line.

5.2 Implementation

The actual implementation assumes that the data are available on a regular square grid. Further, it assumes that the curvature data have been divided into those falling within a given interval, and those not, as done in Section 3 and shown on Figure 8 to the left hand side. The actual input is therefore a matrix of ones (which will be called points) and zeros. The algorithm has the following steps:

Noise reduction: If a point is isolated, has only one nearest neighbour, or no more than two next-to-nearest center(diagonal) neighbours, it is deleted.

Thickening: The nearest and next-to-nearest neighbours of existing points are also set to one. See Figure 8 to the right.

Identification of regions: The set of all points is separated into maximal disjoint connected (via nearest and next-to-nearest neighbours) components.

Thinning: Those regions with less than 100 points are deleted. See Figure 9 to the left.

For each region and initial directions up and down:

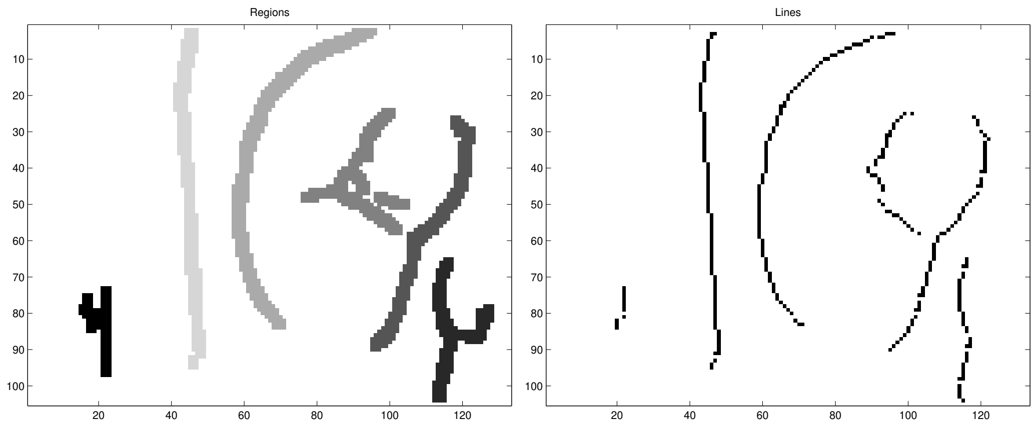

Figure 9: Left: The identified regions. Right: The first iteration of center lines.

- Find a starting point within the region.
- Draw vertical, horizontal and diagonal lines through it.
- Intersect each line with the region.
- Add the midpoint of the shortest line segment to the approximation of the line.
- Set the new current direction to be the normal to the shortest line segment whose scalar product with the current direction is positive (stop if none exists).
- Find the nearest point to the midpoint of the shortest line segment in the direction of the new current direction, and repeat using this point as a new starting point.

The resulting center lines are shown on Figure 9 to the right. There are now two problems left:

(1) To take care of the fact that the feature regions may not be simply connected, simply check for each new center line point whether it is within distance $2e$, say, of any of the points of the center line which are more than five iterations away in the past.

(2) To locate and drill bifurcating branches of the feature region tree, we do as follows: Recall the two pictures on Figure 9. For each constructed center line we remove from the feature region the collection of all points on the shortest intersection lines (constructed in Step 2 of Section 5.1) together with their neighbours within distance $1.5e$, say. Then we are left with a reduced feature region on which

we then run the algorithm again. This process clearly terminates and gives a complete centerline description for the corresponding feature region.

6 Drilling Strategy

On the basis of the center line finding algorithm, in particular from Step 2 of Section 5.1, we can easily parametrize the parallel drilling paths that the drilling tool has to traverse in order to cover the full feature regions: Let $2R$ be the length of the longest line segments found in Step 2 in Section 5.1, let $c(i)$, $i \in [0, N]$ be the center curve, and let $d(i)$, $i \in [0, N]$ be a choice of the direction of the shortest line through $c(i)$ for each $i \in [0, N]$. Define $P : [0, N] \times [-R, R] \to \Omega$ by $P(i, j) = c(i) + jd(i)$. Then the desired tool-path is

$$\begin{array}{ccccccccc}
P(0,R) & \to & P(1,R) & \to & & \cdots & & \to & P(N,R) \\
 & & & & & & & & \downarrow \\
P(0,R-1) & \leftarrow & \cdots & & \leftarrow & P(N-1,R-1) & & \leftarrow & P(N,R-1) \\
\downarrow & & & & & & & & \\
P(0,R-2) & \to & \cdots & & & & & & \\
 & & & & & \cdots & & \to & P(N,-R).
\end{array}$$

In order to get a reasonable tool-path it is probably necessary to check more than four lines in the plane as suggested in Step 1 in Section 5.1.

7 Conclusion

(1) For any given set of discrete scan data corresponding to a sculpture height function we have defined the function $\max_P(x, y)$, the maximal principal curvature function.

(2) For each given spherical milling tool we have, via the function $\max_P(x, y)$, defined the corresponding Feature Regions, which - to the depth and detail of the given tool - capture all visual features of the sculpture.

(3) For each given set of Feature Regions we have developed a Feature Line finding algorithm which will produce the initial track for the corresponding spherical milling tool.

(4) The Feature Line algorithm furthermore provides a full 2-dimensional parametrization of the corresponding Feature Regions. Hence it gives a strategy for

the complete milling process to the depth and detail defined by any spherical milling tool.

Bibliography

[1] M. P. do Carmo, *Differential Geometry of Curves and Surfaces*, Prentice Hall, 1976.

[2] Peter J. Giblin and Guillermo Sapiro, Affine-invariant distances, Envelopes and Symmetry sets, *Geom. Dedicata* 71:237–261, 1998.

The Group Working on the Problem

Gretchen Greene, [ggreene@math.ucla.edu]
Robert Greene, [greene@math.ucla.edu]
Steen Markvorsen, [S.Markvorsen@mat.dtu.dk]
Claus Dybdal Nielsen, [scantech@isa.dknet.dk]
Hans Bruun Nielsen, [hbn@imm.dtu.dk]
Thomas Randrup, [Thomas.Randrup@3dfacto.dk]
Peter Røgen, [Peter.Roegen@mat.dtu.dk]
Marit Hvalsøe Schou, [marit@mip.ou.dk]
Robert Sinclair, [R.M.Sinclair@mat.dtu.dk]
Michael Ungstrup, [M.Ungstrup@mat.dtu.dk]