# Extracting Surface Curvature from Noisy Scan Data

J. Rugis[1,2]

[1]CITR, Dept. of Computer Science, University of Auckland.
[2]Dept. Electrical & Computer Engineering, Manukau Institute of Technology.
Email: john.rugis@manukau.ac.nz

## Abstract

In general, the noise that is present in real-world 3D surface scan data prevents accurate curvature calculation. In this paper we show how curvature can be extracted from noisy data by applying filtering after a noisy curvature calculation. To this end, we extend the standard Gaussian filter (as used in 2D image processing) by taking adjacent point distances along the scanned surface into account. A brief comparison is made between this new *2.5D Gaussian filter* and a standard 2D Gaussian filter using data from the Digital Michelangelo Project.

**Keywords**: Surface curvature, noisy scan data, 3D noise filtering

## 1 Introduction

Three dimensional objects are often digitized in a way that results in surface point data sets [1]. In general, real-world scan data is noisy due to inaccuracies accumulated in the scanning process[1]. Curvature, being a second derivative property, is particularly sensitive to corruption by noise.

A common approach towards solving this problem is to smooth the point data prior to attempting curvature calculations. This approach has a shortcoming in that surface detail can easily be lost. Alternatively, in this paper, we firstly calculate *noisy curvatures* and subsequently apply filtering to these curvature values.

Although the standard 2D Gaussian filter can be used in our approach, it does have the undesirable effect of smoothing edges as well as noise. Edge preserving variants of the 2D Gaussian filter have been developed. The bilateral filter by Tomasi and Manduchi [2] is a 2D filter that employs an edge preserving term that decreases pixel weighting based on pixel intensity differences.

Smoothing of the 3D shape itself is the goal of other noise reduction filters. A curvature and Laplacian operator based diffusion approach was introduced by Desbrun et al. in [3]. Work by Fleishman et al. consists of a mesh de-noising algorithm that operates on a surface predictor geometric component of the mesh [4].

In contrast, even though the approach that is presented in this paper uses 3D surface point position information, it does not alter the position of those points.

[1]In this paper, we will consider noise which has primarily a Gaussian distribution.

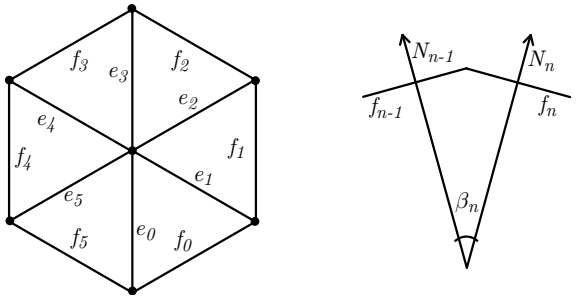


Figure 1: Curvature estimators: point adjacency on the left and face normals on the right.

## 2 Surface curvature estimators

Surface curvature is a well-defined property for continuous smooth surfaces [5]. However, when working with point data sets, many surface properties can only be estimated [6], and there exists a number of different estimators for determining curvature [7].

In this paper, the mean curvature is estimated as done by other authors [8]. With reference to the left side of Figure 1, we consider a point on the surface and, say, six adjacent points. The points are thought to be connected by *edges*, and edges enclose, in this case, six *faces*. We also identify an area $\mathcal{A}(f_n)$ associated with each face $f_n$. On the right side of Figure 1, we identify a surface normal vector associated with each face from an edge-on view point. The angle between adjacent face normals is designated as $\beta$. Angle $\beta$ is positive if the faces form a convex surface (i.e., when viewed

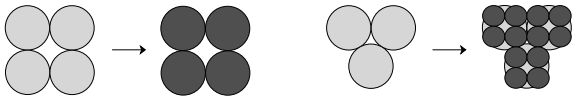

Figure 2: Mappings of orthogonal (on the left) or hexagonal (on the right) grids into an orthogonal grid.

from the outside) and $\beta$ is negative if the faces form a concave surface (i.e., when viewed from the outside).

The mean curvature at the point $P$ is estimated by

$$H(P) = \frac{3\sum ||e_n||\,\beta_n}{4\sum \mathcal{A}(f_n)} \qquad (1)$$

This estimator is generally valid, without change, in the case of adjacency point counts other than six.

## 3 Curvature maps

For 2D visualization purposes it is useful to convert the mean curvature values at surface scan points into a (2D) *curvature map* [9]. If the 3D point data has been acquired in a 3D orthogonal grid, then the curvature mapping is straightforward (defined by orthogonal cuts parallel to coordinate planes, see [10]). For data that has been acquired in a hexagonal grid, a *squashed dot* mapping is used [9].

Mappings for both cases, either orthogonal or hexagonal, are shown in Figure 2. The second mapping has the expense of quadrupling the number of pixels.

## 4 Curvature noise filtering

Consider sampling the planar surface, which has zero curvature everywhere. Noisy sampled data points will exhibit a symmetrical distribution of positive and negative curvatures centered around zero, with the limiting mean value for many points being zero. This suggests filtering that includes a mean calculation.

In this paper we describe and briefly compare two different weighted mean based curvature noise filtering approaches.

### 4.1 2D Gaussian filter

In this approach, we start by converting the noisy curvature values into a 2D curvature map as described briefly in Section 3. See [9] for further details on the curvature map creation process. This conversion into 2D enables the use of standard 2D images processing techniques.

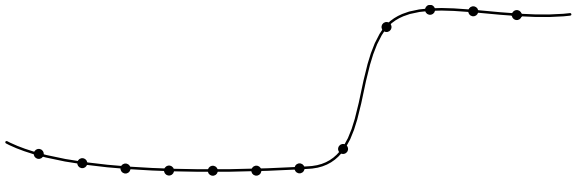

Figure 3: Cross section of a surface fold.

Next we apply a standard 2D image processing Gaussian filter which performs the desired smoothing based on the now fixed adjacency (and distance) relationships in the curvature map. The standard 2D Gaussian filter is implemented as a convolution process with the terms in an $(2m+1)\times(2m+1)$ Gaussian convolution kernel centered at $(0,0)$ being determined using the formula

$$h(n_1,n_2) = h_g(n_1,n_2) \Big/ \sum_{n_1=-m}^{m}\sum_{n_2=-m}^{m} h_g$$

$$\text{with} \quad h_g(n_1,n_2) = e^{-(n_1^2+n_2^2)/2\sigma^2} \qquad (2)$$

where, as usual, the standard deviation $\sigma$ acts as a pixel area smoothing factor. Note that the $(n_1^2+n_2^2)$ term can be thought of as the *adjacency distance* (squared) in a fixed adjacency grid.

### 4.2 2.5D Gaussian filter

As a preliminary motivation, note that the 2D filter, with its innate fixed distance adjacency, has the undesirable effect of smoothing edges which may be present due to silhouettes, occlusions, and surface folding in the scan. For example, Figure 3 shows a cross-section view of a surface fold in which the distances between adjacent scan points are not equal.

We introduce a *2.5D Gaussian filter* which gives consideration to edges. We will apply this filter to the noisy curvature values assigned to each scan point in the 3D domain *before* generating a final 2D curvature map.

In the 3D point space, we define *adjacency point neighborhood rings* and assign subscripts as shown in Figure 4 for the case of hexagonal adjacency. Note that only the inner two rings are shown, but that additional rings may be used. The first subscript identifies the ring and the second subscript identifies each point within a ring. The concept of neighborhood rings applies similarly to orthogonal adjacency where, of course, the number of points in each respective ring will be greater.

We calculate the 2.5D Gaussian filtered mean curvature $\tilde{H}$ at each point $P$ on the surface (indexed in turn as $P_{0,0}$) as follows:

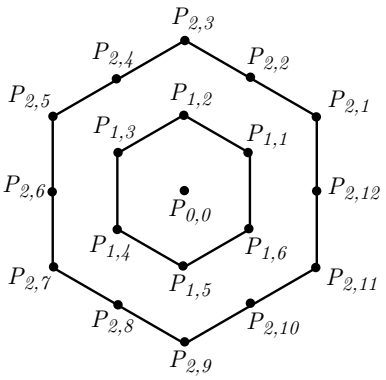


Figure 4: Adjacency point neighborhood rings.

$$\tilde{H}(P_{0,0}) = \sum_m \sum_n (w_{mn} H_{mn}) \Big/ \sum_m \sum_n w_{mn}$$

$$\text{with} \quad w_{mn} = e^{-(P_{mn} - P_{0,0})^2 / 2\sigma^2} \qquad (3)$$

where the values of $H_{mn}$ are the unfiltered mean curvatures. The summations are computed over $m$ neighborhood rings with $n$ points in each ring[2]. Note that $||P_{mn} - P_{0,0}||$ is the physical (Euclidean) distance from the center point to a point in a neighborhood ring, and that the values of $w$ can be thought of as representing filter weights. The standard deviation $\sigma$ is smoothing factor that, in contrast with the 2D Gaussian filter, is now based on (estimated) surface area. The standard deviation retains its usual meaning in that we would expect, for example, the sum of the filter weights assigned to points within a two-sigma radius to be 95.5% of the total filter weight.

We refer to this as a 2.5D filter because it is computed on a 2D surface which is embedded in 3D space. Although there is some similarity to the standard 2D filter, note that, strictly speaking, this is not a convolution process and we have lost the computational efficiency of a fixed convolution kernel.

## 5 Experiments

We have performed experiments using scan data of the David statue from the Digital Michelangelo Project [1]. This data set was acquired with hexagonal point adjacency and thus neighborhood rings as illustrated in Section 4.2 were assigned for the computations. The data contains a moderate level of noise, with the equivalent Gaussian noise level standard deviation being approximately equal to the minimum scan point adjacency distance.

We concentrate on a curl of hair above David's right eye which only just visible at the top edge

[2]Including the center point $P_{0,0}$ as the single element in ring zero.

Figure 5: A photograph of Michelangelo's David.

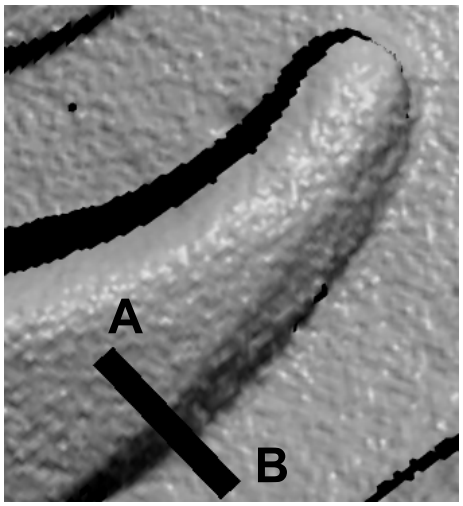


Figure 6: The curl: rendered mesh.

of the photograph shown in Figure 5. Figure 6 is a reference closeup of the curl rendered as a mesh surface [11] constructed from the raw scan data points. The lighting direction in this rendering has been chosen to highlight certain contours. The holes in the scan are due to occlusion. The black bar with ends marked A and B identifies a cut through a folded section of the surface.

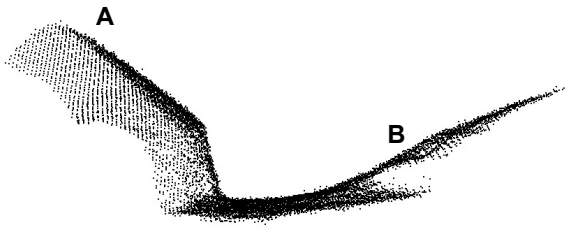


Figure 7: Cross-section of curl scan points.

Figure 7 shows a cross-section slice of the scan points which indicates a folding edge between locations A and B. Traversing from A to B, the curvature starts as slightly positive, is distinctly positive

at the first bend, is then zero, is distinctly negative at the second bend, and then slightly negative.

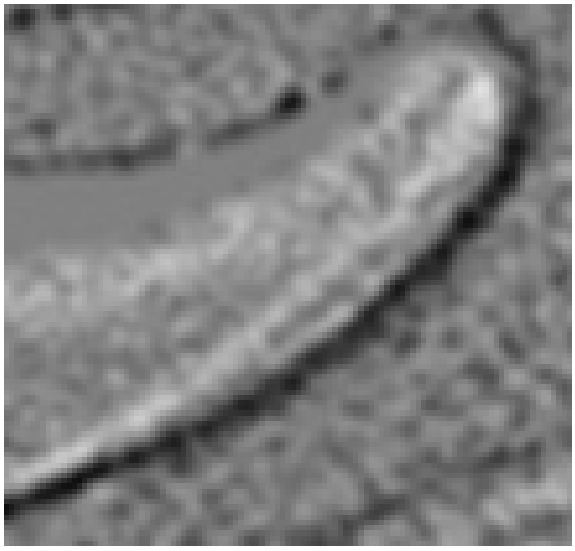

Figure 8: Curvature map: 2D filtering.

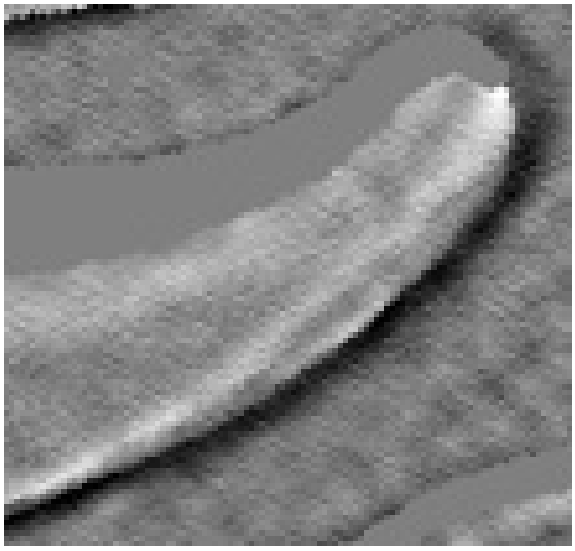

Figure 9: Curvature map: 2.5D filtering.

Figure 8 shows a shading encoded mean curvature map of the curl with 2D filtering. Maximum positive curvature is shading coded as white. Zero curvature is shading coded as medium grey and maximum negative curvature is encoded as black. Note that, because the filtering was done after the squashed dot mapping, there is some spread into regions where there is otherwise insufficient point data for curvature calculation. Figure 9 shows a shading encoded mean curvature map of the curl with 2.5D filtering. Equivalent values for the smoothing factor $\sigma$ were used in both filters.

In Figure 9, it does appear that, as expected, there exists sharper transitions between black and white at the sharp fold edges. To confirm this, we extracted pixels associated with the previously illustrated A-B cut from both the 2D and the 2.5D filtered curvature maps. The extracted pixels are shown in Figure 10.

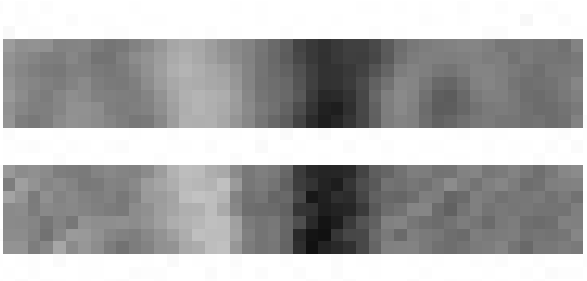

Figure 10: Extracted A-B cut pixels: 2D filter (top), 2.5D filter (bottom).

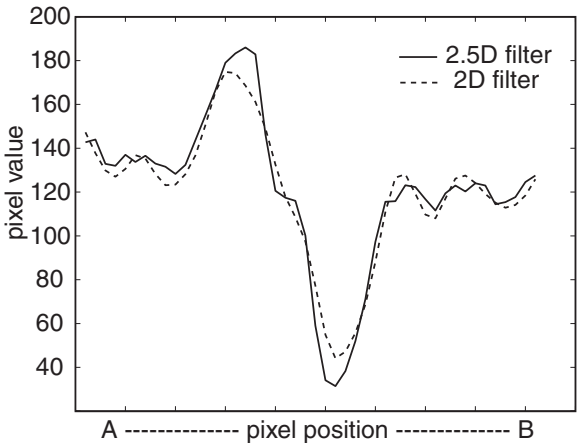


Figure 11: Pixel values along the A-B cut.

We averaged the pixel values across the narrow cut direction and then plotted the resultant values as shown in Figure 11. In the center section of the plot, we see that, in the 2.5D trace, 1) the maximum slope is greater, 2) the magnitudes of both the positive and the negative peaks are greater, and 3) there is a distinct zero curvature shelf.

## 6 Conclusion and Further Work

In the experiment presented, 2.5D filtering results in more representative curvature at a fold edge than does 2D filtering. Further work is anticipated to include additional noise models (such as highly impulsive), additional filtering methods, and additional visualization techniques.

Finally, to illustrate the total size and scale of the scan, Figures 12 and 13 each show a curvature map of the entire scan, one with 2D filtering and the other with 2.5D filtering. Close examination reveals that other regions, such as the eyelid near the corner of the right eye, for example, appear to benefit from improved edge preservation.

## 7 Acknowledgements

The author would like to acknowledge financial support from The Department of Electrical and Computer Engineering at Manukau Institute of Technology. Access to the Digital Michelangelo Project data is courtesy of Stanford University.

## References

[1] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk, "The digital Michelangelo project: 3D scanning of large statues," in *Proc. SIGGRAPH*, pp. 131–144, 2000.

[2] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *ICCV*, pp. 839–846, 1998.

[3] M. Desbrun, M. Meyer, P. Schrder, and A. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Computer Graphics*, no. 33, pp. 317–324, SIGGRAPH 99 Proceedings, 1999.

[4] S. Fleishman, I. Drori, and D. Cohen-Or, "Bilateral mesh denoising," in *ACM Transactions on Graphics 22*, no. 3, pp. 950–953, Proceedings of ACM SIGGRAPH, 2003.

[5] A. Davies and P. Samuels, *An Introducion to Computational Geometry for Curves and Surfaces*. Oxford: Oxford University Press, 1996.

[6] R. Klette, R. Kozera, L. Noakes, and J. Weickert, *Geometric Properties for Incomplete Data*. Dordrecht: Springer, 2006.

[7] R. Klette and A. Rosenfeld, *Digital Geometry*. San Francisco: Morgan Kaufmann, 2004.

[8] L. Alboul and R. van Damme, "Polyhedral metrics in surface reconstruction," in *The Mathematics of Surfaces VI* (G. Mullineux, ed.), (Oxford), pp. 171–200, Clarendon Press, 1996.

[9] J. Rugis, "Surface curvature maps and Michelangelo's David," in *Image and Vision Computing New Zealand 2005* (B. McCane, ed.), pp. 218–222, 2005.

[10] S. Hermann and R. Klette, "Multigrid analysis of curvature estimators," Tech. Rep. CITR-TR-129, Centre for Image Technology and Robotics, University of Auckland, 2003.

[11] J. Foley, A. vanDam, S. Feiner, and J. Hughes, *Computer Graphics: Principles and Practice*. Boston: Addison-Wesley, 1996.

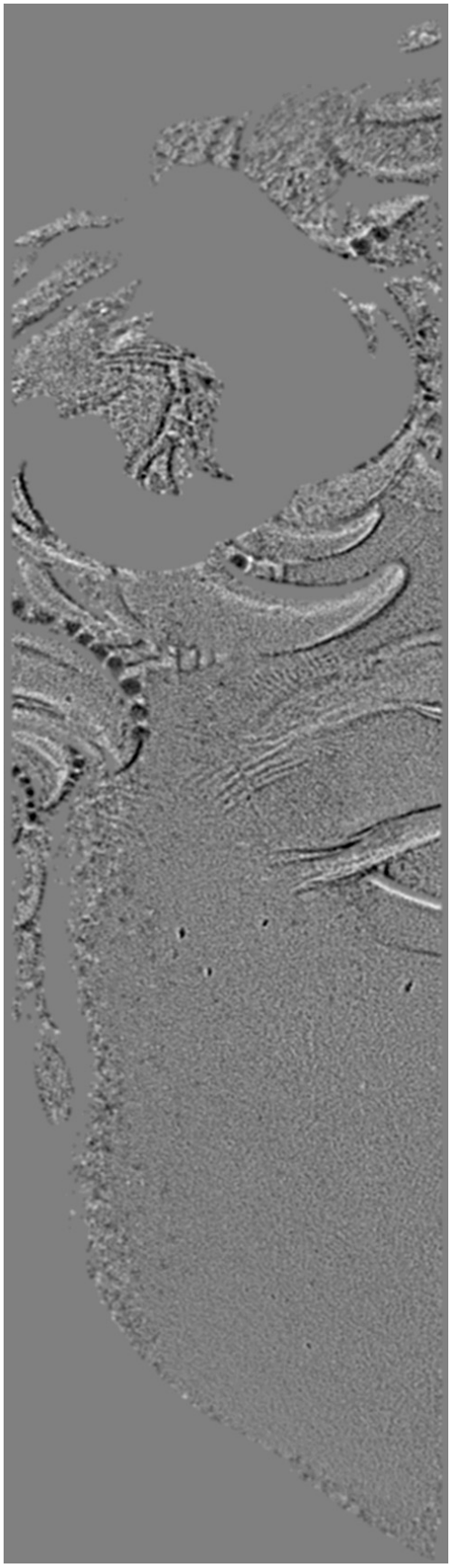

Figure 12: David's face: 2D filtering.

Figure 13: David's face: 2.5D filtering.