

ALSD: An algorithm for noise erasure and structural information recovery for surfaces with structural discontinuities

Abheek Saha, Seema Garg

I. INTRODUCTION

We consider the problem of structural reconstruction of noisy data measured over an underlying space Ω . The data could correspond to any physical parameter of interest which is characterized by the spatial geometry. For example, in the field of terrain mapping, the data points could correspond to altitude, gravity measurements, or albedo. In the field of image processing, the data points could correspond to luminosity or chromaticity. In the field of telecommunications, the data could represent measured interference. For the purpose of this paper, we refer to the empirical data indexed over the space as the *image*. It is known that the image consists of multiple separate smooth sub-surfaces or manifolds, which can meet at abrupt discontinuities. Each manifold can be represented by the tuple $\{f_i(x, y), \Sigma_i\}$, where $f_i(\cdot)$ is a fixed but unknown polynomial $f(x, y) \in C(n)$, defined over the continuous closed set $\Sigma_i \subset \Omega$ within the image. The manifolds do not overlap and the boundaries between individual manifolds may be non-differentiable and discontinuous. The image is represented as a dataset $z(x, y) \forall (x, y) \in \Omega$, sampled at fixed intervals on the X and Y planes at a given resolution. The data is corrupted by a Gaussian noise pattern with a known variance.

We wish to recover the original manifolds which constitute the composite image, both in terms of the polynomials $f_i(x, y) \in C(n)$, corresponding to the i th manifold and its domain Σ_i . N , the maximum order of the polynomial $f_i(\cdot)$ is an input to the algorithm.

In this paper, we have identified a novel technique to simultaneously remove noise and recover the underlying structural information for a noisy image and have described the implementation of the same. We believe that the two functions are complementary to each other and we use information from one to improve the other. We show that we achieve

good scaling on this architecture and the results achieved are comparable to other postulated solutions. We call our algorithm Adaptive Local Surface Deconstruction (ALSD). The algorithm is highly parallelizable and works in $\Theta(n)$ time, where n is the number of sampled points. In a followup paper, we shall discuss the implementation of this algorithm in a Linux cluster and the experience thereof.

The rest of this paper is organized as follows. In the next section we will discuss previous approaches of to this problem, specifically four approaches which we have studied and borrowed elements of. In the subsequent section IV, we shall discuss our proposed algorithm. In section V we shall present some experimental results.

II. PROBLEM DESCRIPTION

Fig 1 shows a surface with the individual manifold, before and after they have been corrupted by noise. A sample input image consists of four sub-surfaces, as follows: The task is to process the image data to remove the noise and identify the constituent manifolds.

Clearly, we are dealing with a combination of two separate problems here. One is to remove the point noise in the input data, with minimal distortion to the original data sequence, which may contain structural breaks and discontinuities. The second is to identify the boundaries of the individual constituent sub-surfaces or manifolds and recover the original model. The first task is covered widely in the field of image reconstruction or inversion which refers to the conversion of a 2-D sample and reconstructing the original three dimensional visual domain which created the image. In case of images, the z data is typically luminosity/chromaticity data and the sample points represent the pixels which compose the image. The problem is well-studied in literature, and

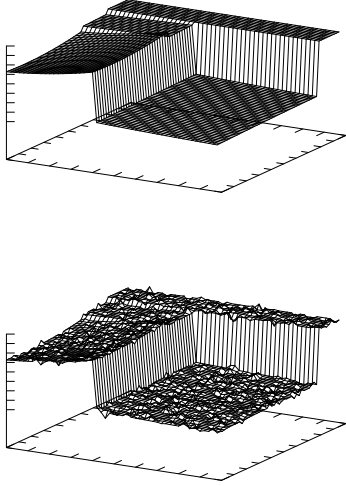


Fig. 1. Original manifolds making up image

has been identified as an “ill-posed” regularization problem [Marroquin,Mitter,Poggio88].

The problem of recovering the original model g , from a set of measurements $z = g(x, y)$ on a certain domain $(x, y) \in \mathcal{D}$ is related to the problem of inversion [Tarantola]. However, inversion holds the relation g constant and tries to find the original model parameters (x, y) . If g is a linear operator G , then we can treat the coefficients of G as the model and (x, y) as the mapping operator, which makes it a standard inversion problem.

Our problem requires an application of two separate methods from two related but different domains. Inversion problems don’t deal with discontinuities and structural breaks and require a good apriori estimate of the model. Noise erasure problems on the other hand, do deal with discontinuities, but don’t give or use any apriori information. In our solution we have used an iterative technique which alternately uses noise reduction and inversion to gradually build up apriori information about the model.

A. Noise reduction and smoothing

Noise reduction or removal is a standard problem of data processing and is typically solved by some form of low-pass/band-pass filtering. This takes into account the fact that the noise signal is typically a wide-band process and will show up at multiple frequencies. If we have a reasonable idea of the nature of the underlying data i.e. its spectrum, a bandpass filter can be appropriately defined to get rid of noise by eliminating

frequencies in which no useful information is present. In other cases where the noise is closely tied to the signal itself, we can use a Kalman-Bucy filter or other kinds of structure based filtering.

The problem in our current case is the presence of discontinuities in the original data itself. Discontinuities, when mapped to the spectral domain, result in very high frequency signatures which will almost certainly be eliminated in any noise filtering approach and can also confuse a tracking approach such as a Kalman filter. Clearly, we need a mechanism which can utilize the structural information about the domain to do a more intelligent noise elimination. Structural information could include direction, continuity and any others specific to the domain. In our current problem, the key information is that discontinuities occur over continuous lines which form the boundaries between two planes.

1) *Tikhonov Regularization*: In computer vision, the noise removal problem is known as the regularization problem. A regularization problem tries to find the actual data z from a noisy sample y by a linear transformation A . i.e. $y = Az$. In the standard form given by Tikhonov, we solve a global optimization problem to find z such that $\|S(z, y)\|^2 + \lambda\|P(z, y)\|^2$ is minimized [Terze88]. S and P are domain specific stabilizing and regularizing functionals respectively, based on known properties of the surfaces we are trying to reconstruct. The first term corresponds to the deformation energy of the surface as a whole (caused by internal discontinuities, creases and surface variations), the second term gives the penalty of deviation from the original data. There are many forms for S and P , each with its own complexities. Once S and P are selected, the problem has to be solved using some form of global optimization, for example, simulated annealing. The challenge is to find the form of S and P which matches our specific domain and then choosing a suitable technique for global optimization of z as per the equation given above.

A key parameter in this equation is $\lambda_{(i,j)}$, which gives the relative weightage between the stabilizing functional and the regularizing function. The higher the value of λ , the higher the local stiffness i.e. penalty given for departure from the original value. We call $\lambda_{(i,j)}$ the noise quotient, because it is tied to the properties of the noise signal i, j .

2) *Controlled Continuity Surface modeling*: A form of the Tikhonov regularizer is given by Terzopoulos [Terze88] using the continuity control functions $\rho(x, y)$ and $\tau(x, y)$, along with

a fixed discrepancy penalty tied to the known variance of the noise signal. It is his interpretation of the continuity control which is most interesting.

Terzopoulos suggests a stabilizing functional as follows:

$$S = \frac{1}{2} \int \int_{\Omega} \rho(x, y) \{ \tau(x, y) (v_{xx}^2 + 2v_{xy}^2 + v_{yy}^2) + (1 - \tau(x, y)) (v_x^2 + v_y^2) \} dx dy$$

$$\rho(x, y) \in [0, 1] \forall (x, y) \in \Omega$$

$$\tau(x, y) \in [0, 1] \forall (x, y) \in \Omega$$

Here $\rho(x, y)$ controls the continuity requirements for smoothing. As $\rho(x, y)$ approaches zero, the penalty due to lack of continuity is reduced. As it approaches 1, the requirement for continuity increases. It therefore represents the forces of cohesion. Further, $\tau(x, y)$ forces the choice between C^0 continuity and C^1 continuity as well as continuity of the first order derivatives. $[1 - \tau(x, y)]$ thus represents the forces of surface tension.

For practical application of this method, we need to have a method computing τ and ρ at each point (x, y) . Terzopoulos suggests an analogue with the physical world, by considering the image as a thin plate or membrane. The bending moment $M(x, y) = -\Delta(x, y)$ at a point measures the local deformation energy and the divergence $G(x, y) = \|\nabla u\|^2$. [Terze88] suggests setting τ and ρ as binary variables. $\tau(x, y)$ is set to 1 if the bending moment at that point is greater than a pre-set threshold. Similarly, ρ is set to 1 if $G(x, y)$ is greater than a preset threshold.

3) *Simulated Tearing*: Simulated tearing, suggested by [Figueiredo and Leital] is a conceptual modification of simulated annealing, and a further expansion of the bending membrane formulation suggested by [Terze88]. The simulated tearing process is similar to taking a membrane or some kind of flexible paper and pressing it down onto the surface below. The membrane will be impervious to noise at individual points i.e. it will deform itself to 'smoothen' them out. However, where there is an actual discontinuity at the boundary between two or more sub-surfaces, the membrane will tear. In the final analysis, the membrane should tear into multiple pieces, each of which corresponds to a given manifold and whose orientation describes that of the manifold itself.

The simulated tearing algorithm, similar to the simulated annealing algorithm, requires the construction of a suitable energy function. In [Figueiredo and Leital], the authors use

the following energy function:

$$E_{\phi}(x, y, h, v) = \frac{1}{2\sigma^2} \sum_{i,j} (x_{i,j} - z_{i,j})^2 + \alpha \sum_{i,j} (h_{i,j} + v_{i,j}) + \mu \sum_{i,j} (x_{i,j} - x_{i,j-1})^2 (1 - h_{i,j}) + \mu \sum_{i,j} (x_{i,j} - x_{i-1,j})^2 (1 - h_{i,j})$$

$h_{i,j}$ and $v_{i,j}$ are the previously mentioned functionals which indicate whether there is a discontinuity on the horizontal and vertical axes. α is the cost of creating a discontinuity. The authors describe an algorithm to simultaneously compute an optimal $x_{i,j} \forall i, j$ and also identifying the discontinuity points. The biggest downside to the algorithm is in the cost of inverting a very large, albeit pentadiagonal matrix. The author's own solution suggests a neural network model for doing this inversion.

Once again, we note the dependence on the selection of $h_{i,j}$ and $v_{i,j}$. Interestingly, [Figueiredo and Leital] does not provide any domain specific method to choose h and v . Instead the algorithm treats the vectors v and h as parameters to the optimization algorithm and estimates them indirectly.

4) *Anisotropic Diffusion*: Anisotropic diffusion is a version of the kind of global optimization attempted in simulated tearing, suggested in [Perona and Malik90]. Anisotropic diffusion attempts to identify individual sub-surfaces by solving the heat diffusion equation

$$\frac{\partial I}{\partial t} = c(x, y, t) \Delta I + \nabla c \nabla I \quad (1)$$

where I is the intensity and c is the heat diffusion coefficient. The authors suggest sub-surface identification by using an appropriate formulation of c , ensuring that heat diffusion coefficient is zero on an edge and maximum within a sub-surface. A suggested formulation by the authors for a 2-d grid as we use is given by:

$$I^{t+1} = I^t + \lambda \{ c_N \nabla I_N + c_S \nabla I_S + c_E \nabla I_E + c_W \nabla I_W \}$$

$$c_{N,S,W,E} = \left(\frac{1}{1 + \left(\frac{\|\nabla I_{N,S,W,E}\|}{K} \right)^2} \right)$$

$$\nabla I_N = I_{i-1,j} - I_{i,j}$$

$$\nabla I_S = I_{i+1,j} - I_{i,j}$$

$$\nabla I_W = I_{i,j-1} - I_{i,j}$$

$$\nabla I_E = I_{i,j+1} - I_{i,j}$$

The interested reader will note that the surface tension and adhesion parameters in this formulation is nothing but the first derivatives around each point. This paper focusses more on the iterative approach of solving the optimization problem as an evolving heat diffusion.

B. An explicit two stage approach

A fairly simple and usable approach has been suggested by [Schunk and Sinha92]. We found the approach appealing because it is simple and simultaneously addresses the dual issues of removing noise as well as identifying surfaces. We have recreated the approach and used it reasonably successfully as we shall describe below, to generate benchmark data, against which we have validated our own algorithm.

The algorithm is a two-stage method. The first stage is a noise-reduction stage. It divides the surface into small sub-surfaces and try to do an optimal plane fit on each sub-surface. An innovation was to use a robust method (MLMS), which also handles the cases where individual points are corrupted with noise, as follows

- 1) Instead of taking all the points within the sub-surface, the algorithm randomly picks out a sub-set of the points
- 2) Using the sampled points, the algorithm using a fitment algorithm to fit the point selected onto a sample curve.
- 3) Based on the results, the algorithm computes and stores the mean square error for the sample
- 4) This process is done repeatedly. For each sample, the MSE is stored
- 5) Finally, the sample with the median MSE is chosen to represent the sub-surface.

After this first step, we will have planar fits for each small sub-surface of the entire grid. This is known as the *clean and fit* stage. This phase is followed by the global optimization stage. The global optimizer uses a global energy function constructed out of B-splines over the basic sub-surface as follows:

$$\mathcal{T}(f) = \sum_{i=1}^n (y_i - f_j(x_i))^2 + \int_{\Omega} w(x)[f''(x)]^2 dx \quad (2)$$

The key advantage that we found in this method is the explicit attention to removing point error (cleaning) before attempting the global optimization. We have recreated it to an extent in our own approach. A second feature, which we have also used, is the use of local linear fit as an indicator of the deformability of the data at a point. We shall discuss this point further below.

III. A BRIEF REVIEW OF THE INVERSION PROBLEM

In a linear inversion problem, the data parameters z , which are the observation coordinates in our case, are related to the model parameters x, y by a function g which is constant over the x, y , at least locally. In our case, we write the relation as $\mathbf{Z} = \mathbf{X}\mathbf{G}$, where $\mathbf{Z} = \{z_0, z_1, \dots, z_n\}$ is the observation vector, the model parameters are coefficients of the linear operator G , and the mapping function is matrix of the form

$$\mathbf{X} = \begin{pmatrix} x_0 & y_0 & x_0^2 & y_0^2 & 1 \\ x_1 & y_1 & x_1^2 & y_1^2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & x_n^2 & y_n^2 & 1 \end{pmatrix}$$

where $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ are the points over which the plane is being constructed. The generalized least squares technique is an accepted method of solving this problem, though it is nearly impossible to justify that the coefficients of \mathbf{G} are sampled from a gaussian mean pdf with a known covariance vector. However, if we have existing estimates $\hat{\mathbf{G}}$, it is not unreasonable to consider that $\mathbf{G} - \hat{\mathbf{G}}$ is a gaussian in R^n and use the least squares algorithm.

IV. THE ALSD ALGORITHM

As described by various researchers, and in our own estimates, the critical choice in any form of solution of the image inversion problem is the trade-off between the adhesion and tension terms respectively. As we can see from the different algorithms given above, each algorithm suggests a different method for selecting it. The proposed ALSD algorithm is a departure from the algorithms studied above, because it uses the knowledge of existing manifolds to determine whether a point is within a manifold or on a boundary, instead of using purely local information from the point i.e. its gradient or divergence, as in the examples above. Since the underlying manifolds which are themselves not necessarily flat, so the divergence for a point is not necessarily an indicator of tension. Instead, it uses an approach of actively discovering the constituent manifolds and measuring the tension/adhesion of each point with respect to the manifold in which it is (possibly) situated. In this respect, it is somewhat similar to the first stage of [Schunk and Sinha92]. However, whereas [Schunk and Sinha92] uses a fixed neighbourhood of a point purely as a way of measuring discontinuities, we all the neighbourhood to dynamically grow into a full manifold. This allows our algorithm to work nearly completely without

any heuristics and without any domain-specific, user-supplied constants; instead, it adapts to the quality of the data and finds the best underlying *structural representation*, as well as the likely noise term.

We will now discuss the algorithm in more detail. Let us recall the problem statement first. As given II, we have a set of sampled data z , which is represented by a set of polynomials $f_i(x, y)$ in the form $z = \sum_i f_i(z, y) I_{\{(x, y) \in \Sigma_i\}}$ where Σ_i are disjoint sets such that $\bigcup_i \Sigma_i = \Omega$. The polynomials $f_i(x, y)$ are not known initially but have to be estimated as part of the problem. This is further corrupted by a noise process of independent variance σ , which is input to the algorithm.

Our algorithm works by choosing the successively running the manifold selection function, parameterized by the maximum tolerance of each manifold as discussed below.

- 1) Start with the input data set z_0 . Identify the component manifolds and their associated ranges, so that each sample point (x, y) in the image space Ω is mapped to a one manifold. The maximum tension in a given manifold is bounded by below by the parameter T . This tension is measured as $1.0 - \exp(-err(\Sigma_i, f_i))$, where the function $err(A, f)$ measures the mean square error for all the points in set A with respect to the defining polynomial f .
- 2) A point is mapped to a manifold if, by including it in the manifold, the tension of the manifold overall is lower than the parameter T . Thus, whether a point is on a boundary or not is a function of both the point and the manifold to which we are trying to map it. This is the principal innovation in this algorithm; instead of using purely local information as in [Terze88] or [Perona and Malik90], we utilize the overall fitment of the point within a particular region.
- 3) When the tension threshold is close to 1, many points may not be mapped to any manifold, since the noise value causes too high an error
- 4) On the other hand, as the threshold drops towards zero, points will get mapped to manifolds even if the resultant error and consequently, surface tension of the manifold is high
- 5) The surface tension energy plays the role of the deformation term in our algorithm. The parameter T plays the role of the stabilizing function.
- 6) Based on the associated manifold function $f_i(x, y)$ for

the i th manifold and the goodness of fit of all points (x, y) mapped to it, we compute a tension energy as the residual sum of squares over the domain of the manifold.

$$d_i = \sum_{x, y \in \Sigma_i} (f_i(x, y) - z(x, y))^2 \quad (3)$$

- 7) If a point is not mapped to a contour, we associate a deformation energy to it as the absolute distance from the mean height of the entire membrane
- 8) The aggregate energy for the membrane is given as the sum of the energies (both deformation and tension) for all manifolds and unmapped points in the data-set
- 9) The algorithm is run with different values of the deformation parameter T and the best solution is taken. It is fairly clear that:
 - a) As T drops, the number of points not mapped to a contour will drop
 - b) As T drops, the deformation in a contour goes up, since it accepts points with higher mean square error
 - c) At some value of T , an optimum will be achieved. We have implemented a gradient search algorithm to compute the optimal value of T
 - d) Thus the global optimization problem becomes a search for optimal T over a convex domain. We can use a binary search or any other algorithm to find the optimal T and the corresponding set of manifolds Σ_i corresponding to this T .
- 10) When the final manifolds are recovered, each point $z_{\{x, y\}} \mapsto \{\Sigma_p, f_p\}$ is projected onto the constituent manifold to get the ‘‘cleaned image’’. $\hat{z}_{\{x, y\}} = f_p(x, y)$

A. The manifold identification algorithm

To identify individual manifolds we propose an algorithm for edge detection using a localized searching technique. In our case, we start with an initial small set of points and start growing a smooth manifold defined over this initial set. At each step, we note not only the manifold coefficients, but also the distortion in the manifold caused by the constituent points. The distortion of the manifold is checked by fitting it onto a smooth polynomial in n dimensions using a best-fit algorithm. The quality of the fitment is verified by measuring the mean square error between the data set and the smooth surface. The domain of the manifold is improved and grown by absorbing points around the edges of the current manifold, and also

eliminating points which increase the distortion. The distortion is limited by the parameter T , supplied to the algorithm (see above).

For a given point, the induced strain caused by it is measured as Δe , the difference in the average MSE in the manifold with and without that point. The probability of adding the point is given by $\exp(-\frac{\Delta e N_d(e)}{N_i})$, where N_i is the number of points in the manifold at the current time and $N_d(e)$ is the number of points in the manifold which are immediate neighbours of the current point.

Since it is expected that boundary points are continuous, the algorithm is more likely to reject a point if it has a neighbour which is also rejected. Points are isolated in the same manner. Individual points which have high noise will be rejected and the algorithm will grow the manifold around them, leaving them as isolated unmapped points. The local search stops when no further points can be added to a manifold. This will typically happen when there are a number of neighbouring points spanning a border of the manifold, all of which are rejected by the local search. This set is naturally identified as the boundary.

The algorithm is as follows:

- 1) Select a local manifold within the overall grid defined over an initial domain $\Sigma_{i,0}$. Choose points in $\Sigma_{i,0}$ so that they are connected and are not currently mapped to any existing manifold (discovered previously)
- 2) At every iteration, use a curve fitting LMS algorithm to find the best-fitting polynomial $f(\Sigma_{i,t})$ in the chosen number of dimensions over this manifold. Store the associated error $\epsilon(N(\Sigma_{i,t}), \Sigma_{i,t})$
- 3) Randomly select a point $p \in \Sigma_{i,t}$ in the manifold and use the rejection metric i.e.

$$\Delta e = \epsilon(f(\Sigma_{i,t} - p), \Sigma_{i,t} - p) - \epsilon(f(\Sigma_{i,t}), \Sigma_{i,t})$$

$$r = \text{rand}()$$
 if $(r > \exp(-\frac{\Delta e}{f(\Sigma_{i,t})}))$ remove point from manifold else retain point in manifold
- 4) Once all points within the manifold S has been checked out, add a ring of points R to this manifold. A ring comprises a set of points which have at least one immediate neighbour which already an element of the existing manifold.

$$\Sigma_{i,t+1} = \Sigma_{i,t} \cup R$$

$$R = \{x \notin \Sigma_{i,t} : W(x) \cap \Sigma_{i,t} \neq \emptyset\}$$

Here, $W(x)$ is the neighbourhood of the point x .

- 5) Now go back to step 3
- 6) The manifold $\Sigma_{i,t}$ is declared complete if $R = \emptyset$ i.e. there are no points which can be added to S .
- 7) Continue this procedure till the entire grid is completed and it is not possible to create any further manifolds.
- 8) Now evaluate all those points which are not an element of any manifold and mark them outliers. Use any smoothing function to filter out outliers.

The principal advantage of this algorithm is that it is possible to honour exact boundaries and it is computationally relatively cheap. The principal cost of computation is the repeated cost of executing the curve-fitting algorithm. The number of times the curve-fitting must be executed is roughly of the order of the number and length of the edges in the grid. In this regard, it is roughly of the same order of complexity as the first stage of the two-stage algorithm suggested by [Schunk and Sinha92]

B. Finding the locally best fitting surface

As stated above, we used a least squares technique to estimate the best coefficients of the manifold function $f_i()$. We experimented with two possible methods. One was the standard generalized linear regression using the Golub Reinsch singular value decomposition method. Here, there are four variables, x , y , x^2 and y^2 . We used rank decimation to determine when the model was over-specified. The second technique tried was a non-linear fitting method, using the Levenberg-Marquardt transform.

Both techniques proved reasonably robust in the face of errors. One of the problems in the overall algorithm is that the data presented to the curve-fitter may have structural breaks. This happens when the initial starting set is incorrectly chosen in such a way that it overlaps two or more contours. Unfortunately, neither of the considered methods are able to natively detect structural breaks; we are thus forced to rely on the residual sum of squares method and the estimated χ^2 value to detect a structural break.

The accuracy of reconstruction of the constituent polynomials is a function both of the noise and the nature of the polynomial itself. Since the values of x and y are very small, most of the surface fitting algorithms we tested (weighted linear fit, nonlinear Levenburg-Marquardt transform) were unable to compute the coefficients for the square terms.

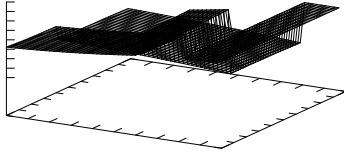


Fig. 2. Original Surface

V. EXPERIMENTAL DATA

We started with the following original surface. The figure is generated using the following four manifolds, defined over the associated regions as given below.

- 1) $f_1(x, y) = 2.3x - 2.2y + 2.92x^2 + 2.67y^2 + 2.455\forall\{(x, y) : 0 < x < 0.3, 0 < y < 0.2\}$: Manifold 1
- 2) $f_2(x, y) = -1.26x + 1.11y - 1.08x^2 - 1.04y^2 - 1.234\forall\{(x, y) : 0 < x < 0.4, 0.2 < y < 0.5\}$: Manifold 2
- 3) $f_3(x, y) = 5.8\forall\{(x, y) : 0.4 < x < 0.5, 0 < y < 0.5\}$: Manifold 3
- 4) $f_4(x, y) = 3.13x + 0.81y + 3.02x^2 + 1.02y^2 + 6.2\forall\{(x, y) : 0.3 < x < 0.4, 0 < y < 0.2\}$: Manifold 4

Note that Ω is defined over $x, y : 0 \leq x \leq 1, 0 \leq y \leq 1$. This data is sampled at 50Hz and then corrupted by a Gaussian noise of known variance over the entire grid. This results in a sampled image of 2500 points as shown below.

A. Analysis of noiseless data

We first tested the algorithm with the clean image, to verify accurate image recovery. The charts below show the original and recovered surfaces. The manifolds were recovered as follows.

- 1) $2.3x + -2.2y + 2.92x^2 + 2.67y^2 + 2.455$: Manifold 1
- 2) $-1.2543x + -0.5759y + -1.0869x^2 - 0.5626$: Manifold 2
- 3) $-1.2611x + 0.0189y + -1.0774x^2 - 0.9524$: Manifold 2
- 4) 5.8000 : Manifold 3
- 5) $7.3083x + 1.1316y + 4.7458$: Manifold 4
- 6) $7.4223x + 1.3202y + 4.6357$: Manifold 4

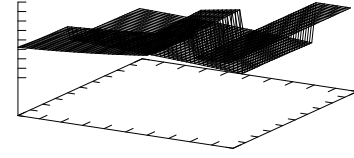
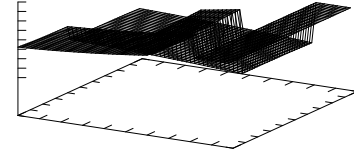


Fig. 3. Analysis of clean data, original and processed

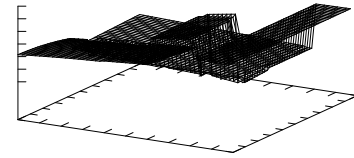
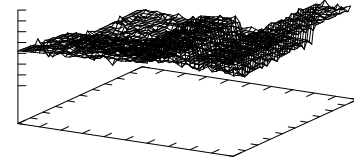


Fig. 4. Analysis of slightly noisy data, original and processed

B. Analysis with slightly noisy data

In the next attempt, have a added a noise of variance 0.5 to the clean image and attempted recovery. The results are shown in 4.

The manifolds returned by the algorithm was as follows. The manifolds were matched by comparing the original domains with the recovered domains.

- 1) $2.1605x + -2.5293y + 0.2369x^2 + 0.7234y^2 + 2.4967$: Manifold 1
- 2) $-1.4291x + 1.0610y + 0.2451x^2 - 1.1700$: Manifold 2
- 3) $1.2713x - 0.5217y + 4.7448$: Manifold 3

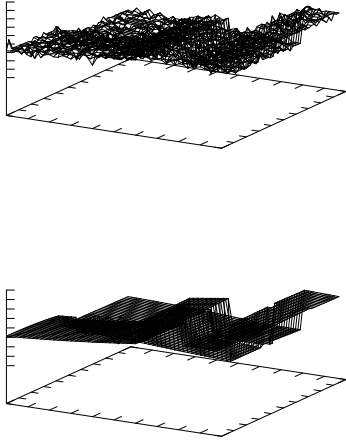


Fig. 5. Analysis of highly noisy data, original and processed

- 4) $0.4716x + 0.3155y + 5.1255$: Manifold 3
- 5) 5.8019: Manifold 3
- 6) $1.9472x + 4.0822$: Manifold 3
- 7) $3.3493x + 0.9138y + 6.0175$: Manifold 4

The mean square error after curve fitting was measured at 0.15.

C. Analysis with noisy data

In the last attempt, we have added noise with a variance of 1.0 and attempted a reconstruction as shown in 5. The algorithm was then run over this data and it resulted in the following output file. A comparison with the original data shows a discrepancy in exactly 11 points (out of 2500) of more than 0.1 (approximately 5% of the mean data value).

The manifolds returned by the algorithm was as follows. The manifolds were matched by comparing the original domains with the recovered domains.

- 1) $2.8278x + 0.5419y + 2.6502x^2 + -5.3279y^2 + 2.2308$: Manifold 1
- 2) $-2.2981x + -0.1727y - 0.6294$: This corresponds to a small region of points whose absolute values (after adding error) was very close to zero and could not be resolved
- 3) $-0.7680x + -0.4917y - 2.1995x^2 - 0.6638$: This corresponds to a small region of points whose absolute values (after adding error) was very close to zero and could not be resolved

- 4) $-0.4600x + -1.6084y - 0.6655$: This corresponds to a small region of points whose absolute values (after adding error) was very close to zero and could not be resolved
- 5) $5.3343x + -1.1404y + 1.6185$: Manifold 2
- 6) $6.6848x + 1.0483y + 5.2052$: Manifold 3
- 7) $0.2015x + 0.6546y + 5.2982$: Manifold 3
- 8) $0.1022x + -0.2056y + 5.7991$: Manifold 3
- 9) $-1.5102x + -0.5141y + 7.6268$: Manifold 4

The mean square error after curve fitting was measured at 0.32. As can be seen above, there were about 33 points which were mapped to a manifold which does not approximately map to any of the original constituent manifolds, since the noise values for these points nearly completely destroyed any useful information. The higher noise also caused less accurate estimation of some of the original

D. Conclusions

We have proposed an algorithm which can be used to analyze noisy data with internal discontinuities to simultaneously remove the added noise as well as recover the original spatial structure of the data. Using synthetic data, we have shown that the noise removal works well in the presence of discontinuities as well as noise. The recovery of structural information, expectedly deteriorates with added noise, but still provides good correlation with the original information.

In future work, we propose to discuss the implementation of this algorithm in a distributed computing environment and also incorporate more sophisticated surface fitting methods, as discussed in IV-B.

ACKNOWLEDGEMENTS

The authors would like to thank Rakesh Mawa and Kartik Sharma for the first level review. Ms. Debdatta Saha gave useful advice on the choice of the multifit algorithm to be used and measures of confidence to be used. Anil Rajput gave valuable advice on L^AT_EX styling and conversion to the IEEE format. The algorithm was implemented using the [GSL, Gnu Scientific Library] for underlying vector, matrix and surface fitting. The authors are grateful to the creators of the GSL package and all the other open source tools used for the purpose of this project.

REFERENCES

- [Terze88] Demetri Terzopoulos, "The Computation of Visible-Surface Representations", IEEE Transactions on Pattern Analysis and Machine Intelligence, July 1988
- [Schunk and Sinha92] Sarvjit Sinha, Brian Schunk, "A Two Stage Algorithm for Discontinuity Preserving Surface Reconstruction", IEEE Transactions on Pattern Analysis and Machine Intelligence, January 1992
- [Figueiredo and Leital] Mario A.T.Figueiredo, Jose M.N. Leita0, "Simulated Tearing: An algorithm for Discontinuity Preserving Visual Surface Reconstruction",
- [Perona and Malik90] Pietro Perona, Jitendra Malik, "Scale-Space and Edge Detection using Anisotropic Diffusion", IEEE Transactions on Pattern Analysis and Machine Intelligence, July 1990
- [Marroquin,Mitter,Poggio88] Marroquin, Mitter, Poggio, "Probabilistic Solution of Ill-Posed Problems in Computational Vision", MIT Artificial Intelligence Laboratory Report 897, March 1988
- [GSL] The Gnu scientific library at <http://www.gnu.org/software/gsl>
- [Tarantola] Albert Tarantola, "Inverse Problem Theory and methods for model parameter estimation", SIAM.