2012 IEEE 27-th Convention of Electrical and Electronics Engineers in Israel

Patch-space Beltrami denoising of 3D point clouds

Aaron Wetzler*
*Faculty of Electrical Engineering
Technion, Haifa 32000, Israel

Guy Rosman†, Ron Kimmel†
†Faculty of Computer Science
Technion, Haifa 32000, Israel

***Abstract*—The Beltrami framework has been shown to be an effective and efficient denoising filter for color images, treating them as two dimensional manifolds embedded in a hybrid spatial-spectral space. Recent work using this framework on the patch-space of an image has demonstrated that including neighboring pixels in the feature space can significantly improve the technique's denoising performance. In this paper we demonstrate an extension of the patch-space Beltrami filter to unstructured point sets. We achieve this by extracting the neighborhood about each point, and using the resulting canonical local frame to perform an explicit iteration of the patch-space Beltrami flow on the normal coordinates. As we demonstrate on real 3D data, the resulting iterative scheme denoises the point set while preserving the underlying manifold structure.**

I. Introduction

Denoising of point clouds, or point set fairing as referred to in the computer graphics literature, is an important post-processing step performed on potentially noisy data obtained from a 3D scanner. Given the drastic increase in the availability of commodity depth cameras the research being carried out in this field is becoming increasingly important.

There are several different solutions to the problem of surface denoising depending on the kind of data. If the scanned data is in the form of a depth image, then many of the methods developed over the years by the image processing community for the purpose of denoising of regular images [20][11][18][1][4] are immediately applicable and require no special treatment apart from possibly scaling the height values.

In a similar, manner many of the same algorithms have been reformulated for use on surface data that is in the form of triangulated meshes and unstructured point sets. Yoshizawa [21] adapted the non-local means algorithm [1] to operate on meshes. Similarly Fleishman [8] showed how the bilateral filter [20][18] could be applied to meshes.

For point sets the method of moving-least-squares (MLS, [10]) is often used to perform smoothing on scanned data. More recently Rosman et al. [13] developed a patch-collaborative method for point set denoising based on the state of the art BM3D [4] algorithm.

More methods for surface fairing are based on diffusion processes. For triangulated meshes, a common diffusion operator is the discrete Laplace-Beltrami cotangent weights formulation of [12], used to perform mean-curvature flow of the surface, including extensions to implicit numerical schemes [5].

For point sets, an anisotropic smoothing process was recently suggested by Lange and Polthier [9]. The method relies on a novel formulation of the underlying surface's Weingarten map which they developed specifically for point clouds.

Recent efforts [14],[19] have shown that denoising of images or depth scans can significantly benefit from operating on so called patch-space within the image. Here we demonstrate how the Beltrami flow operating in patch-space can be used as an anistropic diffusion process that denoises noisy point cloud data.

In the work presented here our contributions are

- An extension of the patch-space based Beltrami method onto unstructured point clouds.
- An exploration of the definition of patches on point cloud data in the spirit of [13], and its use in the context of patch-space Beltrami filtering.
- An investigation of role of normal estimation, and the choice of robust normal estimation [7] as opposed to PCA.

In Section II we shortly describes the Beltrami framework and its patch-space extension. Section III describes the implementation of patch-space Beltrami filtering in point clouds, and details some of the aspects involved therein. In Section IV we demonstrate the results of our algorithm on both real and synthetic noise cases. Section V concludes the paper and describes some of our future directions.

II. The Beltrami framework in patch-space

We now give an overview of the patch-space Beltrami framework. For an in-depth exposition of the mathematical tools required, we refer the reader to standard textbooks in Riemannian geometry [6].

The Laplace-Beltrami is the generalization of the Laplacian on Riemanian manifolds. It has been shown [17] that image denoising can be effectively formulated by considering an image to be a 2D manifold embedded into a higher dimensional spatial-spectral space such as $\{x, y, R, G, B\}$. This embedding can then be traversed iteratively using the so called Beltrami flow which generates a scale-space over the manifold and leads to noise reduction of the image while preserving relevant features such as edges. However an image need not represent color intensity and as such for the following formulation we shall consider a height map which represents the signed distances of points above a plane. In the next section we describe how this graph can be extracted.

We shall start by considering a height map I as being a $2D$ Riemannian manifold embedded in a higher dimensional

space. Similar to previous work [19] we wish to take advantage of the information available in the neighborhood of each point. To this end, we can lift each point into a high dimensional space by considering not only its value but also those of its neighbors within a fixed window known as a patch. For the case of a height map over a discretized domain we thus define the patch-space mapping $P : \Sigma \to M \subseteq \mathbb{R}^{n(2w+1)^2+2}$ such that

$$P(x,y) = \left(x, y, \left\{I^k(x+i, y+j)\right\}\right) \quad , \tag{1}$$

for $i,j = -w,..,w, \quad k = 1,..,n$ where $w \in \mathbb{N}$ is the window size and n is the number of channels we use. For the case of a single height field $n = 1$, however if we were provided with a set of registered scans of a particular surface, n could potentially represent the number of scans. In the following discussion we assume $n = 1$. The manifolds Σ and M are equipped with metrics G and H respectively. We require and thus enforce that lengths measured over both manifolds are the same. To that end, we write that

$$ds^2 = \left(dx\, dy\, dI^k_{i,j}\right) H \begin{pmatrix} dx \\ dy \\ dI^k_{i,j} \end{pmatrix} = (dx\, dy)\, G \begin{pmatrix} dx \\ dy \end{pmatrix}. \tag{2}$$

where $I^k_{i,j}$ is the compact form for $\left\{I^k(x+i, y+j)\right\}$. In reality, the coordinates x and y do not possess the same physical measure as the intensity values of the height field because we are working on an integer-sized grid so we need to introduce a scaling factor β into the patch-space metric H given by

$$h_{pq} = \begin{cases} \delta_{pq} & 1 \le p,q \le 2 \\ \beta^2 \delta_{pq} & 2 < p,q \le n(2w+1)^2 + 2 \end{cases} , \tag{3}$$

where δ_{pq} is the Kronecker delta. We can now use the chain rule $dI^k_{i,j} = \beta^2 I^k_{i,j}{}_x dx + \beta^2 I^k_{i,j}{}_y dy$ from which it follows that when we pullback the metric from the embedding the induced metric tensor is seen to be

$$G = \begin{pmatrix} 1 + \beta^2 \sum_{i,j,k} {I^k_{i,j}}^2_x & \beta^2 \sum_{i,j,k} I^k_{i,j}{}_x I^k_{i,j}{}_y \\ \beta^2 \sum_{i,j,k} I^k_{i,j}{}_x I^k_{i,j}{}_y & 1 + \beta^2 \sum_{i,j,k} {I^k_{i,j}}^2_y \end{pmatrix}. \tag{4}$$

Using this metric we define a measure S on the manifold. For a Euclidean embedding in M, S is none other than the area of the surface as measured in Σ

$$S[\Sigma, G] = Area \propto \iint \sqrt{det(G)} dx dy, \tag{5}$$

where the proportion is up to a scale as a result of the non-unity coefficients of the diagonal entries of H. There is a more general version of the above measure called the Polyakov action which can be useful for non-Euclidean embeddings and details of its application to the Beltrami framework can be found in [17]. We minimize Eq. (5) using variational calculus and then multiply by $g^{-1/2}$ which is permitted to us through the freedom of paramaterization. We eventually arrive at

$$\frac{1}{\sqrt{g}} \mathrm{div}\left(\sqrt{g} G^{-1} \nabla I^k_{i,j}\right) = 0. \tag{6}$$

The left hand operator is recognized to be the Laplace-Beltrami operator and we can now compactly write the reformulation of the Beltrami flow in patch-space as

$$I_t = \Delta_g I. \tag{7}$$

The regular Beltrami flow is a mean curvature flow that preserves intensity edges while eliminating noise. By adding neighborhood pixels and therefore performing the flow in patch-space the aim is to provide additional information that will implicitly extend the weighting of an edge by providing more support. The flow is proportional to $\frac{1}{\sqrt{g}}$ which acts like an edge indicator. It slows down the smoothing of strong ridges on the manifold and in addition accelerates the removal of noisy areas.

III. Algorithm description

In this section we describe one way in which the patch-space version of the Beltrami filter can be applied to a point cloud to perform denoising. Other possible variations on the method are briefly described in the following subsections, as well as in Section V.

The algorithm performs the filtering by extracting a patch around each point in the point set, computing the flow time-step on the patch using the patch-Beltrami filter described in the previous section and then adjusting the current point accordingly. An algorithmic description is given as Algorithm 1.

We now proceed to describe the various substeps of the algorithm.

A. Local neighborhood definition

In a point cloud setting we do not have the true neighborhood relations for each point. We therefore define a neighborhood $\mathcal{N}_i$ of a point p_i to be the set of its k-nearest neighbors intersected with the set of points that lie within a distance R where distance is measured in Euclidean space using the L^2 norm. The main question that arizes now is how to choose R consistently and effectively. We will use $\mathcal{N}_i$ to define a support from which we construct a local frame around p_i. We note that choosing too-big an R will cause the tangent plane estimation to be inaccurate. However choosing R to be too small, it is possible that the noise will dominate the chosen neighborhood leading to an unreliable frame extraction.

In order to mitigate these problems we fix a minimum number of nearest neighbors k_{min} that we would like to have in any neighborhood. We then traverse the point cloud and estimate the average distance $r_{k_{min}}$ to the k^{th}_{min} nearest neighbor of every point. As in almost all denoising algorithms we assume we have some estimate of the level of noise σ that has corrupted the data. Ideally we would like a neighborhood

Algorithm 1 Patch-space Beltrami point set denoising

for iterations $t = 1, 2, \ldots, T$ **do**
  for $i = 1, 2, \ldots, N$ **do**
    Compute point neighborhood $\mathcal{N}_i$.
    Estimate normal direction n_i.
    Use the projection of points in $\mathcal{N}_i$ in the normal direction to obtain a local parameteric surface representation I_i.
    Smooth the neighborhood's representation using the patch-space Beltrami flow.
    Update point i according to its change in I_i in the normal direction.
  end for
end for

to span a larger domain than the strength of the signal and so we define the neighborhood radius R to be

$$R = max\left(5\sigma, r_{k_{min}}\right), \tag{8}$$

which essentially guarantees that we will have at least k_{min} points in any given neighborhood. The number k of neighborhood points is adjusted to reflect the value of R so we set

$$k = \lceil k_{min} \frac{R^2}{r^2_{k_{min}}} \rceil \tag{9}$$

where the squared powers are a result of the very rough assumption that points are distributed more or less evenly per unit area of the tangent plane to each point. In practice the value for k has to be bounded from above because the even distribution assumption is only valid locally and often too many points may be included in some neighborhoods.

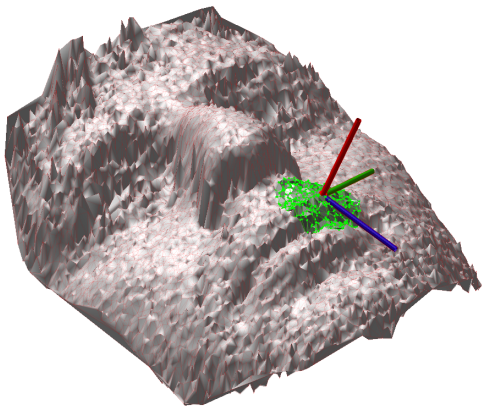

Fig. 1. A local frame and neighborhood estimation result.

We note that the neighborhood is kept constant throughout the all iterations. This is for computational reasons, but this also helps avoid clustering of points together, as noted by Lange and Polthier [9]. Figure 1 demonstrates the selection of a local neighborhood and the accompanying choice of local frame.

B. Frame estimation and local surface interpolation

For point p_i we would like to use $\mathcal{N}_i$ to help us approximate the noisy implicit surface on which p_i is placed. The denoising filter described earlier requires a grid to operate over and as such we need to interpolate the points in $\mathcal{N}_i$ over a discretized domain that spans an approximation of p_i's tangent plane. Approximating the tangent plane requires determining its normal vector n_i. One simple and well established method is to take the third principal component of $\mathcal{N}_i$'s coordinate matrix, performing principal components analysis (PCA) on the local patch. PCA is a linear method and is prone to outlier corruption. In fact the problem of robust normal estimation based on a point's neighborhood is non-trivial and is one of the major influencing factors in the efficacy of any method which requires point normals. Fleishman et al. [7] describe a robust normal estimation procedure which we adopt and discuss its benefits and application later on. Assuming that we have a method to estimate p_i's normal, we define two additional arbitrary vectors u_i, v_i that span the tangent plane and are orthnormal to n_i and to each other. All three vectors are orthonormal and as such form a local tangent frame. We use these vectors and the position of the point p_i to obtain a transformation

$$T_i = \begin{pmatrix} u_i^T & -(p_i)_u \\ v_i^T & -(p_i)_v \\ n_i^T & -(p_i)_n \\ 0\ 0\ 0 & 1 \end{pmatrix}, \tag{10}$$

which will take any point in $\mathcal{N}_i$ in its homogenous representation and transform it into a canonical frame. Provided that the surface does not fold in on itself, the points in this new canonical frame can be interpolated over the domain $[-R, R] \times [-R, R]$ to form a height map I_i for point p_i. In our examples we found a simple bilinear interpolation to be sufficient. Figure 2 demonstrates the estimated patch and local frame.

C. Patch-space update along the normal

The two previous processing steps have transformed $\mathcal{N}_i$ into I_i. In order to update p_i we perform a single Euler step along a discrete version of the flow described by (7), over I_i. We then extract the change in I_i for the central pixel, δ_i, as the central pixel corresponds to p_i. We update point i by the explicit step

$$p_i^{t+1} = p_i^t + \delta_i^t n_i^t, \tag{11}$$

where t indicates the global iteration step. When computing the discrete flow it is important to note that the scaling factor β from the patch-space metric H should have a fixed ratio proportional to the dimensions of the grid and be inversely proportional to the patch radius R because I_i's domain was defined to span twice the patch radius. However this is not necessarily the case in practice so we leave β as a somewhat flexible parameter to the method. We note that for a specific choice of β, the height map update will result in a variant of the mean-curvature flow. Such a flow does not preserve surface edges as is often desired in surface processing. The choice of a high β value, however, along with robust estimation of the surface normals, results in a highly nonlinear flow that preserves edges while removing surface noise.

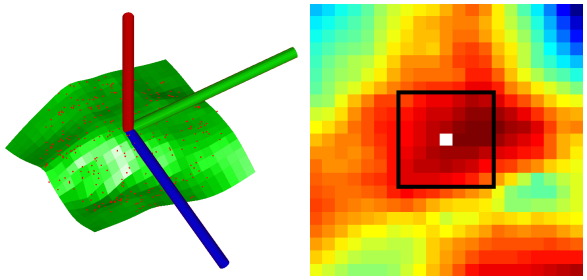

Fig. 2. Left: An estimated local patch and local frame axes. Right: an image representation of the projection of the patch in the normal direction. The black frame represents the patch-space window.

D. Robust normals estimation

The efficacy of the proposed method is largely dependent on how well we can estimate the normals at every point. While for the results shown in Section IV, PCA-based normal estimation was sufficiently accurate, robust normal estimation is desirable. One such method is the method of Fleishman et al. [7] introduced in the context of moving-least-squares approximation.

The gist of the procedure is an iterative RANSAC-like process that builds and tests support sets of points from a neighborhood to try to find the normal vector with the least number of outliers. We found that this method preserved the corners more effectively compared to regular PCA. An example of the difference in the results can be seen in Figure 3

IV. Results

We now proceed to demonstrate the results of our method. We implemented the main algorithm in Matlab and used some functionality from the open source PCL library [16], such as nearest-neighbor indexing. For all experiments we chose a grid size of 21x21 and a window size of 7x7. In principle the grid size should be dependent on the spacing of the data but we found for the data in our experiments it was sufficient to leave it constant.

Figure 4 demonstrates the removal of additive white Gaussian noise from the Bust model. Noise intensity was $\sigma = 0.01$. As can be seen, the resulting surface is quite smooth and the

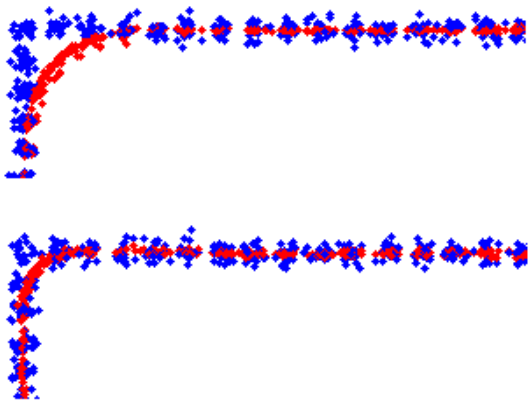

Fig. 3. A comparison of PCA-based and robust normal estimation, in terms of smoothing results. Top: Smoothing results using PCA for normal estimation. Bottom: Results using robust normal estimation.

facial features are preserved despite the strong level of noise, achieving results comparable to other techniques such as MLS [10].

In Figure 5 we demonstrate the denoising of a real 3D scan of a face, taken from a structured light scanner [15]. The results demonstrate the robustness of our method for different noise types, including real scanning artifacts.

Currently, our Matlab implementation is far from optimized, and takes several minutes to run on the Bust model, with 9800 points. The parallel nature of the update step computation makes GPU implementation seems to promise significantly faster results. We intend to pursue this direction in the near future.

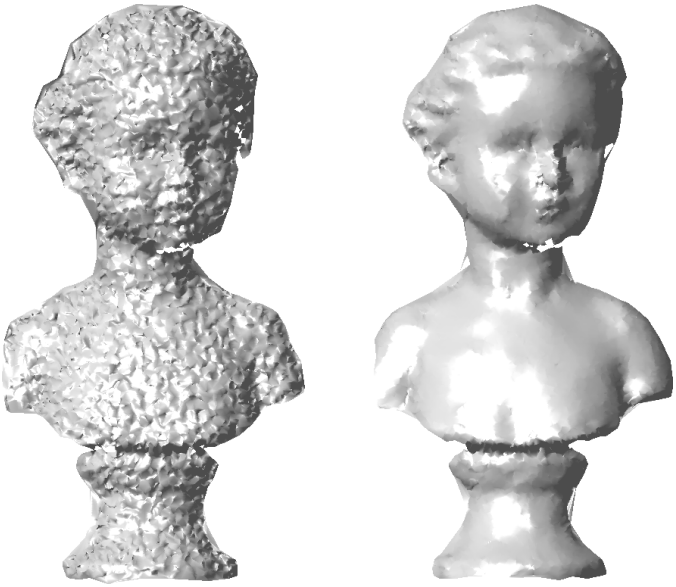

Fig. 4. Left: The standard Bust model perturbed by Gaussian noise with $\sigma = 0.01$ Right: The corresponding model after passing being denoised by the proposed method. Note: The point clouds have been triangulated for the sake of visual clarity only. The proposed method does not assume any mesh topology or triangulation.

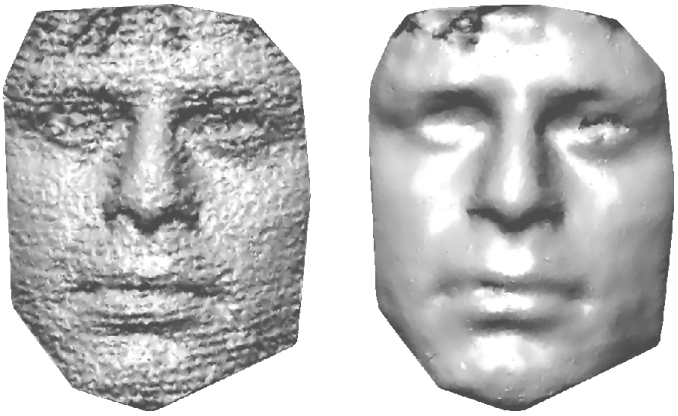

Fig. 5. Left: A noisy point cloud obtained from a structured light scanner. Right: The corresponding point cloud after after passing through the Beltrami filter operating on patch-space using the proposed method.

In all our experiments we kept the neighborhoods of each point constant over all iterations of the algorithm. This dramatically improves the run time of the algorithm and is justified by the fact that we do not expect the neighborhood of a point to change dramatically during the proposed denoising process.

V. Conclusion

In this paper we have presented a method for applying the Beltrami filter to a point cloud using a patch-space based iterative denoising procedure. Furthermore we have investigated the impact of using a more robust frame estimation over a neighborhood of points, compared to regular PCA, and demonstrated that this improves the capacity of the suggested method to restore edges. The results of the experiments are encouraging and open the door for further research into the application of patch and neighborhood based filtering methodologies in point clouds.

Several aspects of this methods are left as open questions which we are investigating. These include different discretizations for the Laplacian operator, and influence of neighboring patches. The choice of similarity measure between patches may include an Iterative Closest Point fitting (ICP, [3], [2]), as shown, for example by Rosman et. al. [13].

Finally, as mentioned before, the algorithm can be extensively parallelized, as we hope to demonstrate in the near future.

Acknowledgements

The authors would like to thank Anastasia Dubrovina and Yonatan Aflalo for stimulating discussions. This research was supported by the European Community's FP7- ERC program, grant agreement no. 267414.

References

[1] B. C. Antoni Buades and J.-M. Morel. A review of image denoising algorithms, with a new one. *SIAM Interdisciplinary Journal*, 4:490–530, 2005.

[2] P. J. Besl and N. D. McKay. A method for registration of 3D shapes. 14(2):239–256, 1992.

[3] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image Vision Comput.*, 10:145–155, April 1992.

[4] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image restoration by sparse 3D transform-domain collaborative filtering. In J. T. Astola, K. O. Egiazarian, and E. R. Dougherty, editors, *Proc. SPIE*, volume 6812, 2008.

[5] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *SIGGRAPH*, pages 317–324, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[6] M. P. do Carmo. *Riemannian Geometry*. Birkhäuser Verlag, Boston, MA, 1992.

[7] S. Fleishman, D. Cohen-Or, and C. T. Silva. Robust moving least-squares fitting with sharp features. *SIGGRAPH*, pages 544–552, 2005.

[8] S. Fleishman, I. Drori, and D. Cohen-Or. Bilateral mesh denoising. *Transactions on Graphics*, 22(3):950–953, 2003.

[9] C. Lange and K. Polthier. Anisotropic smoothing of point sets. *Comput. Aided Geom. Des.*, 22(7):680–692, 2005.

[10] D. Levin. The approximation power of moving least-squares. *Math. Comput.*, 67(224):1517–1531, 1998.

[11] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach Intell*, pages 629–639, 1990.

[12] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

[13] G. Rosman, A. Dubrovina, and R. Kimmel. Patch-collaborative spectral surface denoising. Technical Report CIS-2012-03, Technion, Department of Computer Science, 2012.

[14] A. Roussos and P. Maragos. Tensor-based image diffusions derived from generalizations of the total variation and Beltrami functionals. In *ICIP*, September 2010.

[15] O. Rubinstein, Y. Honen, A. Bronstein, M. Bronstein, and R. Kimmel. 3D-color video camera. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1505 –1509, 272009-oct.4 2009.

[16] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation*, Shanghai, China, May 9-13 2011.

[17] N. Sochen, R. Kimmel, and R. Malladi. A general framework for low level vision. *IEEE Trans. on Image Processing*, pages 310–318, 1998.

[18] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proc. IEEE ICCV*, pages 836–846, 1998.

[19] A. Wetzler and R. Kimmel. Efficient Beltrami flow in patch-space. In *SSVM*, pages 134–143, 2011.

[20] L. P. Yaroslavsky. *Digital Picture Processing*. Springer Verlag New York, Inc., Secaucus, NJ, USA, 1985.

[21] S. Yoshizawa, A. Belyaev, and H. P. Seidel. Smoothing by example: mesh denoising by averaging with similarity-based weights. *Proceedings of International Conference on Shape Modelling and Applications*, pages 38–44, 2006.