A Global 3D Curvature Estimator Applied to Non-Rigid Shape Reconstruction

J.-M. Favreau
Univ. Blaise Pascal - LIMOS - UMR 6158
Campus Scientifique des Cézeaux, 63177 Aubière Cedex, France
Jean-Marie.Favreau@ge.imati.cnr.it

J. Fayad
Queen Mary, University of London
Mile End Road, E1 4NS London, UK

L. Agapito
Queen Mary, University of London
Mile End Road, E1 4NS London, UK

Abstract

Ordering a set of 3D surfaces according to how deformed they are is a non-trivial problem, particularly when the reference “undeformed” surface is unknown. Given a 3D point cloud, in this work we describe a method to estimate its degree of deformation (or how bent a surface is with respect to some rest shape) by estimating the local distortion using normals. First, we describe a novel algorithm to estimate the unoriented normals of a point cloud by using an offset approach. The main contribution of our algorithm is that it can deal with sparse and non-homogeneous 3D point clouds as opposed to previous approaches. We then use the normals to compute the local distortion or deformation - at each point of the surface. Finally, we describe each surface with a unique global distortion measure which is computed from local distortion values. We then apply these global curvature descriptors to the context of 3D reconstruction of non-rigid surfaces from video sequences acquired by a camera observing a 3D deforming shape.

1. Introduction

Designing a method to evaluate the degree of deformation of a surface is not a trivial problem because of the subjectivity inherent to the definition. A more general question would be that of defining a good representation of the shape properties of 3D objects. Extracting global shape descriptors for 3D surfaces described by point clouds or triangulated meshes is a classical problem which has recently attracted increasing attention due to the explosion of research in the area of 3D shape retrieval [14]. A large part of the methods proposed so far aim to extract global information, e.g. to produce a single similarity measure to use in a 3D model search engine. In the specific context of the evaluation of the deformation of an object, a complex descriptor (such as a histogram) would not provide a direct way to sort objects according to such criterion. Even in the case when it were possible to project from the complex shape descriptor space to \( \mathbb{R} \), a single shape feature would always be preferable since it would avoid ambiguities. Lian et al. [8] describe a method to provide a single descriptor (a scalar) to measure how rectilinear a 3D mesh is. However, their descriptor does not measure the degree of deformation of an object with respect to an instance of the rest shape, but rather the straightness of the lines that form the object. In this work we describe a complete method to evaluate the degree of deformation of surfaces with the aim of selecting the least distorted surface from a large sequence of acquired point clouds as a pre-requisite for a non-rigid 3D reconstruction method.

Our global curvature estimation is based on an initial computation of local curvature information. In this paper we focus on the estimation of curvature information for 3D point clouds specifically in the challenging case when the density of the points is low and they are distributed in a non-uniform way. We compute the unoriented normals at each point of the cloud. For this purpose we propose a new offset approach which can be seen as an extension of Chazal et al.’s algorithm [3, 2] to the case of sparse, and non-uniform point clouds. In particular, we propose a dis-
cretization of their algorithm to adapt it to the case of sparse 3D data clouds and we devise a novel adaptive computation of the offset radii to deal with non-uniformly sampled data sets. The result is that each point on the data cloud will be assigned a radius value that takes into account the local geometry of the cloud. The unoriented normals are then combined to define the local curvature descriptors and these in turn are averaged to produce a final global curvature descriptor for the entire surface.

The three original contributions of our work can be summarised as follows. Firstly, we present an original combinatorial approach to compute the unoriented normals of a point cloud, that does not use a Voronoi method, by performing an offset operation in the data set. Critically, our algorithm provides local curvature descriptors in the case of sparse and non-uniformly sampled point clouds overcoming some of the weaknesses of previously proposed methods. Secondly, we propose a new global shape deformation descriptor which is based on the local curvature information given by the normals. We perform experiments where we show that the global descriptor can be used to describe the overall deformation of a surface and to order a collection of deformed instances of a 3D object according to their degree of deformation with respect to a rest shape. Finally we apply the degree of deformation descriptors to select the least distorted shape from a sequence and thus provide an automatic initialization process (i.e. that does not require any a priori knowledge about the manipulated data set) for a non-rigid 3D reconstruction algorithm from video sequences.

2. Related work

Although, to the best of our knowledge, there are no existing works that use a single scalar curvature value as a shape descriptor, some authors use more complex curvature descriptors in the context of 3D shape retrieval. Vandeborre et al. [16] use a curvature histogram to summarize the shape of an object, and to compare it with other shapes. Alternatively, other approaches compute the histogram of the shape index (defined by the two principal curvatures) [18]. The use of a single global descriptor to encapsulate the degree of deformation of a non-rigid object appears to be somewhat novel. Given a sequence of instances of the 3D shape of a single object as it deforms, our global curvature descriptor allows us to select the frame where the object is least deformed (what we describe as its rest shape).

The unoriented surface normal at each point of the cloud is the input to our global distortion estimator. The estimation of surface normals is a classical problem and it is here that the most important contributions of our paper lie, extending the computation of local curvature information to the case of sparse and unevenly distributed point clouds. Dey, Li et Sun [4] underlined that there are two dominant approaches to extract curvature information from a discrete point cloud. The first one operates an optimization method on a tensor system associated to the points [13, 9], leaning on the neighbourhood structure to propagate and smooth the normals. To be able to apply this method, the neighbours of each point of the cloud need to be known in advance. However this a priori knowledge is not available for a point set. The second approach is combinatorial, using Voronoi or Delaunay properties [11, 14]. To this classification we must add the approach of moving least-squares (MLS) [7] in which the normal to the surface is locally estimated by finding the minimum of an energy function. The normal associated to a point is estimated by taking into account the surrounding points in the cloud. The influence of neighbour points is defined by a weighting function, that can be chosen to be the Euclidean or a more specific distance [6]. Contrary to our method, in which the neighbour connections are implicitly deduced, in the MLS approach the computed normals are strongly linked to the way in which the neighbour links are defined by the weights. Finally, an alternative approach to the estimation of the local curvature from a point set would be to first interpolate the data with a global surface [10] and then compute the curvature on the estimated surface [12]. However, this approach suffers from various limitations. First, approximating a surface to a non-homogeneous and noisy point cloud may lead to errors. Secondly, some of these methods require some specific topological properties to be satisfied (e.g. B-splines require quadrangle curve meshes) or imply subtle adjustments of the parameters [1]. Finally, computing geometric information directly on the point cloud simplifies the process since the first approximation step is not required.

In this work we present an alternative approach, by using a discrete implementation of the offset method described by Chazal et al. [3, 2]. Our algorithm can be seen as an extension of theirs where in addition to the discretization that allows practical computations in the case of sparse point clouds, we describe a novel adaptive computation of the offset radii to deal with non-uniformly sampled data sets. We then exploit these results to estimate unoriented surface normals for each point in the cloud. Finally, we apply the global curvature descriptor to select the least deformed instance of a set of 3D shapes obtained from an image sequence. This rest shape is then used to initialise a non-rigid structure from motion algorithm [5] which can recover the 3D shape of a deforming object from image measurements. We show results on sparse 3D data clouds obtained from 3D estimation processes (laser-scans, motion capture data, stereo) on simulated data and parametric surfaces as well as 3D data estimated from 2D image sequences.

3. From point cloud to a global shape estimator

Let $\mathcal{P}$ be the set that describes a sparse cloud of points in $\mathbb{R}^3$. Our aim is to compute a global distortion descriptor of
the 3D surface described by these points. We first compute
the local distortion at each point in the set. The geometric
information needed by this local descriptor are the unori-
tented normals at each point in the 3D cloud.

3.1. Unoriented normal estimation

We first describe the estimation of the unoriented surface
normal at each point of the sparse set \( P \).

3.1.1 Offset approach

Given an offset radius \( r \in \mathbb{R} \), and a point \( q \in P \), let \( K_q^r \)
be the closed ball of radius \( r \) centered at \( q \). Let \( K^r \)
be the union of the balls \( K_q^r, q \in P \). We know that the bor-
der \( \partial K^r \) of \( K^r \) is a spherical polyhedron [3]: its faces are
surface subsets of \( \partial K_q^r \), its edges are arcs of circles con-
tained in \( \partial K_q^r \cap \partial K_j^r \) (with \( i \neq j \)), and its vertices are
points located in the intersection of three or more spheres
\( K_q^r \). We call offset surface of \( p \), denoted \( S_r(p) \), the subset
of \( \partial K^r \) associated to \( K_p \). \( \partial K^r \). It is the union of faces of
the spherical polyhedron \( \partial K^r \). Using the mesh structure
of \( \partial K^r \), we now define the neighbourhood of \( p \in P \) as
\( N_r(p) = \{ q \in P; q \neq p, S_r(p) \cap S_r(q) \neq \emptyset \} \).

Assuming that \( S_r(p) \neq \emptyset \), we define the normal axis as-
so to \( p \) as the mean of all possible lines \( (p, q) \), where
\( q \in S_r(p) \). Figure 2 shows a 2D sketch of the de-
finitions of offset surface \( S_r(p) \) (in red) and the neigh-
bourhood \( N_r(p) = \{ n_0, n_1 \} \) for point \( p \).

3.1.2 Effective computation

From a continuous point of view, the axis is fully defined
by the previous description. However, to deal with sparse
point clouds a discretization process is required to be able
to compute the unoriented normal associated to \( p \) efficiently,
for a given offset radius \( r \).

Starting from an icosahedron, \( N \) subdivisions are ap-
plied to generate a triangulated mesh representation of
\( \partial K_p^r \). The precision of our normal estimator is thus de-
pendent on the value chosen for \( N \). For practical purposes,
we have chosen \( 2 \leq N \leq 5 \) to generate the experimental
results in Section 4.1. Let \( \partial K_p^r \) be the mesh obtained
by subdivision of the icosahedron, and \( V(\partial K_p^r) \) its vertices.

The subset \( S_r(p) \) of vertices \( V(\partial K_p^r) \) that describes
\( S_r(p) \) is computed by comparing the radius \( r \) with each
of the shortest distances between \( v \in V(\partial K_p^r) \) and \( p' \in
P \setminus \{ p \} \). Only vertices that lie outside the other balls form
part of \( S_r(p) \).

The normal axis associated to \( p \) is then computed by us-
ing Principal Component Analysis (PCA) of the lines de-
scribed by \( (p_s - p) \), with \( p_s \in S_r(p) \).

Finally, the neighbourhood of \( p \) \( N_r(p) \) is computed using
the boundary \( \partial S_r(p) \) of \( S_r(p) \), set up by the points \( S_r(p) \)
connected by an edge to a vertex of \( V(\partial K_p^r) \setminus S_r(p) \).
From this border, we build the neighbourhood of \( p \) \( N_r(p) \) by
taking the closest point in \( P \setminus \{ p \} \) to each point in \( \partial S_r(p) \).

Without using a specific data structure to store \( P \) in \( \mathbb{R}^3 \),
every point of \( P \) has to be visited for each vertex in the
mesh \( V(\partial K_p^r) \). If there are \( N \) points in \( P \) and \( M \) vertices in
\( V(\partial K_p^r) \) then the computation of \( S_r(p) \) and the neigh-
bourhood has order of \( O(NM) \) time complexity. If we extend
the computation to all the points in the cloud, we obtain
an algorithm with complexity \( O(N^2M) \). For practical pur-
poses, we have introduced the use of an octree structure to
speed up the algorithm. In Section 4.1 we show how this
allows the algorithm to be visibly accelerated by around 3
orders of magnitude.

3.1.3 Radii estimation

The main parameter to be tuned in the computation of the
unoriented normals is the offset radius \( r \). Chazal et al. [3]
manually set a single radius value for the entire point cloud.
Beyond the question of how to select the best value for each
point cloud, their approach would be valid for a homoge-
neous point cloud, but not for a non-homogeneous one (see
Figure 3), which is the case that interests us.

One approach – which we call the fast algorithm – would be
to adjust the offset radius exploiting the local distance be-
tween points by setting the offset radius of \( p \) to \( r_p = \frac{d_n(p)}{n} \),
where \( n \) is a given parameter, and \( d_n(p) \) is the distance be-
tween \( p \) and its \( n \)-th closest point. Increasing \( n \) has the con-
sequence of smoothing the normals thus reducing the
potential noise in the point cloud.

However, in the context of the application of our algo-
rithm to non-rigid shape estimation (see Section 5), these
smoothing effects would be unwanted since the normals
are used to define accurate local and global curvature de-

Figure 2. Normal axis estimation using an offset method.

Figure 3. Homogeneous radii on a non-homogeneous point cloud.
We choose to impose a unique sphere covering ratio $c \in [0, 1]$ for all the spheres. The radius $r(p)$ associated with a given point $p \in \mathcal{P}$ is then defined as the first $r$ such that $c < |\hat{S}_r(p)|/|V(\partial K^C_p)(p)|$. $c$ thus expresses the percentage of a sphere that is outside other spheres (the red area in figure 4(a)). $r(p)$ can be computed by centering a discrete unit sphere on $p$ and computing, for each vertex of this sphere, the radius for which the corresponding point of $\partial K^C_p$ meets the inside of other spheres.

However, since the point clouds may correspond to open surfaces (possibly with boundary), this method could produce incoherent results. Figure 4(b) shows a sketch of the result obtained with the previously described method for a large radius value. It is clear that it produces some unwanted intersections, e.g. $n_4$ is detected as a neighbor of $p'$.

To solve this limitation, we introduced a maximum acceptable angle $\alpha_{\text{max}} \in [0, \pi/2]$. As illustrated in figure 4(c), the ratio $c$ is only applied to the regions with angles smaller than $\alpha_{\text{max}}$. This adjustment does not modify the inside points (figure 4(a)), but produces coherent results on the boundaries (figure 4(d)). The sphere covering ratio $c$ and the maximum acceptable angle $\alpha_{\text{max}}$ are both free parameters in the algorithm. Tuning these parameters will allow to adjust the surface quality used for normal axis and neighbourhood computation. The result is that each point on the data cloud will be assigned a radius value that takes into account the local geometry of the cloud.

Without using an octree approach for point cloud manipulation, the complexity of this algorithm is of order $O(NM)$, versus $O(N)$ for the fast algorithm described above. In practice, we use an octree structure which allows us to speed up the algorithm by around 3 orders of magnitude (see Section 4.1 and Figure 7).

### 3.2. Local distortion

In the context of our application to non-rigid 3D shape estimation (Section 5) the point clouds are very sparse, therefore a large part of the curvature information must be captured locally. Therefore we choose to exploit this purely local information as a distortion descriptor.

We define the local distortion of the surface at point $p \in \mathcal{P}$ as the weighed mean of the angles described by the normal axis associated to $p$ and those associated to each of its neighbouring points:

$$s(p) = \frac{1}{|N_r(p)|} \sum_{n \in N_r(p)} \frac{d}{||n-p||} \alpha(p, n),$$

where $|| \cdot - \cdot ||$ is the Euclidean distance in $\mathbb{R}^3$, $\alpha(p, n) \in [0, \pi/2]$ is the angle between the normal axis associated to $p$ and $n$, and $d$ is the median of the distances between pairs of neighbouring points over the whole point cloud. We use a weighted mean to reduce the impact of more distant neighbours, taking into account the spatial distribution of the points over the cloud. Points that are closer to $p$ (relative to the median of the pairwise distances over the whole shape) have more influence than distant points.

### 3.3. Global distortion as a shape estimator

The global distortion of the surface is then estimated by computing the mean of the local distortions over the entire surface: $s(\mathcal{P}) = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} s(p)$.

The local distortion estimator will give low values (close to 0) for points lying on flat surface patches, while it will take higher values for points that lie on deformed patches that deviate from a flat configuration. Therefore the mean will capture the global curvature of the surface as a whole. We have applied this specific property in the context of non-rigid 3D shape reconstruction from video sequences to select the least deformed instance of a surface. This rest shape is then used to initialise a non-rigid structure from motion algorithm which can recover the 3D shape of a deforming object from image measurements.

### 4. Experimental results

#### 4.1. Local curvature estimation

We first applied our unoriented normals estimator to a set of complex surfaces, from 3D scanners (figures 5(c) and 5(d)), from artistic modeling (Figure 5(a)), 3D reconstruction (Figure 5(b)) and from parametric surfaces (Figure 5(e)). We have chosen to visualise the normals by associating a sphere to each point on the cloud which is then
splatted along the direction of the normal to a thin disc. The figures show very good qualitative results of the estimation of the nonoriented normals. Except for the parametric surfaces, we are not able to test the algorithm quantitatively. In the example of the torus (Figure 5(e)), the mean angle error on the estimated normals according to the theoretical value is less than 0.009 radians.

To illustrate the suitability of our global distortion measure, we computed its value for a set of point clouds generated from the distortion of an originally planar data set. \( c = 0.3 \).

![a) Original surface. b) Folded surface. c) Bumped surface. d) Twisted surface.](Image)

Figure 6. Global distortion and unoriented normals computed on a set of point clouds generated from the distortion of an originally planar data set. \( c = 0.3 \).

<table>
<thead>
<tr>
<th>Number of points</th>
<th>Mean angular error</th>
</tr>
</thead>
<tbody>
<tr>
<td>86</td>
<td>0.026</td>
</tr>
<tr>
<td>81</td>
<td>0.039</td>
</tr>
<tr>
<td>72</td>
<td>0.057</td>
</tr>
<tr>
<td>64</td>
<td>0.082</td>
</tr>
<tr>
<td>54</td>
<td>0.104</td>
</tr>
<tr>
<td>45</td>
<td>0.120</td>
</tr>
</tbody>
</table>

Table 1. Mean angular error (in radians) for normals on 50 prunings from the same point cloud.

![a) Fast radius estimation algorithm, \( n = 18 \). b) Full radius estimation algorithm, \( c = 0.2 \), \( \alpha_{\text{max}} = 3\pi/8 \).](Image)

Figure 7. Computation time of distortion estimation on various data sets, with fast and full radius estimation, both with and without octree improvements. The sphere is described by 162 points.

normals with respect to the sparsity of the point clouds.

To illustrate the speed-up due to the use of the octree structure, we computed our global estimator on various point clouds generated by subdivisions of the same initial mesh. Figure 7 illustrates the computation time for our algorithms on an Intel@ Core™2 Duo CPU 3.00GHz. As expected, the octree optimizations do not modify the normal values. The mean angle difference between vectors computed with the fast radius algorithm and full estimation is small, e.g. 0.040 radians for a cloud with 6, 273 points, but the estimation of the normals suffers with the fast algorithm when the points are not evenly distributed.

### 4.2. Global curvature estimator on sequences

In Figure 8 we show the results of the global curvature estimator on a synthetic sequence. The synthetic shape simulated a long and thin cylinder to allow strong bending motions. The points were spread around one half of the surface
of the cylinder. The shapes in Figure 8 show the evolution of the cylinder over the sequence. The cylinder started straight and was then bent upwards and then downwards. The plot in Figure 8 shows the computation of the global curvature descriptor for each frame and reflects the symmetry of the sequence. We show that for frames 1 and 26 in which the cylinder was straight, the global curvature descriptor provides the minimum value, close to 0.

In Figure 9 (left) we show some sample frames of the results of our method applied to a real object sequence, acquired with a MOCAP system. This cylindrical object also bends upwards and downwards. Once more the computed normals behave as expected, following the deformations of the shape. In Figure 9 (right) we show the curvature values for the whole MOCAP sequence. The estimates for the curvature roughly follow the deformation of the object. However, due to the small number of points available the curve is not smooth as in the synthetic case, resulting in a noisy curve. Figure 10 shows the estimated normals on sequence of a flag waving in the wind as well as the global curvature values for the whole sequence (data from [17]). The global curvature values for the example frames presented are highlighted on the plot. We can see that these correlate well with the strength of the deformation of the flag.

5. Application to non-rigid shape estimation from image sequences

5.1. The quadratic deformations model

The quadratic deformations model of Fayad et al. [5], similarly to the rigid factorization approach [15], assumes that a set of \( p \) points lying on the surface of a 3D object are viewed by a moving orthographic camera. In the case of a rigid object, the 3D coordinates of a world point \( S_j = [X_j, Y_j, Z_j]^T \) are projected on the image following the orthographic projection equation \( w_{ij} = R_i S_j + T_i \), where \( w_{ij} = (u_{ij}, v_{ij})^T \) are the non-homogeneous coordinates of point \( S_j \) in frame \( i \); \( R_i \) is a \( 2 \times 3 \) orthographic camera matrix that contains the first two rows of a rotation matrix and \( T_i \) is the \( 2 \times 1 \) translation vector.

In the case of deformable objects the observed 3D points change as a function of time, thus this model cannot be used. To tackle this, the quadratic deformations model for non-rigid bodies augments the rigid shape matrix with quadratic and cross-term components. The shape coordinates for point \( j \) would become \( S_j = [X_j Y_j Z_j, X_j^2 Y_j Z_j, (X_j Y_j)(Y_j Z_j)(Z_j X_j)]^T \). The shape matrix \( S \) is then built by stacking the augmented coordinates of all the points on the surface. Applying a quadratic transformation matrix \( A_i \) to the shape matrix \( S \) we obtain the 3D coordinates of the deforming body at each frame \( i \) as \( S_i = A_i S = [L_i, Q_i, C_i] S \) where \( L_i, Q_i \) and \( C_i \) are the \( 3 \times 3 \) transformation matrices associated respectively with the linear, quadratic and cross-term deformations.

Notice that the shape matrix \( S \) is a matrix which encodes the augmented coordinates of the shape at rest, i.e. the undeformed shape. This matrix is fixed for all the frames while the deformation matrix \( A_i \) varies frame-wise. Finally, projecting the 3D shape onto the image gives the 2D coordinates of point \( j \) at frame \( i \) \( w_{ij} = R_i [L_i, Q_i, C_i] S_j + T_i \). The
parameters of the camera matrices, the quadratic deformations and the 3D shape are then estimated using a non-linear optimization approach minimising image reprojection error, defined as the difference between the measured and estimated image points:

$$\arg \min_{R_i, T_i, L_i, q_i, c_i, S_j} \sum_{i,j} \left| \left| w_{ij} - \hat{w}_{ij} \right| \right|^2$$

(2)

Non-linear optimization heavily depends on a good initialization. The camera matrices and the quadratic deformation matrices are initialized assuming the object is rigid.

Unless an a priori 3D model of the undeformed object is provided, the rest shape $S$ must be estimated purely from image data. Fayad et al. assume that the object’s rest shape is observable at the start of the sequence and does not vary for a certain number of frames and use a standard rigid structure from motion algorithm [15] to estimate it from the image observations. In this work we relax this assumption and instead we compute an approximate 3D shape for every frame in the sequence using Tomasi and Kanade’s rigid structure from motion approach [15] over a sliding window of 15 frames. Effectively this gives a collection of 3D shape instances over the sequence. We then use our proposed global curvature descriptor to select the frame with the lowest curvature value and use it to initialise the rest shape in the quadratic deformation model. In this way the algorithm is completely automated.

5.2. Results for non-rigid shape analysis

To illustrate the application of our method, we used the MOCAP sequence of a cylindrical object being bent upwards and downwards (Figure 9). The 3D coordinates of the points were synthetically projected onto 2D images to create a sequence. We then applied Tomasi and Kanade’s rigid factorization algorithm to groups of 15 images, as described in Section 5, to obtain an approximate 3D reconstruction of the shape for each frame in the sequence. In Figure 11 we show the correlation between the global curvature descriptor computed with our proposed algorithm, and the 3D reconstruction error obtained applying Fayad et al.’s method initialised with each frame of the sequence. Figure 11 shows a strong correlation between the two values. We show that low 3D errors are obtained by initialising the algorithm with the shapes that have a low global curvature value. Although the shape with the lowest curvature is not the one that gives the overall lowest 3D error (4.69%), it does provide a very similar error (6.61%). Finally, we show some of the reconstructed frames obtained with the quadratic deformations model when initialized by the shape chosen by our method in Figure 12.
6. Conclusions

In this paper we have presented a new method to provide a global estimator to define the degree of deformation of a surface. First, we described a novel algorithm to estimate the unoriented normals of a point cloud by using an offset approach. The main novelty of our algorithm is that it can deal with sparse and non-homogeneous 3D point clouds. Each surface is then described with a unique global distortion measure which is then used to select the least deformed faces from video sequences acquired by a camera. Since our unoriented normal computation can be affected by large levels of noise, it cannot be easily applied to very noisy and large point sets. However, our approach could be used as the initialization for a tensor voting method to smooth the normals by using the neighbourhoods provided by our method.

7. Acknowledgements

This research has received funding from the European Research Council under ERC Starting Grant agreement 204871-HUMANIS.

References