EUROGRAPHICS 2014 / B. Lévy and J. Kautz
(Guest Editors)

Volume 33 (2014), Number 2

Self-similarity for accurate compression of point sampled surfaces

Julie Digne[1] Raphaëlle Chaine[1] Sébastien Valette[2]

[1]Université Lyon 1; LIRIS; CNRS UMR5205
[2]INSA-Lyon; CREATIS; CNRS UMR5220

Abstract

Most surfaces, be it from a fine-art artifact or a mechanical object, are characterized by a strong self-similarity. This property finds its source in the natural structures of objects but also in the fabrication processes: regularity of the sculpting technique, or machine tool. In this paper, we propose to exploit the self-similarity of the underlying shapes for compressing point cloud surfaces which can contain millions of points at a very high precision. Our approach locally resamples the point cloud in order to highlight the self-similarity of the shape, while remaining consistent with the original shape and the scanner precision. It then uses this self-similarity to create an ad hoc dictionary on which the local neighborhoods will be sparsely represented, thus allowing for a light-weight representation of the total surface. We demonstrate the validity of our approach on several point clouds from fine-arts and mechanical objects, as well as a urban scene. In addition, we show that our approach also achieves a filtering of noise whose magnitude is smaller than the scanner precision.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations I.4.2 [ImageProcessing and Computer Vision]: Compression (Coding)—Approximate methods

1. Introduction

Recent years have witnessed a drastic improvement in scanner acquisition devices which now yield point sets of tens of millions of points at a high precision. The counterpart of this development are datasets requiring ever higher storage capacity. In the absence of efficient yet accurate compression techniques, lower-resolution representations of the shape are commonly used, but at the cost of losing precision.

Furthermore, scanner acquisition devices are often supplied with software packages that exploit proximity and local redundancy of points to reconstruct a polygonal mesh from the point set, with the idea that a triangle is worth a set of points. Therefore research has mainly focused on mesh compression.

Yet, some applications require only a point cloud and do not need a mesh. For example, dense point clouds are better suited to some operations like registration. An acquisition campaign might result in incomplete data, and prematurely building a mesh from these acquisitions is not an optimal strategy. Finally, the global meshing operations are sometimes not accurate enough, and oversmooth the data. Thus,

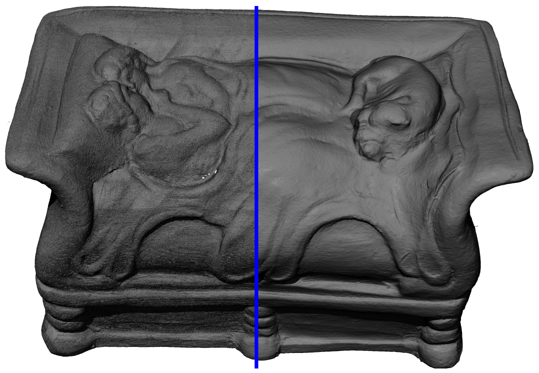

Figure 1: *The Lovers of Bordeaux (*15.8 *million points). Exploiting self-similarity in the model, we compress this representation down to* 1.15 *MB. The resulting model (right) is very close to the original one (left), as the reconstruction error is less than the laser scanner precision (*0.02*mm) for* 99.14*% of the input points.*

there is a dire need for compression methods that preserve the acquisition device precision.

In this paper, we propose a solution to the compression of raw surfaces represented as unorganized point clouds. Unlike meshes, input point sets correspond to samplings of shapes, without any interpretation. From this observation, we build a new compression approach that turns a dense point set surface into another one while keeping the distortion below the input accuracy. We only locally change the sampling on the surface, so as to favor compression strategies, while remaining within the input precision.

The very reason for locally changing the sampling on the surface is to highlight the self-similarity inherent to the shapes, and to use it for compression purpose. The discrete analysis that we perform requires that the different areas of the same object be represented with comparable sampling schemes. Moreover, our goal is not to use a local constraining model as is sometimes done to visualize point set surfaces (Q-Splat [RL00], MLS [ABCO*01]) but to remain closer to the data (shape and sampling), while assuming that each point has a 2-manifold neighborhood at the scale where the self-similarity is analyzed. The outcome is a very efficient and faithful compression approach that performs much better than previous algorithms to compress unstructured surface point sets. Strictly speaking, this method is not lossless since the exact input point cloud cannot be recovered. Yet, the distortion introduced by the compression can be set below the scanner accuracy, as we will illustrate experimentally.

To sum up, our contributions are the followings:

- We introduce a way of encoding point clouds based on shape self-similarity, while being respectful of the scanner accuracy.
- This compression is done at the resolution of the scanner enabling improved control of the point cloud resolution.
- The input point clouds can be oriented or non-oriented.

Figure 1 shows the difference between an original point set and its decompression. Our method achieves a 0.59 bits per point (bpp) bitrate on this model while keeping the reconstruction error below the scanner precision for 99.14% of the points. As a side-effect, noise and registration artifacts are filtered out.

The remainder of this paper is organized as follows: section 2 reviews the related work on 3D compression and self-similarity analysis. Section 3 presents the working assumptions for our framework. Sections 4 and 5 describe respectively the compression and decompression methods, and, finally, section 6 shows quantitative and qualitative results and comparisons.

2. Related Work

2.1. 3D Compression

3D compression has been deeply explored during the two last decades. For our purpose, we can split 3D compression in 3 different categories : point cloud compression, mesh compression and shape compression. Point cloud compression approaches have mostly dealt with coordinates quantization via recursive space partitioning [SK06, GD02, HPKG06, SPS12]. In a nutshell, these approaches consist in inserting the points in a space partitioning data structure (e.g. octree, kd-tree) of given depth, and to replace them by the center of the cell they belong to.

Other approaches encode the shape represented by the point cloud and use this encoding to generate a different point cloud in the decompression phase. Kalaiah et al. [KV05] propose to define a level-of-detail hierarchy of the point cloud by computing the PCA of the points and splitting along the axis corresponding to the largest variation. The final points are then generated by Gaussian sampling. Schnabel et al. [SMK08] segment the point cloud into canonical geometric shapes. Each part is then efficiently encoded individually using vector quantization. Our approach also uses local height maps, but no segmentation or model regression is needed. Instead, the shape will define its own analysis space by itself. Compression based on surface similarity was proposed in [HMHB08], and we will come back extensively on it in the next sections. Our approach will also encode the shapes and generate a different point cloud from this shape encoding.

Mesh compression was initiated by Deering [Dee95], and has been greatly improved over the years. The rationale for mesh compression is that some applications need the original mesh connectivity unchanged, e.g. for simulation applications, or for cases when attribute data are present. Valence-based coding [TG98] has proven to be nearly optimal for single-resolution compression. Progressive approaches such as [AD01, MCAH12] aim at refining the mesh geometry and connectivity in a progressive way, until full resolution is reached. Karni & Gotsman [KG00] decompose the input geometry on a spectral basis and the complexity of this approach is alleviated via mesh partitioning. Space-partitioning approaches have also been proposed, such as [GD02, PK05] and have the ability to encode meshes with complex topologies. However, these approaches give mixed performance at low bitrates.

When only the shape needs to be transmitted, without the need to keep acquisition or sampling data, resampling approaches for shape compression have proven to be the most efficient approaches. Prior to compression, the input model is remeshed into a new structured surface (e.g. mesh with subdivision connectivity). This remeshing step has the advantage to remove tangential information which does not contribute to the general aspect of the shape. Wavelets [KSS00, GVSS00, GGH02] and bandelets [PM05] exploit the fact that meshes exhibit smooth parts, and offer good performance at very low bitrates. However, few robust approaches were proposed for structured remeshing like [LSS*98] and dealing with data generated from complex meshes might be problematic.

2.2. Self-similarity for images

Self similarity of measured signals has gained interest over the past decade: research on signal processing as well as image processing has accomplished outstanding progress by taking advantage of the self-similarity of the measure. In the image processing field, the idea originated in the non-local means algorithm [BCM05]: instead of denoising a pixel using its neighboring pixels, it is denoised by exploiting pixels of the whole image *looking similar*. The similarity between pixels is computed by comparing patches around them. Behind this powerful tool lies the idea that pixel far away from the considered area might entail information that will help processing it, because of the natural self-similarity of the image. This idea gave birth to a thread of work proposing patch-based methods for a wide variety of problems including denoising, super-resolution or inpainting. Recently, the Patch Match algorithm made the similar patch search much faster [BSFG09]. Following the development of the compressive sensing theory [CRT06], the idea arose that there exists spaces, in which the signals would be sparsely represented, that are especially well suited for processing the signals. In particular, [AEB06] introduces the idea of designing dictionaries over which each patch of the image is represented as a sparse linear combination of the dictionary atoms. We will describe more precisely this method below.

As a consequence of its success in image processing, this idea was also introduced for surfaces, defined as meshes or point clouds.

2.3. Self-similarity for surfaces

Self-similarity of surfaces has mainly been exploited for surface denoising applications: the non local means filter has been adapted for surfaces be it meshes [YBS06] or point clouds [AGDL09, Dig12]. It was also used to define a Point Set Surface variant [GAB12] exhibiting better robustness to noise. Self-similarity of surfaces is obviously not limited to denoising purposes. For example, analyzing the similarity of a surface can lead to detect symmetries or repetition structures in surfaces [MGP06], [PMW*08]. An excellent survey of methods exploiting symmetry in shapes can be found in [MPWC12]. The similarity can also be analyzed between surfaces [BLSC*11]. More related to our approach is the method of [ZSW*10] where self-similarity is used to consolidate scans of urban scenes.

To the best of our knowledge, self-similarity for compression has only been proposed once before and in a different setting [HMHB08]. After decomposition of the surface into patches, these are clustered by similarity, and are replaced by the representative of each cluster during decompression. Our approach is more pliant: each patch is represented as a sparse linear combination over a dictionary, allowing for a better depiction of the variety of the patches, and permitting a very high resolution. We will compare our approach to this method and show a clear gain both numerically and visually.

2.4. K-SVD

The K-SVD algorithm is a method for building representations of finite discrete signals as sparse linear combinations over an ad hoc dictionary. It was introduced in [AEB06], and has since then be used for various purposes including denoising [EA06] and image analysis [MBP*09].

Let Y be a $k \times n$ matrix, whose columns are n training signals $(y_i)_{i=1,\cdots n}$, each of them represented by k samples. The idea is to find a dictionary D, composed of d signal atoms, over which each signal y_i can be represented as a linear combination of the dictionary atoms d_j. In other words, one can find a vector of coefficients x_i such that $y_i = D \cdot x_i$. Let us call X the $d \times n$ matrix whose columns are the x_i, then both X and D $(k \times d)$ are solved for by computing:

$$\min_{D,X} \|Y - DX\| \quad s.t. \quad \forall i, \|x_i\|_0 \leq T_0$$

The $\|\|_0$ represents the number of nonzero coefficients in x_i. It is sometimes abusively called the L^0 norm, though it is not a norm (it is not positive-homogeneous). It measures the sparsity of a decomposition (a decomposition is said to be sparse if a lot of the coefficients are zero and thus its L^0 "norm" is small). T_0 constrains the number of nonzero coefficients.

The K-SVD alternates between finding the best sparse representation for Y on D using a pursuit algorithm (e.g. orthogonal matching pursuit) and updating the dictionary. The dictionary update consists in considering iteratively each atom d_k from the previous step: consider the group of samples that use d_k in their representation and compute the representation error E_k^R of the samples that use atom d_k on dictionary $D \setminus d_k$. Then E_k^R is decomposed using Singular Value Decomposition $E_k^R = U\Delta V^T$. The updated atom d_k is the first column of U while the updated coefficients is the first column of V multiplied by $\Delta(1,1)$ (first singular value). This iterative process needs an initial dictionary which is a random subset of the input training signals $(y_i)_{i=1\cdots N}$. In this paper we will use a fast version of this algorithm, with complexity $O(2n^2d)$, that was introduced in [RZE08].

3. Working assumptions

Our algorithm takes as input a point cloud $\mathcal{P}$ which is supposed to be dense enough to unambiguously represent an interpolating surface $\mathcal{M}$ at a fixed resolution coarser than the point set density. It can handle possibly non-oriented point clouds, by estimating possibly inexact normal directions from local neighborhood. If the input data is provided with oriented normals, instead of estimating the normal directions, the provided ones are used.

The algorithm also requires a radius R that corresponds to the scale at which the self-similarity of the surface is to be analyzed. Our framework is based on some precise assumptions on R with respect to the input point set.

- *Topological condition:* R must be set such that $\mathcal{P}$ can be covered by the set of R-neighborhoods corresponding to a subset of seed points in $\mathcal{P}$. Additionally, each R-neighborhood should delimit a topological disk on the underlying surface $\mathcal{M}$ (to enable parameterization over the tangent plane).
- *Sampling condition:* The R-neighborhood of a seed point must contain enough points so that a meaningful patch of surface can be computed. This implies R to be sufficiently large with respect to the scanner resolution.
- *Noise level:* The method assumes that noise magnitude is strictly below radius R.

In the examples provided in this paper, the R-neighborhood of a seed point on the surface is approximated by the R-neighborhood in the ambient space. Therefore, the topological condition amounts to taking the radius R below the reach of the underlying shape, *i.e.* below the distance between a surface point and the shape medial axis. In the case of sharp edges, it is sometimes possible to find a valid covering with nonzero radius. Otherwise, the approximation of the R-neighborhood on the surface should be replaced by a more sophisticated approach. However, it should be noted that the current approximation approach performs well in practice, even when there are sharp edges and invalid neighborhoods (Figures 7, 12), or when dealing with surfaces with boundaries.

4. Compression

The compression algorithm consists in three successive steps: the first one selects a subset of points (the *seeds*) that will serve as center points to cover the surface with local patches and obtain an atlas description. The second one computes a discrete description of the local neighborhood around a seed. This patch description is obtained by sampling the local surface height over a local frame: each neighborhood is resampled and described as a height map over a radial grid (Fig. 2). Finally, the third step compresses the description of the patches by exploiting their self-similarity and building a custom dictionary, over which all descriptors will be decomposed *sparsely*.

4.1. Seeds selection

The first step of the compression algorithm selects a subset of *seed* points. This subset $\mathcal{S}$ has to satisfy the following property: its dilatation of radius R must contain the whole point cloud $\mathcal{P}$. More formally:

$$\forall p \in \mathcal{P}, \exists s \in \mathcal{S}, \|p - s\| \leq R.$$

Each point p can be covered several times. The subset is constructed in a dart-throwing fashion: we pick a first seed, tag all the other points in its R-neighborhood as covered and continue traversing the points until a non-covered point is found, then a new seed is added and the process continues until all points are covered. In order to discard outliers in the point cloud, we require that a point with few neighbors cannot be considered as a seed, at the cost of breaking the covering condition above, and omitting some points (in most cases these outliers represent around 0.1% of the points). In fact, the outliers will only be problematic if there is a cluster of outliers resulting in a spurious patch, however it is very unlikely in real-life cases. On the Lovers pointset (Fig. 1) containing 15.8 million points, using a radius of $0.5mm$, 89256 seeds are selected and each point is covered in average 2.6 times.

4.2. Neighborhood description

Once the seeds are selected, their local R-neighborhoods are sampled on a local pattern, yielding a sampling of the surface local graph on a radial grid (see Fig. 2). To perform this resampling properly and consistently over the surface, the normal $\vec{n}$ and first principal direction $\vec{t}_1$ are first estimated from local covariance analysis on each local neighborhood. If the normal is available (in case of an oriented point set), then only $\vec{t}_1$ is computed. It should be noted that during decompression, $\vec{n}$ and $\vec{t}_1$ can be predicted using covariance analysis on the seeds only, and only three difference angles need to be stored for correction.

Once both $\vec{n}$ and $\vec{t}_1$ are available, a radial grid $(r_i, \theta_j)_{i,j=0\cdots N_{bins}-1}$ is computed such that:

$$r_i = (\frac{1}{2} + i) \cdot \frac{R}{N_{bins}}; \theta_j = j \cdot \frac{2\pi}{N_{bins}}$$

A different local sampling pattern could have been used (e.g. rectangular grid, Poisson sampling), provided the same pattern is used for the whole surface and the resolution of the pattern is close to the point cloud resolution. This radial pattern is well adapted since more points are sampled near the seed, where it is likely that there is less multiple coverage. The seeds coordinates are encoded using a kd-tree based approach similar to [GD02]. The difference angles are quantized and compressed using arithmetic coding. For the Lovers of Bordeaux, bitstream sizes are 312KB for the seeds coordinates and 297KB for the angle differences, both quantized on 8 bits.

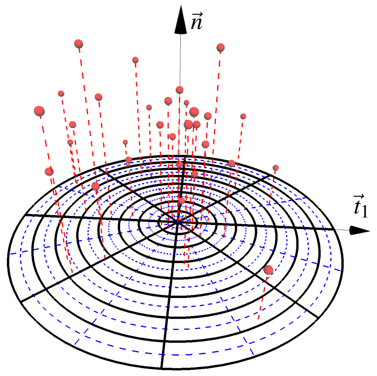

Figure 2: *Local neighborhood description: a height map over a radial grid*

4.3. Self-similarity compression

To exploit self-similarity between neighborhoods, our approach relies on building a dictionary, a set of atoms over which the patches will be decomposed. Both the dictionary atoms and the coefficients of the patch decomposition are solved for by the K-SVD algorithm [AEB06], which minimizes the error while maintaining the sparsity of the coefficients, as explained in section 2.4. This algorithm yields sparse linear decompositions of the patches (*i.e.* with few nonzero coefficients). In addition, the nonzero coefficients distribution presents itself with a sharp peak centered around 0, making them suitable for compression. Therefore we use scalar quantization followed by entropy coding to compress the coefficients. In addition to its purpose of reducing the file size, this quantization improves the sparsity of the coefficients. On the Lovers pointset, the percentage of nonzero coefficients is about 30% before quantization and 20% after quantization (on 8 bits).

Our content-based self-similarity approach compares favorably with the Discrete Cosinus Transform (DCT) which is known to concentrate the energy on the low frequencies but is fixed independently of the signal. This DCT is performed on the same neighborhood description and the same amount k of nonzero coefficient is kept (corresponding to the k DCT basis functions accumulating the highest energy). Then the coefficients are quantized using the same scalar quantization on 8 bits. The Root Mean Square Error (RMSE) is higher for the DCT: 0.14 against 0.01 for our approach.

Figure 3 shows examples of built dictionaries for two shapes. For the Lovers of Bordeaux, the dictionary and coefficients are respectively encoded on 18KB and 507KB.

Tuning parameters and controlling the accuracy

The proposed approach is driven by 5 parameters : the patch radius R, the patch discretization size N_{bins}, the number of atoms in the dictionary N_{atoms}, and the quantization for the seeds coordinates and patch coefficients.

In all our experiments, we set $N_{bins} = 16$ (i.e. the average

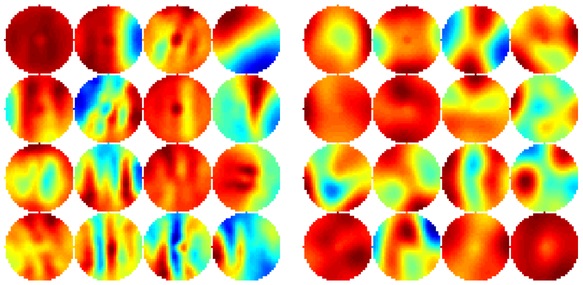

Figure 3: *Dictionaries built for two different shapes: a geometrical one (the mire, left) and a fine art one (the Lovers, right). The atoms are shown by order of importance (total absolute weight in the linear decompositions).*

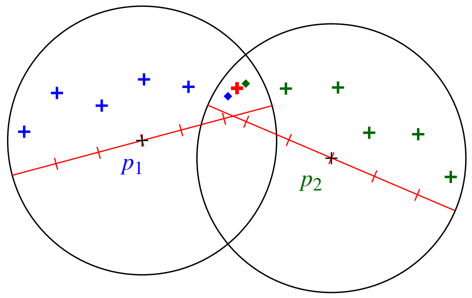

Figure 4: *Recovering the points from the patches: Seeds p_1 and p_2 potentially contribute an amount of N_{bins}^2 points. To avoid oversampling in overlapping areas, each seed provides a position (blue and green diamonds) that will contribute to the final red bold point.*

number of points per neighborhood is around N_{bins}^2) and set the radius accordingly, so that the resolution of the sampling is close to the resolution of the scanner. This gives a rough way of computing the radius parameter.

The number of atoms is also easy to set: it will impact directly on the preservation of the details. If the surface has low detail variation (for example a mechanical artifact or a shape where all details have roughly the same size), 16 atoms is enough to represent the shape. In case of more detailed surfaces (e.g. the Tanagra with details at various size) we took 32 atoms. For a more precise control on the accuracy of the compression, the number of atoms in the dictionary can also be gradually increased until the k-SVD approximation error drops below the accuracy of the scanner. As long as the target accuracy is not reached, atoms are added to the current dictionary which is further updated. This approach is valid since it should be noted that the approximation error decreases towards zero for a dictionary in which every seed is an atom.

Quantization was set to 16 bits for the seeds coordinates and 8 bits for the patch coefficients.

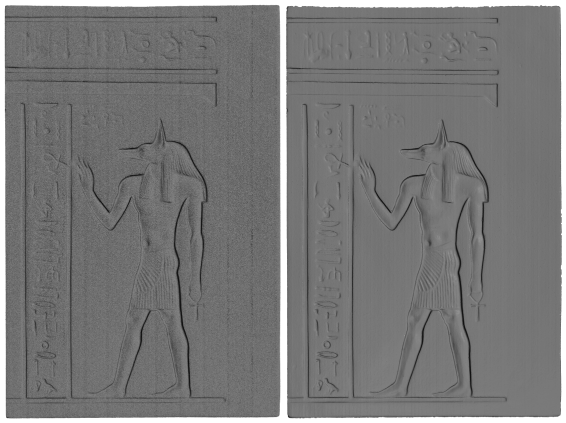
Figure 5: *Compression and decompression of the Anubis point set (Left:original, right: decompression).* $R = 0.7mm, N_{bins} = 16$

5. Decompression

The decompression algorithm consists in two steps: the first one decompresses the patches given the dictionary and the coefficients. The second step translates the set of seeds and the decompressed patches into the final point cloud.

Given the dictionary D and the coefficients matrix X, the reconstructed patches are computed as $Y_{rec} = D \cdot X$, therefore for each seed, the corresponding decompressed patches are recovered. The next step is to go from the set of decompressed patches to the final point cloud. In order to avoid high variations of sampling density in areas of overlap between neighboring patches, we perform a consolidation of the samplings in those areas.

Each bin (r, θ) of the sampling pattern $\mathcal{F}_s(r, \theta)$ corresponding to seed s yields potentially a point, with coordinates in the local frame $(s, t_1(s), t_2(s), n(s))$:

$$(r\cos(\theta), r\sin(\theta), \mathcal{F}_s(r, \theta))$$

which are translated in the global coordinate system as: $s + r\cos(\theta)t_1(s) + r\sin(\theta)t_2(s) + \mathcal{F}_s(r, \theta)n(s)$. Since the patches overlap, one needs a merging strategy in overlapping areas (Fig. 4). More precisely, when adding a decompressed point p, we count the number of seeds whose neighborhood contains p. If there is a single seed the point is kept as is. Otherwise, p is projected onto the parameterized tangent planes of each of the seeds. Each of those seeds s can then predict the position of p from its own patch, yielding a position p^s, and the final position of p is the mean of the predicted positions p^s. On a side note, using radial grids makes the overlap computation easier than with square grids.

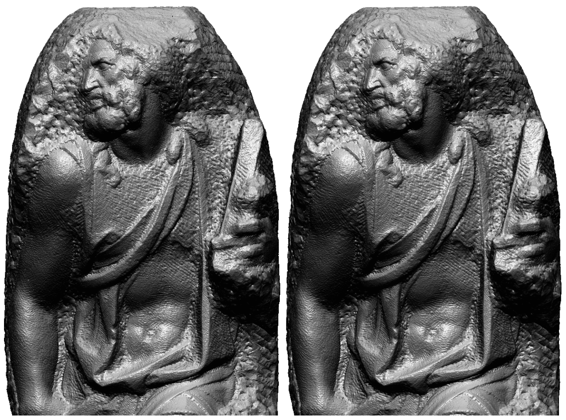
Figure 6: *Rendering of the original (left) and decompressed Stanford St Matthew (right), both have* 93.4 *million points.* $R = 3mm, N_{bins} = 16$

6. Results and comparisons

The compression was tested on shapes from the Farman Institute dataset [DAL*11], the Stanford Michelangelo Project and the Robotic 3D Scan Repository (University of Osnabrück). Oriented normals were provided in all the shapes except for the Bremen dataset and the St Matthews. As for computation times, for the 90 million points St Matthew pointset, the seed selection (800000 seeds) and local patches computations take around 10 minutes. The K-SVD algorithm takes 3 additional minutes, while the rest of the operations is almost instantaneous. For the David pointset, the whole compression takes around 8 min. In comparison [SMK08] needs 11 min just to decompose the 28M points David model into height maps. Moreover, compressing this dataset with the less efficient bzip2 algorithm takes about 6 minutes. The decompression step takes more time: while decoding the coefficients and local patterns is instantaneous, the non-optimized point generation takes around 15 min for the St Matthew point set and 10 minutes for the David point set. Due to the locality of this step, further optimizations and parallelization would undoubtedly drastically reduce decompression time.

Figure 5 shows the detail recovery as well as the denoising obtained through a compression/decompression cycle. One can see that the details are very well-preserved, while the noise is removed. One can check that only noise is removed by comparing the error (obtained with the fixed parameters proposed) with the scan resolution: in most cases this error is below the scanner resolution. Figure 6 shows the compression/decompression result on the dense St Matthew pointset [LPC*00], which has been reduced to 93,4M points due to memory limitations (4GB). One can see that the details are very well preserved. Figure 7 shows the behavior of the decompressed surface with respect to a standard reconstruction algorithm [KH13]. Quasi-identical results are

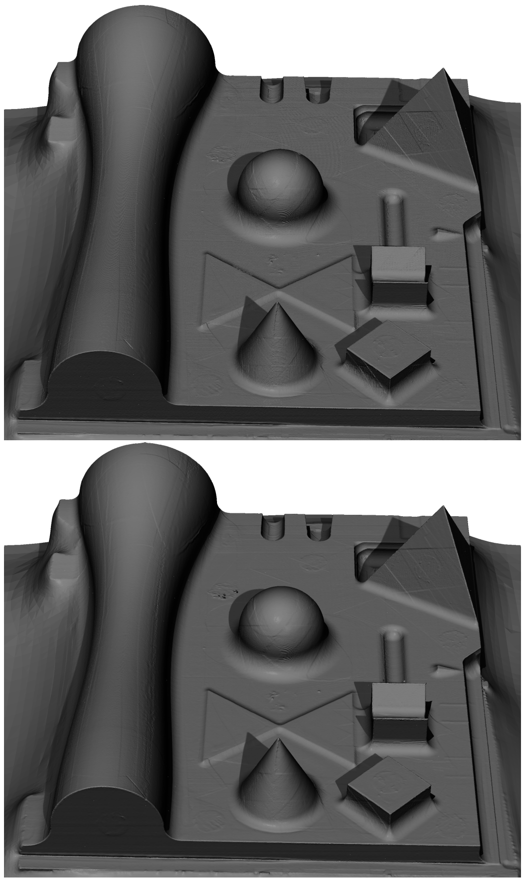

Figure 7: *Original mire pointset (top) and decompression (bottom). Both pointsets were reconstructed using Screened Poisson Reconstruction [KH13].* $R = 0.6mm, N_{bins} = 16$

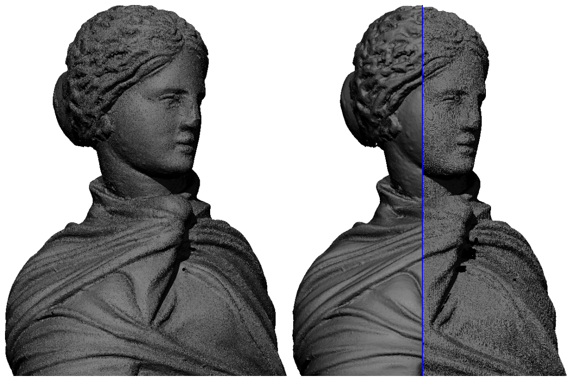

Figure 8: *Comparison with kd-tree based coding. Left : original point cloud. Right : comparative decompression. Even with more bits per point (*4.83 *against* 0.6 *in our method), the right part encoded with [GD02] is less accurate than our approach (left part).* $R = 0.7mm, N_{bins} = 16$

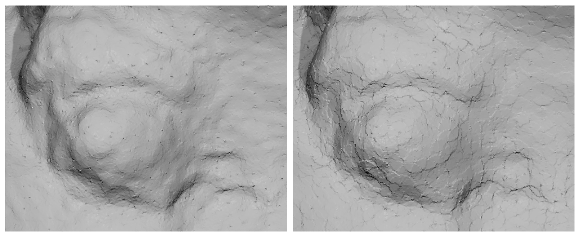

Figure 9: *Comparison with an approach similar to [HMHB08] (right). Our method (left) avoids creating artifacts by being more pliant. Quantitatively, the RMSE for the compared method is* 0.12*mm* (0.01*mm in our case).* $R = 0.5mm, N_{bins} = 16$

obtained with the original and with the decompressed point sets.

For the following comparisons, we measured decompression error as the root mean square normal distance between two point clouds (except for the curves of Fig 10, where it is measured as the root mean square distance to the nearest point, to be consistent with the results presented in the corresponding papers). We have compared our method with a kd-tree based approach similar to [GD02] which compresses unoriented point clouds. At 4.13bpp, [GD02] yields a RMSE of 0.066mm. In comparison our approach gives a RMSE of 0.010mm at 0.6bpp. The results are compared visually on figure 8. We also have compared our approach with an approach close to [HMHB08], by computing self-similarity clusters and replacing each patch by the representative of its cluster. We allowed for 16 clusters (the same size as our dictionary). We can see that numerous artifacts appear (Figure 9) which translates into a higher error (0.12mm vs 0.01mm with our method for the Lovers point set). Figure 10 compares our approach with previous works [KV05,SMK08,HMHB08] in terms of rate/distortion. For our approach, we obtained several bitrates simply by varying the patch radius R. Note that our approach clearly outperforms previous works. Finally, Table 1 shows quality measures for the compression/decompression of several point clouds. In particular, it shows that the amount of points that have higher decompression error than the initial resolution is low, illustrating therefore that the compression does not hinder precision.

Figure 11 shows the behavior of the algorithm with respect to added outliers. The only outliers that hinder the process are the ones that fall within a distance R to the surface. Their neighborhoods might indeed contain enough points to create a patch and therefore decompression artefacts. Those outliers can be considered as large noise. This confirms our working assumptions (section 3). Outliers lying far from the surface are well discarded by the algorithm.

On figure 12 we show the compression results on a very challenging data set of 70 million points that clearly breaks our working assumption: some linear structures exist and

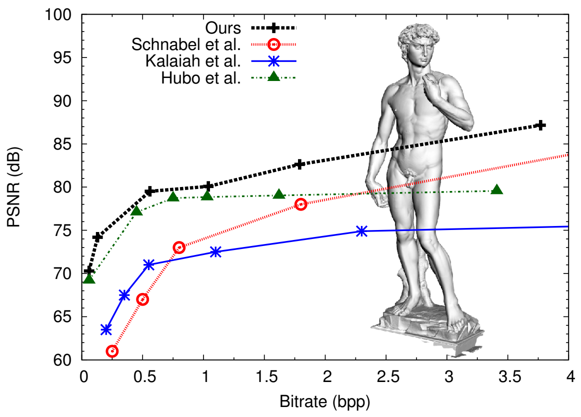

Figure 10: *Comparison with previous works in terms of rate/distortion on the David model. The different bitrates were obtained by increasing the radii of the patches and the size of the descriptors.*

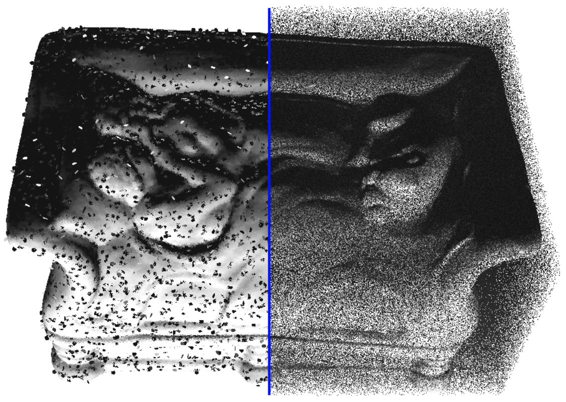

Figure 11: *Compressing the Lovers with* 10% *of outliers (Left:decompressed; Right:initial)*

there are some low density areas. The result is still valid in densely sampled areas. However, sparsely sampled areas may be filtered out by the preprocessing step (section 4.1), showing the limits of the algorithm.

7. Conclusion

We introduced a very efficient framework for compressing dense point sets describing a surface. First, the method selects a subset of the points, it then computes local descriptions of the selected points and use the similarity between the descriptions to encode them.

Our approach encodes tens of millions of points scanned over a surface with less than 1 bit per point. Furthermore, the difference with the input point set is related to noise. So, the decompressed point set behaves as well, if not better, than the original point set, for the purposes of a reconstruction algorithm or visualization.

There are several ways in which our compression scheme could be improved:

- Exploiting patch-based representation, artifacts may appear in case of boundaries, which could be dilated throughout decompression. One could mitigate this issue by adjusting the patch size (clipping some outer grid cells) along boundaries. This would require to store one small integer for each patch, at a small cost.
- Other seed picking strategies could be implemented, for example by placing the seeds so that they minimize the local error, in the spirit of [OBS06].
- Encoding per-point attribute such as normals and colors is possible with the same similarity-based coder.

Perspectives : Although our algorithm is based on the exploitation of self-similarity on the whole surface, most of the involved treatments remain local. This is a good prospect for handling data of ever increasing size, using streaming processes. This is particularly important at a time when the geometric digitization campaigns sometimes cover entire cities.

Acknowledgements

The authors wish to thank Pierre-Marie Gandoin for providing his compression code and valuable discussions. The Bremen dataset (Fig. 12) is courtesy of Dorit Borrmann and Jan Elseberg from Jacobs University Bremen, Germany (the dataset is available from the Robotic 3D Scan Repository of Universität Osnabrück). The Saint Matthew (Fig. 6) and David (Fig. 10) point sets are courtesy of the Digital Michelangelo Project, Stanford University. This work was partially supported by the Région Rhône-Alpes (ARC6).

References

[ABCO*01] Alexa M., Behr J., Cohen-Or D., Fleishman S., Levin D., Silva C. T.: Point set surfaces. In *Proc. Vis '01* (Washington, DC, USA, 2001), IEEE Computer Society, pp. 21–28. 2

[AD01] Alliez P., Desbrun M.: Progressive encoding for lossless transmission of 3d meshes. In *ACM Siggraph Conference Proceedings* (2001), pp. 198–205. 2

[AEB06] Aharon M., Elad M., Bruckstein A.: K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Trans. Sig. Proc. 54*, 11 (Nov. 2006), 4311–4322. 3, 5

[AGDL09] Adams A., Gelfand N., Dolson J., Levoy M.: Gaussian kd-trees for fast high-dimensional filtering. *ACM Trans. Graph., Proc. SIGGRAPH 28* (July 2009), 21:1–21:12. 3

[BCM05] Buades A., Coll B., Morel J.-M.: A non-local algorithm for image denoising. In *In CVPR* (2005), pp. 60–65. 3

[BLSC*11] Boyer D. M., Lipman Y., St. Clair E., Puente J., Patel B. A., Funkhouser T., Jernvall J., Daubechies I.: Algorithms to automatically quantify the geometric similarity of anatomical surfaces. *Proceedings of the National Academy of Sciences 108*, 45 (2011), 18221–18226. 3

[BSFG09] Barnes C., Shechtman E., Finkelstein A., Goldman D. B.: Patchmatch: A randomized correspondence

Pointset	number of points	R	compressed size (bytes)	RMSE (% of diagonal)	Percentage of points with error above sampling precision	bpp
Anubis	$9,9M$	$0.7mm$	$1,201,636$	0.01mm (0.003%)	1.23%	0.96
Lovers of Bordeaux	$15,8M$	$0.5mm$	$1,152,245$	0.01mm(0.006%)	0.86%	0.59
Mire	$16,1M$	$0.6mm$	$1,480,118$	0.03mm (0.011%)	1.30%	0.73
Tanagra	$16,4M$	$0.7mm$	$1,238,271$	0.01mm (0.004%)	1.56%	0.60
David	$28,2M$	$10mm$	$2,150,711$	0.24mm (0.004%)	0.75%	0.61
Bremen	$69,9M$	$18cm$	$6,699,915$	1.48cm (0.005%)	not available	0.76
St Matthew	$93,5M$	$3mm$	$9,780,886$	0.05cm (0.002%)	not available	0.83

Table 1: *Number of bytes compressed versus decompressed. The last column uses the acquisition device precision, information that we lacked for both the Bremen and St Matthew point clouds. The St Matthew pointset (Fig 6) was first reduced to 93,4M points due to memory limitations (4GB).*

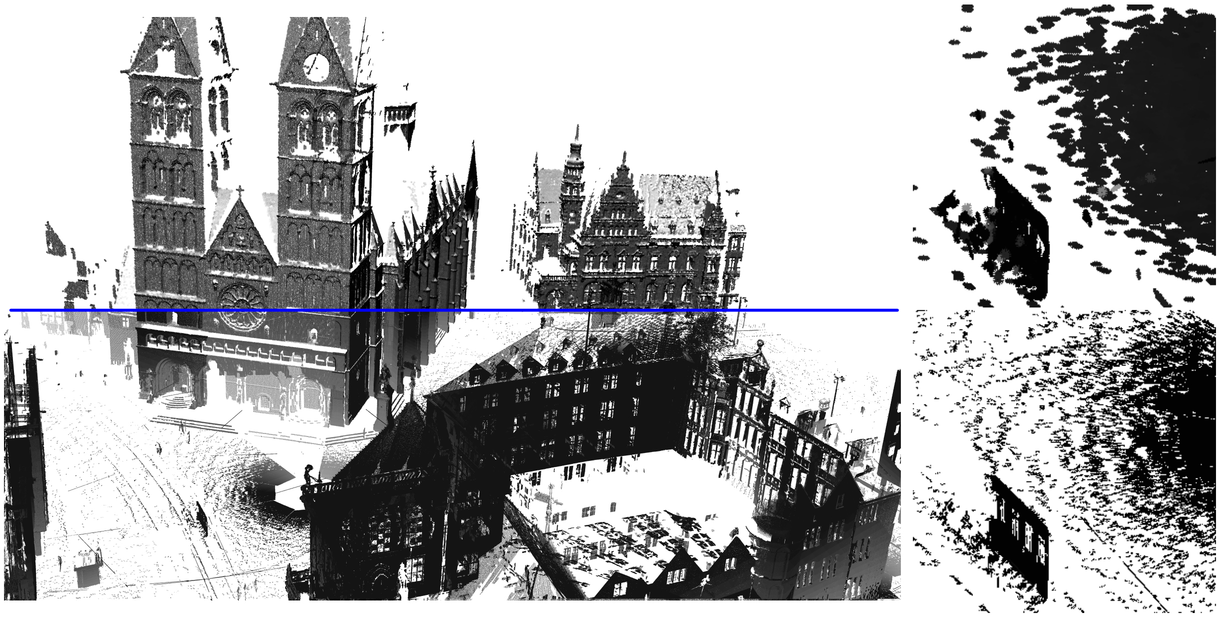

Figure 12: *Left : the Bremen point cloud (bottom) and the decompressed result (top). Right : closeup views : original (bottom) and decompressed (top) $R = 18cm, N_{bins} = 16$. In sparsely sampled areas, two phenomena occur: if the sampling is too low, the points are removed during the pre-processing step and if the sampling is above the limit, there is a dilatation effect clearly visible in the closeups (right).*

algorithm for structural image editing. *ACM Trans. Graph. 28*, 3 (July 2009), 24:1–24:11. 3

[CRT06] CANDÈS E. J., ROMBERG J. K., TAO T.: Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics 59*, 8 (2006), 1207–1223. 3

[DAL*11] DIGNE J., AUDFRAY N., LARTIGUE C., MEHDI-SOUZANI C., MOREL J.-M.: Farman Institute 3D Point Sets - High Precision 3D Data Sets. *Image Processing On Line 2011* (2011). 6

[Dee95] DEERING M.: Geometry compression. In *ACM Trans. Graph. Proc. SIGGRAPH '95* (New York, NY, USA, 1995), ACM, pp. 13–20. 2

[Dig12] DIGNE J.: Similarity based filtering of point clouds. In *CVPR Workshops* (2012), IEEE, pp. 73–79. 3

[EA06] ELAD M., AHARON M.: Image denoising via learned dictionaries and sparse representation. In *In CVPR* (2006), pp. 17–22. 3

[GAB12] GUILLEMOT T., ALMANSA A., BOUBEKEUR T.: Non local point set surfaces. In *Proc. 3DIMPVT 2012* (2012), pp. 324–331. 3

[GD02] GANDOIN P.-M., DEVILLERS O.: Progressive lossless compression of arbitrary simplicial complexes. *ACM Trans. Graph., Proc. SIGGRAPH 02*, 3 (July 2002), 372–379. 2, 4, 7

[GGH02] GU X., GORTLER S. J., HOPPE H.: Geometry images. *ACM Trans. Graph. Proc. of SIGGRAPH 02) 21*, 3 (2002), 355–361. 2

[GVSS00] GUSKOV I., VIDIMČE K., SWELDENS W., SCHRÖDER P.: Normal meshes. In *ACM Trans. Graph. Proc. SIGGRAPH '00* (New York, NY, USA, 2000), pp. 95–102. 2

[HMHB08] HUBO E., MERTENS T., HABER T., BEKAERT P.:

Self-similarity based compression of point set surfaces with application to ray tracing. *Comput. Graph. Point-Based Graphics 32*, 2 (Apr. 2008), 221–234. 2, 3, 7

[HPKG06] HUANG Y., PENG J., KUO C.-C. J., GOPI M.: Octree-based progressive geometry coding of point clouds. In *Proc. Point-Based Graphics 06* (2006), SPBG'06, Eurographics, pp. 103–110. 2

[KG00] KARNI Z., GOTSMAN C.: Spectral Compression of Mesh Geometry. In *ACM Siggraph 00 Conference Proceedings* (2000), pp. 279–286. 2

[KH13] KAZHDAN M., HOPPE H.: Screened poisson-surface reconstruction. *ACM Transactions on Graphics* (2013). 6, 7

[KSS00] KHODAKOVSKY A., SCHRÖDER P., SWELDENS W.: Progressive Geometry Compression. *ACM Trans. Graph. Proc. SIGGRAPH 00* (2000), 271–278. 2

[KV05] KALAIAH A., VARSHNEY A.: Statistical geometry representation for efficient transmission and rendering. *ACM Trans. Graph. 24*, 2 (Apr 2005), 348–373. 2, 7

[LPC*00] LEVOY M., PULLI K., CURLESS B., RUSINKIEWICZ S., KOLLER D., PEREIRA L., GINZTON M., ANDERSON S., DAVIS J., GINSBERG J., SHADE J., , FULK D.: The digital michelangelo project. In *ACM Trans. Graph. Proc SIGGRAPH 00* (2000), pp. 131–144. 6

[LSS*98] LEE A. W. F., SWELDENS W., SCHRÖDER P., COWSAR L., DOBKIN D.: Maps: multiresolution adaptive parameterization of surfaces. ACM, pp. 95–104. 3

[MBP*09] MAIRAL J., BACH F., PONCE J., SAPIRO G., ZISSERMAN A.: Non-local sparse models for image restoration. In *ICCV* (2009), pp. 2272–2279. 3

[MCAH12] MAGLO A., COURBET C., ALLIEZ P., HUDELOT C.: Progressive compression of manifold polygon meshes. *Computers & Graphics, Proc. SMI 2012 36*, 5 (2012), 349 – 359. 2

[MGP06] MITRA N. J., GUIBAS L. J., PAULY M.: Partial and approximate symmetry detection for 3d geometry. *ACM Trans. Graph. Proc SIGGRAPH 06* (2006), 560–568. 3

[MPWC12] MITRA N. J., PAULY M., WAND M., CEYLAN D.: Symmetry in 3d geometry: Extraction and applications. In *EUROGRAPHICS State-of-the-art Report* (2012). 3

[OBS06] OHTAKE Y., BELYAEV A., SEIDEL H.-P.: A composite approach to meshing scattered data. *Graph. Models 68*, 3 (May 2006), 255–267. 8

[PK05] PENG J., KUO C.-C. J.: Geometry-guided progressive lossless 3d mesh coding with octree decomposition. *ACM Trans. Graph., Proc. SIGGRAPH 05,*, 3 (July 2005), 609–616. 2

[PM05] PEYRÉ G., MALLAT S.: Surface compression with geometric bandelets. In *ACM Trans. Graph. Proc. SIGGRAPH '05* (2005), ACM, pp. 601–608. 2

[PMW*08] PAULY M., MITRA N. J., WALLNER J., POTTMANN H., GUIBAS L. J.: Discovering structural regularity in 3d geometry. *ACM Trans. Graph., Proc SIGGRAPH'08*, 3 (2008), 43:1–43:11. 3

[RL00] RUSINKIEWICZ S., LEVOY M.: Qsplat: a multiresolution point rendering system for large meshes. SIGGRAPH '00, pp. 343–352. 2

[RZE08] RUBINSTEIN R., ZIBULEVSKY M., ELAD M.: *Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit*. Tech. rep., 2008. 3

[SK06] SCHNABEL R., KLEIN R.: Octree-based point-cloud compression. In *Symposium on Point-Based Graphics 2006* (July 2006), Botsch M., Chen B., (Eds.), Eurographics. 2

[SMK08] SCHNABEL R., MÖSER S., KLEIN R.: Fast vector quantization for efficient rendering of compressed point-clouds. *Computers & Graphics 32*, 2 (2008), 246 – 259. 2, 6, 7

[SPS12] SMITH J., PETROVA G., SCHAEFER S.: Progressive encoding and compression of surfaces generated from point cloud data. *Computers & Graphics 36*, 5 (2012), 341–348. 2

[TG98] TOUMA C., GOTSMAN C.: Triangle Mesh Compression. *Graphics Interface 98 Conference Proceedings* (1998), 26–34. 2

[YBS06] YOSHIZAWA S., BELYAEV A., SEIDEL H.-P.: Smoothing by example: Mesh denoising by averaging with similarity-based weights. In *SMI '06* (Washington, DC, USA, 2006), IEEE, p. 9. 3

[ZSW*10] ZHENG Q., SHARF A., WAN G., LI Y., MITRA N. J., COHEN-OR D., CHEN B.: Non-local scan consolidation for 3d urban scenes. *ACM Trans. Graph., Proc. SIGGRAPH 10*, 4 (2010), 94:1–94:9. 3