Progressive low bit rate coding of simple 3D objects with Matching Pursuit

Ivana Tosic, Pascal Frossard and Pierre Vandergheynst
Signal Processing Institute, EPFL
Lausanne 1015, Switzerland
{ivana.tosic, pascal.frossard, pierre.vandergheynst}@epfl.ch
Fax: +41 21 693 7600, Phone: +41 21 693 2601

Abstract

This paper presents a low rate progressive 3D mesh compression scheme for simple genus-zero 3D objects. The proposed scheme is based on signal representation using redundant expansions. 3D model representation is constructed using a Matching Pursuit algorithm, with an over-complete dictionary of atoms defined on a sphere. A specific dictionary construction is proposed, that is adapted to the characteristics of 3D models. The dictionary is built on both low-frequency atoms, and anisotropic refinement to capture singularities of the signal, living on a 2-D sphere. The novel coding method is shown to favorably compare to state-of-the-art compression schemes at low bit rate, while providing a flexible and progressive representation. It therefore represents an very interesting alternative for simple 3D model representations, especially in view-dependent or scalable applications.

I. Introduction

The widespread use of 3D data in many industries has created an essential need for efficient compression of 3D data representation. The basic tool for 3D data representation are polygonal meshes. Due to the large amount of data necessary for representing polygonal meshes, there is still a need for high compression ratios for network transmission of 3D graphic models. On the other side, the large variety of receivers with different performance and network resources, imposes a need for multi-resolution and progressive models, as well as for low-complexity decoders.

The first progressive transmission scheme for multi-resolution triangular manifold meshes was introduced by Hoppe [1]. Triangular manifold mesh is represented by a base mesh followed by a sequence of successive vertex split refinements. Taubin [2] introduced the Progressive Forest Split (PFS) scheme which highly reduces the number of levels of detail, that are usually not needed. A forest split operation is represented by a group of consecutive edge split operations. Together with Topological Surgery (TS) [3], PFS is a base for 3D mesh coding in the MPEG-4 standard.

One of the problems in 3D data representation is that data is generally not defined on a regular grid, in contrast to the cases of speech, image and video signals. In the new coding scheme described in this paper, we move away from restrictive representation techniques by resampling 3D data on a regular spherical grid, thus reducing the dimension of the input data into a 2D data set. This representation can therefore give us the advantage of

This work has been partly supported by the Swiss National Science Foundation.

using various 2D signal transform coding techniques. We represent a simple genus-zero 3D model[1] as a function on a sphere. By a simple genus-zero surface we assume a surface which has only one intersection point with each radial line from the center of the point cloud.

One of the first work based on representing functions defined on a surface, particularly on a sphere, was done by Schröder and Sweldens [4]. They introduced a lifting scheme to construct biorthogonal spherical wavelets with customized properties. Shape compression using spherical wavelets has become recently an active area of research. The progressive coding scheme introduced by Khodakovsky et al. [5] uses wavelet transform, zerotree coding and subdivision based reconstruction to improve the compression ratio. Hoppe and Praun [6] described a shape compression technique using spherical geometry images, which represents the surface remeshed into a regular 2D grid. They show a comparison between compression of geometry images using ordinary image wavelets and spherical wavelets.

Instead of using the wavelet transform for compression of a function defined on a sphere, we try to exploit some good properties of non-orthogonal transforms. Non-orthogonal decompositions over a anisotropically refined redundant dictionary has been shown to lead to improved approximation rate for the representation of multidimensional signals. They provide good compression efficiency, especially at low bit rates where most of the signal energy is captured with only few elements. In this paper, we propose to extend the use of *Matching Pursuit* [7] to the representation of simple 3D objects. Compared to classical shape compression methods, it offers better compression performance at low rate, while providing an interesting scalability property.

This paper is organized as follows. Section II describes the 3D object encoder based on Matching Pursuit. It starts with a short overview of the Matching Pursuit algorithm, and then more particularly focuses on a detailed description of the dictionary construction, that is adapted to 3D object properties. Section III presents experimental results, and comparisons with state-of-the-art algorithms. Finally, Section IV concludes the paper.

II. 3D OBJECTS MATCHING PURSUIT ENCODER

A. Matching Pursuit

Matching Pursuit (MP) is an algorithm that iteratively decomposes a signal into a linear combination of waveforms. These waveforms, also called atoms, form a dictionary, that is generally constructed to efficiently represent the signal characteristics. Interestingly, very few restrictions are imposed on the dictionary construction, besides the fact that it should at least span the space of the signal to represent. In other words, the dictionary is defined as a set of vectors $D = (g_\gamma)_{\gamma\in\Gamma}$ in Hilbert space H. In order to be able to represent each vector in H as a linear combination of unit norm vectors in D, the dictionary must satisfy the completeness property (i.e., $Span(D) = H$).

Let $f \in H$ denote a function which we want to approximate with a linear expansion over D. With MP, an N-terms linear expansion is obtained by successive approximations of $R^n f$ through orthogonal projections on dictionary vectors:

$$f = \sum_{n=0}^{N} \langle R^n f, g_{\gamma_n} \rangle \, g_{\gamma_n} + R^{N+1} f, \quad (1)$$

[1]A mesh has a genus g, iff one can cut the mesh along 2g closed loops without disconnecting the mesh

where $R^n f$ is the residue after $n-1$ iterations of the algorithm ($R^0 f = f$). To minimize the approximation error, one must choose, at each iteration, the atom that best approximates $R^n f$, with the maximal projection $|\langle R^n f, g_{\gamma_n}\rangle|$ over the dictionary:

$$|\langle R^n f, g_{\gamma_n}\rangle| = sup_{\gamma\in\Gamma}|\langle R^n f, g_{\gamma}\rangle|. \tag{2}$$

When $N \to \infty$, under assumption that the dictionary is complete, it can be shown that:

$$f = \sum_{n=0}^{\infty} \langle R^n f, g_{\gamma_n}\rangle g_{\gamma_n}. \tag{3}$$

Interestingly, it has been shown that the residue decays exponentially in a finite dimensional space, assuming complete dictionary [7]. Under the same assumption, the signal can be exactly recovered after a possibly very large number of iterations, i.e., $R^{N+1} f \to 0$ when $N \to \infty$. The decay rate depends on the correlation between the residue and the dictionary elements. The construction of an efficient dictionary, adapted to the very structure of the signal f, becomes therefore crucial in coding applications.

B. Dictionary construction

1) Preliminaries: The construction of an over-complete dictionary adapted to the coding of simple 3D models represented on a sphere, involves the three following steps:

- to define the generating function on the sphere
- to move atoms on the sphere, and to rotate them around their axis
- to implement the anisotropic scaling of atoms

Combining motions and scaling, an overcomplete set of atoms g_γ is finally derived from the chosen generating functions g, where $\gamma = (\theta, \varphi, \psi, a_1, a_2) \in \Gamma$, respectively represent the position of the atom on the sphere, its orientation, and the scaling parameters.

Since the signal to be approximated is a function $f(\theta, \varphi) \in L^2(S^2)$ defined on a 2-sphere, the atoms have also to live in the same space, i.e., $g(\theta, \varphi) \in L^2(S^2)$. In order to map the atoms on the sphere, we consider the stereographic projection [8] at the North pole $\Phi : S^2 \to \mathbb{C}$, where $\mathbb{C}$ represents the complex plane. This is shown on the Figure 1. The stereographic projection is given by:

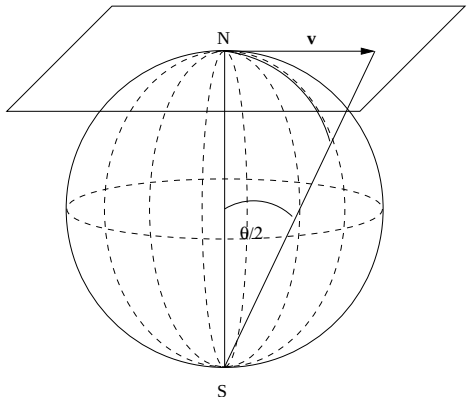


Fig. 1. Stereographic Projection

$$\Phi(\omega) = \vec{v} = \rho e^{j\varphi} = 2tan\left(\tfrac{\theta}{2}\right) e^{j\varphi}, \quad \omega \equiv (\theta, \varphi), \quad 0 \le \theta \le \pi, -\pi \le \varphi \le \pi, \tag{4}$$

For a function defined on a tangent plane $\mathbb{C}$, (ρ, φ) are actually its polar coordinates, and $\vec{v} = (\rho\, cos\varphi, \rho\, sin\varphi)$.

2) Generating functions: In order to efficiently capture the particular characteristics of 3D models (e.g., singularities over pieces of great circles), we propose to use the following generating function, defined in the space $L^2(\mathbb{R}^2)$, is:

$$g_{rect}(\vec{v}) = -\frac{1}{K}\left(4x^2 - 2\right) exp\left(-\frac{(x^2+y^2)}{4}\right), \tag{5}$$

where $\vec{v} = (x, y)$ is a vector in $\mathbb{R}^2$, and K is a normalization factor. Note that this function is very similar to the one used for images, which is Gaussian in one direction and its second derivative in the other direction:

$$g_{image}(\vec{v}) = \frac{1}{K_0}\left(4x^2 - 2\right) exp\left(-\left(x^2+y^2\right)\right). \tag{6}$$

Atoms derived from this function have shown good approximation properties for images, capable of catching contour-like discontinuities [9], [10]. The function defined in Eq. (5) differs however from Eq. (6), since it generates longer atoms (slower decay) in the direction of Gaussian, but with the same sharp decay in the direction of its derivative. This leads to improved approximation of singularities. The generating function can now be expressed in polar coordinates:

$$g_{rect}(\rho, \varphi) = -\frac{1}{K}\left(4\rho^2 cos^2\varphi - 2\right) exp\left(-\frac{\rho^2}{4}\right). \tag{7}$$

We can do the inverse stereographic projection $\Phi^{-1} : C \rightarrow S^2$ and obtain the function on the sphere:

$$g(\theta, \varphi) = -\frac{1}{K_1}\left(16tan^2\frac{\theta}{2}cos^2\varphi - 2\right) exp\left(-tan^2\frac{\theta}{2}\right), \tag{8}$$

where K_1 is a normalization constant. Eq. (8) defines an atom that is centered exactly on the North pole.

The generating function g described above has shown however not to be able to efficiently represent low frequency features of the signal. We thus extend the MP dictionary with a set of atoms derived from a two-dimensional Gaussian generating function in $L^2(S^2)$:

$$g_{LF}(\theta, \varphi) = exp\left(-tan^2\frac{\theta}{2}\right). \tag{9}$$

3) Motion: The generating functions defined above are now translated on the sphere to generate the MP dictionary. Motion of atoms is realized by rotations $\varrho \in SO(3)$[2], applying the unitary operator Π_ϱ on the matrix of Cartesian (x, y, z) coordinates of the atom, denoted as P:

$$P_r = \Pi_\varrho P = k(\psi)U(\theta)k(\varphi)P, \quad \varrho \in SO(3), \tag{10}$$

[2] $SO(3)$ is the rotation group in $\mathbb{R}^3$

where $\{P\}_{3\times N}$ is the matrix of (x, y, z) coordinates of the non-transformed atom, and $\{P_r\}_{3\times N}$ is the matrix of (x, y, z) coordinates of the transformed atom. Matrix $k(\psi)$ performs the rotation of the atom around its axis, and is given by:

$$k(\psi) = \begin{pmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The matrices $U(\theta)$ and $k(\varphi)$ introduce motion of the atom over the sphere, with angles θ and φ. The matrix k is already defined above, and $U(\theta)$ is given by :

$$U(\theta) = \begin{pmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{pmatrix}$$

The order of this matrices is strictly defined, with first rotation of the atom on the North pole, and afterwards the motion of the atom to a particular point (θ,φ) on the sphere. The generating function defined on the sphere in Eq. (15) is applied to the (θ, φ) grid obtained after this transformation.

4) Anisotropic scaling: Anisotropic refinement is now applied to the generating function g, with different behavior in two orthogonal directions. Scaling is first performed on a tangent plane at the North pole, and then projected by inverse stereographic projection on the sphere S^2, as explained before. If $\vec{v} = (x, y)$ is a vector in a tangential plane, anisotropic scaling operator is introduced as:

$$D(a_1, a_2)g_{rect}(\vec{v}) = C_0 g_{rect}(a_1 x, a_2 y), \tag{11}$$

where the constant C_0 is the normalizing factor. Coordinates of the scaled vector $\vec{v_s}$ become :

$$x_s = a_1 x = a_1 \rho cos\varphi, \quad y_s = a_2 y = a_2 \rho sin\varphi, \tag{12}$$

which translates, in polar coordinates, into :

$$\rho_s = \sqrt{x_s^2 + y_s^2} = \rho\sqrt{a_1^2 cos^2\varphi + a_2^2 sin^2\varphi}, \quad \varphi_s = arctan\frac{y_s}{x_s} = arctan\frac{a_2 sin\varphi}{a_1 cos\varphi}. \tag{13}$$

Substituting polar coordinates into (5), we get:

$$g_{rect}(\rho, \varphi) = -\frac{1}{K}\left(4a_1^2\rho^2 cos^2\varphi - 2\right) exp\left(-\frac{\rho^2\left(a_1^2 cos^2\varphi + a_2^2 sin^2\varphi\right)}{4}\right). \tag{14}$$

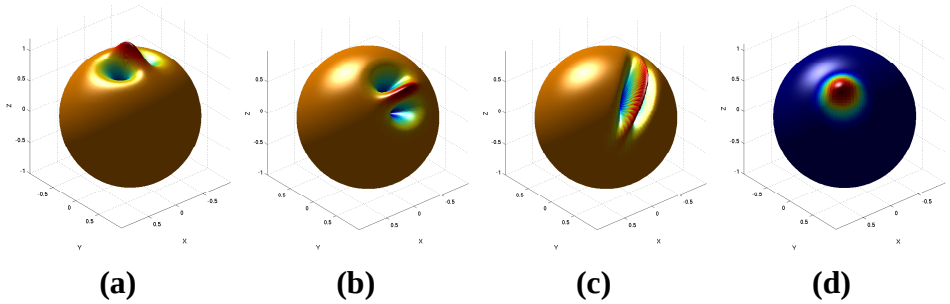
Finally, by inverse stereographic projection $\Phi^{-1} : \mathbb{C} \to S^2$, the anisotropically refined atom on the sphere can be written as :

$$g(\theta, \varphi) = -\frac{1}{K_2}\left(16a_1^2 tan^2\frac{\theta}{2}cos^2\varphi - 2\right) exp\left(-tan^2\frac{\theta}{2}\left(a_1^2 cos^2\varphi + a_2^2 sin^2\varphi\right)\right) \tag{15}$$

where K_2 is a normalization factor.

For LF atoms we have :

$$g_{LF}(\theta, \varphi) = exp\left(-tan^2\frac{\theta}{2}\left(a_1^2 cos^2\varphi + a_2^2 sin^2\varphi\right)\right). \tag{16}$$



(a) **(b)** **(c)** **(d)**

Fig. 2. Anisotropic atoms: a) on the North pole ($\theta = 0, \varphi = 0$), $\psi = 0, a_1 = 8, a_2 = 8$; b) $\theta = -\frac{\pi}{4}, \varphi = \frac{\pi}{2}, \psi = 0, a_1 = 8, a_2 = 8$; c) $\theta = -\frac{\pi}{4}, \varphi = \frac{\pi}{2}, \psi = \frac{\pi}{4}, a_1 = 16, a_2 = 4$; d) Low frequency atom: $\theta = -\frac{\pi}{4}, \varphi = \frac{\pi}{4}, \psi = \frac{\pi}{4}, a_1 = 8, a_2 = 8$

5) The MP dictionary: In order to obtain a dictionary of finite size we obviously need to discretize the atom parameters $\theta, \varphi, \psi, a_1, a_2$. We will use the spherical grid that is obtained by uniformly dividing θ axis in the interval $[0, \pi]$ and φ axis in $[-\pi, \pi)$, thus covering the complete sphere. The rotation parameter ψ is uniformly discretized in the interval $[0, \pi)$, with step of $\frac{\pi}{18}$. Scaling parameters are uniformly distributed on the logarithmic scale from one to half of the size of the input signal (function on the sphere), with one third of octave of granularity. The largest atom thus covers half of the sphere. In general, increasing the granularity or the size of dictionary allows for better approximation, but also to higher search complexity. For low pass atoms, the maximal scale is chosen to be 1/16 of the signal size. Motions and rotations are discretized the same way as for anisotropic atoms. Sample atoms are represented in Figure 2.

C. Encoding scheme

For compression purposes, Matching Pursuit coefficients and corresponding parameter values need further encoding, in order to form a compact representation. Matching Pursuit coefficient values are first quantized, unlike other parameters which are already discretized and can be sent directly to the encoder. In this paper we used linear piecewise approximation of the coefficient values curve (after reordering them in decreasing order) as an upper bound for quantization, inspired from what has been proposed in [11]. After quantization, the coefficient and atom index values are finally encoded with an arithmetic coder [12].

III. Experimental results

A. Implementation

Before analyzing the performance of the proposed encoding scheme, we briefly address here three essential implementation issues, that are the generation of the input data and the mesh connectivity, and the particular MP search implementation.

The input data is a set of radius values representing a function $f : S^2 \rightarrow \mathbb{R}$ defined on 2-sphere S^2 as $R_{int} = f(\theta, \varphi)$. We use uniform θ and uniform φ grid. The signal value for each point on the grid is obtained by performing nearest neighbor interpolation within four points from the model, with maximal projection on the desired direction (θ, φ).

Then, since we have chosen remeshing compression approaches, we have to define different mesh connectivity than the original one. We can use the advantage of a-priori knowledge of θ and φ coordinates of each vertex. Having in mind that we are dealing with simple models, we can construct a *semi-regular* connectivity structure. A mesh with semi-regular connectivity has almost all vertices of valence[3] 6, except few isolated extra-ordinary vertices with valence$\neq$6. In order to obtain a semi-regular mesh, we can divide the spherical grid into rings limited with two successive values of θ, and then triangulate each ring to produce a triangular strip. All vertices are of valence 6, except 2 poles, thus our mesh is semi-regular.

As an example, the original Venus model is shown on the Figure 3 (f), while the interpolated model, with resolution $N_\theta \times N_\varphi = 128 \times 128$ is displayed on the Figure 3 (e). The interpolation error is expressed with relative L^2 error in units of 10^{-4} and in PSNR[dB]. Relative L^2 error is actually a ratio of RMS - Root Mean Square Error (that measures the squared symmetric distance between two surfaces averaged over the first surface) relative to a bounding box diagonal. The PSNR is simply given by $20\ log\left(\frac{1}{L^2}\right)$. These two values are obtained using the MESH software (http://mesh.epfl.ch/).

In the experiments presented below, the Matching Pursuit has been implemented with a full search over the dictionary. The position of the best matching atom can be found by computing the convolution on the sphere[4] with the atom placed at the North pole ($\theta = 0, \varphi = 0$). The convolution coefficient with the maximum absolute value correspond to the position parameters of the best matching atom. Rotation and scaling are determined by finding the maximal inner product (i.e. maximal projection) of the model residual function with the atom. The inner-product of two functions f and g defined on the sphere is given by :

$$\langle f, g \rangle = \int_\theta \int_\varphi f(\theta, \varphi) g(\theta, \varphi) sin\theta d\theta d\varphi. \tag{17}$$

The algorithm starts with a search of the LF dictionary and then switches to the anisotropic dictionary. The switch point is detected when the residue energy goes into saturation, or more precisely when:

$$\frac{|C_n|}{\|R_n\|_2} \rightarrow const, \tag{18}$$

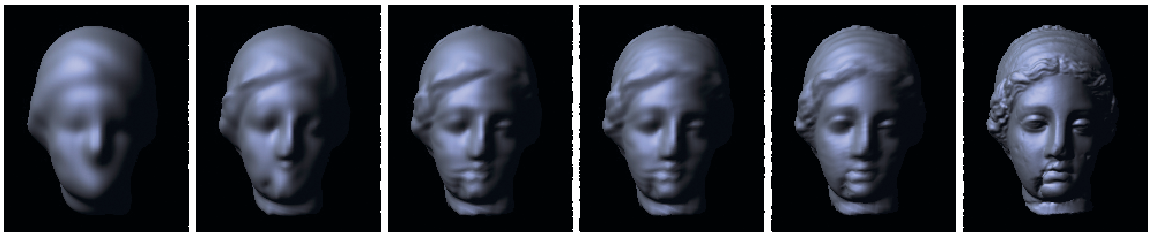
where C_n denotes a projection after n-1 iterations.

B. Numerical results

The representation of the Venus model using the proposed Matching Pursuit encoder is shown on Figure 3, with different numbers of coefficients. It can be seen that Matching Pursuit rapidly captures the most important features of the 3D model, and progressively refines the representation with finer details. The type of coding artifacts is quite different than the degradations observed in mesh-based coders, and visually less annoying at low rate. It can be seen also that the gain in representation accuracy is less important when the number of iterations increases, which renders Matching Pursuit mostly efficient for low bit rate representations.

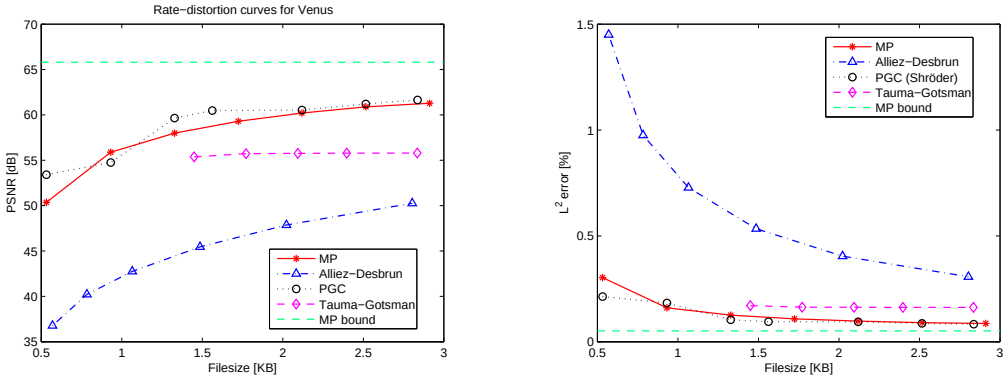
[3]A valence of a vertex is a number of edges incident to that vertex

[4]http://fyma.fyma.ucl.ac.be/projects/yawtb/



(a) **(b)** **(c)** **(d)** **(e)** **(f)**

Fig. 3. MP results: Recovered model of Venus: a) 100 coefficients; b) 200 coefficients; c) 400 coefficients; d) 600 coefficients; e) interpolated model PSNR=65.7983dB, $L^2 = 5.1296$ f) original model.



(a) **(b)**

Fig. 4. Rate-distortion performance for the Venus model (a) PSNR, (b) L^2 error.

Figure 4 compares the rate-distortion performance of the proposed MP algorithms (by means of PSNR (a) and L^2 error (b)) with the following state-of-the-art schemes:

- TG: Tauma-Gotsman non-progressive coding [13]
- Alliez-Desbrun progressive coding [14]
- PGC: Progressive coding scheme by Khodakovsky et al. [5]

Comparison with some other state-of-the-art-schemes still lacks due to different format use and software unavailability.

The input model for TG and Alliez-Desbrun methods is the model that has been used to obtain the interpolated version for Matching Pursuit encoder[5]. For TG, we decimated the Venus model to 1400 faces (using Qslim software[6]), in order to have a comparison in the same rate region. Different rates for the TG algorithm are obtained by changing the number of bits per vertex for encoding. As PGC uses its own mesh format, the input model is downloaded from the PGC website[7]. Note that the rate is actually given with filesize not with bits per vertex, since the proposed MP coding scheme uses one single mesh (128x128 vertices in the current implementation) throughout progressive compression. MP significantly outperforms the classic 3D compression methods TG and Alliez-Desbrun, but it still performs worse than the PGC wavelet coder. However, when

[5]http://www.cyberware.com/samples/index.html
[6]http://graphics.cs.uiuc.edu/ garland/software/qslim.html
[7]http://www.multires.caltech.edu/software/pgc/

interpreting these results, one must keep in mind that the starting model for the MP coder is actually the interpolated model and that the approximation converges during iterations towards this approximated model, and not the original one. Therefore, its performance is upper bounded by the PSNR introduced by interpolation (PSNR=65.7983dB in this case). This bound is also displayed on the Figure (4). The power of MP coder should thus be interpreted as its capability to approach this bound. Altogether, the proposed encoder is shown to offer an interesting alternatives to classical approaches, since it provides interesting compression performance at low rate, and still brings an inherently progressive representation.

IV. Conclusions and Future work

A novel approach to the coding of simple 3D objects has been proposed, that uses a completely different representation than common encoders. An algorithm based on Matching Pursuit has been used to obtain good signal approximations at low bit rate. A dictionary has been specifically designed as an overcomplete family of atoms living on the sphere. It has been shown that such a dictionary is able to represent simple genus zero models, with good visual quality. Our method has been compared to state-of-the-art encoders (two progressive compression schemes and one non-progressive), where has been shown to offer very good compression performance, that are however limited by the quality of the input model. Matching Pursuit has the advantage of providing on intrinsically progressive scheme, that is additionally very flexible. Such advantages become very important in view-dependent streaming of model information, or in scalable applications.

Finally, the proposed algorithm still leaves numerous possibilities for improvement. For example, the interpolation step, which uses the very basic nearest neighbor method, has an important influence on the compression results. We are also currently working on improving the dictionary, and on the extension of the proposed scheme to the coding of more complex 3D models.

References

[1] H. Hoppe, “Progressive meshes,” *Siggraph ’96 Conference Proceedings*, pp. 99–108, August 1996.

[2] G. Taubin, A. Guéziec and W. Horn, “Progressive forest split comprression,” *Siggraph ’98 Conference Proceedings*, pp. 123–132, July 1998.

[3] G. Taubin and J. Rossignac, “Geometry compression through topological surgery,” *ACM Transactions on Graphics*, vol. 17, no. 2, pp. 84–115, April 1998.

[4] P. Schröder and W. Sweldens, “Spherical wavelets: Efficiently representing functions on the sphere,” *Siggraph ’95 Conference Proceedings*, pp. 161–172, August 1995.

[5] A. Khodakovsky, P. Schröder and W. Sweldens, “Progressive geometry compression,” *Siggraph ’00 Conference Proceedings*, July 2000.

[6] H. Hoppe and E. Praun, “Shape compression using spherical geometry images,” *Symposium on Multiresolution in Geometric Modeling. Cambridge*, September 2003.

[7] S.G. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, December 1993.

[8] J. Antoine and P. Vandergheynst, “Wavelets on the 2-sphere : a group theoretical approach,” *Applied and Computational Harmonic Analysis*, vol. 7, pp. 1–30, November 1999.

[9] P. Vandergheynst and P. Frossard, “Efficient image representation by anisotropic refinement in matching pursuit,” *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Salt Lake City*, vol. 3, pp. 1757–1760, May 2001.

[10] R. Figueras i Ventura, P. Vandergheynst and P. Frossard, “Low rate and scalable image coding with redundant representations,” *No TR-ITS-2003.002*, June 2003.

[11] P. Frossard, P. Vandergheynst, R.M. Figueras i Ventura and M. Kunt, “A posteriori quantization of progressive matching pursuit streams,” *IEEE Transactions on Signal Processing*, vol. 52, no. 2, pp. 525–535, February 2004.
[12] I.H. Witten, R.M. Neal and J.G. Cleary, “Artihmetic coding for data compression,” *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, June 1987.
[13] C. Touma and C. Gotsman, “Triangle mesh compression,” *Graphics Interface Conference Proceedings, Vancouver*, June 1998.
[14] P. Alliez and M. Desbrun, “Progressive compression for lossless transmission of triangle meshes,” *Siggraph ’01 Conference Proceedings*, August 2001.