مجلة جامعة سبها للعلوم البحتة والتطبيقية
Sebha University Journal of Pure & Applied Sciences

Journal homepage: www.sebhau.edu.ly/journal/index.php/jopas

Efficient 3D Surface Patch Compression and Reconstruction using Parametric Descriptions and Transform Techniques

Abdusslam Osman Beitalmal

Department of Mathematics, Sebha University, Sebha, Libya

Keywords:
Compression techniques
Computational Geometry
Object Modeling
Discrete Cosine Transform
Discrete Fourier transform
Discrete Wavelet Transform
3D data structures
Partial Differential Equations

ABSTRACT
This paper proposes and demonstrates novel methods for compressing and reconstructing 3D surface patches typically obtained from scanners that rely on stereo vision, structured light, or time-of-flight techniques. The methods involve applying a polygon reduction to the mesh to get a set of vertices lying in structured planes of a sparse, regular grid. The data in each plane is then described parametrically, and a comparative analysis is conducted using the Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), and Discrete Wavelet Transform (DWT). Quality factors are employed to further process the transform coefficients, resulting in substantial reductions in the amount of data saved to disk. The paper also defines file formats with the necessary parameters for a complete reconstruction of the sparse mesh. Finally, elliptic Partial Differential Equations (PDE) are used to represent the reconstructed data, and the Laplace equation is iteratively solved between adjacent planes to recover the vertex density of the original mesh. Experimental results demonstrate the effectiveness of the proposed methods, achieving compression rates of over 98% compared to the OBJ file format and over 91% compared to a list of vertices in ASCII format.

كفاءة ضغط التصحيح السطحي D3 وإعادة الإعمار باستخدام الأوصاف البارامترية وتقنيات التحويل

عبدالسلام عصمان بيت المال

قسم الرياضيات، كلية العلوم، جامعة سبها ، سبها ، ليبيا

الكلمات المفتاحية:
تقنيات الضغط
الهندسة الحاسوبية
نمذجة الكائنات
تحويل جيب التمام المنفصل
تحويل فوريه المنفصل
تحويل الموجات المنفصلة
هياكل البيانات D3
المعادلات التفاضلية الجزئية

الملخص
في هذه الورقة البحثية تم اقتراح طرق جديدة لضغط البيانات ثلاثية الأبعاد وإعادة بنائها. هده الطرق مناسبة للصور والاسطح ثلاثية الأبعاد ، وحيث ان بيانات هدة الصور والاسطح تأخذ بواسطة ماسحات ضوئية وكمرات مخصصة تطلق ضوء منتظم تخزن على شكل بيانات شبكية. أولاً، يتم تطبيق تقليل مضلع على الشبكة مما يؤدي إلى مجموعة من الرؤوس ملقاة في مستويات منظمة من شبكة منتظمة متفرقة. يتم تعريف المتجهات التي تصف البيانات في كل مستوى بشكل بارامتري ويتم تقديم تحليل مقارن عبر تحويل فوريه المنفصل (DFT) وتحويل جيب التمام المنفصل (DCT) وتحويل موجة منفصلة (DWT). تتم معالجة معاملات التحويل بشكل أكبر وفقًا لعامل الجودة الذي يقلل بشكل كبير من كمية البيانات المحفوظة في القرص. يتم تعريف تنسيقات الملفات مع المعلمات اللازمة لإعادة البناء الكامل للشبكة المتناثرة. أخيرًا، من أجل استعادة كثافة رأس الشبكة الأصلية، يتم تمثيل البيانات المعاد بناؤها بالمعادلات التفاضلية الجزئية (PDE) ويتم حلها بشكل متكرر بين المستويات المجاورة فيما يتعلق بمعادلة لابلاس. توضح التجارب فعالية الطرق التي تسمح بمعدلات ضغط تزيد عن 98٪ مقارنة بتنسيق ملف OBJ وأكثر من 91٪ مقارنة بقائمة الرؤوس في تنسيق ASCII.

Introduction

As 3D data becomes increasingly prevalent in various applications, such as computer graphics, virtual reality, and medical imaging, the need for efficient storage and transmission of 3D data has become a pressing issue. 3D data is typically large in size and requires substantial computing resources to process and analyse. Therefore, the development of efficient compression and reconstruction techniques

*Corresponding author:
E-mail addresses: abd.beitalmal@sebhau.edu.ly

Article History : Received 02 October 2022 - Received in revised form 12 June 2023 - Accepted 30 June 2023

for 3D data has become an active research area in recent years. In a latest 2012 article [1], Siew and Rahman present an evaluation of techniques for 3D data compression during the last decade. It is mentioned that the pervasiveness of XML in distributed applications makes it a natural candidate to represent the semantics of the 3D data structure. However, XML has a tendency to generate huge files that, mixed with geometry statistics might render any such scheme impractical. The essential issue lies in locating approaches to compress the geometry and connectivity of the data before coding the semantics into XML or ASCII. In a 2005 assessment article [2], Peng et al. evaluation technologies for three-D data compression with a selected consciousness on triangular meshes. A beneficial class of algorithms is offered either as connectivity-driven or geometry-driven. In an earlier 2003 survey of 3D compression methods [3], Alliez and Gotsman focus on compression techniques for single-rate and modern mesh coding based totally on geometry and connectivity. Compression techniques are therefore, focused on representing the geometry and connectivity of the vertices inside the triangulated mesh. Geometrical approaches aim to reduce the size of the mesh by simplifying its geometry and approaches include geometry coding [4], Generalized Triangle Mesh [5], triangulated model techniques [6], and quantization techniques [7], [2] where rates of over 80:1 have been achieved. Examples of coding connectivity include the topological surgery algorithm [8], and the Edgebreaker algorithm [9], [10]. Products additionally exist within the market that says a 95% lossless file reduction which includes 3D compression technology [11] for normal geometric shapes.

The 3D compression methods proposed in this paper are derived from our studies on fast 3D reconstruction using structured light methods [12], [13], [14], [15], [16], [17], [18]. One of our main application areas concerns 3D face recognition and the exchange of data over the network is a major requirement. In [16] we proposed a technique for 3D data compression and reconstruction based on polynomial interpolation. The method is suitable for data that can be defined as a single-valued function, which is typical of data acquired by standard 3D scanners based on stereo vision, structured light or time-of-flight techniques. In this study, we explored the possibility of using arbitrary face meshes, not necessarily obtained from our scanner, by sampling the mesh to conform to a rectangular pattern. This allowed us to evaluate compression rates and test the usage of polynomial interpolations of various degrees. Our findings indicated that the polynomial compression method is suitable for smooth data, but when dealing with complex data, such as facial data, the required high degree of polynomials made the method unstable. Therefore, we suggested that alternative approaches should be investigated for such cases.

The objective of this paper is to find ways to compress large data files that contain 3D models, while still maintaining the quality of face recognition techniques. The data model used in this study is a connected mesh of vertices with triangular faces, which is a common format for 3D computer-generated models. This format is supported by several standard file formats such as Collada, Java 3D OBJ, VRML, and Wavefront OBJ [19],[20],[21].The proposed methods in the study rely on a structured re-meshing of the surface, which involves polygon reduction to simplify the mesh. This results in a specific shape of vertices that leads to a simpler and more reliable triangulation system, compared to an arbitrarily related mesh. Arbitrary meshes may require complex triangulation algorithms such as Delaunay [22], which can increase the computational complexity and potentially reduce the accuracy of the reconstructed model.

The polygon reduction approach employed in the study involves using sets of vertices that lie within a plane, and subjecting them to DFT (Discrete Fourier Transform), DCT (Discrete Cosine Transform), and DWT (Discrete Wavelet Transform) transforms. The resulting coefficients are then compressed using a quality factor, which is defined in Sections 2.2, 2.3, and 2.4 of the paper. The compressed coefficients are saved to ASCII files with specific structures that contain the necessary information to allow for reconstruction using inverse transforms of DFT, DCT, and DWT, padded with zeros where required. However, the issue of recovering the original mesh density before polygon reduction is addressed by defining the set of structured vertices as boundary conditions to elliptic PDEs (Partial Differential Equations), as described in Section 2.5 of the paper. The PDEs are then iteratively solved through the Laplace equation, which allows for the reconstruction of the original mesh density while preserving the quality of the compressed data. This approach aims to strike a balance between compression and accuracy, resulting in a more efficient and reliable method for reconstructing large data files containing 3D models. Section 3 of the paper presents the experimental results obtained from using 16 high-density facial models. The original and reconstructed models are visualized under various quality parameters, allowing for a qualitative assessment of the compression and reconstruction methods employed. The paper also estimates error surfaces, along with their corresponding root mean square errors (RMSE), to provide a more objective assessment of quality. Furthermore, the paper provides information on common compression quotes for fine parameters ranging from 5 to 100 over the 16 models. Finally, the conclusions drawn from the study are presented in Section 4 of the paper.

2. Method

2.1. Simplifying Polygons by Structured Vertex Reduction

The method presented in this paper is applicable to surface patches that can be represented as a single-valued function in one dimension, which is expressed as an explicit function of two independent variables. An example of such data, which was captured using the structured light scanning technique employed in this study, is shown in Figures 1 and 2, with mesh details illustrated in Figure 3. In a surface patch, the vertical position of a point is determined by its "-value". This means that the height of a function at any given point on the surface can be expressed as a function of its two-dimensional coordinates. The advantage of using a single-valued function is that it has a simple parametric form of;

$$P(u,v) = \big(u, v, f(u,v)\big) \tag{1}$$

which makes it easier to manipulate and analyse. With normal vector $\mathbf{n}(u,v) = (-\delta f/\delta u, -\delta f/\delta v, 1)$. Both u and v are the dependent variables for the function and u-contours lie in planes of constant x, and v-contours lie in planes of constant y.

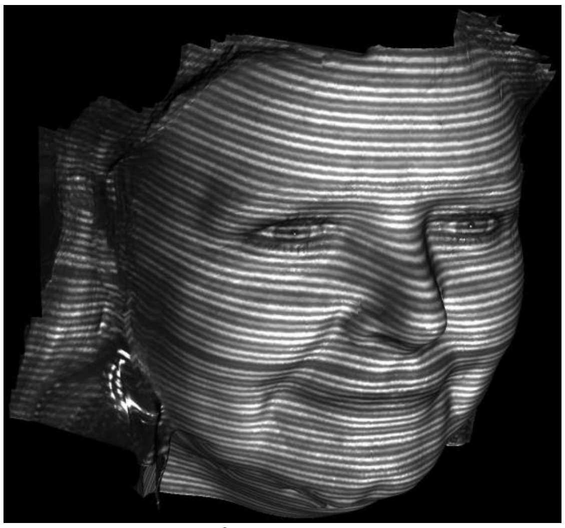


Fig. 1: An instance of a 3D model with texture mapping

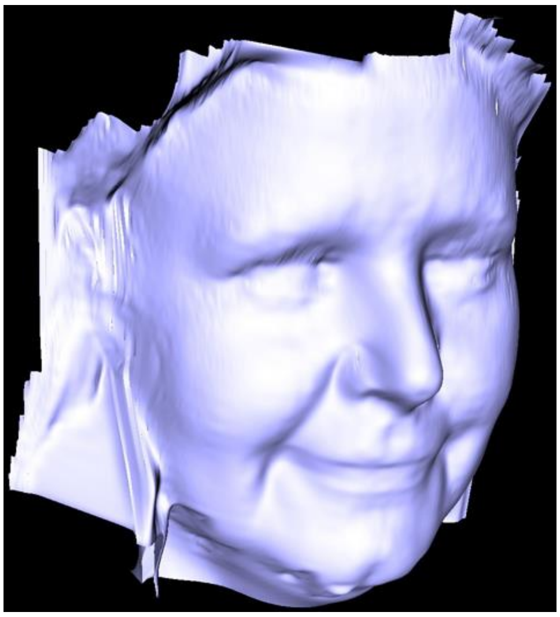


Fig. 2: Shaded 3D model generated using structured light method.

When such patch is visualized in 3D using quads for instance, each edge of the polygon is a trace of the surface cut by a plane with $x = k_1$ and $y = k_2$ for some values of k_1 and k_2. In a polygon reduction by explicit structured vertices, a randomly oriented surface patch is described in relation to a global coordinate system. The minimum bounding box of the patch in 3D is estimated by geometric algorithms (e.g. [23]). The patch must be rotated until its bounding box edges are aligned with the $x-, y-$, and z-axes of the global coordinate system. For the face models used to demonstrate the concepts in this paper, the smallest dimension of the bounding box is aligned with the z-axis as this can assure that the face models are orientated successfully.

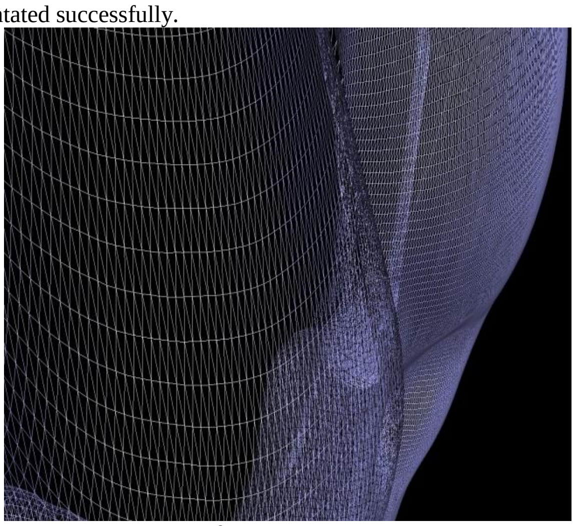


Fig. 3: An example of a textured 3D model.

The correct orientation of the bounding box depends on the characteristics of the data but the general principle is that the z-axis is normal to the image sensor in standard 3D scanners

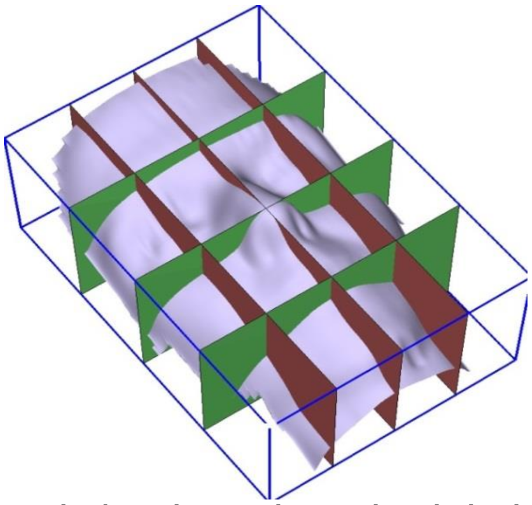


Fig. 4: Bounding box and structured cutting planes displayed.

A number of structured cutting planes are defined within the boundaries of the bounding box; let us call these 'horizontal' and 'vertical' cutting planes as illustrated in Figure 4 where, for clarity, only a few planes are shown. These planes are as a consequence, described as parallel to one of the x or y-axes with normal vectors $(1,0,0)$ and $(0,1,0)$ respectively. The intersection of any two planes defines a line, and the point where such line intersects the mesh is defined as a structured vertex. Consequently, the number or the density of structured vertices can be controlled by the number of planes in either direction. An issue right here is that we cannot assure that the intersection of two planes on the mesh will rest on a vertex. More likely, it will intersect somewhere on a polygon's face somewhere between vertices. A great approximation is to discover the three vertices at the mesh that are the nearest to the intersection line. Such vertices define a plane and it then becomes straightforward to determine the intersection point. Assume that the line has a starting point S and direction $\boldsymbol{c}$. The intersection line is given through

$$L(t) = S + ct \tag{2}$$

The answer most effective involves finding the intersection point with the generic plane [24]. The generic plane is the xy-plane or $z = 0$. The line $S + \boldsymbol{c}t$ intersects the generic plane when

$S_z + \boldsymbol{c}_z t_i = 0$ where t_i is t 'intersection':

$$t_i = \frac{S_z}{C_z} \tag{3}$$

From Equation 2, the line hits the plane at point P_i:

$$P_i = S - C\left(\frac{S_z}{C_z}\right) \tag{4}$$

The set of all points belonging to a particular plane is a subset of structured vertices [25]. Depending on the characteristics of the surface patch, the set of vertices lying in either horizontal or vertical planes can be selected. If the selected points lie in a plane with normal vector $(0,1,0)$, the distance between structured vertices in that plane is the distance between planes with normal $(1,0,0)$ and vice-versa. Calling these distances D_1 and D_2, the x and y coordinates of any structured vertex can be recovered for all planes k:

$$x_r = \{ rD_1, | r = 1,2, \dots, k_1 \} \tag{3}$$
$$y_c = \{ \mathrm{c}D_2, | \mathrm{c} = 1,2, \dots, k_2 \} \tag{4}$$
$$\mathrm{z} = \{ z_i, \ | \mathrm{i} = 1,2, \dots, k_1 k_2 \} \tag{5}$$

where (r, c) are the indices of the planes. This is significant as, in a stroke, $2/3$ of the 3D data can safely be discarded in a sense that it is not necessary to save the actual values (x, y) of each vertex; instead, only D_1, D_2, k_1 and k_2 are kept for each plane. The number of structure planes is controlled by experimentation bearing in mind that the resulting structured vertices should still be representative of the original mesh. The z-values can be expressed by Equation 1 as a single-valued function and estimated using Equation 4 for each combination of (x_r, y_c). If we choose to represent these as the set of structured vertices belonging to planes with normal (0,1,0) this is reduced to a $2D$ case in which on the horizontal axis we have exactly k_2 points with the constant step of $D_2$and on the vertical axis we have their corresponding z-values. The above operations mean that starting from a surface patch with a complex polygonal arrangement, we achieve an established mesh in which the quantity of polygons is decreased and triangulation will become a trivial task, as it's miles most effective vital to connect vertices from adjacent planes. In other phrases, the mesh now incorporates an underlying express structure for triangulation.

2.2 The DFT Technique

Once experimental data are represented by the z-values of each structured plane as specified in Section 2.1, the vertices lying in each plane can be considered as a one-dimensional signal and treated as a Fourier series [26]. The usefulness of the Fourier analysis is that we can break upany arbitrary periodic function into a set of simple terms that can be solved individually and then recombined to reconstruct the original signal to a high degree of accuracy. The Fourier transform is essentially a universal problem-solving technique. Its significance is based totally on the fundamental property that one can examine a particular relationship in the time domain from an entirely different viewpoint in the frequency domain. Simultaneous visualization of a function and its Fourier transformis often the key to successful problem solving [27]. The continuous Fourier transform is defined as

$$\mathrm{f(v)} = \mathcal{F}_t[\mathrm{f(t)}](\mathrm{v}) \quad (6)$$

$$= \int_{\infty}^{-\infty} \mathrm{f(t)e^{-2\pi ivt}d\,t} \quad (7)$$

If the integral exists for every value of the parameter f then Equation 9 defines $f(v)$, the Fourier transform of $f(t_k)$. Now consider the generalization to the case of a discrete function, $f(t) \to f(t_k)$ by letting $f_k \equiv f(t_k)$, where $t_k \equiv k\Delta$, with $k = 0,1,\dots.N-1$. Writing this out gives the discrete Fourier transform $F_n = \mathcal{F}_k[\{f_k\}_{k=0}^{N-1}](n)$as

$$\mathrm{F_n} = \sum_{\mathrm{k=0}}^{\mathrm{N-1}} \mathrm{f_k e^{-2\pi ink/N}}. \quad (8)$$

The inverse transform $f_k = \mathcal{F}_n^{-1}[\{F_n\}_{n=0}^{N-1}](k)$ is then

$$\mathrm{f_k} = \frac{1}{\mathrm{N}} \sum_{\mathrm{n=0}}^{\mathrm{N-1}} \mathrm{F_n e^{2\pi ikn/N}}. \quad (9)$$

Discrete Fourier transforms are extremely beneficial due to the fact they reveal periodicities in input data as well as the relative strengths of any periodic components. There are a few subtleties in the interpretation of discrete Fourier transforms, however. In general, the discrete Fourier transform of a real sequence of numbers will be a sequence of complex numbers of the same length. Especially, if f_k are real, then F_{N-n} and F_n are associated by using

$$\mathrm{F_{N-n}} = \overline{\mathrm{F}}_\mathrm{n} \quad (10)$$

for $n = 0,1,\dots\dots,N-1$, where $\overline{F}$ denotes the complex conjugate. This means that the component F_0 is always real for real data. As a result of the above relation, a periodic function will contain transformed peaks in not one, but two places. This happens because the periods of the input data become split into `positive' and `negative' frequency complex components.

Table 1: Text file format for 3D compression using DFT

Line number	ASCII data info						
1	$k1$	$k2$	$D1$	$D2$	Q		
2	$v1$	$v2$	$a0$	$a6$	L	an	bn
...	...						
N	$v1$	$v2$	$a0$	$a6$	L	an	bn

The set of Fourier coefficients are estimated for each plane and saved in plain ASCII format onto a file with N lines of text where the first line contains header information followed by:

$(N-1)$ lines of data as defined in Table 1 where:

1	line 1 contains header info,
2 – N	lines 2 to N contain data,
k_1, k_2	are the scale factors or distance between two consecutive horizontal and twoconsecutive vertical planes in mm,
D_1, D_2	are the dimensions of the data in number of rows and columns,
v_1, v_2	are the first and last valid vertices for each row of data,
Q	the quality of the compression in percentage from 1 to 100,
a_0, a_6	are the scalar Fourier coefficients for each row of data,
L	the vector length of Fourier coefficients,
a_n, b_n	the vector real and imaginary Fourier coefficients for each row of data.

The parameter Q is defined as the quality of the compression and is expressed in percentage. It refers to the percentage of coefficients to keep and it is applied slightly differently for DFT, DCTand DWT. A compression of DFT coefficients only applies to the set of imaginary coefficients. Commonly, most imaginary coefficients for high frequency signals are zero or close to zero, and the most significant ones are the few first ones. So, the options we faced were either to force any value below a certain threshold to zero or simply discard a percentage from the end of the vector which is the chosen option for its simplicity of operation. In this manner, we guarantee to maintain the most relevant ones even for low values of quality. A quality $Q = 100$ means do not discard any coefficient while $Q = 30$ means discarding 70% of them returned to the front.

2.3 The DCT approach

The DCT transform and its variants have been used in a variety of contexts most notably in image and video compression [28],[29],[30]. DCT is a close relative to the DFT transform as it defines a sequence of data in terms of the sum of the cosine functions at different frequencies. It can beseen as the 'real' version of the DFT in which the basis vectors contain only co-sinusoidal patterns. Even as a DFT consists of real and imaginary components the DCT operates on data with even symmetry which means that a DCT is equivalent to a DFT with approximately twice the length of the data. In practice, it would be equivalent to a DFT by means of doubling the sampling data and transferring the added data to theend of the signal. There are numerous variations of the DCT and here we use the unitary discrete cosine transform as defined in Matlab [31]. The DCT transform of a one-dimensional signal z representing the depths on each structured plane is expressed as:

$$\mathrm{y(k)} = \mathrm{w(k)} \sum_{\mathrm{n=1}}^{\mathrm{N}} \mathrm{z(n)} \cos\left(\frac{\pi(2\mathrm{n}-1)(\mathrm{k}-1)}{2\mathrm{N}}\right) \quad (11)$$

for $k = 1,2,\dots N$ where N is the length of the signal and

$$w(k) = \begin{cases} 1/\sqrt{N} & for\ k = 1 \\ \sqrt{2/N} & for\ 2 \le k \le N \end{cases} \quad (12)$$

The length of the coefficients y is the same as the original signal z. The advantage here is that it is only necessary for a few coefficients to reconstruct the signal. Most signals can be described with over 99% accuracy by using only a handful of coefficients.

The inverse cosine transform recovers the original signal from the set of coefficients $y(k)$:

$$z(n) = \sum_{k=1}^{N} w(k)y(k) \cos\left(\frac{\pi(2n-1)(k-1)}{2N}\right) \quad (13)$$

for $n = 1,2, \ldots N$ where N is the length of the coefficients in Equation 13.

Table 2: Text file format for 3D compression using DCT

Line number	ASCII data info				
1	$k1$	$k2$	$D1$	$D2$	Q
2	$v1$	$v2$	$B1$		
.\.\.	.\.\.				
N	$v1$	$v2$	BN		

The parameters depicted in Table 2 are saved in simple ASCII format where B_i is the set of DCT coefficients estimated by Equation 13 and shortened by using parameter Q. In different words, the number of coefficients i to be saved is defined by the floor of $i = kQ/100$. The alternative parameters in Table 2 are as defined in Section 2.2.

2.4 The DWT Method

The DWT transform [32-35] is a time-scale representation of a signal acquired through digital filtering techniques wherein the signal to be analysed is passed through filters with different cut-off frequencies at different scales. The approach is realised by iteration and the resolution of the signal which determines the amount of information in the signal can be controlled by subsampling(up and down) operations. For a given signal two sets of coefficients are computed referred to as the approximation coefficients A and detail coefficients$D$. The A coefficients are obtained by convolving the signal with a low-pass filter and the D are obtained by convolving with a high-pass filter. As the signal is decomposed by the half band filters this results in signals spanning only half the frequency band. This doubling of frequency resolution reduces uncertainty in frequency by half. Following Nyquist's rule, the signal can now be down sampled by discarding half the samples with no loss of information. The result is that while the half band low pass filtering removes half the frequencies thus halving the resolution, a decimation by 2 halves the time resolution and thus doubles the scale. Convolving the signal $z(n)$ with a half band digital low pass filter with impulse response $h(n)$ can be described in discrete time as:

$$\mathrm{x(n) * h(n)} = \sum_{\mathrm{k}=-\infty}^{\infty} \mathrm{x(k)\,.\,h(n-k)} \qquad (14)$$

Applying the Nyquist rule by subsampling the signal by 2 can be represented as

$$\mathrm{y(n)} = \sum_{\mathrm{k}=-\infty}^{\infty} \mathrm{h(k)\,.\,x(2n-k)} \qquad (15)$$

Equation 17 is used for both high pass and low pass filtering operations. This one level decomposition can be expressed as:

$$\mathrm{y_{high}} = \sum_{\mathrm{n}} \mathrm{x(n)\,.\,g(2k-n)} \qquad (16)$$

$$\mathrm{y_{low}} = \sum_{\mathrm{n}} \mathrm{x(n)\,.\,h(2k-n)} \qquad (17)$$

where y_{high} and y_{low} are the outputs of high and low pass filters after decimation by 2. In order to reconstruct the original signal, the procedure is straightforward given that halfband filters form orthonormal bases. At every level of decomposition, the signal is up sampled by two, filtered through high pass and low pass synthesis filters $g'(n)$ and $h'(n)$ and then summed over. Thus, for every level of decomposition the recovered signal is represented as

$$\mathrm{x(n)} = \sum_{\mathrm{k}=-\infty}^{\infty} \left(\mathrm{y_{high}(k).\,g(-n+2k)}\right)\left(\mathrm{y_{low}(k).\,h(-n+2k)}\right) \qquad (18)$$

It is crucial to note that if the filters are not ideal half band, then perfect reconstruction is not possible. Even as it is clear that perfect filters aren't feasible to comprehend, some filters under some conditions can offer the best reconstruction. The maximum used and correct ones are Daubechies filters additionally known as Daubechies wavelets [34] and these are the ones used in the experimental results described in the next section. Moreover, with the purpose to save the DWT coefficients to a text file for subsequent reconstruction it is necessary to decide on the number of levels of decomposition. We set for 3 levels as no significant gain is achieved with further levels for the examined facial data.

Table 3: Text file format for 3D compression using DWT

Line number	ASCII data info							
1	$k1$	$k2$	$D1$	$D2$	Q			
2	$v1$	$v2$	$L1$	$L2$	$L3$	$L4$	$L5$	C
...	...							
N	$v1$	$v2$	$L1$	$L2$	$L3$	$L4$	$L5$	C

The parameters $k_1, k_2, v_1, v_2, D_1, D_2$ and Q in Table 3 are defined in Section 2.2 and the introduced parameters are related to the approximation and detail coefficients for a 3-level decomposition as follows

$L_1$is the length of approximation coefficients level 3 $(A3)$,
L_2 is the length of detail coefficients level 3
$(D3)$, L_3 is the length of detail coefficients level
2 $(D2)$, $L_4$is the length of detail coefficients level
1 $(D1)$, L_5 is the length of the original signal,
C are the coefficients to save

Note that the number n of coefficients to save depends on the quality factor and it is defined as the floor of (length(C) $Q/100$)as before. For any value of quality $Q < 100$ implies discarding some of the detail coefficients $D1, D2$, and $D3$ in that order. Upon reconstruction, these are padded with zeros. The algorithm for discarding coefficients is as follows:

1. Estimate $(d = (length\ C) - n)$ as the number of detail coefficients to discard.
2. If $d > L_4 + L_3$ discard all from $D1$ and $D2$ plus some or all from $D3$,
3. If $d > L_4$ discard all from $D1$ plus some or all from $D2$,
4. If $d \le L_4$ discard some or all from $D1$.

Observe that the approximation coefficients level 3 are not subject to compression. In that case, the quality of the reconstructed data is largely deteriorated. The method proposed here uses a 3-level decomposition;if further levels are required then the saved data and the algorithm above needs to be adjusted accordingly.

2.5 The PDE approach

Consider the high-density mesh depicted in Figure 3. The polygon reduction by structured planes as described in Section 2.1 will substantially reduce the number of vertices (by a factor of 4 inthe experiments described in Section 3). Upon compression by DFT, DCT and DWT, we wouldnot be able to recover the original mesh density due to the very nature of polygon reduction. We propose that using structured planes as boundary conditions to elliptic Partial Differential Equations can successfully be used to recover the original mesh density. PDEs have been used to solve a variety of problems in science and engineering [36], [37], which include 3D data parameterization and reconstruction and image processing [38]. The problem we are trying to solve has no dependency on time, as we wish to generate a solution between two functions geometrically defined as single-valued. To this end, we can represent the problem by an elliptic PDE without a derivative in by the Laplace equation that's the simplest elliptic PDE with a properly-advanced idea:

$$\Delta u = u_{xx} + u_{yy} = 0 \qquad (19)$$
$$\text{if}\, x \in R, y > 0$$

Where and are spatial independent variables in Cartesian coordinates and u_{xx} and u_{yy} are the second derivatives. Note that with no derivatives in elliptic PDEs require no initial conditions, i.e., they are entirely boundary value PDEs. In order to solve this class of PDE, Schiesser [37] describes the Method of Lines (MOL) which is based on a well-established numerical procedure for solving PDEs in rectangular and cylindrical coordinates. The method generates surfaces from the solutions to elliptic PDEs, where boundary conditions are used to control the surface shape. The MOL is regarded as a special finite difference method (FDM) however is more powerful with respect to accuracy and computational time than the ordinary FDM [25].

By using the method of separation of variables the solutions to Equation 21 are subject to the set boundary conditions as in Figure 4. Subsequently, the solution will be a harmonic function in the upper half-plane. On the x-axis we require Dirichlet boundary conditions of the form:

$$R = \{(x,y) \mid 0 \le x \le a, 0 \le y \le b\} \tag{20}$$

Where two cases are identified:

$$u(0,y) = u(x,b) = u(x,0) = 0 \tag{21}$$

For $u(x,y)$ satisfying the homogeneous boundary conditions, and

$$u(a,y) = f(y) \tag{22}$$

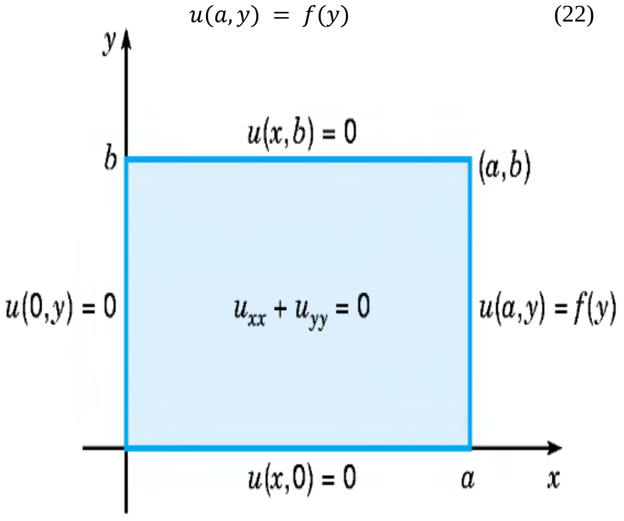


Fig. 5: Dirichlet boundary problem for a rectangular domain.

for $u(x,y)$ satisfying the non-homogeneous boundary condition, where f is a given function. Note that the Laplace equation itself and the homogeneous boundary conditions satisfy the super- position principle; this means that if u_1 and u_2 satisfy these conditions, so does $c_1u_1 + c_2u_2$, for any choice of constants c_1 andc_2. To satisfy the homogeneous boundary conditions of Equation 23, the solution to Equation 21 is given as:

$$\mathrm{Y}(\mathrm{y}) = \mathrm{A}\cosh(\mathrm{n}\pi\mathrm{y}/\mathrm{b}) + \mathrm{B}\sinh(\mathrm{n}\pi\mathrm{y}/\mathrm{b}) \tag{23}$$

where A and B are constants of integration, and the boundary condition $Y(0) = 0$ implies that $A = 0$. Thus, we find that the nontrivial product solutions to Laplace's equation together with the homogeneous boundary conditions are constant multiples of

$$\mathrm{u_n}(\mathrm{x},\mathrm{y}) = \sin(\mathrm{n}\pi\mathrm{x}/\mathrm{b})\sinh(\mathrm{n}\pi\mathrm{y}/\mathrm{b}). \tag{24}$$

These functions satisfy the differential Equation 21 and all the homogeneous boundary conditions for each value of n. To satisfy the remaining non-homogeneous boundary condition at $x = a$ we assume that we can represent the solution $u(x,y)$ in the form

$$\begin{aligned}\mathrm{u}(\mathrm{x},\mathrm{y}) &= \sum_{\mathrm{n}=1}^{\infty} \mathrm{c_n}\,\mathrm{u_n}(\mathrm{x},\mathrm{y}) \\ &= \sum_{\mathrm{n}=1}^{\infty} \mathrm{c_n}\sin(\mathrm{n}\pi\mathrm{x}/\mathrm{b})\sinh(\mathrm{n}\pi\mathrm{y}/\mathrm{b}).\end{aligned} \tag{25}$$

The coefficients c_n are determined by the boundary condition

$$\mathrm{u}(\mathrm{a},\mathrm{y}) = \sum_{\mathrm{n}=1}^{\infty} \mathrm{c_n}\sin(\mathrm{n}\pi\mathrm{a}/\mathrm{b})\sinh(\mathrm{n}\pi\mathrm{y}/\mathrm{b}) = \mathrm{f}(\mathrm{y}). \tag{26}$$

Therefore, the quantities $c_n \sinh(n\pi a/b)$ must be the coefficients in the Fourier sine series of period $2b$ for f and are given by

$$\mathrm{c_n} sinh\frac{\mathrm{n}\pi\mathrm{a}}{\mathrm{b}} = \frac{2}{\mathrm{b}}\int_0^{\mathrm{b}} \mathrm{f}(\mathrm{y})\sin\frac{\mathrm{n}\pi\mathrm{y}}{\mathrm{b}}\mathrm{dy} \tag{27}$$

Thus, the solution to Equation 21 satisfying the boundary conditions of Equation 23 is given by Equation 27 with the coefficients c_n computed from Equation 29. From Equations 27 and 29 we see that the solution contains the factor $\sinh(n\pi x/b)\,/\sinh(n\pi a/b)$. To estimate this quantity for large values of n we can use the approximation $\sinh\theta \cong e^{\theta}/2$, and thereby obtain

$$\frac{\sinh(\mathrm{n}\pi\mathrm{x}/\mathrm{b})}{\sinh(\mathrm{n}\pi\mathrm{a}/\mathrm{b})} \cong \frac{\exp(\mathrm{n}\pi\mathrm{x}/\mathrm{b})/2}{\exp(\mathrm{n}\pi\mathrm{a}/\mathrm{b})/2} = \exp[-\mathrm{n}\pi(\mathrm{a}-\mathrm{x})/\mathrm{b}] \tag{28}$$

Thus, this factor has the character of a negative exponential; consequently, the series of Equation 27 converges quite rapidly unless $a - x$ is very small. Results are described in the next Section.

3. Experimental Procedures and Findings

Our experimental setup consisted of three sets of tests. In the first set, we applied DFT, DCT, and DWT transforms to points situated in a single plane. The second set involved performing the same transforms on multiple planes, while the third set focused on using DFT, DCT, and DWT on multiple planes in conjunction with PDE reconstruction. In our experiments, we employed a set of 16 3D models to evaluate the performance of different compression techniques. Figure 6 shows one of these models, a human head with high complexity and detail, which we used for illustration purposes throughout our analysis. This particular model was chosen for its challenging characteristics, making it a useful test case for evaluating the effectiveness of different compression algorithms for 3D data. Detailed statistics and error analysis for each model are presented in the following sections.

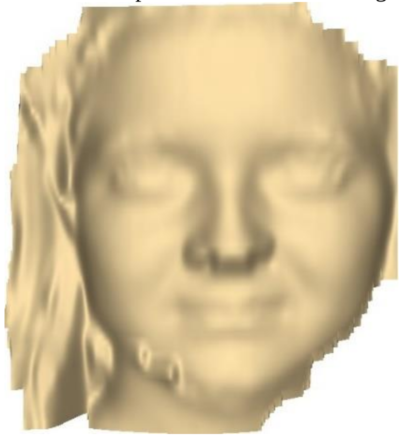


Fig. 6: The 3D model used for illustration of compression techniques.

3.1 Compression of 3D Data by Applying DFT, DCT, and DWT to Vertices Lying in a Single Plane.

Our initial experiment aimed to evaluate the effectiveness of our method and computer programs when applied to a set of vertices situated on a single plane. To select a representative complex curve from the dataset, we chose a plane that intersected the 3D model shown in Figure 6 and included the tip of the nose. We then applied each compression technique in sequence and recorded the results, which are displayed in Figures 7 and 8, that shows the three-level DWT decomposition and reconstruction of the planar 3D data, demonstrating the steps involved in applying DWT to the vertices and showing the reconstructed data. We compared the performance of this method to other compression techniques in our analysis.

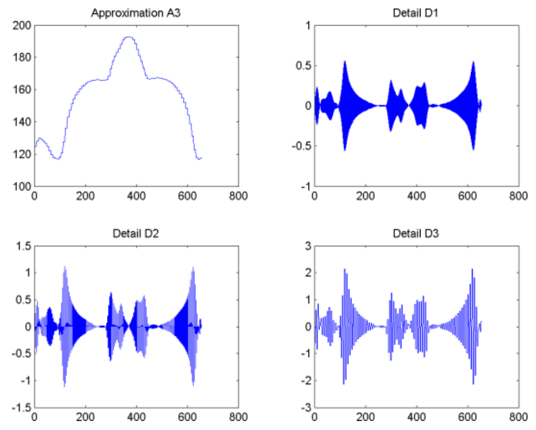
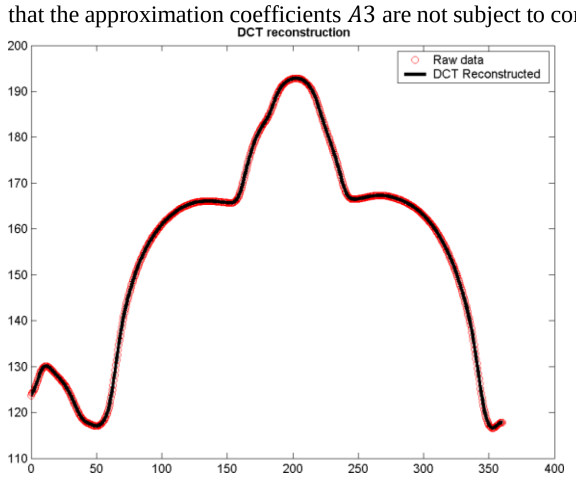


Fig. 7: DWT 3-level decomposition and reconstruction of vertices in a single plane.

During this stage of the experiment, the quality parameter was set to $Q = 100$, as no compression was applied to the data. It's worth noting that the last point of the DFT transform is spurious due to periodicity, as reconstruction is performed within the range $0.0, \dots, 2\pi$. Due to the periodicity of the DFT transform, the last point of each reconstructed plane will join up with the first point. Therefore, the last point is redundant and needs to be deleted from the reconstructed data for each plane.

To clarify how compression would be applied to each of these curves with a quality parameter $Q = 50$, we can examine the specific modifications required for each transform method. For the DFT curve, the bottom half of the imaginary components would be discarded, meaning that these coefficients would not be saved to disk. When reconstructing the data using the inverse DFT, the missing coefficients are padded with zeros. For DCT, the bottom half of the coefficients in Equation 13 would simply be discarded and then padded with zeros upon reconstruction using the inverse DCT. For the DWT method, only the detail coefficients $D1, D2,$ and $D3$ from the high pass filter are subject to compression (as shown in Equation 18). In this case, 50% of all detail coefficients would be discarded, starting with D3, then $D2$, and finally $D1$. It's important to note that before discarding any of $D2$, all coefficients from $D1$ must be discarded, and the same applies to $D3$ in relation to $D2$. Upon reconstruction with the inverse DWT, all missing coefficients are padded with zeros. It's worth noting that the approximation coefficients $A3$ are not subject to compression.



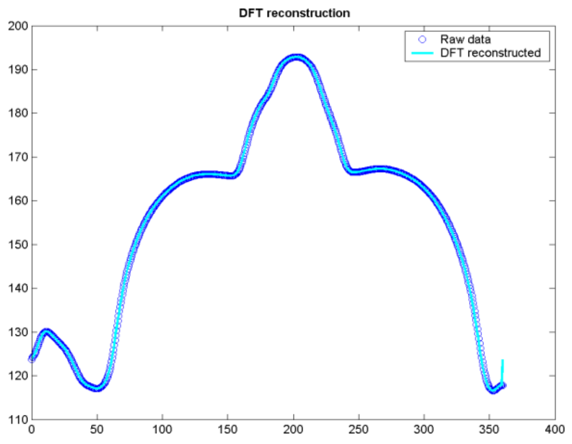


Fig. 8: Reconstruction of 3D models using DFT and DCT compression methods applied to vertices lying in a single plane.

3.2 Extending Transform Methods to Multiple Planes for Compression of 3D Data: DFT, DCT, and DWT.

In order to evaluate the ability of DFT, DCT, and DWT transforms to handle more complex datasets, we extended their application to multiple planes intersecting the 3D model shown in Figure 6 and analysed the results. Our objective was to illustrate the quality of the resulting reconstructed models and error surfaces under different compression rates. We used quality parameters of $Q = 100$ (no compression) and $Q = 50$ (50% compression) for both qualitative and quantitative assessments of the models. Two sets of experiments were conducted: the first to test the effectiveness of the techniques on a sparse mesh with simple compression and uncompressing, and the second to recover the high-density mesh using the PDE technique. It's important to note that in both cases, the compressed model is the sparse mesh, but the PDE technique allows us to recover the original mesh density. Without PDE, we can only recover the sparse mesh.

In order to implement our approach, we first performed polygon reduction as defined in Section 2.1. The density of the resulting sparse mesh is determined by the number of structured planes, which is a function of the horizontal and vertical distances between planes ($D1$ and $D2$ in Equations 5 and 6). For the face models, we found that spacing each horizontal plane 3mm apart was adequate, with 0.25mm (or 4 points per millimetre) data points along each plane to capture all the nuances of the face.

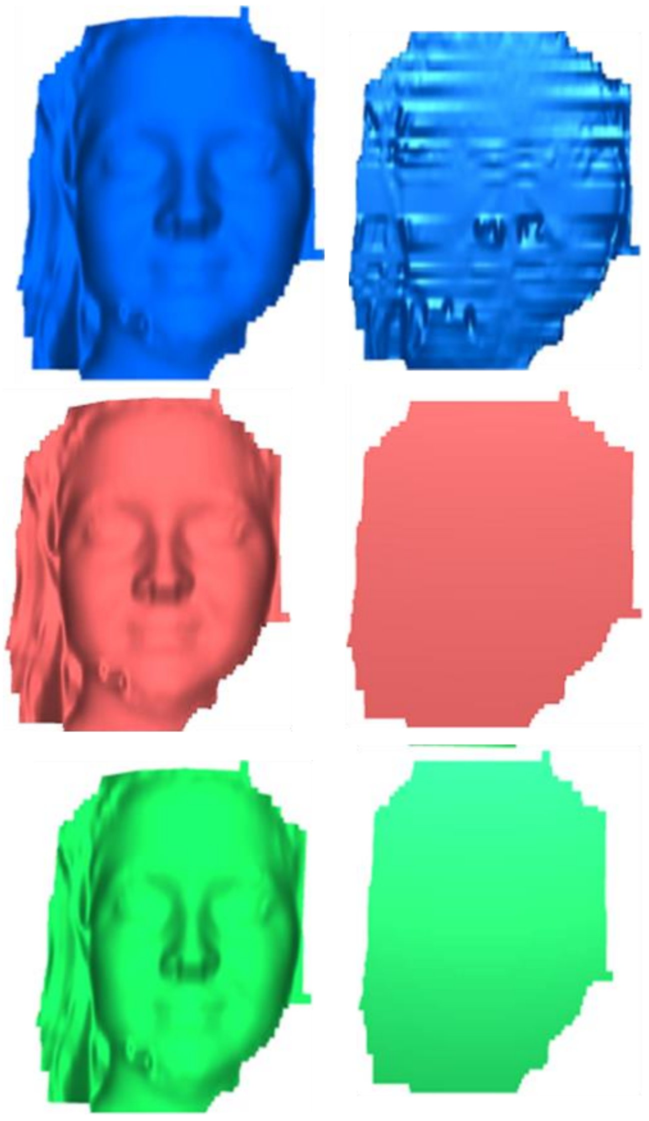


Fig. 9: Quality Q = 100. Top row: reconstructed models DFT (blue) DCT (red) and DWT (green). Bottom row: respective error surfaces

These choices reduced the number of vertices by a factor of 4 from the original dense mesh shown in Figure 3. Similar distances between planes were used for each model tested, with the actual number of planes depending on the original extension of its bounding box.
After compression and saving to disk, we evaluated the resulting uncompressed models with quality parameter $Q = 100$ and their respective error surfaces (a numerical quantification of error surfaces is presented below) in Figure 8. Overall, we found that all three compression techniques could effectively compress and uncompress 3D data, as evidenced by the general appearance of the reconstructed models. The lowest row of Figure 8 shows the error surfaces, which were estimated by subtracting the reconstructed version from the sparse mesh.

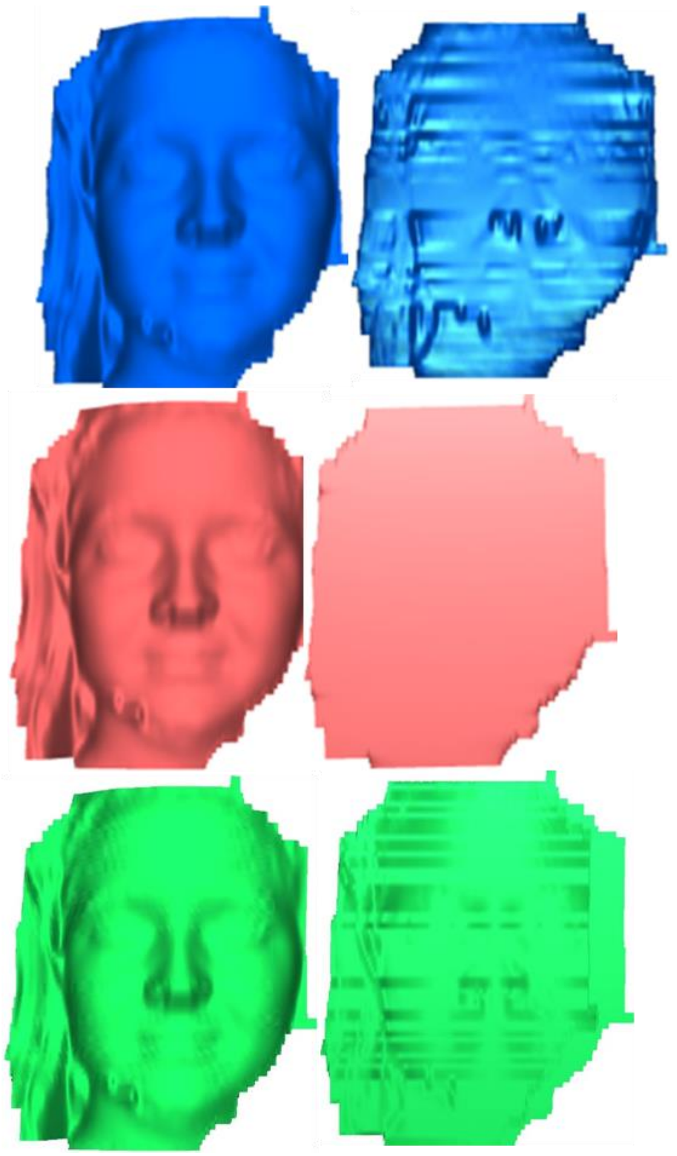


Fig. 10: Quality Q = 50. Top row: reconstructed models DFT (blue) DCT (red) and DWT (green). Bottom row: respective error surfaces.

A perfect match between the two versions would result in an error surface lying in the xy plane with all coordinates $z = 0$. The error surface of the DFT method shows relatively large errors in more complex areas of the face, such as around the nose and at the boundaries. In contrast, both DCT and DWT exhibit flat surfaces with errors at or near zero. Based on these observations, it is clear that DCT and DWT are better techniques, and they appear to be equivalent for compression and uncompressing with a quality parameter of 100. The results for compression using a quality parameter of 50 are shown in Figure 10. Once again, the DFT approach exhibits a relatively larger error surface, highlighting the superiority of both DCT and DWT techniques for compression. The DCT transform shows small errors mostly at the boundary of the model, while the DWT technique has the error distributed along the surface, and there is a noticeable high-frequency ripple pattern in the green model. The next step was to recover the original mesh density using the PDE method described in Section 2.5. For each pair of structured planes, we used them as boundary conditions for an elliptic PDE, and the distance $D1$ was used to estimate the discrete step Δx between any two planes. Since the distance between planes is 3mm, it was necessary to solve the Laplace equation using 5 steps (2 at the boundaries and 3 internal steps), resulting in a mesh density with an average quad face area of exactly $0.75 \times 0.25mm$. This is comparable to the original high-density mesh, where the average area of each quad face is $0.75 \times 0.26mm$. The outcomes of using the PDE method are illustrated in Figures 11 and 12 for quality parameters of 100 and 50, respectively. One primary observation is that solving the Laplace equation over the mesh creates a higher level of noise compared to the sparse mesh shown earlier. Although this may not be readily apparent in the 3D models, the error surfaces indicate the introduction of higher levels of noise. The DFT method is clearly the most affected, but now the DCT and DWT methods also exhibit ripples across the face caused by the bluntness of the Laplace solution. For quality parameter 50, the effects are similar, except that the error surfaces are larger than expected. The advantages of using the PDE approach are that, while the compressed file sizes are the same as for the sparse mesh, the uncompressed mesh has a density comparable to the original model. These advantages are due to the fact that high-density meshes can be compressed and recovered using relatively few structured planes. We quantified the error surfaces as a function of quality by running experiments where we set the quality parameter $Q = [5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]$. Each quality parameter was applied in turn to the 16 models, and summary statistics were computed for both with and without PDE-based reconstruction.

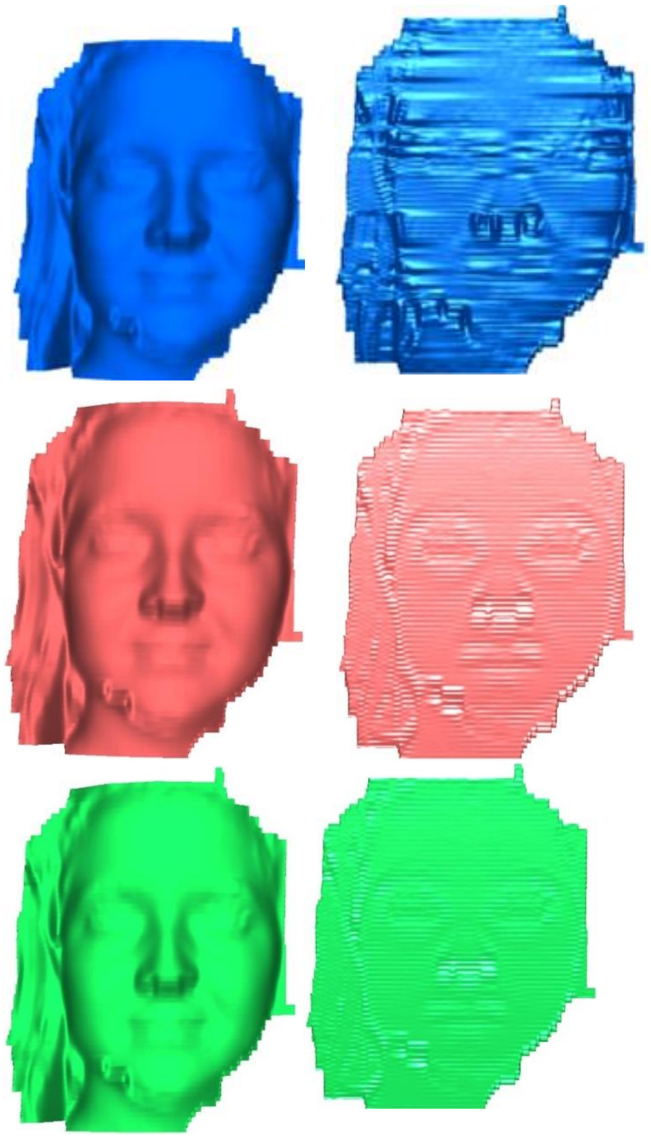


Fig. 11. Quality $Q = 100$ with PDE based reconstruction. Top row: DFT (blue) DCT (red) and DWT (green). Bottom row: respectiveerror surfaces

The root mean square error (RMSE) of each error surface was estimated, and the averages over the 16 models are shown in Figure 13. The left panel shows the error surfaces estimated by simply compressing the data from the multiple planes followed by reconstruction and direct comparison between the two datasets. This provides a direct comparison of the effectiveness of the techniques, although the original mesh density is not recovered. It is clear from the results that the DCT technique is the most appropriate, as errors are very small up to a compression rate of 80% ($Q = 20$). For any larger compression rate, surface errors grow exponentially. For large compression rates over 90%, the DWT technique is more stable and exhibits relatively smaller error surfaces. The DFT method is the worst performer, consistently showing larger errors. However, it is worth noting that the RMSE of all three techniques stays near $0.5mm$ over a long range of compression rates between $0 - 80\%$.

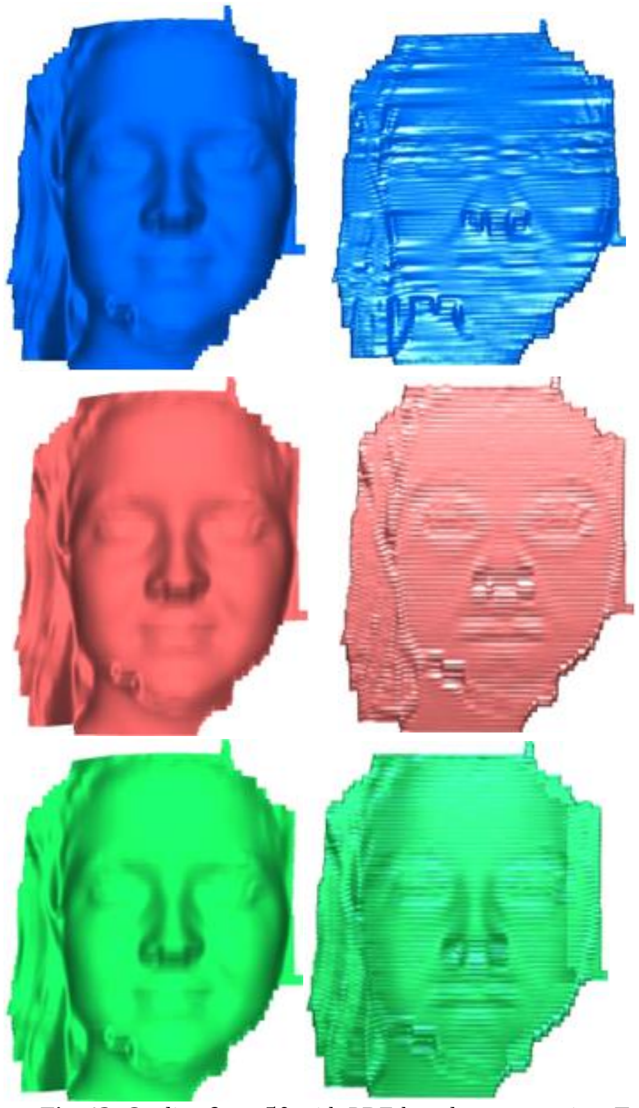


Fig. 12: Quality $Q = 50$ with PDE based reconstruction. Top row: DFT (blue) DCT (red) and DWT (green). Bottom row: respectiveerror surfaces.

The right panel of Figure 13 shows the results for reconstruction using the PDE method to recover the original mesh density, which exhibits a similar behaviour but with larger RMSEs.
This behaviour is expected, as the uncertainty of data points estimated by the PDE method is compounded by the initial errors. As a result, a comparison of the original dense mesh with the PDE mesh will show errors introduced by the Laplace approximation added to the underlying errors of the previous sparse mesh reconstruction.

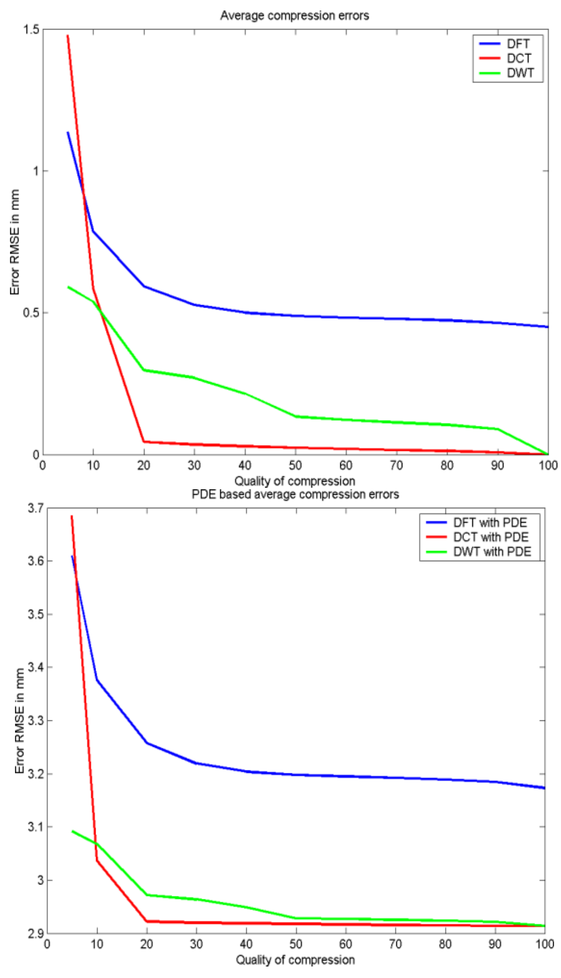


Fig. 13: Average RMSE errors of uncompressed data for quality parameter $5 \leq Q \leq 100$. Left, standard DFT, DCT and DWT. Right, PDE based reconstruction.

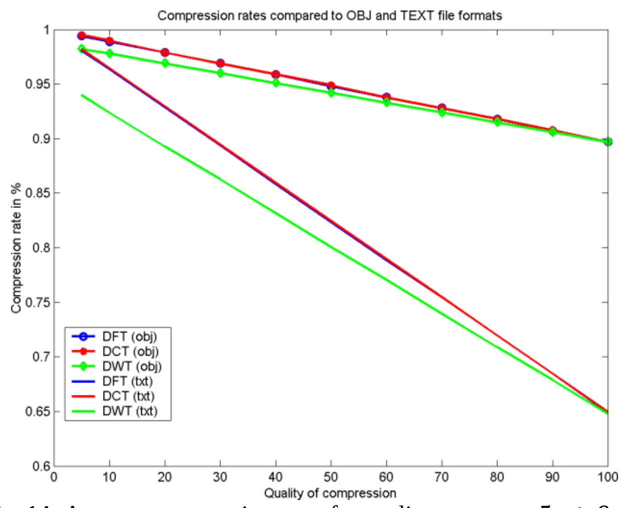
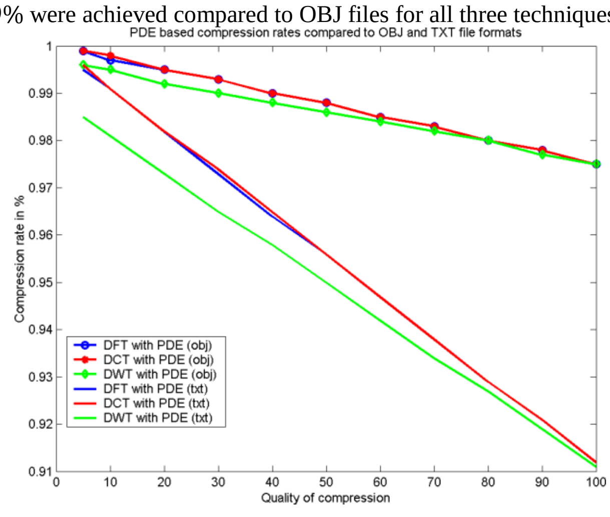


Fig. 14: Average compression rates for quality parameter $5 \leq Q \leq 100$. Left panel shows results for the sparse mesh, and the right panel shows results for the sparse mesh with PDE reconstruction.

Finally, we conducted a comparative analysis of file sizes as specified in Tables 1, 2, and 3. All compressed data were saved in plain ASCII format, and the comparison was made with the Wavefront OBJ file format and a simple triplet of (x, y, z) floating points capable of holding equivalent 3D data in ASCII format.

Again, we used the quality parameter $Q = [5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]$ and compressed each of the 16 files in turn. Figure 13 depicts the results for both the sparse mesh and PDE reconstruction. While both images show the same behavior, the difference lies in the compression rates achieved. For the sparse mesh (left image), in which the density of triangular faces is similar to the sparse and uncompressed meshes, compression rates from $90 - 99\%$ were achieved compared to OBJ files for all three techniques.



Compared to the equivalent text file, compression rates of $68 - 98\%$ for DFT and DCT, and $68 - 94\%$ for DWT, were achieved for the sparse mesh. Using the PDE method yields higher compression rates (right panel), ranging from 97.5-99% compared to OBJ files for all three techniques. Compression rates using the PDE method range from $91 - 99.5\%$ for DFT and DCT, and from $91 - 98.5\%$ for DWT, compared to the equivalent text file. The overall observed pattern is that for sparse meshes, where the density of the structured planes is similar to the original density of the mesh (i.e., no significant polygon reduction applied), error surfaces are generally very small with good compression rates. For high-density meshes, errors tend to increase as polygon reduction introduces errors, which are made worse by the Laplace approximation. The advantage is that now we can achieve higher compression rates while the perceived quality of the mesh does not deteriorate significantly.

4. Conclusion

In conclusion, the proposed method in this study involves several steps to compress and reconstruct 3D data. The first step is to reduce the number of vertices in the mesh using a polygon reduction technique that produces a sparse mesh. The remaining vertices are then analysed using DFT, DCT, or DWT to obtain their change coefficients, which are stored in ASCII format along with scale data and indices to constitute the compressed data. To recover the density of the original mesh, the Laplace equation is solved iteratively over the sparse mesh using the PDE approach. This approach increases the total vertex density of the original mesh and can achieve very large compression rates compared to the OBJ file format. The study also conducted a sensitivity analysis on the coefficients and proposed a quality parameter Q ranging from $1 - 100$ to compress the coefficients further. The results showed that DCT was the superior method, with coefficients compressible by up to 80% without significant loss in mesh quality. DWT was recommended for compressing coefficients over 90%. However, the PDE method was found to increase the root mean square error (RMSE) of the reconstructed model, suggesting that it may not be an optimal tool for improving the unique mesh density. The authors are therefore working on a moving 4th-order polynomial interpolation applied to 4 adjacent vertices on 4 consecutive planes and recursively estimating the missing vertices to improve the technique.

Future work will also encompass conducting a sensitivity analysis of face recognition algorithms operating on compressed meshes and exploring how the proposed methods can scale up to handle complete 3D models. This will further investigate the applicability and performance of the compression techniques in practical scenarios.

In summary, this study introduces novel approaches for 3D data compression that can achieve significant compression rates while preserving mesh quality and density. The proposed methods, involving polygon reduction, coefficient analysis using DFT, DCT, or DWT, and PDE-based reconstruction, provide promising avenues for efficient storage and transmission of 3D models. The study contributes valuable insights and techniques to the field of 3D data compression and lays the foundation for further advancements in this area.

5. Acknowledgment

The authors acknowledge the financial support provided by Professor Marcos Rodrigues, which helped to complete the research work. They also express gratitude towards the GMPR Geometric Modelling and Pattern Recognition Group and Sheffield Hallam University for their support and the data used in the research work.

References

[1]- Siew, C.B. and A.A. Rahman (2012). Compression Techniques for 3D SDI. *FIG Working Week 2012 Knowing to manage the territory, protect the environment, evaluate the cultural heritage*. Rome, Italy, 6-10 May 2012.

[2]- Peng, Jingliang and Kim, Chang-Su and Kuo, C-C Jay (2005). Technologies for 3D mesh compression: A survey. *Journal of visual communication and image representation. Elsevier (2005)* 16/6 688–733.

[3]- Alliez, P. and C. Gotsman (2003). Recent Advances in Compression of 3D Meshes. *Inria Sophia Antipolis Research Report* 4966, Oct 2003, 26pp.

[4]- Taubin, G., Horn, W., Lazarus, F., Rossignac, J. (1998), Geometry coding and VRML, *Proceedings of The IEEE*, 86(6)

[5]- Deering, Michael (1995). Geometry Compression, *SIGGRAPH 95 Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques.* 13-20.

[6]- Shikhare, D., S.V. Babji, and S.P. Mudur (2002). Compression techniques for distributed use of 3D data: an emerging media type on the internet, *15th International Conference on Computer Communication*, India, pp 676–696.

[7]- Qian, Shen-En and Hollinger, Allan B and Williams, Dan and Manak, Davinder (1998). 3D data compression of hyperspectral imagery using vector quantization with NDVI-based multiple codebooks, *IEEE International Geoscience and Remote Sensing Symposium IGARSS'98*, volume 5, 2680–2684.

[8]- Taubin, Gabriel and Rossignac, Jarek, Geometric compression through topological surgery, *ACM Transactions on Graphics (TOG)*. 17 (2) (1998) 84–115.

[9]- Szymczak, A., D. King and J. Rossignac (2000). An Edgebreaker-Based Efficient Compression Scheme for Regular Meshes,
12th Canadian Conference on Computational Geometry, pp 257–265.

[10]- Szymczak, Andrzej and Rossignac, Jarek and King, Davis, Piecewise regular meshes: Construction and compression, *Graphical Models* 64(3-4), *Elsevier 2002*, 183–198.

[11]- 3DCT (2012). 3D Compression Technologies (3DCT), *www.3dcompress.com/web/default.asp*, accessed Oct 2012.

[12]- Brink, Willie and Robinson, Alan and Rodrigues, Marcos A. (2008). Indexing Uncoded Stripe Patterns in Structured Light Systems by Maximum Spanning Trees, *British Machine Vision Conference BMVC 2008*, *Citeseer* volume 2018, 1–10 Sep 2008

[13]- Robinson, Alan and Alboul, Lyuba and Rodrigues, Marcos (2004). Methods for Indexing Stripes in Uncoded Structured Light Scanning Systems, *Journal of WSCG*, 12(3), 2004, pp 371–378.

[14]- Rodrigues, Marcos and Robinson, Alan (2011a). Real-time 3D Face Recognition using Line Projection and Mesh Sampling. In: *EG 3DOR 2011 - Euro graphics 2011 Workshop on 3D Object Retrieval*, Llandudno, UK, 10th April 2011. Euro graphics Association. p9–16.

[15]- Rodrigues, Marcos A and Robinson, Alan (2011b). Fast 3D recognition for forensics and counter-terrorism applications. In: AKHGAR, Babak and YATES, Simeon, (eds.) *Intelligence management: knowledge driven frameworks for combating terrorism and organized crime. Advanced information and knowledge processing*, London, Springer-Verlag, 95–109.

[16]- Rodrigues, Marcos A and Robinson, Alan and Osman, Abdulsslam (2010a). Efficient 3D Data Compression Through Parameterization of Free-Form Surface Patches, 2010 International Conference on Signal Processing and Multimedia Applications (SIGMAP), 26-28 July 2010, p130–135.*IEEE*

[17]- Rodrigues, Marcos and Robinson, Alan (2010b). Novel methods for real-time 3D facial recognition. In: SARRAFZADEH, Majid and PETRATOS, Panagiotis, (eds.) *Strategic Advantage of Computing Information Systems in Enterprise Management*, Athens, Greece, ATINER, 169–180.

[18]- Rodrigues, M.A., A. Robinson, and W. Brink (2008). Fast 3D Reconstruction and Recognition, in *New Aspects of Signal Processing*, *Computational Geometry and Artificial Vision*, 8th WSEAS ISCGAV, Rhodes, 2008, 15–21

[19]- Collada (2012). Digital Asset and FX Exchange Schema, *https://collada.org*, accessed Oct 2012.

[20]- VRML2: The Virtual Reality Modeling Language Specification, *graphcomp.com/info/specs/sgi/vrml/spec/* accessed Oct 2012.

[21]- Wavefront and Java 3D .OBJ Format, *www.eg-models.de/formats/Format Obj.html* accessed Oct 2012.

[22]- N. P. Weatherill, N.P. and O. Hassan (1994). Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints, *International Journal for Numerical Methods in Engineering*, John Wiley & Sons, Ltd., 37(**12**)1981–2150.

[23]- Lahanas, Michael and Kemmerer, Thorsten and Milickovic, Natasa and Karouzakis, Kostas and Baltas, Dimos and Zamboglou, Nikolaos (2000). Optimized bounding boxes for three-dimensional treatment planning in brachytherapy, *Medical Physics* 27(10):2333–2342 *Wiley Online Library* 2000.

[24]- Hill Jr, Francis S (2008). *Computer Graphics Using OpenGL*, 2nd edition, *Pearson Education*. Prentice-Hall Inc, 922pp.

[25]- Rodrigues, Marcos and Osman, Abdusslam and Robinson, Alan (2013). Partial differential equations for 3D data compression and reconstruction. *ADSA Advances in Dynamical Systems and Applications Research India Publications*, Vol 8(2),303–315.

[26]- Wolfram MathWorld. Generalized Fourier Series. Wofram MathWorld Calculus and Analysis, math*world.wolfram.com/GeneralizedFourierSeries.htm* accessed Oct 2012.

[27]- Weinberger, H.F. (1995). *A first course in partial differential equations with complex variables and transform methods*, Dover Publications (New York).

[28]- Belkasim, Saeid (2011). Multi-resolution Analysis Using Symmetrized Odd and Even DCT Transforms, *Data Compression Conference (DCC)*, *IEEE*447pp.

[29]- Kim, Daewon and Shin, Daekyu (2003). Energy-based adaptive DCT/IDCT for video coding, *2003 International Conference on Multimedia and Expo. ICME'03. Proceedings (Cat. No. 03TH8698)*, *IEEE*. Vol 1, 557–560.

[30]- Gharge, Saylee and Krishnan, Shoba (2007). Simulation and Implementation of Discrete Cosine Transform for MPEG-4, *International Conference on Computational Intelligence and Multimedia Applications*, *IEEE Computer Society 2007*. Vol 4, 137– 141.

[31]- Matlab Revision R2012b Documentation, *MathWorks Online Documentation*, www.mathworks.co.uk/help/ accessed Oct 2012.

[32]- Nicholl, Paul and Ahmad, Afandi and Amira, Abbes (2010). Optimal discrete wavelet transforms (DWT) features for face recognition, *2010 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 132–135.

[33]- Talukder, K.H. and K. Harada (2011). Enhancement of Discrete Wavelet Transform (DWT) for Image Transmission over Internet, *2011 Eighth International Conference on*

Information Technology: New Generations (ITNG), 1054–1055.

[34]- Vonesch, C., T. Blu and M. Unser (2007). Generalized Daubechies Wavelet Families, *IEEE Transactions on Signal Processing*, 55(9)4415–4429.

[35]- Wali, M.K. M. Murugappan, R.B. Ahmad, B.S. Zheng (2012). Development of Discrete Wavelet Transform (DWT) toolbox for signal processing applications, *2012 International Conference on Biomedical Engineering (ICoBE)*, 211–216.

[36]- Lapidus, Leon and Pinder, George F (2011). *Numerical solution of partial differential equations in science an engineering*, *John Wiley & Sons* 2011.

[37]- Schiesser, W.E. and G.W. Griffiths (2009). *A Compendium of Partial Differential Equation Models: Method of Lines Analysis with Matlab*, ISBN-13 978-0-521-51986-1, Cambridge University Press, Cambridge, UK.

[38]- Osman, Abdusslam (2014). 3D modelling using partial differential equations (PDEs), *Sheffield Hallam University (United Kingdom)*.