Geometry Prediction for High Degree Polygons

Martin Isenburg* Ioannis Ivrissimtzis† Stefan Gumhold‡ Hans-Peter Seidel§

Max-Planck-Institut für Informatik
Saarbrücken, Germany

Abstract

The parallelogram rule is a simple, yet effective scheme to predict the position of a vertex from a neighboring triangle. It was introduced by Touma and Gotsman [1998] to compress the vertex positions of triangular meshes. Later, Isenburg and Alliez [2002] showed that this rule is especially efficient for quad-dominant polygon meshes when applied "within" rather than across polygons. However, for hexagon-dominant meshes the parallelogram rule systematically performs miss-predictions.

In this paper we present a generalization of the parallelogram rule to higher degree polygons. We compute a Fourier decomposition for polygons of different degrees and assume the highest frequencies to be zero for predicting missing points around the polygon. In retrospect, this theory also validates the parallelogram rule for quadrilateral surface mesh elements, as well as the Lorenzo predictor [Ibarria et al. 2003] for hexahedral volume mesh elements.

Keywords: mesh compression, predictive geometry coding

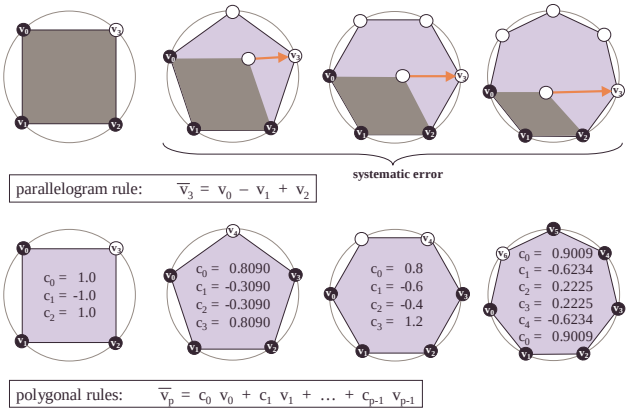


Figure 1: The parallelogram rule of Touma and Gotsman [1998] perfectly predicts the missing vertex of a regular quadrilateral but performs systematic miss-predictions for regular polygons of higher degree. Our polygonal rules predict such polygons exactly.

1 Introduction

Limited transmission bandwidth hampers working with large polygon meshes in networked environments. This has motivated researchers to develop compressed representations for such data sets. The two main components that need to be compressed are the mesh *connectivity*, that is the incidence relation among the vertices, and the mesh *geometry*, that is the specific location of each individual vertex. Since connectivity is of combinatorical and geometry of quantative nature they are compressed by different processes.

Traditionally, mesh compression schemes have focused on purely triangular meshes. Probably the most popular compressor for triangle meshes was proposed by Touma and Gotsman [1998]. The bitrates achieved by their connectivity coder have remained the state-of-the-art for single-pass compression and decompression [Isenburg and Snoeyink 2005]. Also their geometry coder, despite its simplicity, delivers competitive compression rates that continue to be a widely-used benchmark for geometry compression.

But a significant number of polygon meshes is not completely triangular. Meshes found in 3D model libraries or output from modeling packages contain often only a small percentage of triangles. The dominating element is generally the quadrilateral, but pentagons, hexagons and higher degree faces are also common. The first approaches to compress polygonal meshes triangulated the input mesh and then used a triangle mesh coder. Later it was shown that polygon meshes can be compressed more efficiently directly in the polygonal representation [Isenburg and Snoeyink 2000]. The connectivity coder as well as the geometry coder of Touma and Gotsman [1998] have since been extended to the polygonal case [Isenburg 2002; Khodakovsky et al. 2002; Isenburg and Alliez 2002].

The extension of the geometry coder by Isenburg and Alliez [2002] aims at compressing quad-dominated meshes. It shows that using the parallelogram rule as often as possible within a polygon rather than across two polygons significantly improves compression. The authors report that the parallelogram rule gives the best predictions for quadrilaterals but that the prediction error increases for higher degree polygons. This is to be expected, since predicting the position of a vertex using the parallelogram rule within a regular, high degree polygon, which is illustrated in Figure 1, makes a systematic prediction error that increases with the degree of the polygon.

In this paper we describe an extension of the parallelogram rule to higher degree polygons that is based on the Fourier spectrum of a polygon. We design our prediction rules such that they place missing points of a polygon in positions that set the highest frequency to zero. We show that this theory is consistent with both the parallelogram rule and the Lorenzo predictor of [Ibarria et al. 2003], giving them retroactively another theoretic blessing.

2 Previous Work

Traditionally, the compression of mesh connectivity and geometry is done by clearly separated (but often interwoven) techniques. Recent schemes for connectivity compression use the concept of region growing [Isenburg 2002] where polygons adjacent to an already encoded region are processed one after the other until the entire mesh is conquered. The traversal order this induces on the vertices is typically used to compress their associated positions. These are predicted from the partially decoded mesh and only a corrective vector is stored. Because the correctors tend to spread around zero

*e-mail: isenburg@cs.unc.edu
†e-mail: ivrissim@mpi-sb.mpg.de
‡e-mail: sgumhold@mpi-sb.mpg.de
§e-mail: hpseidel@mpi-sb.mpg.de

and have a lower entropy than the original positions they can be efficiently compressed with an arithmetic coder [Witten et al. 1987].

Recently we have seen schemes that also exploit the redundancy between geometry and connectivity for improved connectivity coding. Such schemes use already decoded geometric information to improve the traversal [Lee et al. 2002], to predict local incidence [Coors and Rossignac 2004], or to have additional contexts for arithmetic coding [Kälberer et al. 2005]. In addition, they often express the prediction errors in a local coordinate frame to decorrelate the prediction error in tangential and normal direction.

Other, even more complex approaches for compressing mesh geometry include spectral methods [Karni and Gotsman 2000] that perform a global frequency decomposition of the surface, space-dividing methods [Devillers and Gandoin 2002] that specify the mesh connectivity relative to a geometric triangulation of connectivity-less coded point cloud, remeshing methods [Khodakovsky et al. 2000] that compress a re-parameterized version of the original mesh, feature-based methods [Shikhare et al. 2001] that find and instantiate repeated geometric features in a model, and high-pass methods [Sorkine et al. 2003] that quantize coordinates after a basis transformation with the Laplacian matrix.

We do not attempt to improve on any of these schemes. Instead, we generalize a simple and widely-used rule for predicting vertex positions in triangular [Touma and Gotsman 1998] and quad-dominated [Isenburg and Alliez 2002] meshes to achieve better compression performance on general polygon meshes such as hex-dominated meshes that result, for example, from discretization of a Voronoi diagram or from dualization of a triangle mesh.

2.1 The Parallelogram Rule

The simple linear prediction coder introduced by Touma and Gotsman [1998] predicts a position to complete a parallelogram that is spanned by the three previously processed vertices of a neighboring triangle. This scheme represents the best overall trade-off between simplicity of implementation, compression and decompression speed, and achieved bit-rate—especially for encoding large data sets. If compressor and decompressor are to achieve throughputs of millions of vertices per second, the per-vertex computation has to be kept to a minimum [Isenburg and Gumhold 2003].

The parallelogram rule gives poor predictions if used across triangle pairs that are highly non-planar or non-convex. Therefore, Kronrod and Gotsman [2002] suggest to first locate good triangle pairs and then use a maximal number of them. On meshes with sharp features, such as CAD models, they report improvements in compression of up to 40 percent. However, the construction of the *prediction tree* for directing the traversal to the good triangle pairs makes this scheme prohibitively complex for large models. Instead of using a single parallelogram prediction, Cohen-Or et al. [2002] propose to average multiple predictions. On smooth meshes this approach improves compression rates by about 10 percent.

For non-triangular meshes, Isenburg and Alliez [2002] show that compression improves on average by 20 percent if the parallelogram rule is applied *within* a polygon instead of *across* two polygons because non-triangular polygons tend to be "fairly" planar and convex. The authors note that "predictions within quadrilaterals have the smallest average prediction error" and that "the error becomes larger as the degree of the polygon increases." They perform experiments that show that "degree-adapted prediction rules result in small but consistent improvements in compression" but conclude that "these gains are bound to be moderate because on average more than 70 percent of the vertices are predicted within a quadrilateral."

The parallelogram rule can be written as the linear combination $v_0 = c_0 * v_0 + *c_1 * v_1 + c_2 * v_2$ where $c_0 = c_2 = 1$ and $c_1 = -1$ Simply by using different c_0, c_1, and c_2 we can formulate a *pentagon rule* or a *hexagon rule*. Such rules are not limited to base their predictions on only three vertices. The prediction of the last unknown vertex within a hexagon, for example, can use a linear combination of all five vertices. The challenge is to find *generic* coefficients that improve compression on all typical meshes. Isenburg and Alliez [2002] compute coefficient triples c_0, c_1, and c_2 for predicting the next vertex from three consecutive vertices for each polygon degree. They do this simply by minimizing the total Euclidean error over all potential predictions in their set of test meshes and report improved compression rates even on meshes that were not part of the test set. We describe how to derive such generic coefficients from the Fourier basis of a polygon, which turns out to be especially successful in case only one vertex is missing.

3 Polygonal Prediction Rules

The classic parallelogram rule can be thought of as predicting the missing vertex of a quadrilateral to be in a position such that the highest frequency of the quadrilateral's Fourier basis is eliminated. With this concept in mind it becomes straight-forward to design prediction rules for higher degree polygons that have just one missing vertex — we also assume their highest frequency to be zero.

Consequently, if more than one vertex of a polygon is missing we assume the second highest frequency of the polygon to be zero as well, then the third highest and so on. While conceptually simple, some practical complications arise from the fact that most frequencies come in pairs of two. We can not set only one of them to zero, as this would result in complex coefficients in our prediction rule.

3.1 The Fourier Basis of a Polygon

A polygon P of degree n can be described as a sequence (vector) of n points

$$P = [p_0, p_1, p_2, \ldots, p_{n-1}]^T \tag{1}$$

In 2D, each point corresponds to a complex number and the polygon is a complex vector, i.e. an element of $\mathbf{C}^n$.

With this notation, the discrete Fourier transform (DFT) in $\mathbf{C}^n$ has a polygonal equivalent which is written

$$\begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega & \omega^2 & \ldots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \ldots & \omega^{2n-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega^{n-1} & \omega^{2n-2} & \ldots & \omega \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ \vdots \\ z_{n-1} \end{bmatrix} = \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_{n-1} \end{bmatrix} \tag{2}$$

where $\omega = e^{\frac{2\pi i}{n}}$. The matrix $A = (a)_{ij}$ on the left is called the Fourier matrix of order n. The polygon $[z_0, z_1, \ldots, z_{n-1}]^T$ is called the Fourier transform of $[p_0, p_1, \ldots, p_{n-1}]^T$.

Many times it is convenient to write Eq. 2 in the form

$$z_0 \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + z_1 \begin{bmatrix} 1 \\ \omega \\ \omega^2 \\ \vdots \\ \omega^{n-1} \end{bmatrix} + \cdots + z_{n-1} \begin{bmatrix} 1 \\ \omega^{n-1} \\ \omega^{n-2} \\ \vdots \\ \omega \end{bmatrix} = \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_{n-1} \end{bmatrix} \tag{3}$$

showing that DFT is equivalent to writing the initial polygon in the Fourier basis of polygons shown in Eq. 3. All the polygons in the Fourier basis come in conjugate pairs, except of the one corresponding to z_0 and (when n is even) the one corresponding to $z_{n/2}$.

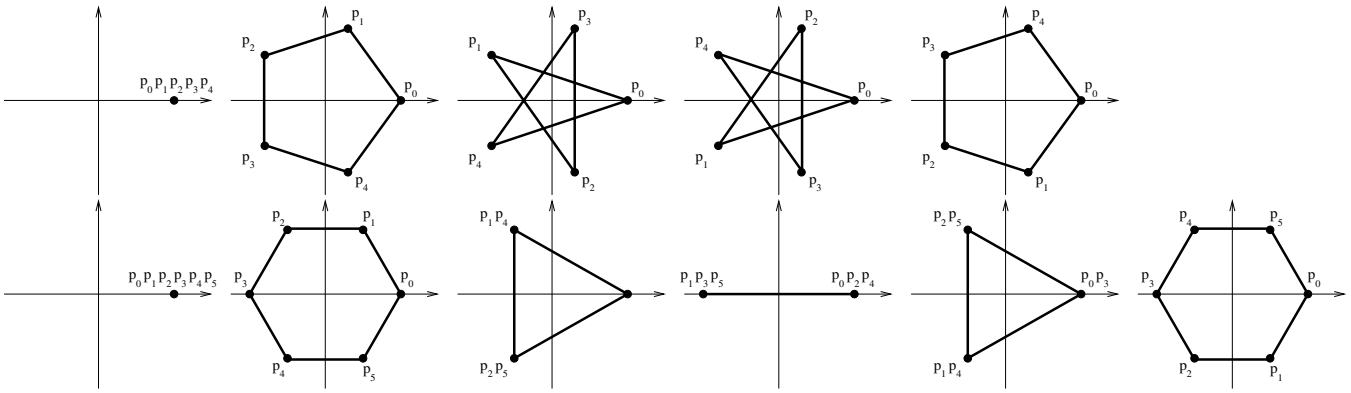


Figure 2: The Fourier basis of the pentagon (top) and the hexagon (bottom).

The number $f = 0, 1, \ldots, \lfloor n/2 \rfloor$ is called the frequency of the components z_f and z_{n-f}. In Figure 2 we give a visual illustration of the Fourier basis of pentagon and hexagon. See [Berlekamp et al. 1965] for a detailed treatment of this geometric interpretation of the DFT. In Figure 2 we give a visual illustration of the Fourier basis of pentagon and hexagon.

3.2 Predicting high Frequencies as Zero

Here we use the DFT in the context of geometry prediction. We assume that we know the position of k points of P and want to predict the position of another point of P as a linear combination of these known points. Working with the Fourier transform we construct a polygon $\hat{P}$ which is identical to P on the k known vertices. The latter property leaves $n-k$ degrees of freedom for the complete definition of $\hat{P}$, and we use them to make as many of the high frequencies of $\hat{P}$ as possible equal to zero. By eliminating the high frequencies our predictor will work well with nicely shaped polygons.

The main complication arises from the fact that even though we assume that $\hat{P}$ is planar, (as it is the case with the parallelogram in the classic prediction rule we generalize), still it is a polygon embedded in 3D. That means that the linear combination of points of $\hat{P}$ giving another point of $\hat{P}$ should have real coefficients. This can be easily achieved when the non-zero frequencies of $\hat{P}$ come in conjugate pairs. But, the latter happens only when k is odd. Thus, below we separate the cases k odd and even.

Odd known points

Due to the inner workings of the connectivity coder we assume that the $k = 2m+1$ known points are consecutive. For symmetry reasons we also assume that p_0 is at the center of this configuration, i.e. the known points are

$$[p_{n-m}, \ldots, p_{n-1}, p_0, p_1, \ldots, p_m]^T \tag{4}$$

From Eq. 2, after putting zero all the frequencies higher than m, we obtain the system

$$B[z_{n-m}, \ldots, z_0, \ldots, z_m]^T = [p_{n-m}, \ldots, p_0, \ldots, p_m]^T \tag{5}$$

where B is the $k \times k$ submatrix of the Fourier matrix $(a)_{ij}$ obtained for $i, j = n-m, \ldots, n-1, 0, 1, \ldots, m$.

From Eq. 5 we get,

$$[\hat{z}_{n-m}, \ldots, \hat{z}_0, \ldots, \hat{z}_m]^T = B^{-1}[p_{n-m}, \ldots, p_0, \ldots, p_m]^T \tag{6}$$

and $[0, \ldots, 0, \hat{z}_{n-m}, \ldots, \hat{z}_0, \ldots, \hat{z}_m, 0, \ldots, 0]^T$ is $\hat{P}$ written in the Fourier basis. Finally, using Eq. 3 we can compute any point of $\hat{P}$ as a linear combination of the known points

$$\hat{p}_l = \sum_{j=0}^{n-1} a_{lj} \hat{z}_j \tag{7}$$

For example if $n = 7$ and $k = 3$ we have

$$B = \begin{bmatrix} \omega & 1 & \omega^6 \\ 1 & 1 & 1 \\ \omega^6 & 1 & \omega \end{bmatrix}$$

where $\omega = e^{\frac{2\pi i}{7}}$, giving,

$$\begin{aligned} p_{i+3} &= p_i - 2.247 p_{i+1} + 2.247 p_{i+2} \\ p_{i+4} &= 2.247 p_i - 4.0489 p_{i+1} + 2.8019 p_{i+2} \end{aligned} \tag{8}$$

Notice that the further the predicted point is from the known points, the larger become the coefficients. This means that the predictions of points far away from the known ones are more prone to result in large prediction errors. Our strategy is to predict points that are next to the known ones, which creates increasingly long chains of known points around the polygon.

Even known points

Let now the $k = 2m$ known points be

$$[p_{n-m}, \ldots, p_0, \ldots, p_{m-1}]^T \tag{9}$$

After eliminating the $2m-1$ frequencies less than $m-1$, we cannot eliminate only one of the two m frequencies without introducing complex numbers in the prediction rules. One easy solution is to use the $k = 2m-1$ predictor, ignoring the point p_{n-m}. Then, we can use the $(2m+1)$-point rule twice to obtain $(2m+3)$ points and so on.

The obvious disadvantage of this approach is that half of the time we do not use one of the known points of the polygon, thus do not exploit all available data for prediction. The solution we adopt here is to find a $2m$-point rule as a linear combination of the two $2m-1$-point rules given by the points

$$[p_{n-m-1}, \ldots, p_0, \ldots, p_{m-1}]^T \tag{10}$$

and

$$[p_{n-m}, \ldots, p_0, \ldots, p_{m-2}]^T \tag{11}$$

Obviously the weights λ and $(1-\lambda)$ of the two predictions should be chosen carefully. Given that the latter prediction is usually inferior, if it is overweighted, then the combined $2m$-point prediction may become worse than a single $(2m-1)$-point prediction.

In the special case of a $2m$ prediction within a polygon of order $2m+1$ the choice $\lambda = \frac{1}{2}$ is obvious due to cyclical symmetry. This case appears most often in practice as the 4-point prediction within a pentagon or the 6-point prediction within a heptagon.

For the general case we choose the λ that minimizes the norm of the vector

$$[\alpha_{n-m}, \ldots, \alpha_0, \ldots, \alpha_{m-1}]^T \tag{12}$$

where

$$\alpha_{n-m} p_{n-m} + \cdots + \alpha_0 p_0 + \cdots + \alpha_{m-1} p_{m-1} \tag{13}$$

is the combined prediction.

Even though the heuristic argument of using small coefficients for good predictions is intuitively appealing, we do not claim that this choice is anyhow optimal and the experimental results where mixed. However, we found that this choice consistently improves over the $(2m-1)$-point rule in the case of polygons of small order which appear most often in practice.

3.3 Validation of Existing Predictors

The main property of the above predictors is the elimination of the highest frequencies of a geometric object (here a polygon) in the Fourier domain. Following this methodology we obtain not only the classical parallelogram rule for quadrilateral surface elements of [Touma and Gotsman 1998] but also the Lorenzo predictor for hexagonal volume elements and its generalizations over the n-dimensional hypercube [Ibarria et al. 2003].

There, the vertices of the hypercube are labelled by the elements of $\mathbf{Z}_2^n = \{0,1\}^n$. Then, assuming that we already know the positions of $2^n - 1$ vertices, the position of the unknown vertex

$$v = \{1\}^n = [1,1,\ldots,1] \tag{14}$$

is predicted by

$$v = \sum_{u \in U} (-1)^{c_0(u)+1} u \tag{15}$$

where $U = \mathbf{Z}_2^n - \{v\}$ and $c_0(u)$ denotes the number of coordinates of u that equal zero. Equivalently, the Lorenzo predictor eliminates the vector

$$\sum_{u \in \mathbf{Z}_2^n} (-1)^{c_0(u)+1} u \tag{16}$$

which is the highest frequency component of the Fourier transform over $\mathbf{Z}_2^n$. The Fourier transform over $\mathbf{Z}_2^n$ is described in [Chazelle 2000] by the Hadamard matrix $H^{(n)}$. The highest frequency component of Eq.(16) is given by the last column of $H^{(n)}$.

4 Results

In Table 1 we detail the polygonal composition of the hex-dominant meshes that we used in our experiments. Five of these meshes are the duals of isotropic remeshings created by Surazhsky et al. [2003], five of them are the duals of well-known standard models. The table compares the compressed size of connectivity and geometry of the dual mesh with the primal mesh. The connectivity coder of Isenburg [2002] can adapt to the duality in degrees and delivers similar compression rates for either mesh. Both meshes contain the same "amount" of connectivity (i.e. the same number of edges). For compressed geometry the comparison is not as meaningful because dual and primal contain a different number of vertices and because

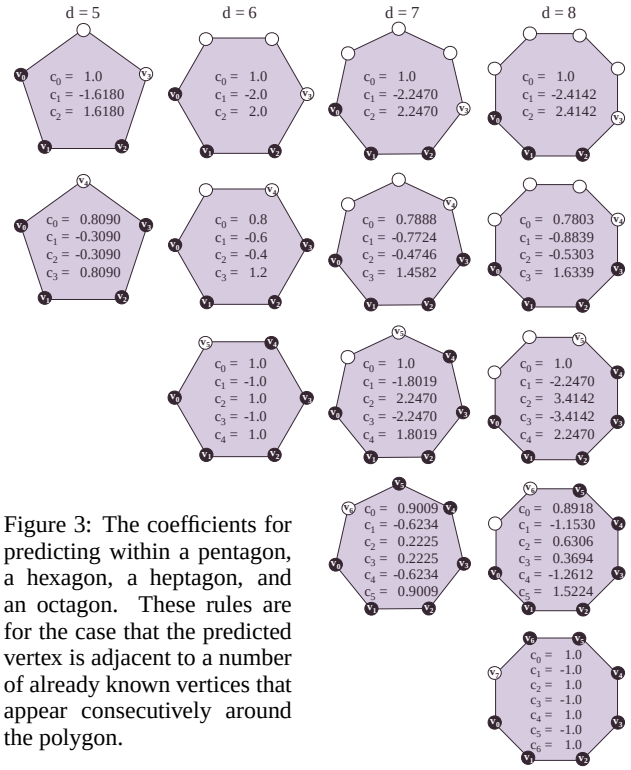


Figure 3: The coefficients for predicting within a pentagon, a hexagon, a heptagon, and an octagon. These rules are for the case that the predicted vertex is adjacent to a number of already known vertices that appear consecutively around the polygon.

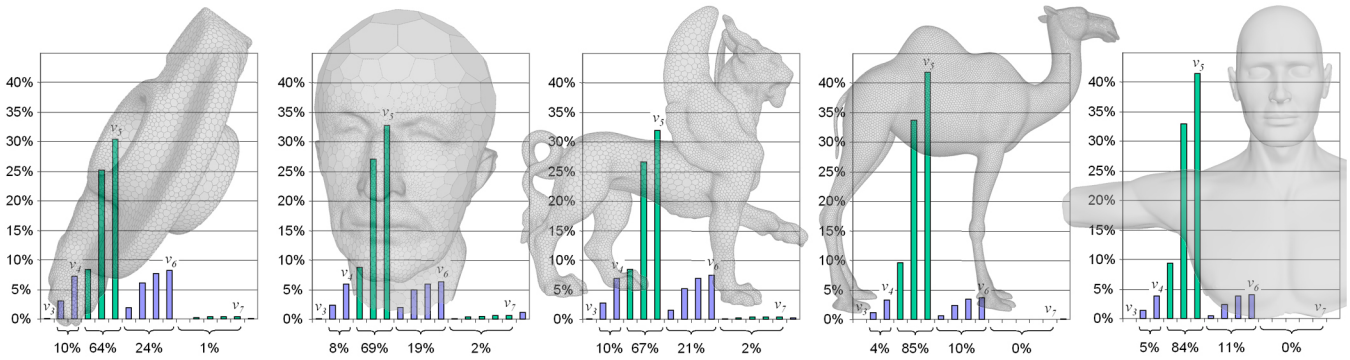
the bit-rates of predictive geometry coding are not directly proportional to the prediction error. However, in contrast to the standard parallelogram rule our polygonal rules have a similar "flavor" of duality. Despite compressing double the number of compressed vertices, the total size increases by only 10 to 40 percent.

In Table 2 we compare the bit-rates achieved using parallelogram predictions with those of using polygonal predictions within each polygon for different levels of precision. Note that due to the systematic miss-predictions of the parallelogram rule there is little additional compression gain as the precision increases. Increasing the precision by two bits per coordinate, often increases the resulting bit-rate by about six bits per vertex. The polygonal predictions, on the other hand, are usually still able to exploit some of the redundancy in the more precise vertex positions. For them, the resulting bit-rate only increases by about three to four bits per vertex.

In Table 3 we analyze the prediction error of each polygonal rule. To be independent of the specific traversal order chosen by the connectivity coder, we averaged the result for each rule over *all* possible applications. For each polygon type and each possible number p of prediction vertices we report the averaged error for prediction of an unknown vertex v_p from its p preceding vertices v_0 to v_{p-1}. By far the best predictions are achieved within polygons where only one vertex is missing. We notice that in general the predictions become better as we have more vertices to base the prediction on. An exception are the prediction errors for v_4 in the heptagon and the octagon, which in both cases are worse than the prediction errors for v_3. This needs further investigation. However, in practice these prediction rules are rarely used. The few cases that occur are at the moment predicted with the prediction rule for v_3.

5 Discussion

In Table 3 we notice that the error is especially low when the order n of the polygon is even and only one vertex is missing. We will show that for n even, the $(n-1)$-point prediction is geometrically



dual meshes	number of vertices	number of polygons	polygons of degree 3	4	5	6	7	8	9+	conn [KB] dual	conn [KB] prim	geom [KB] dual	geom [KB] prim	geom [KB] diff
rocker arm	11,108	5,554	–	46	947	3573	939	47	2	1.0	1.0	21.7	16.7	30%
max plank	31,704	15,803	24	117	2,291	11,083	2,070	194	24	2.9	2.6	51.3	40.4	27%
feline	41,262	20,629	–	90	3,382	13,814	3,135	186	22	3.6	3.4	66.8	53.4	25%
camel	78,162	39,083	–	10	2,930	33,260	2,842	31	10	3.8	3.7	85.5	75.6	13%
torso	284,692	142,348	2	27	11,550	119,234	11,483	44	8	14.2	14.1	233	208	12%
cow	5,804	2,904	7	87	514	1,796	364	98	38	0.7	0.6	13.1	10.3	27%
horse	96,966	48,485	13	745	7,481	32,230	7,154	792	70	8.5	8.5	132	93.6	41%
dinosaur	112,384	56,194	–	1,355	10,734	32,511	10,044	1,316	134	11.7	11.4	146	119	23%
armadillo	345,944	172,974	41	4,440	36,780	90,928	35,088	5,326	151	37.4	36.8	443	320	38%
isis	375,284	187,644	–	2,293	31,521	120,408	30,896	2,450	76	33.0	32.9	352	255	38%

Table 1: We report the size of the compressed **dual** and the compressed **prim**al for ten different models. The histograms illustrate the percentage of vertices that is predicted with each polygonal rule. For **conn**ectivity we use the Degree Duality coder of Isenburg [2002]. For **geom**etry of we use the polygonal rules described here for the dual mesh and the standard parallelogram rule for the primal mesh. The vertex positions are uniform quantized to 14 bits of precision. The first five meshes (shown) are dual to isotropic remeshings created with the method of Surazhsky et al. [2003]. The bottom five meshes (not shown) are dual to well-known standard models.

dual meshes	12 bit para	12 bit poly	12 bit gain	14 bit para	14 bit poly	14 bit gain	16 bit para	16 bit poly	16 bit gain
rocker arm	20.4	11.0	46 %	27.2	16.0	41 %	33.3	21.4	36 %
max plank	16.6	9.1	45 %	22.8	13.2	42 %	28.9	18.1	37 %
feline	17.4	8.7	50 %	23.5	13.3	43 %	29.5	18.3	38 %
camel	14.6	5.9	60 %	20.6	9.0	56 %	26.7	13.2	51 %
torso	11.1	5.0	55 %	17.0	6.7	61 %	23.3	10.4	55 %
average			51 %			49 %			43 %
cow	22.6	13.5	40 %	29.9	18.5	38 %	35.1	23.7	32 %
horse	13.7	7.2	47 %	19.1	11.2	41 %	24.9	15.7	37 %
dinosaur	13.1	6.6	50 %	19.0	10.6	44 %	25.2	15.7	38 %
armadillo	11.8	6.5	45 %	15.7	10.5	33 %	19.0	15.0	21 %
isis	9.5	5.4	43 %	15.0	7.7	49 %	21.0	11.9	43 %
average			45 %			41 %			34 %

Table 2: Compression rates for **para**llelogram predictions and for **poly**gonal predictions within the polygons of hex-dominant dual meshes at different levels of quantization and the achieved gains.

exact on the meshes produced by barycentric dualization. Thus, any reported error is the result of the uniform quantization of vertices.

The standard dualization algorithm we use introduces new vertices at the barycenters of the triangles of the primal mesh. Let O be a vertex of the primal mesh. The corresponding face of the dual is related to the boundary of the 1-ring neighborhood of O through a simple linear transformation which can be described in two steps.

First, join in order the midedges of the boundary of the 1-ring neighborhood of O to obtain a new polygon. This is known in the literature as the *midedge transformation*, see [Berlekamp et al. 1965]. Then, scale the new polygon, (i.e. apply a similarity transformation), with center O and ratio 2/3. Fig. 4 shows these two transformations on a hexagon and a quadrilateral.

mesh name	quad v_3	pent v_3	pent v_4	hexa v_3	hexa v_4	hexa v_5	hept v_3	hept v_4	hept v_5	hept v_6	octa v_3	octa v_4	octa v_5	octa v_6	octa v_7
rocker arm	1	127	76	190	169	1	245	259	108	47	273	329	227	158	1
max plank	3	174	102	323	288	4	729	784	305	131	703	732	488	365	5
feline	1	107	62	161	146	2	202	220	94	40	254	286	200	148	2
camel	1	44	25	49	44	2	75	76	43	19	172	214	166	106	2
torso	14	17	1	21	18	2	31	32	20	9	70	79	61	37	2

Table 3: The averaged prediction error of each polygonal rule over all possible ways of predicting missing vertices v_p within polygons using the vertices v_0 to v_{p-1} with the coefficients c_0 to c_{p-1} from Figure 4. The averaged prediction errors have been scaled per model with 1 corresponding to the smallest prediction error.

The effect of the similarity transformation on the eigencomponents of the polygon is trivial as it scales all of them by $2/3$. On the other hand, the midedge transformation corresponds to the matrix

$$C = \begin{bmatrix} 1/2 & 1/2 & 0 & \dots & 0 & 0 \\ 0 & 1/2 & 1/2 & \dots & 0 & 0 \\ & & \dots & & & \\ 0 & 0 & \dots & 0 & 1/2 & 1/2 \\ 1/2 & 0 & 0 & \dots & 0 & 1/2 \end{bmatrix}. \qquad (17)$$

The matrix C is *circulant* because each row is the previous row cycled forward one step. By a well-known theorem, the Fourier basis in Eq.(3) is a set of orthogonal eigenvectors for any circulant matrix, see [Davis 1979]. That means that the effect of the midedge transformation on the eigencomponents of a polygon is a scaling by a ratio equal to the corresponding eigenvalue.

To compute the eigenvalues of C recall that any circulant matrix is

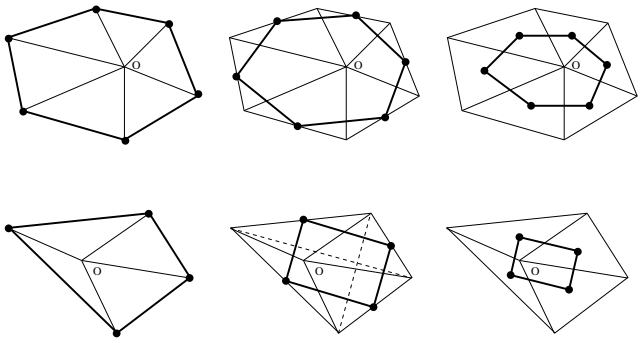


Figure 4: Left: The boundary of the 1-ring neighborhood of O in bold. Middle: The midedge transform in bold. Right: The face of the dual in bold. Top: The hexagon. Bottom: the quadrilateral.

completely defined by its first row

$$(c_0, c_1, c_2, \ldots, c_{n-1}) \tag{18}$$

and thus, it is also completely defined by the associated *generating polynomial*

$$p(z) = c_0 + c_1 z + c_2 z^2 + \cdots + c_{n-1} z^{n-1} \tag{19}$$

The eigenvalues are obtained by evaluating $p(z)$ at the n-th roots of unity

$$\lambda_j = p(\omega^j), \quad j = 0, 1, \ldots, n-1 \tag{20}$$

see [Davis 1979].

In the case of the midedge transformation the generating polynomial is

$$\frac{1}{2}(1+z) \tag{21}$$

If n is even, then -1 is an n-th root of unity, corresponding to the highest frequency $n/2$. By Eq.(21) the eigenvalue corresponding to this highest frequency is zero. That means that the midedge transformation, and thus our dualization process, eliminates the highest frequency component from the polygons of even order. That makes the $(n-1)$-point predictions on these polygons exact.

For $n = 4$ in particular, this means that the parallelogram rule is exact on dual meshes. This corresponds to a famous theorem of elementary Euclidean geometry, that the midedge transform of a (not necessarily planar) quadrilateral is a parallelogram. Indeed, the opposite edges of the midedge transform are both parallel (and half in length) to a diagonal of the original quadrilateral (see Fig. 4).

6 Summary

We have presented a general method for extending the parallelogram rule for more efficient predictions within pentagons, hexagons, and other high degree polygons that are especially prominent after dualization. Our method is based on the decomposition of the polygons into the Fourier basis and the prediction that missing points will be in a position that does not introduce high frequencies. We have reported the prediction coefficients that result from this analysis for the most common polygon types and we have validated their effectiveness on two different sets of dual meshes.

Previous work has developed connectivity coders that have practically the same compression rate for both the primal and the dual of a mesh [Isenburg 2002; Khodakovsky et al. 2002]. Although the direct comparison of primal and dual geometry rates is less "pure" because the number of predicted entities (i.e. vertex positions) increases significantly and because compression rates in predictive coding are generally not directly proportional to the prediction error, our polygonal prediction rules have the flavor of a "similar" duality for geometry compression.

Acknowledgements

We thank Pierre Alliez for providing us with the primals and the duals of the five isotropically remeshed models we used in our experiments.

References

BERLEKAMP, E., GILBERT, E., AND SINDEN, F. 1965. A polygon problem. *Amer. Math. Monthly 72*, 233–241.

CHAZELLE, B. 2000. *The Discrepancy Method — Randomness and Complexity*. Cambridge University Press.

COHEN-OR, D., COHEN, R., AND IRONY, R. 2002. Multi-way geometry encoding. Tech. rep., Computer Science, Tel Aviv University.

COORS, V., AND ROSSIGNAC, J. 2004. Delphi: Geometry-based connectivity prediction in triangle mesh compression. *The Visual Computer 20*, 8-9, 507–520.

DAVIS, P. 1979. *Circulant matrices*. Wiley-Interscience.

DEVILLERS, O., AND GANDOIN, P.-M. 2002. Progressive and lossless compression of arbitrary simplicial complexes. In *SIGGRAPH'02 Conference Proceedings*, 372–379.

IBARRIA, L., LINDSTROM, P., ROSSIGNAC, J., AND SZYMCZAK, A. 2003. Out-of-core compression and decompression of large n-dimensional scalar fields. In *Eurographics'03 Proceedings*, 343–348.

ISENBURG, M., AND ALLIEZ, P. 2002. Compressing polygon mesh geometry with parallelogram prediction. In *Visualization'02 Conference Proceedings*, 141–146.

ISENBURG, M., AND GUMHOLD, S. 2003. Out-of-core compression for gigantic polygon meshes. In *SIGGRAPH'03 Proceedings*, 935–942.

ISENBURG, M., AND SNOEYINK, J. 2000. Face Fixer: Compressing polygon meshes with properties. In *SIGGRAPH'00 Proceedings*, 263–270.

ISENBURG, M., AND SNOEYINK, J. 2005. Early-split coding of triangle mesh connectivity. manuscript.

ISENBURG, M. 2002. Compressing polygon mesh connectivity with degree duality prediction. In *Graphics Interface'02 Proceedings*, 161–170.

KÄLBERER, F., POLTHIER, K., REITEBUCH, U., AND WARDETZKY, M. 2005. Freelence: Compressing triangle meshes using geometric information. Tech. Rep. ZR–04–26 (revised), Zuse Institute Berlin.

KARNI, Z., AND GOTSMAN, C. 2000. Spectral compression of mesh geometry. In *SIGGRAPH'00 Conference Proceedings*, 279–286.

KHODAKOVSKY, A., SCHROEDER, P., AND SWELDENS, W. 2000. Progressive geometry compression. In *SIGGRAPH'00 Proceedings*, 271–278.

KHODAKOVSKY, A., ALLIEZ, P., DESBRUN, M., AND SCHROEDER, P. 2002. Near-optimal connectivity encoding of 2-manifold polygon meshes. *Graphical Models 64*, 3-4, 147–168.

KRONROD, B., AND GOTSMAN, C. 2002. Optimized compression of triangle mesh geometry using prediction trees. In *International Symposium on 3D Data Processing Visualization and Transmission*, 602–608.

LEE, H., ALLIEZ, P., AND DESBRUN, M. 2002. Angle-analyzer: A triangle-quad mesh codec. In *Eurographics'02 Proceedings*, 198–205.

SHIKHARE, D., BHAKAR, S., AND MUDUR, S. 2001. Compression of 3D engineering models using automatic discovery of repeating geometric features. In *Proc. of Vision Modeling and Visualization'01*, 233 – 240.

SORKINE, O., COHEN-OR, D., AND TOLEDO, S. 2003. High-pass quantization for mesh encoding. In *Proceedings of Symposium on Geometry Processing'03*, 42–51.

SURAZHSKY, V., ALLIEZ, P., AND GOTSMAN, C. 2003. Isotropic remeshing of surfaces: a local parameterization approach. In *Proceedings of 12th International Meshing Roundtable*, 215–224.

TOUMA, C., AND GOTSMAN, C. 1998. Triangle mesh compression. In *Graphics Interface'98 Conference Proceedings*, 26–34.

WITTEN, I. H., NEAL, R. M., AND CLEARY, J. G. 1987. Arithmetic coding for data compression. *Communications of the ACM 30*, 6, 520–540.