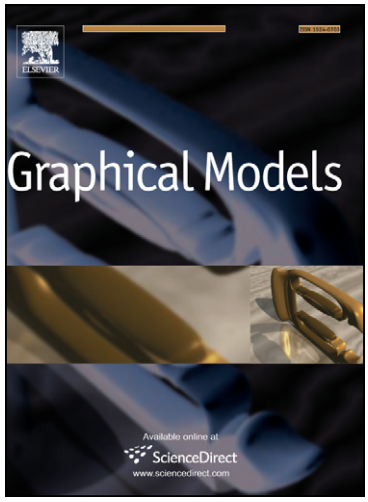

Vertex Location Optimisation for Improved Remeshing

Yu-Kun Lai, Ralph R. Martin

PII: S1524-0703(12)00031-8
DOI: http://dx.doi.org/10.1016/j.gmod.2012.04.011
Reference: YGMOD 809

To appear in: *Graphical Models*

Received Date: 4 March 2012
Revised Date: 12 April 2012
Accepted Date: 23 April 2012

Please cite this article as: Y-K. Lai, R.R. Martin, Vertex Location Optimisation for Improved Remeshing, *Graphical Models* (2012), doi: http://dx.doi.org/10.1016/j.gmod.2012.04.011

Vertex Location Optimisation for Improved Remeshing

Yu-Kun Lai [1], Ralph R. Martin [1]

[1] Cardiff University, UK

Abstract

Remeshing aims to produce a more regular mesh from a given input mesh, while representing the original geometry as accurately as possible. Many existing remeshing methods focus on where to place new mesh vertices; these samples are placed exactly on the input mesh. However, considering the output mesh as a piecewise linear approximation of some geometry, this simple scheme leads to significant systematic error in non-planar regions. Here, we use parameterised meshes and the recent mathematical development of orthogonal approximation using Sobolev-type inner products to develop a novel sampling scheme which allows vertices to lie in space near the input surface, rather than exactly on it. The algorithm requires little extra computational effort and can be readily incorporated into many remeshing approaches. Experimental results show that on average, approximation error can be reduced by 40% with the same number of vertices. A similar technique can also be applied to surface normals to provide more accurate rendering results with the same number of vertices.

1 Introduction

With the maturity of 3D acquisition and modelling techniques, 3D meshes are now widely available. Polygonal meshes, especially triangular meshes, and to a lesser extent quad meshes, are widely used in digital geometry processing for applications such as modelling and animation. They are also popular in finite element analysis for engineering simulation. In all of these applications, the quality of meshes is of importance. Higher quality meshes typically improve the efficiency of processing because fewer elements suffice to accurately represent the geometry. In simulation, they also improve the robustness and stability of numerical computation. Low quality meshes, on the other hand, may make certain algorithms behave poorly or even work incorrectly due to irregular connectivity, badly shaped triangles etc. Unfortunately, meshes obtained directly from reconstruction of scanned range images are often of low quality. Thus, *remeshing* is commonly used for mesh preprocessing before further operations.

Although different applications may have different requirements, mesh *regularity* is often desirable. For triangular meshes for example, elements should be close to equilateral and, at least for some purposes, have similar size. The degree of each vertex is ideally 6. This leads to a large class of remeshing methods which attempt to perform (quasi-)isotropic remeshing (e.g. [2, 30, 29]). On the other hand, the error with which the output mesh approximates the input mesh should be low. If low triangle counts are important, this may be achieved via curvature adapted sampling—using smaller triangles in regions with high curvature and larger triangles for low-curvature regions.

Most remeshing methods assume that new vertices are directly placed on the input mesh. This strategy is simple and intuitive, but causes significant systematic approximation error in non-planar (convex or concave) regions. For example, if the input is a mesh representing a convex body, it is not hard to see that such remeshing will lead to a new mesh entirely contained within the input mesh, hence having lower volume. The output mesh is actually a piecewise linear representation with greater error at the centre of each linear element. Some methods such as [30] assume that the *input* triangle mesh (rather than the *output* triangle mesh) is a piecewise linear approximation to some smooth surface. Thus, instead of sampling vertices on faces of the input mesh, vertices are placed on local estimates of the original surface produced by fitting. A simple and effective approach proposed in [32] substitutes each triangle with a so-called curved PN triangle, i.e. an interpolated three-sided cubic Bézier patch with a quadratically varying normal, given positions and normals at three corner points. G^1 continuity can be obtained by making normals identical where adjacent triangles meet. A more complex approach in [33] guarantees G^1 continuity at the cost of more complicated computation.

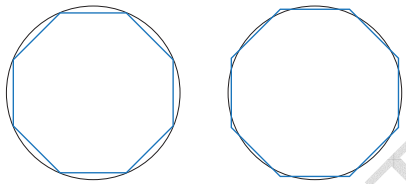

Figure 1: Two approaches to piecewise linear approximation. The right hand approach has lower error.

The input geometry to our algorithm is usually represented as a triangle mesh M_I. If M_I is sufficiently dense, direct use of it can give a reasonable approximation to the input surface. Otherwise, if the assumption that vertex samples lie *on* the input surface can be made, we use curved PN triangles to better approximate the geometry, for simplicity. Alternative approaches such as those in [33] could also be used. We use $\tilde{M}_I$ to represent the (approximated) input geometry in either case.

Even if the input surface can be well approximated from the input mesh, systematic errors still exist. A simple 2D example is shown in Fig. 1, where in this case a smooth curve (a circle) is approximated by a 2D piecewise linear shape (a polygon rather than a mesh). Previous remeshing approaches essentially perform a sampling of the kind shown in Fig. 1(left) where sample points (vertices) are located exactly on the shape. The approximation error is unnecessarily large as the piecewise linear polygon lies entirely within the shape to be approximated. A more accurate solution can be achieved with the same number of linear elements using the polygon in Fig. 1(right). The vertices are now located off the shape, and the output polygon is as close as possible to the shape—each linear element lies partly inside and partly outside the shape. Similar errors arise whether the input shape is exact, or a mesh approximating that shape.

In this paper, we propose a novel algorithm to optimise vertex locations near to, but not necessarily on, the input shape, in order to significantly reduce the approximation error with respect to the input surface when constructing meshes—piecewise linear representations. The paper fo-

cuses on triangular remeshing for simplicity. The concepts proposed in the paper can be incorporated into various remeshing approaches, or used as a post-processing stage operating on the output of traditional remeshing methods.

Previous work has considered optimising linear approximation in the settings of shape approximation [10] and linearisation of splines [26], but it appears that it has not previously been used for optimising vertex locations to provide improved piecewise linear approximation *in remeshing*. We give a robust and efficient algorithm for achieving this. Similar to triangular remeshing algorithms, an iterative algorithm is used to balance the regularity of the output mesh and the approximation error. Within each iteration, we adapt the recent mathematical development of piecewise linear orthogonal approximation defined over triangulations of domains in $\mathbb{R}^d$ [5] to more general 2D triangular meshes embedded in 3D space. By constructing specific Sobolev-type inner products, hat functions defined on the triangulation are made orthogonal, leading to explicit solutions to the approximation minimisation problem. Unlike using least-squares fitting with appropriate constraints, no linear system needs to be solved, and thus it can be computed very efficiently.

2 Related Work

Remeshing is an active research topic in digital geometry processing; we refer readers to [4] for a comprehensive survey. Most work considers triangular remeshing. A variety of algorithms has been proposed to achieve (quasi-)isotropic remeshing of models with, ideally, equilateral triangles of the same size, or size adapted according to local curvature. Alliez et al. [3] use error diffusion to initially distribute new sample vertices over the input surface. Their positions are then optimised using a centroidal Voronoi diagram in the global conformal parameter domain. Constrained Delaunay triangulation is finally used to build the connectivity. Although it is possible to convert an arbitrary mesh into a topological disk, the method does not work so well for high-genus models due to the introduction of seams, and a greater degree of parametric distortion. The work in [30, 29] uses local parameterisations to reduce artefacts introduced by global parameterisation. Another approach to isotropic remeshing proposed in [9] utilises iterative local modifications to incrementally improve the regularity. The method proposed in [19] uses a two-stage approach. In the first stage, vertices are sampled over the surface and their positions optimised by minimising a spring-like energy [35]. The connectivity is then re-established through constrained Delaunay triangulation in local (quasi-)conformal parameter domains. The work in [7, 11] produces isotropic remeshing using Delaunay refinement. Under certain assumptions such methods can produce meshes with bounded approximation error. Yan et al. [36] gave an efficient and exact implementation of a restricted Voronoi diagram (an approximation of a Voronoi diagram over surfaces but using Euclidean distance instead of geodesic distance) for improved isotropic remeshing. The proposed method exploits the improvement of (isotropic) remeshing by allowing vertices to stay off the input surface. This can potentially be combined as a post-processing step with various remeshing algorithms, as demonstrated by various examples later in the paper.

Regularity and geometric fidelity are always a trade-off. With downstream applications such as multi-resolution modelling in mind, various algorithms attempt to achieve semi-regular remeshing (often called remeshing with *subdivision connectivity*), which requires the output mesh to have a small number of extraordinary vertices while all other vertices are completely regular. This kind of remeshing can be achieved via a parameterisation (mapping) of the input mesh to some simplified base mesh. An algorithm proposed in [20] constructs the base mesh and builds the parameterisation in the same process. Kobbelt et al. [17] give an approach based on simulating the physical process of shrink wrapping to obtain an optimised base mesh for remeshing with subdivision connectivity. A completely regular remeshing is also possible. Geometry images [15] use global stretch minimisation parameterisation to map an arbitrary surface to a square domain. By carefully controlling the boundary parameterisation, the geometry can be reduced to an image-like regular grid of 3D coordinates (mimicking the red, green, blue components of colours), with fully implicit connectivity. Spherical parameterisation can also be used for producing geometry images [27]. It is particularly suitable for genus-zero surfaces since distortion in parameterisation can usually be kept small. Achieving better regularity often implies sacrificing a certain degree of geometric accuracy for a given number of samples.

Other research focuses more on reducing the approximation error than on achieving regularity. Although curvature-adapted sampling is widely used [3, 30, 29] (with larger triangles in flat regions and smaller triangles in regions of high curvatures), such an approach has limited usefulness for regions having only one large principal curvature. For sharp edges, methods like the one in [6] can not only preserve but potentially enhance sharp edges in the remeshed models. To deal with both smooth and sharp features, [19] suggests an approach to isotropic remeshing using a feature sensitive metric which takes into account both positions and normals—it robustly adapts triangle shape according to the local principal curvatures. As isotropic sampling is performed in a specifically defined metric, regularity in connectivity is well preserved. However, the resulting triangles can significantly deviate from equilateral in feature regions.

In all of the above approaches, a certain balance between regularity and geometric approximation error is achieved. Our method, on the other hand, can be applied to different remeshing algorithms, reducing the approximation error while keeping regularity.

Quad (dominant) remeshing has also received wide attention. Quads are particularly suitable for representing local principal directions, and more naturally represent many man-made, nearly rectilinear, structures. Various methods have been proposed: streamline integration [1, 23], the gradient of a smooth harmonic field designed by the user [13] or derived from Laplacian eigenfunctions [12], combinatorial optimisation [24], iterative local incremental improvement [18], and global parameterisation [28, 16, 8]. Quad dominant meshes with edges well aligned with principal directions tend to have quasi-planar quads as the sum of inner angles is close to 2π. Liu et al. [22] give an optimisation process to improve the planarity of quad meshes. Although our current method focuses on improving triangle remeshing, our ideas can potentially be extended to quasi-planar quad remeshing, as both use piecewise linear shapes to approximate underlying geometry.

Using our method provides more accurate meshes for the same number of triangles, or for a given accuracy, provides meshes with fewer triangles, which can both be of benefit to downstream applications. Our method can be considered as a mesh post-processing algorithm for improved remeshing. From this perspective, it is related

to [34]. That work, however, does not take into account the piecewise linear nature of the mesh, but instead minimises an energy combining both fidelity (measured using two-sided Hausdorff distances between vertices of the input and remeshed models) and regularity (measured by the shapes of triangles). If a dense input mesh is used, it can produce results that bear some similarities with the proposed method. However, significant conceptual differences exist between the two approaches: our method explicitly considers piecewise linear approximation by the remeshed model while [34] uses dense point sampling to approximate integration. A dense input mesh is thus essential for their method to work effectively which is not required by our method. Our method further uses an orthogonal weighted Sobolev metric to explicitly calculate the updated positions and the computational cost mainly depends on the size of the *remeshed* model. On the other hand, [34] needs to solve a much larger linear system in each iteration, which is related to the sizes of both the *input* and the *remeshed* meshes. Our method is typically tens or even hundreds of times faster. Instead of optimising Hausdorff distance, our method minimises some average distance which is preferable at least for certain applications, as it takes into account the distribution of all the vertices. Further discussion will be given in Sec. 4. A typical 40% reduction in approximation error from the input can result with the same number of vertices.

3 Algorithm

This section presents a detailed algorithm for vertex location optimisation based on piecewise linear (PL) approximation. Pseudocode is listed in Algorithm 1. Our algorithm can be incorporated in various existing remeshing algorithms. One important observation is that to reduce the PL approximation error, only local considerations are needed for adjustment of vertex positions. For greater efficiency, our method can be applied in just the last few iterations of traditional iterative remeshing algorithms. Many remeshing algorithms use a two-stage strategy, and connectivity is only re-established in the second stage after position optimisation has finished. Since our modification is assumed to be local, we make a further assumption that only vertex positions need to be optimised and the connectivity can be kept unchanged. This is in fact necessary, since connectivity information is needed to determine the optimal PL approximation.

The input to our algorithm includes the input mesh to be remeshed M_I and the initial remeshing result M_0. Optimising a PL approximation changes the positions of vertices. Since the desired regularity (e.g. shape of triangles) is defined with respect to the output triangulation, an iterative process is used to balance goodness of approximation and regularity. A sequence of iteratively optimised meshes M_t is produced, until convergence (the maximum change in any vertex position between successive meshes in the sequence is below a threshold ε), or a set maximum number of iterations has been reached. During each iteration, vertex locations are optimised to reduce PL approximation error followed by regularity optimisation (e.g. to keep the new mesh regular in terms of uniform edge lengths). This may be considered as an extension of traditional iterative optimisation of point samples, but instead of optimisation in a search space with points constrained to lie on the surface, we instead optimise regularity in the search space of optimised PL approximation. Similar convergence to remeshing algorithms can be obtained with our proposed approach.

Algorithm 1 Vertex location optimisation for piecewise linear approximation

input:
  the input mesh M_I to be remeshed
  the initial remeshing result M_0
output: mesh M_T with improved vertex locations
$t \Leftarrow 0$
Fit curved PN triangles to M_I
repeat
  $t \Leftarrow t+1$
  for each vertex v_i in mesh M_{t-1} **do**
    Project it onto curved PN triangles representing M_I to get the position $\mathbf{p}_i$ and normal $\mathbf{n}_i$ (see Sec. 3.1)
  end for
  for each edge e_j in mesh M_{t-1} **do**
    Calculate the edge midpoints m_j
    Project midpoint m_j onto curved PN triangles representing M_I to obtain the position $\bar{\mathbf{p}}_j$ (see Sec. 3.1)
  end for
  for each vertex v_i in mesh M_{t-1} **do**
    Calculate the optimised position $\hat{\mathbf{p}}_i$ based on PL approximation (see Sec. 3.2)
  end for
  for each vertex v_i in mesh M_{t-1} **do**
    Update $\hat{\mathbf{p}}_i$ to minimise the spring-like energy (see Sec. 3.3)
  end for
  Modify M_{t-1} with updated positions $\hat{\mathbf{p}}_i$ to give M_t
until $|M_t - M_{t-1}| < \varepsilon$ or or $t > t_{\max}$
Return the last M_t as M_T

3.1 Sampling by Projection

A triangulated mesh is a 2D PL discrete surface embedded in 3D space. Since the connectivity is assumed to be pre-determined, we use the concept of a *parameterised mesh* [31] to treat the triangular mesh as a 3-dimensional signal $\mathbf{p}_i = (x_i, y_i, z_i)$ defined over the abstract mesh structure, where each position $\mathbf{p}_i$ corresponds to sampling of a signal, i.e. projection of sample points on the input surface. Normals $\mathbf{n}_i$ can also be treated as a 3-dimensional signal as well. If applicable (see Sec. 1), curved PN triangles [32] are used to estimate the smooth surface described by the input triangular mesh. The coefficients of the Bézier patches in the curved PN triangles can be precomputed for each triangle of M_I. After that the position and normal of a point on the surface can be evaluated using the face that it lies on and the corresponding barycentric coordinate. As described in the pseudocode (details given in the next subsection), an iterative algorithm is used to update the positions of the output mesh M_t and in each iteration, we need the position $\mathbf{p}_i$ and normal $\mathbf{n}_i$ for each mesh vertex v_i, and the position $\bar{\mathbf{p}}_j$ of edge midpoint m_j for each mesh edge e_j (derived from the mesh in the previous iteration M_{t-1}). We find these as follows.

Take the midpoint initially as the average of the positions of two end points of the edge e_j. This point is not usually on the input surface $\tilde{M}_I$. Vertex positions from the previous iterations also in general lie off $\tilde{M}_I$. Since the signal is defined over the input surface, we move these points back on to it. Euclidean projection is used to find the closest point on the input surface in both cases. Finding the projection on M_I can be accelerated using an approximate nearest neighbor (ANN) library [25] to find the mesh triangle containing the footpoint. In later iterations, the face on M_I containing a vertex's footpoint is unlikely to change, and quickly testing this eliminates most ANN queries.

3.2 Approximated Piecewise Linear Optimisation

To estimate the optimal PL approximation to an input surface while avoiding solving expensive equations, we utilise the recent mathematical development in [5]. Given an arbitrary function $f(p)$ defined over some domain $\Omega \subset \mathbb{R}^2$, and some triangulation over the domain with $v_i \in \mathbf{V}$ representing the set of vertices, suppose we wish to find some optimal PL approximation. The space of PL functions can first be represented as a linear combination of hat functions $h_i(p)$, which are 1 on vertex v_i and 0 on every other vertex $v_j \neq v_i$. Finding the optimal PL approximation then is equivalent to finding the best coefficients for such a linear combination that the approximation error between the combined PL function and f is minimised. By using appropriate weights $\mathcal{W}$ to form weighted Sobolev-type inner products $< \cdot, \cdot >_{\mathcal{W}}$, PL hat functions h_i in this metric are mutually orthogonal. Doing this, an arbitrary function defined over the domain can be explicitly decomposed as a combination of PL hat functions, with coefficients $< f, h_i >_{\mathcal{W}} / < h_i, h_i >_{\mathcal{W}}$. This solution is optimal with respect to the weighted Sobolev metric, rather than the Euclidean metric, but experiments show that this is a good approximation in many practical situations. Please refer to [5] for more details.

We adapt the original method for PL approximation from $\mathbb{R}^2$ to a triangulated mesh using the concept of a parameterised mesh. To avoid expensive integration, using discrete approximation and the assumption that the surface can locally be well approximated by a quadric, the updated position of v_i, with respect to one of the triangles $T_{i,s}$ within its 1-ring neighbour $\mathcal{N}(v_i)$ $(s = 1, 2, \ldots, |\mathcal{N}(v_i)|)$ can be explicitly computed [5] as

$$\mathbf{p}_{i,s} = \frac{4}{15}\mathbf{p}_i - \frac{2}{15}\mathbf{p}_j - \frac{2}{15}\mathbf{p}_k + \frac{11}{15}\bar{\mathbf{p}}_{i,j} + \frac{11}{15}\bar{\mathbf{p}}_{i,k} - \frac{7}{15}\bar{\mathbf{p}}_{j,k}, \quad (1)$$

where v_j and v_k are the other two vertices of $T_{i,s}$, $\mathbf{p}_i$, $\mathbf{p}_j$ and $\mathbf{p}_k$ are the projections of vertices v_i, v_j and v_k, respectively and $\bar{\mathbf{p}}_{i,j}$, $\bar{\mathbf{p}}_{i,k}$, $\bar{\mathbf{p}}_{j,k}$ are projections of the midpoints of edges (v_i, v_j), (v_i, v_k) and (v_j, v_k).

To achieve the overall minimal approximation error, the optimal position under PL orthogonal approximation is estimated by an area weighted combinations of these positions:

$$\hat{\mathbf{p}}_i = \frac{\sum_s A(T_{i,s}) \cdot \mathbf{p}_{i,s}}{\sum_s A(T_{i,s})}, \quad (2)$$

where $A(T_{i,s})$ is the area of the triangle $T_{i,s}$.

We initially update vertex v_i to $\hat{\mathbf{p}}_i$ to fully minimise approximation error. After a certain number of iterations, to promote convergence, we restrict the vertex position update to lie along the normal direction, discarding any tangential movement: $\hat{\mathbf{p}}_i \Leftarrow \mathbf{p}_i + ((\hat{\mathbf{p}}_i - \mathbf{p}_i) \cdot \mathbf{n}_i)\,\mathbf{n}_i$. Changing the number of initial iterations will lead to different balance of approximation error and regularity. 20 initial iterations were used in the experiments reported here. Our energy minimisation scheme is similar to that used in [35, 19] for regularity optimisation. Since movement along the normal direction does not change the footpoint, similar convergence can be obtained.

3.3 Regularity Optimisation

As well as providing geometric closeness, a remeshing process is usually expected to optimise regularity in some sense (e.g. the triangles should be close to equilateral). Updating vertices to provide an optimised PL approximation usually decreases regularity. We thus now use a step to improve regularity. The overall algorithm alternates between the previous step and this step to give the final mesh.

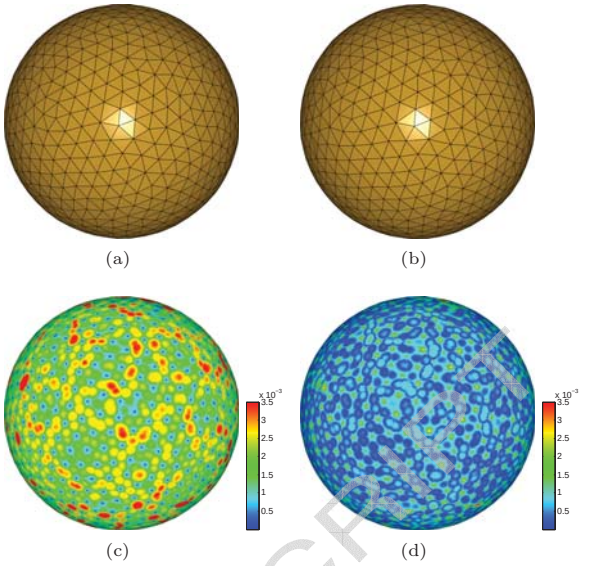

Figure 2: Vertex location optimisation for a sphere: (a) mesh before optimisation; (b) mesh after optimisation; (c) and (d) corresponding error rendering for (a) and (b). Colour indicates error, with blue, green, yellow, orange and finally red indicating increasing errors.

The usual regularity requirement is to provide isotropic remeshing, and we use this as an example in this subsection. To optimise isotropic resampling, a spring-like energy similar to that in [35, 19] is defined between vertices v_i and v_j sharing an edge:

$$E_{ij} = \exp(-\frac{|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j|^2}{2\sigma^2}), \quad (3)$$

where σ can be set to the constant $0.3\sqrt{A/N}$, where A is the overall surface area and N is the number of output vertices [35]. The force from v_j exerted on v_i minimising the spring-like energy satisfies

$$\mathbf{F}_i = \sum_{v_j \in \mathcal{N}(v_i)} (\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j) E_{ij}, \quad (4)$$

where $\mathcal{N}(v_i)$ is the 1-ring neighbor of vertex v_i. The updated equilibrium vertex position is computed as $\hat{\mathbf{p}}_i \Leftarrow \hat{\mathbf{p}}_i + \rho\mathbf{F}_i$, where ρ is a step size. Simply choosing $\rho = 1$ works well for most cases and is used throughout the paper. Incorporating this into the iterative system together with PL approximation leads to a system similar to the traditional projected gradient descent solver. The gradient descent step optimises the energy derived from regularity, while the optimised PL approximation step brings the sampled points back to a state that minimises PL approximation errors.

Unlike many other spring-based approaches for remeshing, we already have the mesh connectivity, so we can use 1-ring neighbours instead of k-nearest neighbours, which is faster and more reliable. As the vertex distribution approaches equilibrium, the force tends to approach zero and the result converges.

As new vertices are allowed to move off the surface, in some vary rare situations (usually in rather bumpy surface regions), it is possible that triangle flips may occur. While geometric closeness and regularity are still well achieved, such flips are not ideal in practical applications. We use a simple and effective strategy to detect and resolve this

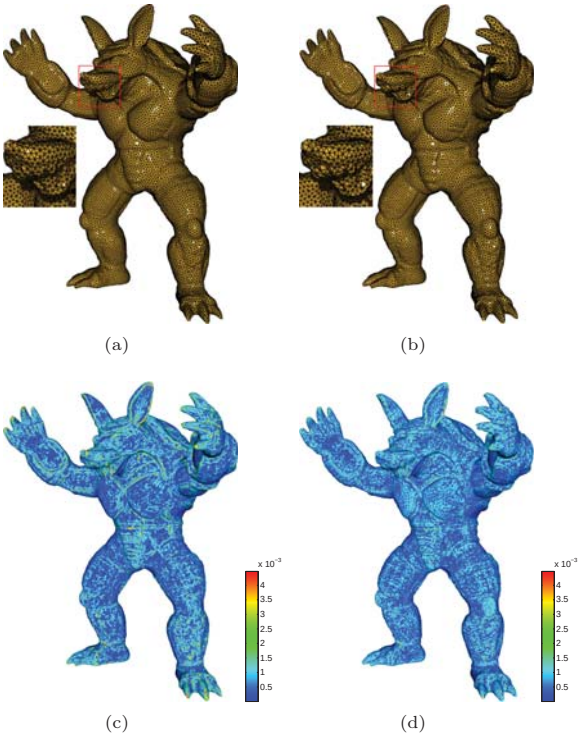

Figure 3: Vertex location optimisation for Armadillo: (a) isotropic mesh before optimisation; (b) mesh after optimisation; (c) and (d) corresponding error rendering for (a) and (b).

problem. At the end of each iteration, we determine if any normal direction has significantly changed before and after optimisation, allowing us to count for each vertex v_i the number C_i of flipped faces adjacent to it. Let n_i be the number of adjustments applied to v_i. We put all v_i with $C_i > 0$ in a priority queue with $C_i - n_i$ as the key, sorted by decreasing key. We successively remove the first element in the queue, reduce the update of this vertex by a half, and update C_j for all the vertices v_j adjacent to v_i as well as C_i and n_i; the priority queue is also updated. This heuristic assumes that vertices related to more flips are typically the key vertices to be corrected. This further allows vertices of similar priority to be adjusted in succession. This simple heuristic is guaranteed to terminate, as when all the problematic vertices revert back to sufficiently close to the input mesh, no flip exists. The strategy to halve the position change is effective in practice as the change is reduced quickly and it is typically sufficient to avoid flips in just a few iterations. The use of this heuristic may cause a slight increase in approximation error. In practice, only very few vertices need to be corrected, thus producing a valid remeshing without losing the overall improvement. For models such as Armadillo in Fig. 3 and Fandisk in Fig. 7, no corrections are needed. Even for models such as Lucy in Fig. 5 with lots of bumpy details, less than 0.1% of the total number of vertices are corrected in each iteration on average. As only a few vertices are affected, this has a negligible effect on the average approximation error. This correction is in spirit similar to mesh inversion prevention in mesh simplification [14]—mesh simplification also allows new vertices to lie off the original surface.

3.4 Optimised Normal Sampling

In certain applications, rendering quality can be improved by storing at each vertex not only the position but the normal as well. Treating normal also as a three dimensional signal defined on the parameterised mesh, our method can be used to find a better estimate of per-vertex normal. Unlike when dealing with positions, we do not apply regularity optimisation but instead normalise the normal vector to be unit length. The normal vector at an arbitrary sample point on the mesh is calculated using a linear combination of normals at the three vertices, with the barycentric coordinates as weights, followed by a normalisation. As demonstrated in the experimental results (Sec. 4), optimised normal sampling tends to reduce approximation error of normals.

3.5 Applications to Other Remeshing

Previous sections assumed isotropic triangular remeshing. Our method can be incorporated into other remeshing approaches. One example is to use geometry images [15], a completely regular representation of geometry. Although geometry images can be considered as a regular grid of quads, for rendering purposes, each quad can be divided into two triangles along the shorter diagonal [15]. Sideband information in geometry images is used to merge split boundaries to form a logically closed shape. Since geometry images pay more attention to approximation error than regularity, optimising the latter is not needed. We will show that our method can also reduce approximation errors for models represented in geometry images.

4 Experimental Results

Various examples are presented in this section demonstrating the effectiveness of the proposed algorithm. Although our algorithm optimises approximation error in a Sobolov metric, our evaluation here is based on Euclidean distance. For an input surface M_I and a remeshed output surface M_O (either M_0 for the initial mesh before optimisation, or M_T after optimisation) represented as a piecewise linear mesh, the position error at $\mathbf{p} \in M_O$ is defined as $e_{\mathbf{p}} = \min_{\mathbf{q} \in M_I} ||\mathbf{p} - \mathbf{q}||_2$ where $\mathbf{p}, \mathbf{q}$ represent all points, not just mesh vertices, of each surface. The maximum and average errors for M_O with respect to M_I can thus be defined as $m_{M_O} = \max_{\mathbf{p} \in M_O} e_{\mathbf{p}}$, and

$$a_{M_O} = \frac{\int_{\mathbf{p} \in M_O} e_{\mathbf{p}} dA(\mathbf{p})}{\int_{\mathbf{p} \in M_O} dA(\mathbf{p})}, \tag{5}$$

where $dA(\mathbf{p})$ is a surface area element at point $\mathbf{p}$. The sphere model in Fig. 2 is a discrete unit sphere. All the other models are scaled to fit to a unit cube. The input surfaces M_I are treated as PN curved meshes in the experiments. They are often much denser than the initial (M_0) and output (M_T) meshes, giving a reasonably accurate representation of the desired surfaces. For both initial and output meshes, the approximation error (w.r.t. the input surface) is calculated and visualised at vertices of a subdivided mesh (with 16 times more vertices) and interpolated for triangle shading. Detailed statistics of various examples are given in Table 1. Our method is applicable to a variety of input triangle meshes. Meshes used for our testing were produced using isotropic remeshing [19] (Figs. 2-4), feature sensitive remeshing [19] (Fig. 5), centroidal Voronoi diagram based remeshing [21] (Fig. 7) and geometry images [15] (Fig. 8). A fixed set of parameters as described above was used in all experiments.

(a) (b) (c) (d) (e)

Figure 4: Vertex location optimisation for the David model. (a) high resolution model; (b) isotropic mesh before optimisation; (c) isotropic mesh after optimisation; (d) isotropic mesh after optimisation with normal; (e) result of (d) with mesh displayed.

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

Figure 5: Vertex location optimisation for Lucy model remeshed with feature sensitive metric: (a) dense mesh ($21K$ vertices) before optimisation; (b) coarse mesh ($13.5K$ vertices) after optimisation; (c)(d) corresponding error rendering for (a) and (b).

The first example is a discrete sphere with about 1000 vertices. The input mesh representing the underlying surface is much denser, with 19200 vertices. As shown in Fig. 2, the shape is only slightly changed with almost the same regularity, but the approximation error is significantly reduced, maximum error by more than 40% and average error reduced by about 75%. We use the same colour coding scheme throughout the paper, with blue, green, yellow, orange and finally red indicating increasing errors. The same scale is used to render both initial and optimised meshes. As a sphere is a simple shape, the exact continuous sphere could be used to calculate the approximation error. We prefer to use the input mesh for error calculation as this is consistent with other experiments. Since the input mesh is sufficiently dense, a similar approximation error reduction result as using the exact sphere for error calculation is obtained. Since a sphere is uniformly non-planar everywhere, a significant reduction of approximation error is observed. Another example is shown in Fig. 3 that optimises an isotropic remeshing with about $20K$ vertices (a). The obtained result after optimisation is shown in (b). Colour coding shows that particularly in feature regions, approximation error is significantly reduced. The average error is reduced by just over 50% and the maximum error by about a third. Note that for these and other examples, as our method aims at reducing some sort of average error (average error in the weighted Sobolev metric), the average error is usually reduced more than the maximum error *percentagewise* while in terms of the *actual* error, the maximum error is usually reduced more than the average error.

We applied our method to the David model produced with isotropic remeshing. The whole model was optimised but only the upper half is shown so that the differences are more visible, as illustrated in Fig. 4. An average 44% reduction in approximation error has been achieved. (a) is the original high resolution reference model, (b) is the initial isotropic remeshing, (c) is the result after position optimisation and (d) is the result after both position and normal optimisation. The average normal deviation in (c) is 5.44 degrees, reduced from 5.91 degrees in (b). By incorporating an optimised normal map, the normal deviation can be further reduced to 4.45 degrees. This is more accurate than direct sampling of normals on the input surface, which has a 4.67 degrees of normal deviation with a stored normal map. Results with increasing visual details can be observed from (b) to (d). Feature regions and regions with highlights show the most noticeable differences. The optimised mesh of the result (d) is presented in (e), which shows isotropy is well preserved.

We have applied our method to the Lucy model, this time remeshing with a feature sensitive metric [19] ($w = 0.05$). The same model was remeshed at 5 different resolutions. Detailed statistics of approximation errors are given in Table 2. The average approximation error reduces with the increasing number of vertices; doubling the number of

Example	num. vertices	A_b	A_a	A_R	M_b	M_a	M_R	α_b	α_a
Sphere (Fig. 2)	1064	19.04	4.85	74.5%	61.54	36.22	41.1%	51.91	52.23
Armadillo (Fig. 3)	20019	8.56	4.16	51.4%	75.85	51.26	32.4%	51.37	48.16
David (Fig. 4)	21110	4.30	2.41	44.0%	159.5	133.7	16.2%	50.73	48.40
Fandisk (Fig. 7)	2003	1.96	1.47	25.0%	72.14	61.28	15.1%	51.94	52.09
Dragon (Fig. 8)	65×65	31.88	20.98	34.2%	342.1	286.4	16.3%	–	–

Table 1: Comparisons of approximation errors. A: average error. M: maximum error. α: average minimum angle (degrees). b: error before optimisation; a: error after optimisation; R: percentage reduction in error. All values in units of 10^{-4}. The sphere model is a discrete unit sphere and all the other models are scaled to fit in a unit cube.

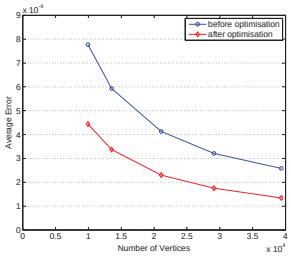

Figure 6: Average approximation errors for the Lucy model remeshed using different resolutions in Fig. 5 before (blue circles) and after (red diamonds) optimisation. See also Table 2.

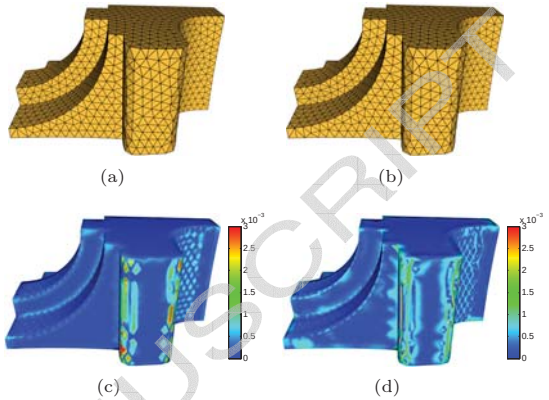

Figure 7: Vertex location optimisation for the fandisk model: (a) isotropic mesh before optimisation; (b) mesh after optimisation; (c)(d) corresponding error rendering for (a) and (b).

vertices roughly halves the approximation error. Using our approach, the average approximation error is reduced by more than 40%, compared with the mesh with the same resolution before optimisation. It can also be observed from the curve that more than 40% fewer vertices can be used for the same approximation error. A comparative rendering is shown in Fig. 5 which demonstrates that using our approach, a coarse remesh (b) with only $13.5K$ vertices actually leads to lower average approximation error than an unoptimised dense mesh with $21K$ vertices. Errors are colour-coded in (c) and (d) (lower error is darker blue). As illustrated in Fig. 6, our method is consistently effective at reducing approximation errors for both coarse and fine output meshes alike.

Our method works well also for CAD models; these may have sharp features. A typical example (Fandisk) is shown in Fig. 7. For the input isotropic remesh (a), we apply our vertex position optimisation to obtain the result (b) which shows that sharp features and isotropy are well preserved. The error distribution is shown as colour coding in (c) and (d). For relatively flat regions, the approximation error is small before optimisation and it is not surprising that our method cannot improve these regions much. For planar regions, no systematic error exists for traditional sampling. It can be observed that the approximation error for curved regions is significantly reduced. Since the fandisk model contains quite large planar regions, the average error is reduced by only 25.0% and the maximum error is reduced by 15.1% (see Table 1). Like other examples, vertex shift is relatively small and the original connectivity is well suitable after optimisation.

Our method mainly aims to reduce the average (mean) error (as in Eqn. 5). Although we explicitly minimise the error in a weighted Sobolev metric, our method effectively achieves this goal. As demonstrated in Tables 1 and 2, the average errors are typically reduced by about 40% after optimisation. The maximum errors are reduced by 15%–41% in these examples. Comparatively, since [34] minimises the Hausdorff distance, the maximum error can be reduced more significantly (e.g. about 50%), however, the mean error only reduces by about 10%–15% and sometimes may even increase. Our method can in general preserve the isotropy of the meshes. One indication is to use the average minimum angle of triangles (ideally 60° for isotropic meshes). As shown in Table 1, the average minimal angle may increase or decrease slightly but is generally close to that of the initial mesh.

We applied our method to geometry images of the dragon model in [15] with high resolution of 257×257 (a)(b) (as input geometry), but down-sampled to 65×65 to obtain a low-resolution representation. As shown in Fig. 8, the initial low resolution representation (c) has large errors, but by using our approach, the generated geometry image (d) reduces approximation error significantly with the same resolution.

The experiments were carried out on an Intel Core2Quad 2.66GHz computer. The algorithm is quite fast. Our algorithm is iterative and in practice about 30 iterations are sufficient for most examples. Assume the numbers of vertices in the input mesh and the output mesh are denoted as N_I and N_O respectively. The time taken in each iteration of our method is $O(N_O \log N_I)$. Thus the running time is proportional to the size of the output mesh and grows very slowly with the increasing size of the input. To remesh the David model in Fig. 4 with about $217K$ vertices ($434K$ triangles) in the input mesh and about $21K$ vertices ($42K$ triangles) in the output, each iteration took only about 0.1s and the whole algorithm took less than 3s. Comparatively, in each iteration, [34] needs to solve a linear system the size of which is proportional to $N_I + N_O$. The performance reported in the paper shows that for a mesh with $173K$ vertices, each iteration takes 26.5s. Our method is tens or even hundreds of times faster, especially for cases with relatively dense input meshes.

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

Figure 8: Vertex location optimisation for the Dragon geometry image. (a) high resolution geometry image (b) corresponding shape (c) approximation error with initial low resolution geometry image; (d) approximation error with optimised low resolution geometry image.

Vertices	9935	13534	21076	29102	39353
A_b	7.77	5.93	4.13	3.21	2.58
A_a	4.44	3.37	2.30	1.75	1.34
A_R	42.9%	43.2%	44.3%	45.5%	48.1%

Table 2: Remeshing of Lucy model using feature sensitive metric. A_b, A_a: average error before, after optimisation; A_R: percentage reduction in error. All values in units of 10^{-4}.

5 Conclusions

In this paper, we have given a fast novel vertex location optimisation algorithm for remeshing based on Sobolov products. A typical reduction in average approximation error of 40% is achieved for various models. If a fixed accuracy is desired, many fewer vertices suffice, leading to cost saving for various applications ranging from rendering to simulation. One limitation of our method is that we cannot guarantee the optimised mesh to be free of self-intersections. As vertices only move slightly off the surface, this does not seem to cause visible problems in most cases. We have focused on improving piecewise linear approximations in triangular remeshing. Quad remeshing algorithms following surface features also tend to have almost planar faces. We would like to extend our current method to handle such remeshing. Although piecewise linear primitives are widely used due to their simplicity, another possible direction of reducing the number of elements in certain applications is to use higher order primitives. We expect to investigate this in future.

Acknowledgements

We would like to thank anonymous reviewers for their helpful suggestions. The models in this paper are courtesy of AIM@SHAPE Repository and Stanford Computer Graphics Laboratory. We thank Bruno Lévy and Yang Liu for the remeshed fandisk model and Xianfeng Gu for geometry images of the dragon model. This work was supported by One Wales Research Institute for Visual Computing (RIVIC) and EPSRC Grant No. EP/I000100/1.

References

[1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. In *ACM SIGGRAPH*, pages 485–493, 2003.

[2] P. Alliez, E. C. de Verdière, O. Devillers, and M. Isenburg. Isotropic surface remeshing. In *IEEE Shape Modeling International*, pages 49–58, 2003.

[3] P. Alliez, E. C. de Verdière, O. Devillers, and M. Isenburg. Centroidal Voronoi diagrams for isotropic surface remeshing. *Graph. Models*, 67(3):204–231, 2005.

[4] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene. Recent advances in remeshing of surfaces. In *Shape Analysis and Structuring*, pages 53–82, 2008.

[5] A. App and U. Reif. Piecewise linear orthogonal approximation. *SIAM J. Numerical Analysis*, 48(3):840–856, 2010.

[6] M. Attene, B. Falcidieno, J. Rossignac, and M. Spagnuolo. Edge-sharpener: recovering sharp features in triangulations of non-adaptively re-meshed surfaces. In *Symp. Geometry Processing*, pages 62–69, 2003.

[7] J.-D. Boissonnat and S. Oudot. Provably good sampling and meshing of surfaces. *Graph. Models*, 67:405–451, 2005.

[8] D. Bommes, H. Zimmer, and L. Kobbelt. Mixed-integer quadrangulation. *ACM Trans. Graph.*, 28(3):Article No. 77, 2009.

[9] M. Botsch and L. Kobbelt. A remeshing approach to multiresolution modeling. In *Symp. Geometry Processing*, pages 189–196, 2004.

[10] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM Trans. Graph.*, 23(3):905–914, 2004.

[11] T. K. Dey, G. Li, and T. Ray. Polygonal surface remeshing with Delaunay refinement. In *International Meshing Roundtable*, pages 343–361, 2005.

[12] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart. Spectral surface quadrangulation. *ACM Trans. Graph.*, 24(3):1057–1066, 2006.

[13] S. Dong, S. Kircher, and M. Garland. Harmonic functions for quadrilateral remeshing of arbitrary manifolds. *Computer Aided Geometric Design*, 22(5):392–423, 2005.

[14] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *ACM SIGGRAPH*, pages 209–216, 1997.

[15] X. Gu, S. J. Gortler, and H. Hoppe. Geometry images. *ACM Trans. Graph.*, 21(3):355–361, 2002.

[16] F. Kälberer, M. Nieser, and K. Polthier. QuadCover–surface parameterization using branched coverings. *Computer Graphics Forum*, 27(3):375–384, 2007.

[17] L. P. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel. A shrink wrapping approach to remeshing polygonal surfaces. *Computer Graphics Forum*, 18(3):119–130, 1999.

[18] Y.-K. Lai, L. Kobbelt, and S.-M. Hu. Feature aligned quad dominant remeshing using iterative local updates. *Computer-Aided Design*, 42(2):109–117, 2010.

[19] Y.-K. Lai, Q.-Y. Zhou, S.-M. Hu, J. Wallner, and H. Pottmann. Robust feature classification and editing. *IEEE Trans. Vis. Comp. Graph.*, 13(1):34–47, 2007.

[20] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. MAPS: multiresolution adaptive parameterization of surfaces. In *ACM SIGGRAPH*, pages 95–104, 1998.

[21] B. Lévy and Y. Liu. l_p centroidal voronoi tesselation and its applications. *ACM Trans. Graph.*, 29(4):Article No. 119, 2010.

[22] Y. Liu, H. Pottmann, J. Wallner, Y.-L. Yang, and W. Wang. Geometric modeling with conical meshes and developable surfaces. *ACM Trans. Graph.*, 25(3):681–689, 2006.

[23] M. Marinov and L. Kobbelt. Direct anisotropic quad-dominant remeshing. In *Pacific Graphics*, pages 207–216, 2004.

[24] M. Marinov and L. Kobbelt. A robust two-step procedure for quad-dominant remeshing. *Computer Graphics Forum*, 25(3):537–546, 2006.

[25] D. Mount and S. Arya. ANN: A library for approximate nearest neighbor searching, 2005.

[26] J. Peters. Efficient one-sided linearization of spline geometry. In *Mathematics of Surfaces*, pages 297–319, 2003.

[27] E. Praun and H. Hoppe. Spherical parametrization and remeshing. In *ACM SIGGRAPH*, pages 340–349, 2003.

[28] N. Ray, W. C. Li, B. Lévy, A. Sheffer, and P. Alliez. Periodic global parameterization. *ACM Trans. Graph.*, 25(4):1460–1485, 2006.

[29] V. Surazhsky, P. Alliez, and C. Gotsman. Isotropic remeshing of surfaces: a local parameterization approach. In *International Meshing Roundtable*, pages 215–224, 2003.

[30] V. Surazhsky and C. Gotsman. Explicit surface remeshing. In *Symp. Geometry Processing*, pages 20–30, 2003.

[31] G. Taubin. A signal processing approach to fair surface design. In *ACM SIGGRAPH*, pages 351–358, 1995.

[32] A. Vlachos, J. Peters, C. Boyd, and J. L. Mitchell. Curved PN triangles. In *Symp. Interactive 3D Graphics*, pages 159–166, 2001.

[33] D. Walton and D. Meek. A triangular G^1 patch from boundary curves. *Computer-Aided Design*, 28(2):113–123, 1996.

[34] T. Winkler, K. Hormann, and C. Gotsman. Mesh massage: A versatile mesh optimization framework. *The Visual Computer*, 24(7–9):775–785, 2008.

[35] A. Witkin and P. Heckbert. Using particles to sample and control implicit surfaces. In *ACM SIGGRAPH*, pages 269–277, 1994.

[36] D.-M. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. In *Symp. Geometry Processing*, pages 1445–1454, 2009.

Highlights

- Efficiently reducing remeshing errors by allowing vertices to stay off the surface.
- Explicit use of piecewise linear nature of meshing in vertex optimisation.
- Using the weighted Sobolev-type inner products to avoid solving linear equations.
- The approach can be combined with various mesh generation methods.