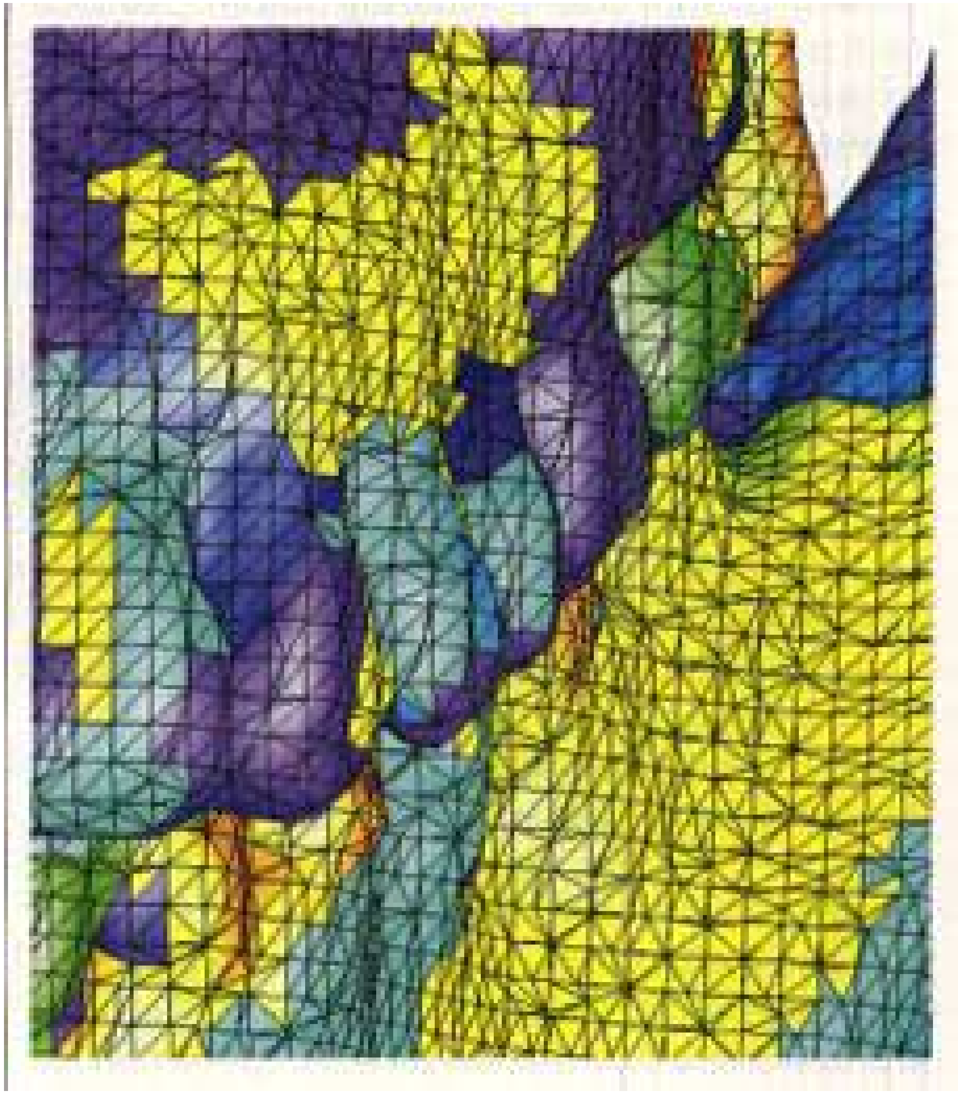
Superfaces: Polyhedral Mesh Simplification with Bounded Error

Alan D. Kalvin

Russell H. Taylor

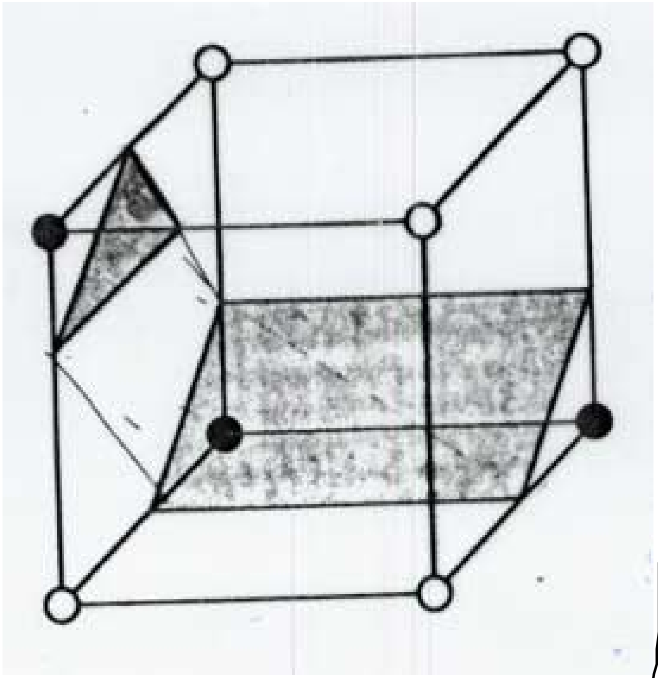
Problems with Polyhedra

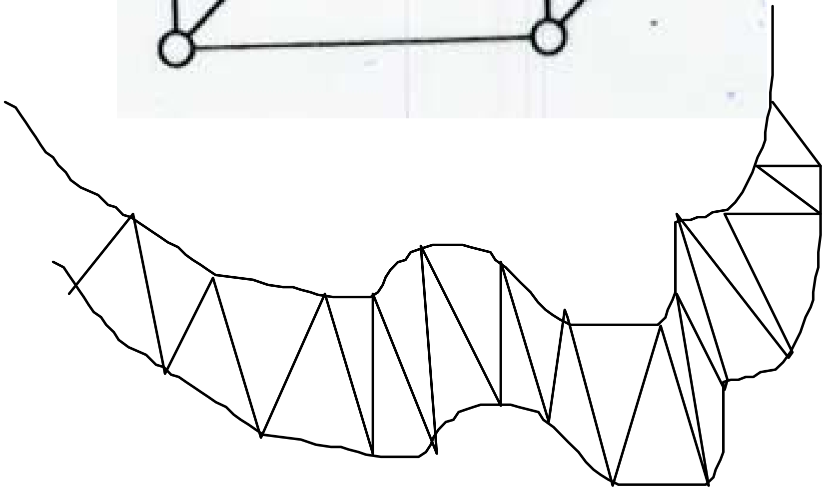
- Common 3D image to surface boundary reconstruction algorithms produce many small faces
- Shapes are complex
- Voxel-based methods cannot span > 1 voxel
- Contour tiling methods cannot span > 2 slices



Problems with Polyhedra

- Common 3D image to surface boundary reconstruction algorithms produce many small faces
- Shapes are complex
- Voxel-based methods cannot span > 1 voxel
- Contour tiling methods cannot span > 2 slices





Superfaces Algorithm: Summary

Automatic simplification of complex polyhedral models with bounded error:

- Applied to biomedical models derived from CT
- Useful on other models with similar characteristics
- Typical performance on 350K triangle skull model and 1 voxel diameter error bound:
  - 4:1 reduction in triangle count
  - 7:1 reduction data structure size
  - 6 minutes on (slow) RS/6000

Superfaces Algorithm: Summary

Automatic simplification of complex polyhedral models with bounded error:

- Applied to biomedical models derived from CT
- Useful on other models with similar characteristics
- Typical performance on 350K triangle skull model and 0.5 pixel units error bound:
  - 3:1 to 6:1 reduction in triangle count
  - Mean approx. error 0.05-0.09 pixel units
  - Run time 8.5 to 9.5 minutes on (slow) RS/6000

11 Original skull model (349,792 triangles).

12 Simplified skull (a) mesh and (b) color-coded approximation errors in pixel units: $\varepsilon = 0.5$ (36.60 percent of original triangles).

13 Simplified skull (a) mesh and (b) color-coded approximation errors in pixel units—with aggressive border straightening: $\varepsilon = 0.5$ (15.58 percent of original triangles).

Algorithm Properties

- Fast, “greedy” method
- Preserves geometric error bound
- Preserves topology
- Simplified model is imbedded in original
- Applicable to any polyhedral model
- No *a priori* knowledge of surface required

Related Work

- Schmidt, Barsky, Du (1986)
  - top-down refinement of surface of bicubic patches
  - for objects $z=f(x,y)$
- Kalvin (1991)
  - adaptive merging of redundant faces
- Schroeder, Zarge, Lorensen (1992)
  - “triangle decimation” to reduce size by given percentage
- Turk (1992)
  - retiling surface by triangulating new set of vertices
- Rossignac & Borel (1993)
  - multi-resolution 3D approximations

Related work

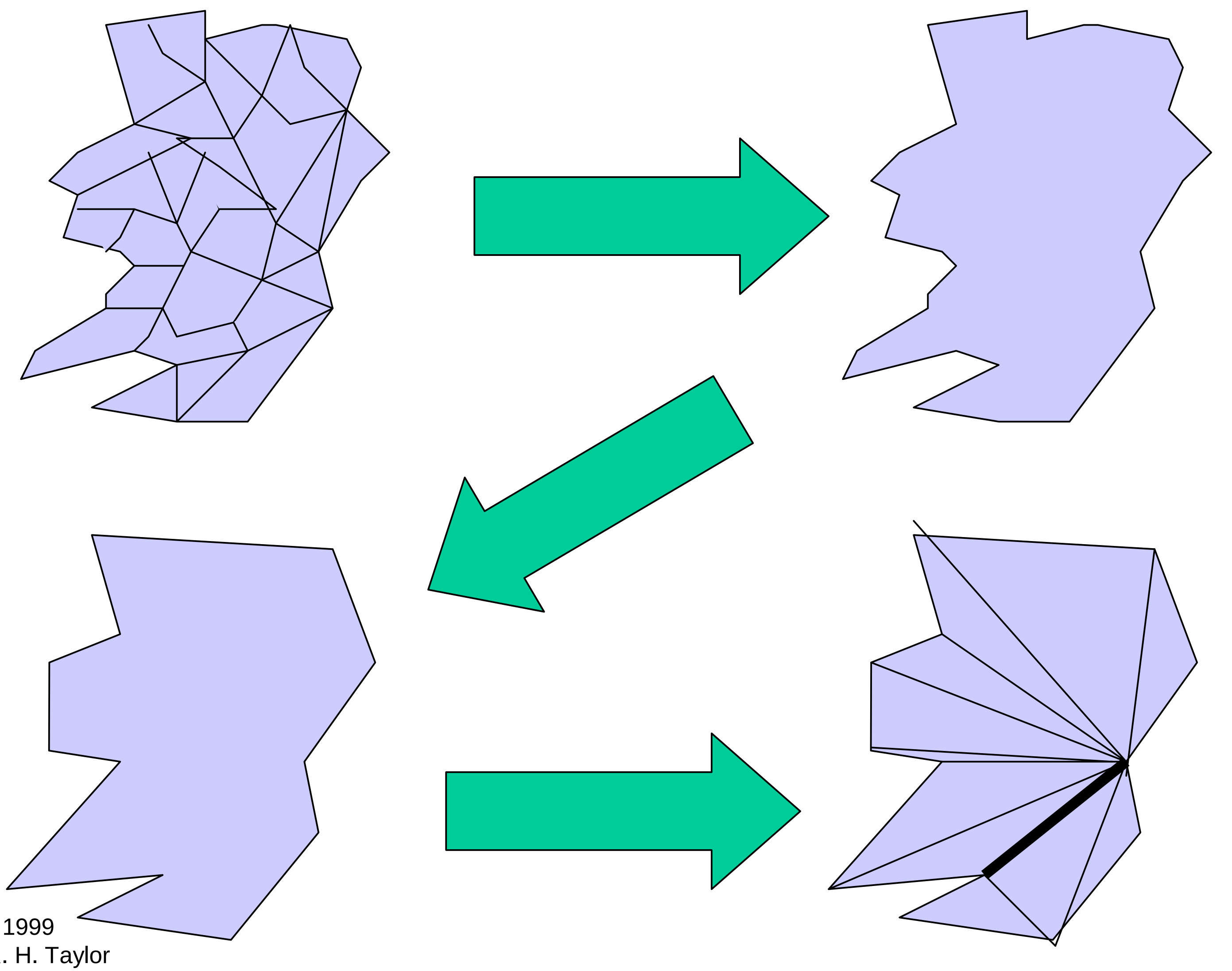
- Cutting, et. al (1991)
  - Registration to anatomical atlas
- Bloomenthal (1988)
- Hall and Warren (1990)
- Ning and Hasselink (1991)
- Gueziec (1996)
- Cohen (1997?)

- many more

Algorithm Outline

- **Phase 1: Merge faces into superfaces**
  - **Greedy, bottom-up algorithm**
  - **Runs in O(n) time, where n=number of faces**
- **Phase 2: Straighten borders**
  - **Create “superedges”**
  - **Several variations with different degrees of agressiveness**
- **Phase 3: Pick triangulation points**
  - **Usually, triangulation is not done explicitly**

Algorithm outline



Phase 1: Greedy Merging

1. Pick a seed face to start a new superface
   - Options include random choice, pincushion search, etc.

2. Keep adding adjacent faces to the superface as long as can find a feasible approximating plane & meet some other technicalities

3. Repeat steps 1 & 2 as long as there are faces not assigned to superfaces

Phase 1: Quasi-planar merging

Consider the approximating plane P of a superface:

$$P = \{(x, y, z) | ax + by + z = d\}$$

in some local coordinate system of the face. The parameters (a, b, d) thus represent P.

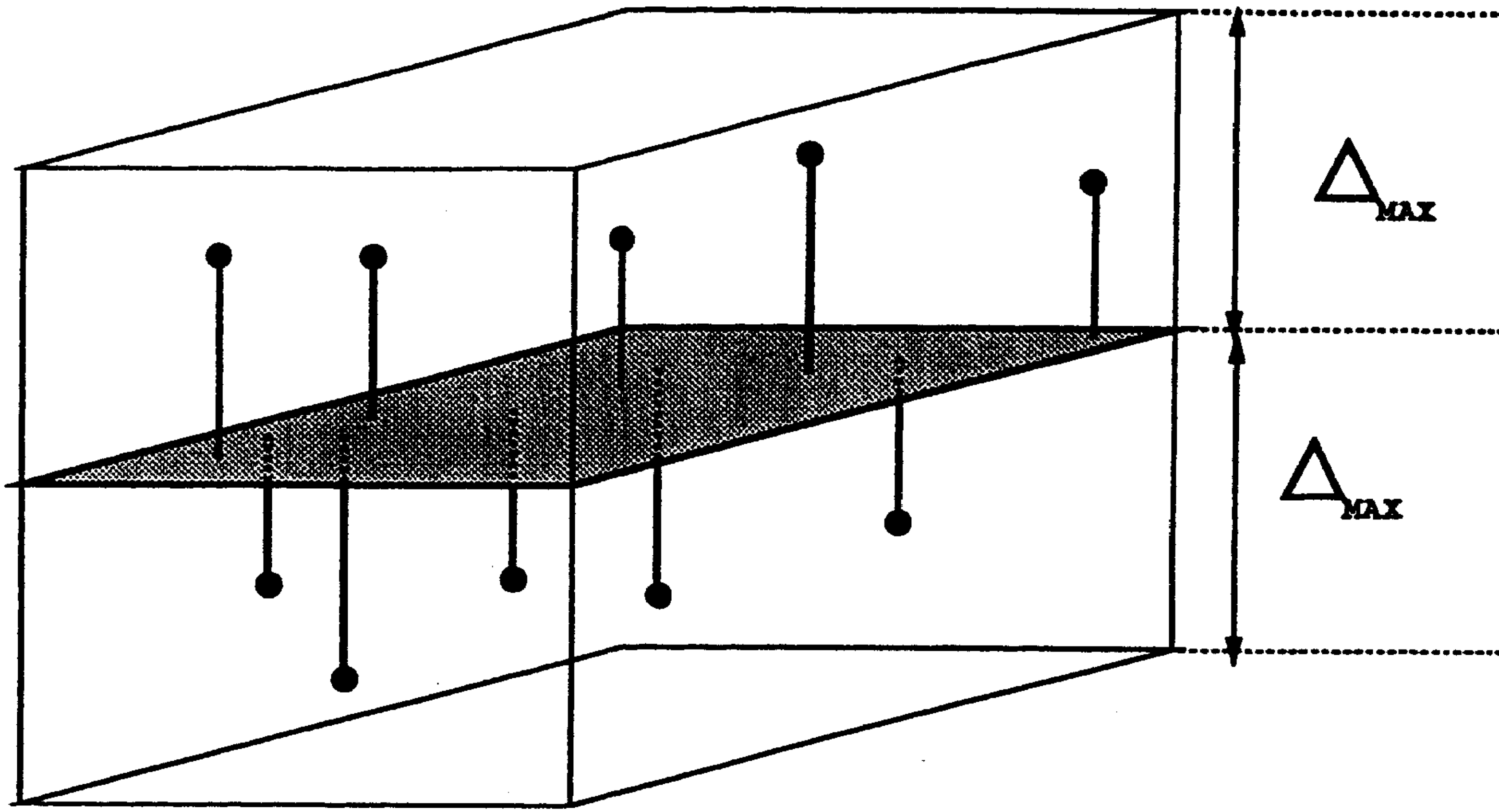
Note the duality: (a, b, d) constrains (x, y, z), but (x, y, z) also constrains (a, b, d).

In general, (a, b, d) will obey the bounded approximation constraints

$$-\varepsilon - z \leq ax + by - d \leq \varepsilon - z$$

and some other (linear) constraints to be discussed later

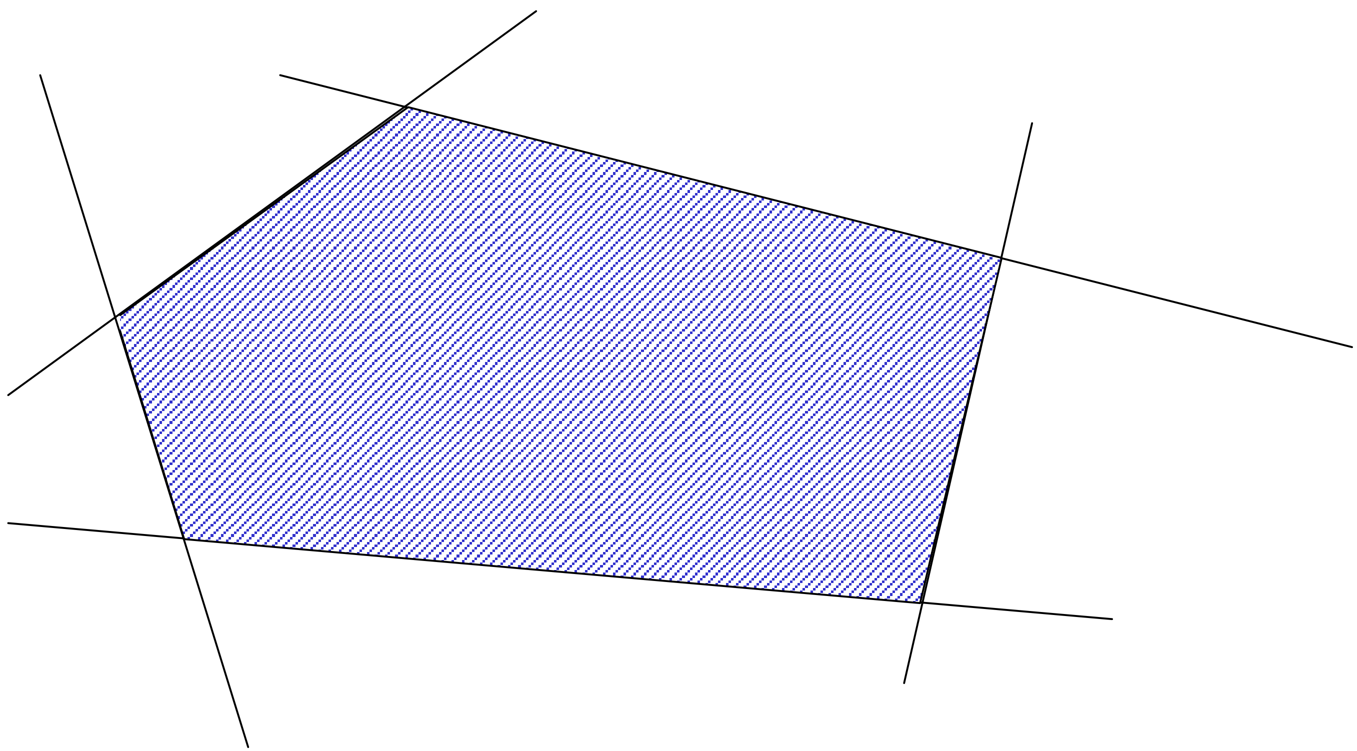
Approximating Plane



Set of feasible approximating planes

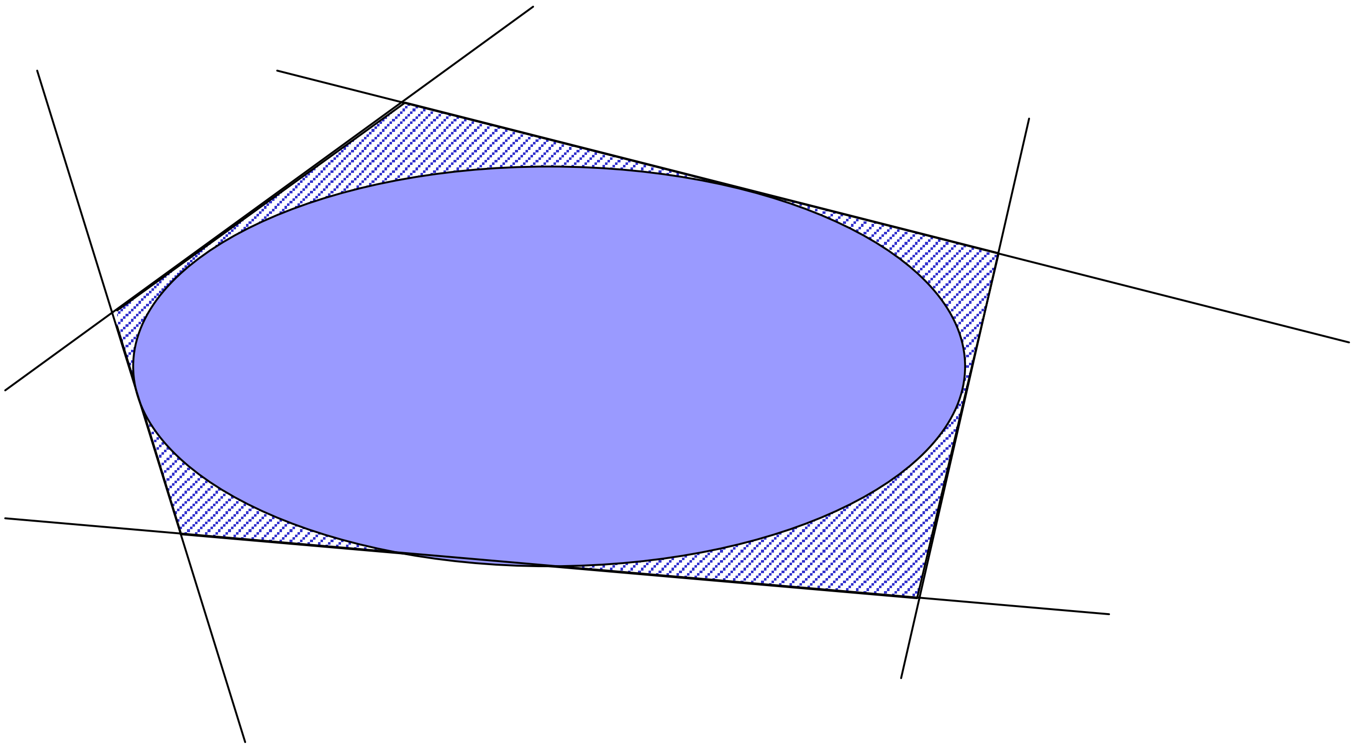
The set of feasible planes is described by a polytope

$$E = \left\{ (a,b,d) \mid C \bullet (a,b,d)^{\mathrm{T}} \leq \bar{g} \right\}$$



Set of feasible approximating planes

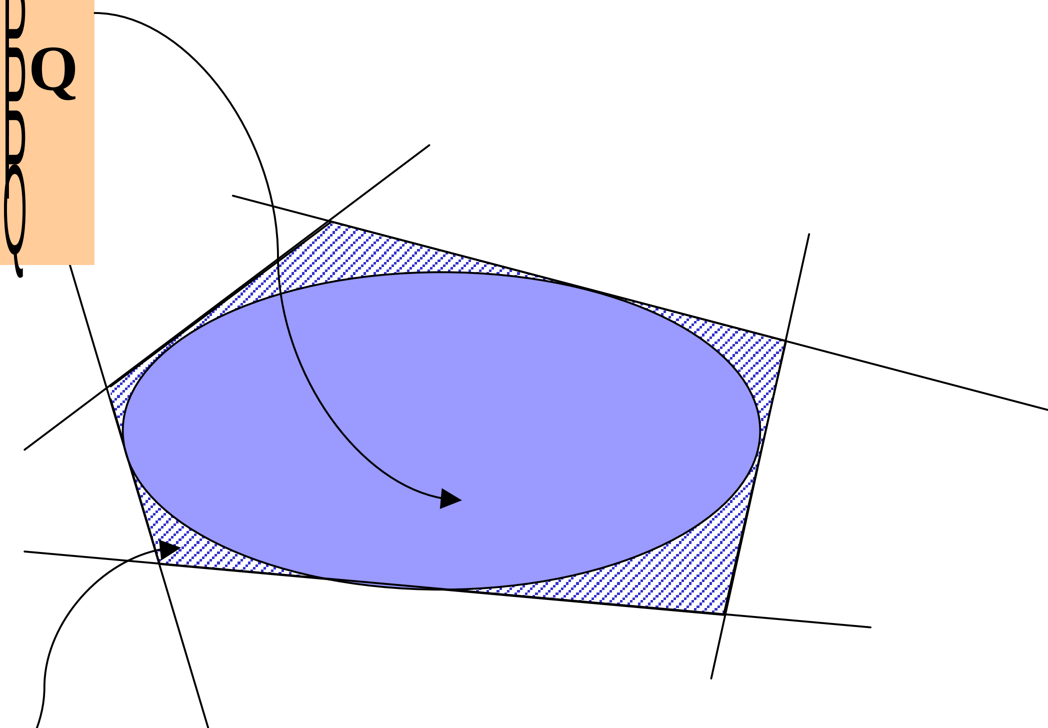
Conservatively approximate *E* by an ellipsoid



Ellipsoidal approximation

$$E^* = \left\{ \mathbf{k} \,|\, (\mathbf{k}-\mathbf{k}_0)^{\mathrm{T}} \mathbf{M} (\mathbf{k}-\mathbf{k}_0) \le 1 \right\}$$

$$\mathbf{M} = \mathbf{Q}^{\mathrm{T}} \begin{bmatrix} \beta_1{}^2 & & & \\ & \beta_2{}^2 & & \\ & & \ddots & \\ & & & \beta_n{}^2 \end{bmatrix} \mathbf{Q}$$



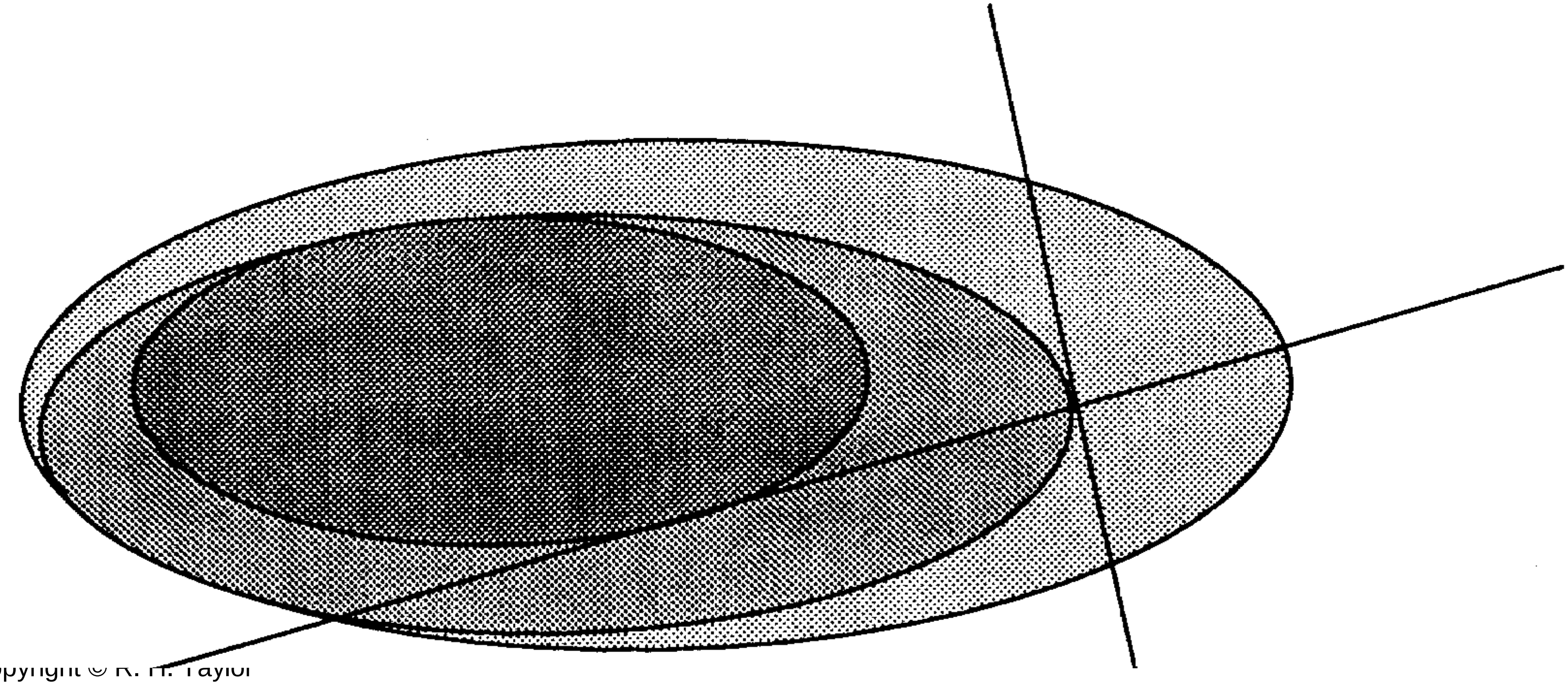
$$E = \left\{ \mathbf{k} \,|\, \mathbf{C} \bullet \mathbf{k} \le \mathbf{g} \right\}$$

Growing a Superface

1. Select f_b face on current perimeter

   $\Rightarrow f_b$ generates new linear constraints $\{C_j\}$

2. Compute Σ' = **linear-time** adjustment of ellipsoid Σ based on $\{C_j\}$



Growing a Superface

3. **if** $\Sigma' \neq \{\}$ **then** f_b satisfies merging criteria
   - merge f_b into superface
   - $\Sigma \leftarrow \Sigma'$
4. Iterate above until:
   - no more acceptable faces to merge
   - bad aspect ratio

Merging Rules

1. Planarity rule:
   - All vertices of f_b must be within bounded distance of approximating plane p
     - **2 constraints:** $\|(a,b,1,-d).(v_x, v_y, v_z, 1)\| \leq \Delta_{max}$
2. Face-axis rule:
   - orientation $f_b \approx$ orientation p
     - **constraint:** $an_x + bn_y \geq \cos(\theta_{max}) - n_z$
     - (n_x, n_y, n_z) = outward-facing normal of f_b
3. No-foldover rule:
   - f_b cannot "tuck-under" superface F
   - $\forall v \in f_b$, v outside F_p (projection of perim F into p)
     - **constraint:** $aK_1 + bK_2 \leq K_3$

Gerrymandering check

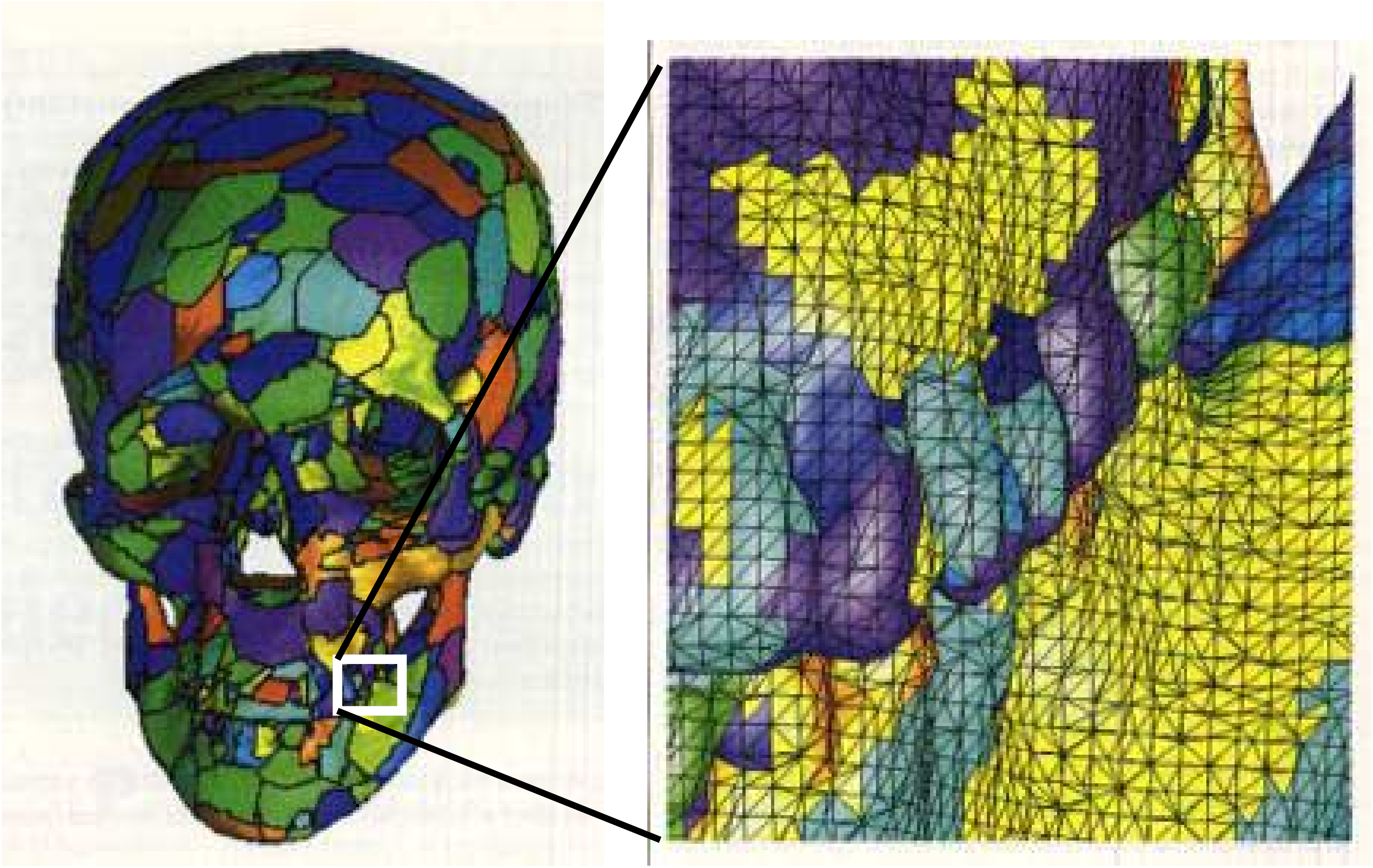
Optional constraint

- to prevent "irregular" shaped superfaces
- stop growing if Irreg(F) > $Irreg_{max}$
- simple estimate: Irreg(F) = **perim²/area**

Polyhedra from Alligator algorithm

- perim ≈ no. edges
- area ≈ no. faces
- Irreg(F) needs 3 floating point ops.

Phase 1 output



Phase 2 Strategy

1. Replace edges between adjacent superfaces with single superedge
2. Recursively split the big superedge into smaller edges until every boundary vertex in any edge merged into superedge is within a bounded distance of one of adjacent superfaces
3. Repeat until done

Alternative: Merge agressively and check all subsumed vertices. Only split if boundedness condition is violated.

Phase 2

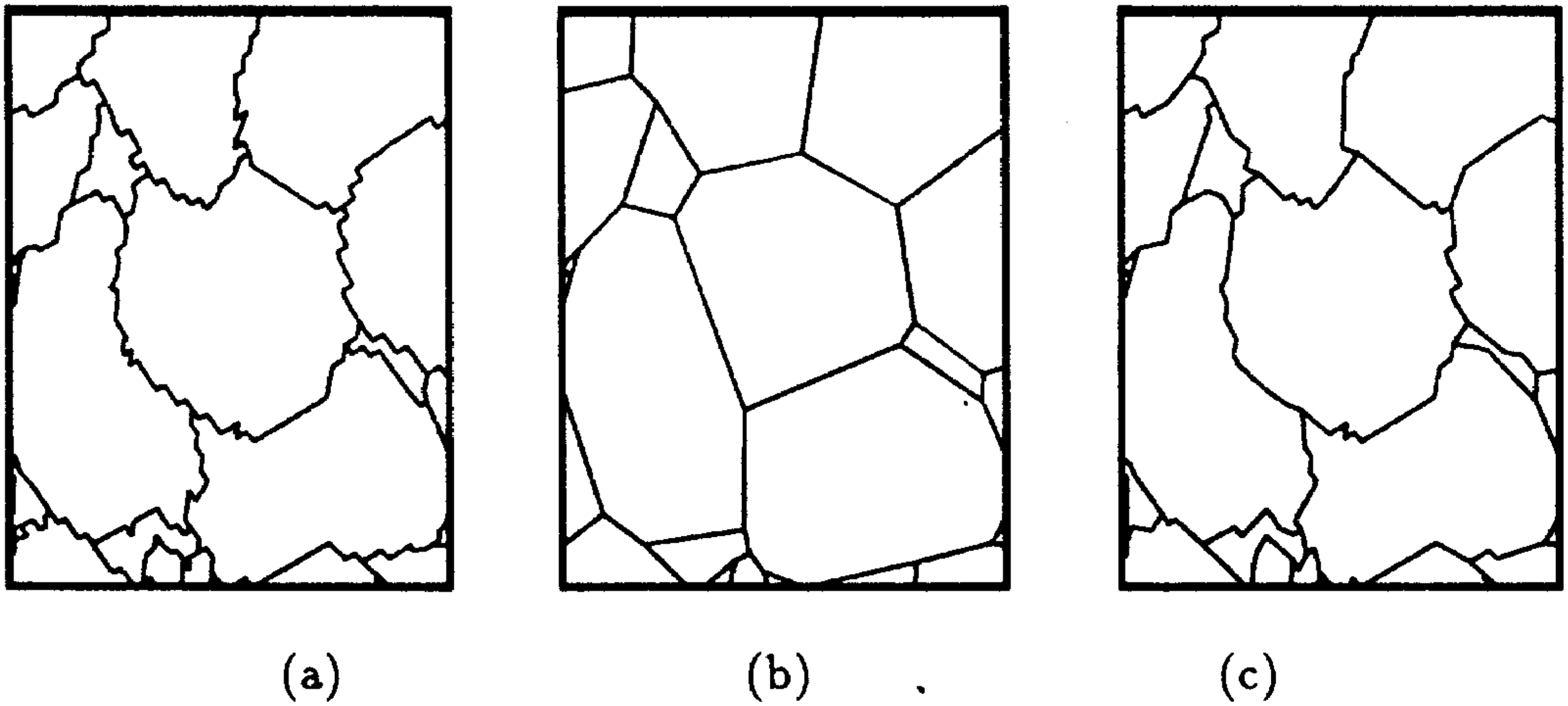


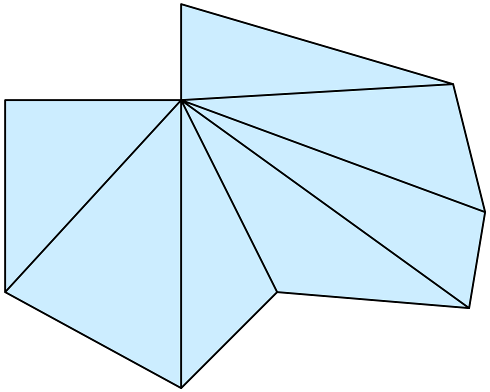
Figure 5: Superface borders (a) before straightening, (b) after edge merging, (c) after edge splitting.

Phase 3: Triangulation

1. Project superface perimeter into the nominal approximating plane

2. Decompose 2D polygon into <u>star polygons</u>

3. (Implicitly) triangulate star polygons



Star Polygon Decomposition

Method 1 (fast, try first):

1. Determine conservative approximation for the polygon kernel of projection of superface onto its approximating plane.
2. Pick point in polytope.

Method 2 (if that fails)

1. Decompose superface into monotone polygons.
2. Determine triangulation point for each monotone polygon.

Star Polygon Decomposition

- Avis & Toussaint (1981) - $O(n \log n)$
  - efficient
  - does not attempt to limit number of star polygons
  - does not handle polygons with holes
- Keil (1985) - $O(n^5 k^2 \log n)$
  - minimizes number of star polygons
  - does not handle polygons with holes
- What we did - $O(n^2)$
  - Attempts to limit number of star polygons
  - Can handle polygons with holes
  - First step is decomposition into monotone polygons
  - Second step is conversion of monotone polygons to star polygons

Monotone polygons & Star Decomp.

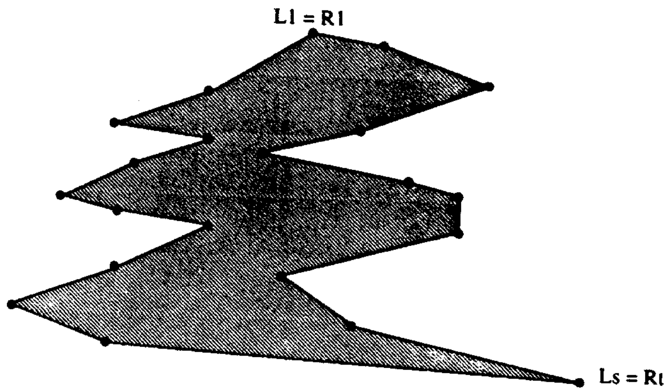


Figure 6: Monotone polygon P.

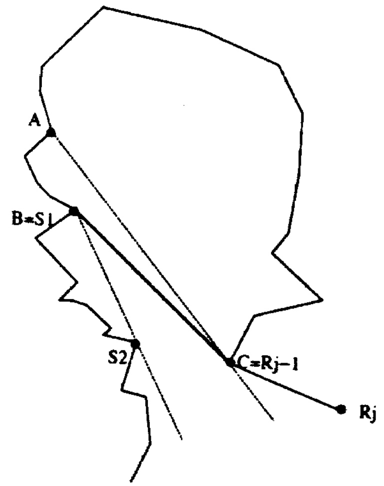


Figure 7: Creating internal diagonal $[B, C]$.

Results for skull

error bound ε	approx. error		triangles		running time (m:ss)
	mean	max.	count	% of original	
0.5	0.0544	0.4723	128,040	36.60	9:52
1.0	0.1289	0.9231	78,002	22.30	8:03
1.5	0.2017	1.4387	50,442	14.42	7:34
2.0	0.2559	1.8690	37,438	10.70	6:41
3.0	0.3088	2.6119	28,388	8.12	6:26
4.0	0.3358	2.7684	24,170	6.91	6:00

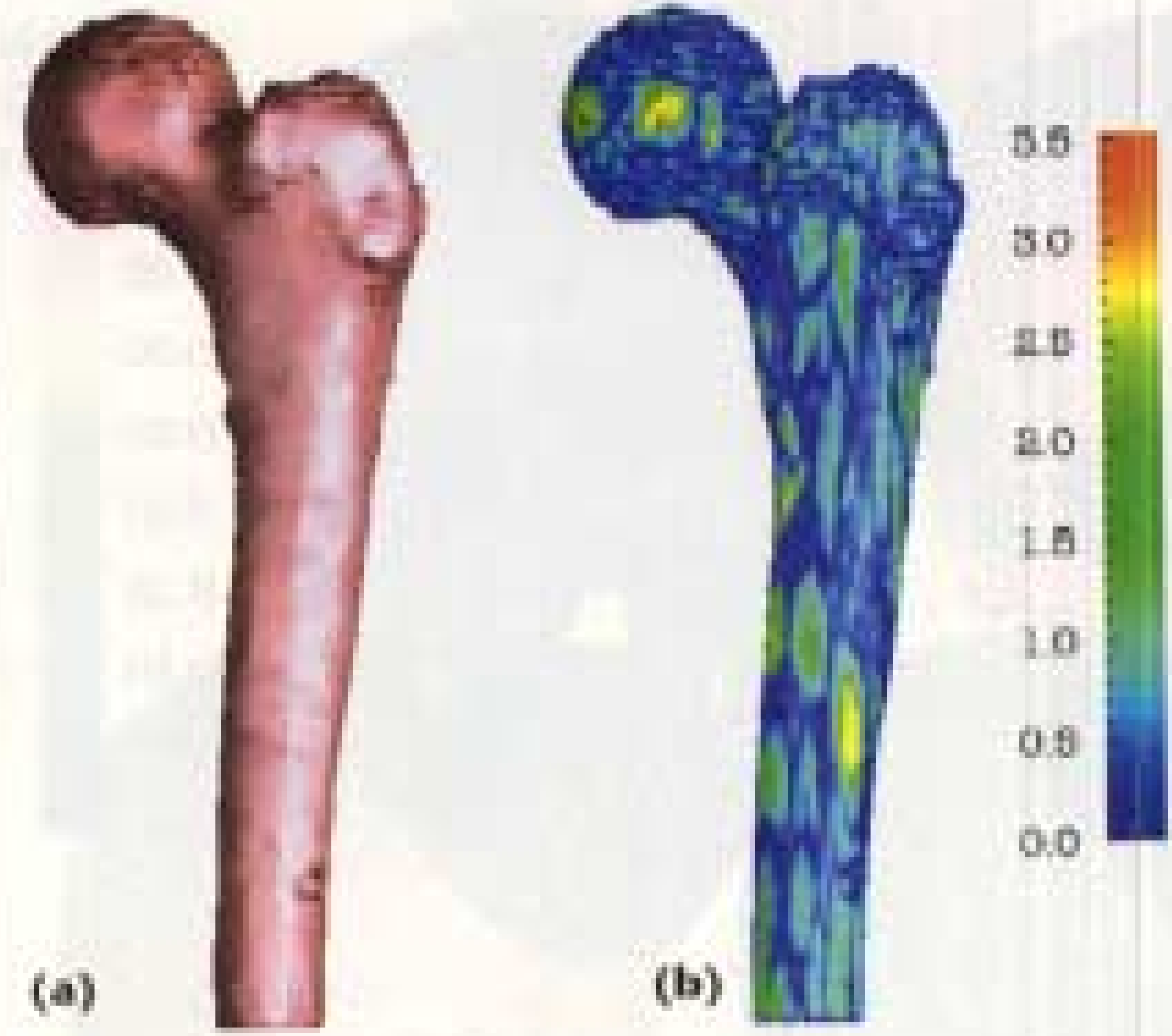
Table 1: Results of simplifying the skull mesh of 349,792 triangles.

error bound ε	approx. error		triangles		running time (m:ss)	vertices above ε limit	superfaces to adjust / total superfaces
	mean	max.	count	% of original			
0.5	0.0947	1.4240	53,790	15.38	8:49	31	18 / 14,403
1.0	0.2187	1.3973	23,704	6.78	7:12	36	9 / 5,626
1.5	0.3402	2.3713	15,470	4.42	6:28	56	7 / 3,606
2.0	0.4523	5.8117	11,994	3.43	6:20	184	5 / 2,738
3.0	0.5984	3.3584	9,820	2.81	6:03	19	1 / 2,299
4.0	0.6714	3.6544	8,934	2.55	5:52	0	–

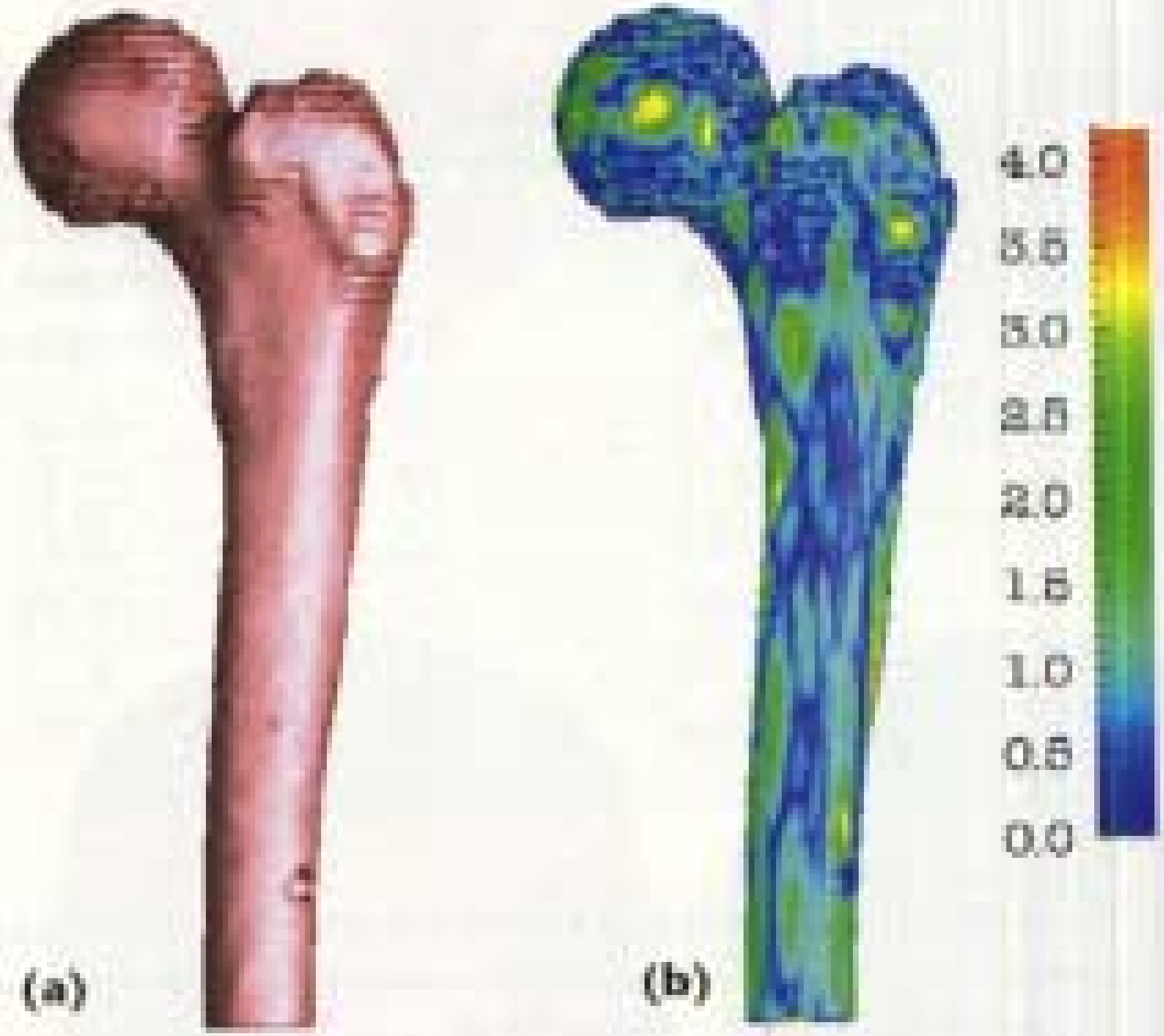
Table 2: Results of simplifying the skull mesh of 349,792 triangles – with aggressive border straightening).

16 Original femur model (179,916 triangles).

17 Simplified femur (a) mesh and (b) color-coded approximation errors in pixel units: ε = 4.0 (12.14 percent of original triangles).



18 Simplified femur (a) mesh and (b) color-coded approximation errors in pixel units—with aggressive border straightening: $\varepsilon = 4.0$ (4.41 percent of original triangles).



Results for femur

error bound ϵ	approx. error mean	approx. error max.	triangles count	triangles % of original	running time (m:ss)
0.5	0.0378	0.4826	97,010	53.92	4:54
1.0	0.1060	0.8821	66,318	36.86	3:31
1.5	0.1708	1.3265	46,748	25.98	3:10
2.0	0.2263	1.7860	36,018	20.02	2:54
2.5	0.2745	2.2530	30,766	17.10	2:45
3.0	0.3196	2.7049	26,832	14.91	2:36
4.0	0.3921	3.5481	21,840	12.14	2:28

Table 3: Results of simplifying the femur mesh of 179,916 triangles.

error bound ϵ	approx. error mean	approx. error max.	triangles count	triangles % of original	running time (m:ss)	vertices above ϵ limit	superfaces to adjust / total superfaces
0.5	0.0733	0.9509	56,294	31.29	4:22	12	10 / 18,792
1.0	0.1797	2.1814	28,216	15.68	3:07	17	8 / 7,599
1.5	0.2778	1.6823	19,348	10.75	2:46	2	2 / 4,908
2.0	0.4000	2.3609	12,762	7.09	2:33	42	4 / 3,016
3.0	0.5516	3.7760	9,046	5.03	2:21	59	2 / 2,152
4.0	0.6797	4.1972	7,942	4.41	2:17	4	1 / 1,885

Table 4: Results of simplifying the femur mesh of 179,916 triangles – with aggressive border straightening).

19 Map of topographic data of the earth.

20 Simplified map of topographic data of the earth: $\varepsilon = 32.0$ meters.

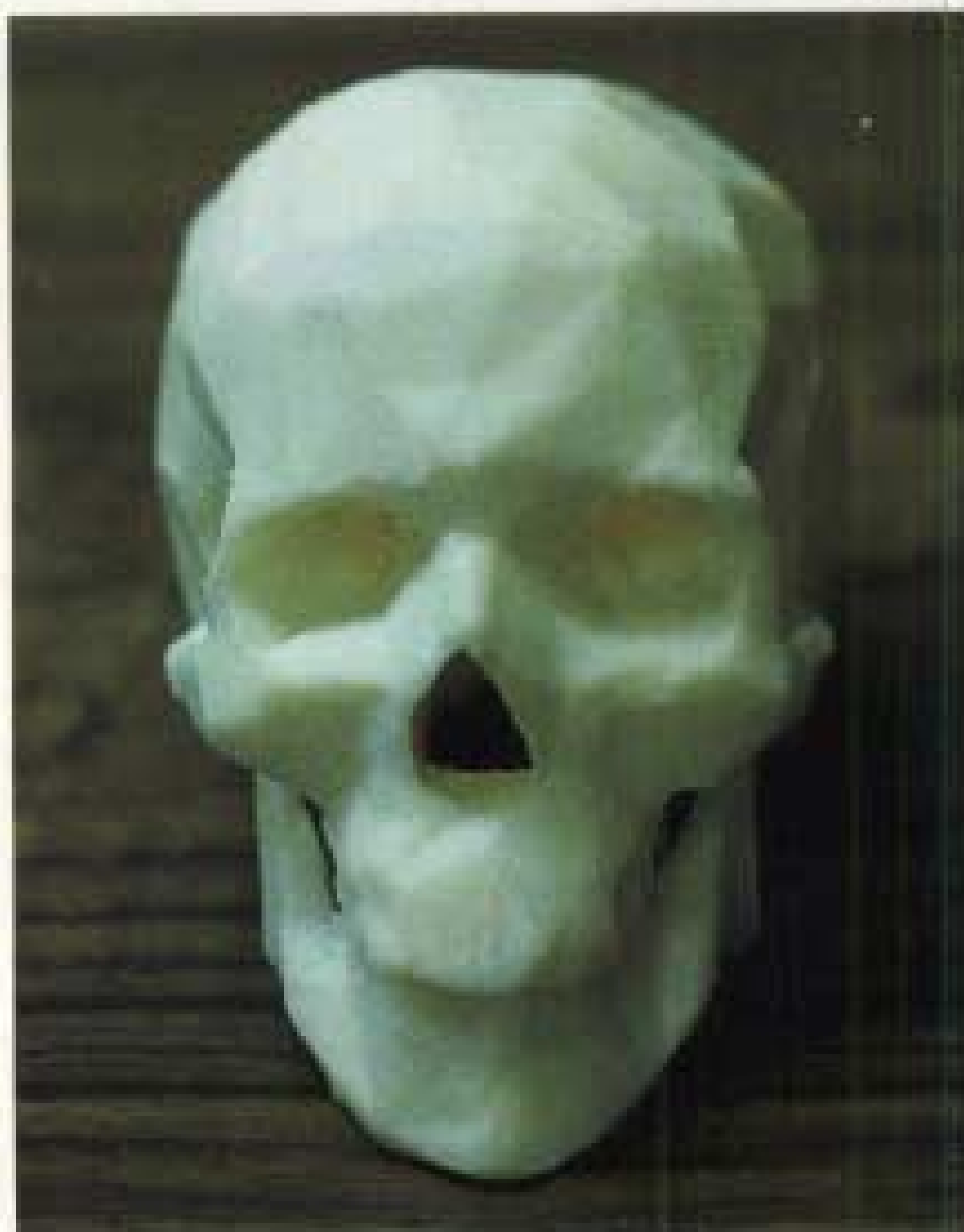
Some published comparisons

Algorithm Hardware Configuration	*triangles*		*running times (seconds)*
	original mesh	*reduced mesh*	
Superfaces `IBM RS/6000 (model 550)`	30,876 179,916 349,792	2,038 21,840 24,170	27 148 360

Triangle Decimation [13]
`SGI Onyx Reality Engine`
`2-processor model`
	38,394
	186,630
	334,643
	1,049,476
		17,799
		71,485
		84,342
		29,507
			8
			43
			90
			322

Geometric Optimization [10] `(hardware not specified)`	315,812 1,019,373	295,636 642,204	96 538
Mesh Optimization [7] `DEC Alpha`	3,832 18,272	432 1,348	600 2820
Multi-resolution Approximation [9] `IBM RS/6000 (model 560)`	349,792	N/A	80

Table 5: Running times of different mesh simplification algorithms.



21 3D hard copy of a simplified skull (9,820 triangles).