CHALMERS | GÖTEBORG UNIVERSITY

PREPRINT 2006:36

Airbag Folding Based on Origami Mathematics

CHRISTOFFER CROMVIK
KENNETH ERIKSSON

Department of Mathematical Sciences
Division of Mathematics
CHALMERS UNIVERSITY OF TECHNOLOGY
GÖTEBORG UNIVERSITY
Göteborg Sweden 2006

Preprint 2006:36

Airbag Folding Based on Origami Mathematics

Christoffer Cromvik and Kenneth Eriksson

CHALMERS | GÖTEBORG UNIVERSITY

Department of Mathematical Sciences
Division of Mathematics
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg, Sweden

Göteborg, December 2006

Preprint 2006:36
ISSN 1652-9715

Matematiska vetenskaper
Göteborg 2006

AIRBAG FOLDING BASED ON ORIGAMI MATHEMATICS

CHRISTOFFER CROMVIK AND KENNETH ERIKSSON

Abstract. A new algorithm for folding three-dimensional airbags is presented. The method is based on Origami mathematics combined with nonlinear optimization.

The airbag is folded to fit into its compartment. Simulating an inflation therefore requires an accurate geometric representation of the folded airbag. However, the geometry is often specified in the inflated three-dimensional form, and finding a computer model of the folded airbag is a non-trivial task. The quality of a model is usually measured by the difference in area between the folded and the inflated airbags.

The method presented here starts by approximating the geometry of the inflated airbag by a quasi-cylindrical polyhedron. Origami mathematics is used to compute a crease pattern for folding the polyhedron flat. The crease pattern is computed with the intention of being fairly simple and to resemble the actual creases on the real airbag.

The computation of the crease pattern is followed by a computation of the folding. This is based on solving an optimization problem in which the optimum is a flat folded model. Finally, the flat airbag is further folded or rolled into its final shape (without using Origami).

The method has been successfully applied to various models of passenger airbags, providing more realistic geometric data for airbag inflation simulations.

1. Introduction

Simulating a crash when the crash test dummy hits the airbag while it is still expanding remains a challenge to the industry. This situation is called out-of-position (OOP), reflecting that the airbag was not designed for occupants that are sitting too close or for some other reason hit the airbag before it is fully inflated.

The difficulty with an OOP situation compared to an in-position situation is that the inflation of the folded airbag is much more important. It has to be realistically computed, since it affects the impact of the dummy. Attaining a realistic simulation means starting with a correct geometry of the folded airbag and simulating the inflation with correct gas dynamics. Several commercial software packages exist that can simulate the inflation process of an airbag, e.g., the explicit Finite Element (FE) code `LS-DYNA` [5].

This work aims at developing an algorithm for computing an accurate geometry of the flat folded airbag. Different airbags are folded by different methods and with different numbers and types of foldings. The airbags are often folded by both machines and humans according to a folding scheme. Still, the creases are not entirely deterministically positioned. It is very difficult to control the placement of smaller creases. The folding schemes all assume that the airbag lies flat and stretched in some direction. In this position, different foldings are executed until the dimension of the folded airbag is small enough so that it fits into the airbag compartment. The foldings can be a combination of simple folds, but also roll folds.

Some preprocessors to `LS-DYNA`, e.g., `EASi-FOLDER` [4] and `OASYS-PRIMER` [1] contain software for folding a (nearly) flat FE airbag mesh. They are capable of executing the type of foldings that are normally used in production on flat airbags, e.g., roll-fold, z-fold. However, they are not as accurate when folding an airbag in its three dimensional shape to a flat airbag.

Some airbag models have a simple construction, e.g., the driver model which is made of two circular layers sewn together. It is essentially two-dimensional. Passenger airbags are often more complicated. They are made of several layers sewn together in a three dimensional shape, with no trivial two-dimensional representation. See Figure 1 for an example.

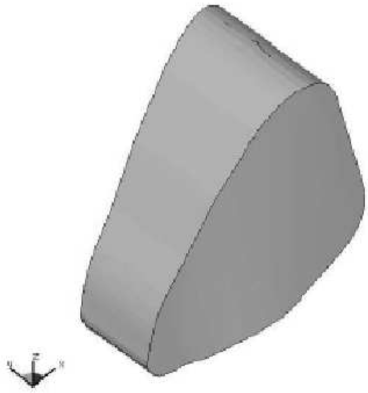

FIGURE 1. A CAD model of a passenger airbag.

In the present work, the computation of the geometry of the flat folded airbag is organized into two steps. First a crease pattern is computed on a polyhedral approximation of the airbag. Second, a nonlinear optimization problem is formed and solved for the purpose of finding the flat geometry. The accuracy of the computed approximation is measured by comparing its area to the area of the inflated model.

2. Crease Pattern

A crease pattern is first designed for a tetrahedron. We present a series of proofs for different types of polyhedra. The proofs are constructive, and their results can be used to design a crease pattern for our application.

Flat foldability, meaning that the polyhedron can be flattened using a fixed crease pattern, is achieved by cutting along the crease lines, folding the resulting object, and then gluing the cut-up faces back according to the correct connections.

Theorem 2.1. *The tetrahedron can be folded flat.*

Proof. The proof is organized in a sequence of figures shown in Figure 7, each visualizing the cutting and folding. Consider the tetrahedron with vertices A, B, C, D as in the figure. Cut up the triangle BCD of the tetrahedron, with straight cuts from a point E on the face, to the three vertices B, C, D, respectively, as in the figure.

Then open up the tetrahedron by rotating the triangular patches BDE, BCE, and CDE around the axes BD, BC, and CD, respectively, until these triangles become parts of the three planes through ABD, ABC, and ACD, respectively, as in the figure.

Cut the quadrilateral surface with vertices A, B, E', D along a straight cut from E' to A, and then rotate the resulting triangular faces ABE' and $AE'D$ around the axes AB and AD, respectively, until these faces become parts of the two planes ABC and ACD, respectively, as in the figure.

We choose the point E such that the edge BE' after rotation coinsides with BE'' and DE' with DE'''. The condition for this is that $\angle ABD+\angle DBE = \angle ABC+\angle CBE$ and $\angle ADB+\angle BDE = \angle ADC + \angle CDE$.

Using this, we may now (partly) restore the surface of the tetrahedron by joining the surfaces ABE'' and $ABE''C$ along the edge BE'', and the surfaces ADE''' and $ADE'''C$ along the edge DE'''.

Finally we rotate the (partly double layered) surface $ADE'''C$ around the axis AC until it coincides with the plane through A,B and C as in the figure. To conclude the proof of the flat foldability of the tetrahedron we now note that the point E''' after rotation coincides with E''. We may therefore now completely restore the topology of the original tetrahedron by joining the edges AE'' and AE''' (after rotation) and the edges CE'' and CE''' (after rotation). □

Note that the proof is based on cutting and gluing. It does not reveal if there is a continuous deformation to a flat shape.

Remark 2.1. *Concerning the line AE' we remark that the angles $\angle BAE'$ and $\angle DAE'$ satisfy $\angle BAE' + \angle DAE' = \angle BAD$ and $\angle BAC - \angle BAE' = \angle CAD - \angle DAE'$, as in the figure, and are thus independent of the plane BCD. We further note that we may also consider rotating the triangles BDE, BCE and CDE in the opposite direction, again until they become parts of the planes ABD, ABC and ACD, respectively, as in figure. We now choose the point E so that $\angle ABD - \angle DBE = \angle ABC - \angle CBE$ and $\angle ADB - \angle BDE = \angle ADC - \angle CDE$. Continuing from the figure we may then again make a straight cut from E' to A (partly double layered). Again, when we now rotate around the axes AB and AD as before the (rotated) point E' will coincide with E'' and E''' respectively, and we can partly restore the tetrahedron by joining along the edges. Finally, after rotation around AC we may completely restore the topology of the surface of the tetrahedron by joining along the edges. Concerning the crease line from A to E'' we note that again the angles $\angle BAE'$ and $\angle DAE'$ must satisfy the same equations $\angle BAE' + \angle DAE' = \angle BAD$ and $\angle BAC - \angle BAE' = \angle CAD - \angle DAE'$ as before and therefore must be the same as above. We therefore conclude that this crease line is independent of both direction of rotation of the triangles BCE, BDE and CDE, and of the position and orientation of the plane BCD (as long as the angles at A are unchanged).*

We now proceed by cutting the tetrahedron by a plane, see Figure 2. We call the cut-off tetrahedron a prism type polyhedron.

Theorem 2.2. *The prism type polyhedron can be folded flat.*

Proof. Consider a tetrahedron $ABCD$ with the crease pattern from the proof of Theorem 2.1. Cut the tetrahedron with a plane, see Figure 2. In the cut, insert two additional triangular surfaces, such that the two cutoff parts are closed, but not separated. The "smaller" cutoff part is a tetrahedron, and the "bigger" part is a prism type polyhedron. Let the vertices of the smaller tetrahedron be a, b, c, d, where $A = a$, b lies on the edge AB, c on AC and d on AD.

Remark 2.1 shows that the crease line from A to E', see Figure 7, is independent of how the inserted triangular face of the "smaller" tetrahedron is folded. Let it be folded to the interior of the "smaller" tetrahedron. This means that a crease pattern can be constructed which will coincide with the crease pattern of the original tetrahedron, i.e., the crease line which is constructed by drawing a straight line from a to e' will coincide with the crease line that was created from the line segment from A to E' in the proof of Theorem 2.1.

Now, make an identical copy of the crease pattern on the inserted triangular face belonging to the prism. Folding the original tetrahedron with its inserted triangular faces is possible by the construction of the crease pattern. Let the two polyhedra be separated by moving the tetrahedron in the plane. By the foldability of the tetrahedron, both the smaller tetrahedron and the prism can be folded flat. □

Next, we cut the prism type polyhedron by a plane, see Figure 3. We call the cut-off prism a box type polyhedron.

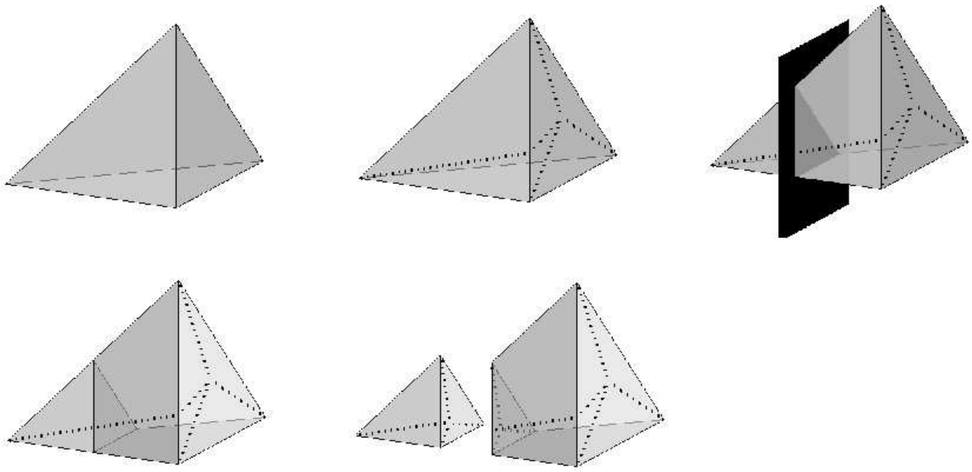

FIGURE 2. A tetrahedron is cut, and in the cut two additional interior triangular faces are created. Identical crease patterns are created on both interior faces, and the tetrahedron is separated into two parts: a smaller tetrahedron and a prism. The flat foldability of the prism follows from the foldability of the tetrahedron.

Theorem 2.3. *The box type polyhedron can be folded flat.*

Proof. Let the prism from the cut-off tetrahedron, with its crease pattern, be cut by a plane, see Figure 3. In the cut insert one additional quadrilateral surface which is only connected to the prism by its four vertices. Along the inserted surface put a crease line γ. Its position is only determined by the position of the upper and lower face of the prism. When the prism (with its cut) and the additional inserted surface are folded, there will be a gap along the sides of the prism, see Figure 4. Let the crease line on the side of the original prism be called ξ. Also, let the point where the crease γ meets ξ unfolded be called p_1, see Figure 4. The gap can be closed by forming two triangles: from a point p, see Figure 4, somewhere along ξ, to the intersection where ξ meets the inserted surface p_2, to B respectively C.

Note that the lengths Cp_1 and Cp_2 are the same, as well as the lengths Bp_1 and Bp_2, and the length Cp is shared by both the gap and the new triangles. Let C_1 and C_2 be positioned according to Figure 4. If the point p is chosen such that $\angle C_1Cp_1 + \angle p_2Cp = \angle C_1CC_2 + \angle C_2Cp$, then the new triangles are an identical match to the gap. By Theorem 2.2, the prism is foldable, so the full construction is foldable, and since the cut does not influence its foldability, and its gap is filled, therefore the box type polyhedron is flat foldable. □

In the proof of Theorem 2.3, a prism was cut off the polyhedron. The process of cutting off a prism can be repeated to create other types of polyhedra.

Definition 2.1. *A quasi-cylindrical polyhedron is a closed cut-off cylinder with a polygonal cross-section.*

Theorem 2.4. *Convex quasi-cylindrical polyhedra are flat foldable.*

Proof. This follows by the proof of Theorem 2.3. In each step, cut off a prism from the polyhedron, until the result forms the given shape.

□

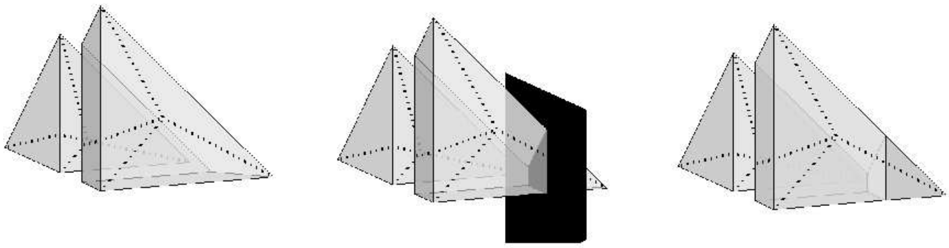

FIGURE 3. The prism from Figure 2 is cut, and in the cut, an additional interior quadrilateral surface is created. The flat foldability of the box type polyhedron follows from the foldability of the prism and the tetrahedron.

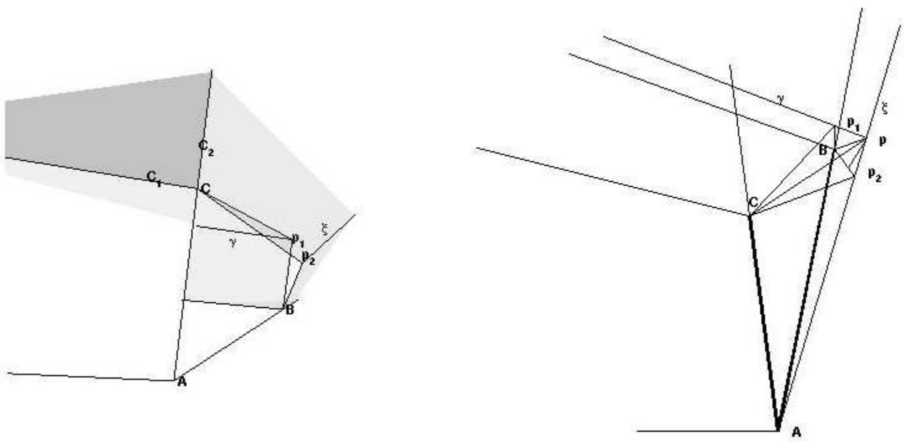

FIGURE 4. The left figure shows the gap around the inserted additional surface from the cut. The right figure shows the same object from above.

Airbags are usually quasi-cylindrical. There are cases, e.g. non-convex polyhedra, for which the technique for generating a crease pattern does not work. These situations might be avoided by slicing the polyhedron, and computing a crease pattern for each part.

Theorem 2.4 provides an algorithm for designing a crease pattern. Given a quasi-cylindrical polyhedron, we can extend it gradually using prisms until it reaches the shape of a tetrahedron. In each step, we apply the theory for flat foldability, creating a working crease pattern.

3. FOLDING

For airbags, there are various alternatives for simulating the folding process. This is specially due to the fact that the problem is artificial in the sense that the folding need not be realistic, e.g., there is no need to introduce the concept of time. The objective is to create a flat geometry which is physically correct, not to fold it in a realistic way.

Our algorithm for folding the polyhedron is based on solving an optimization problem. A program is formulated such that the optimal solution represents a flat geometry. The target function, to be minimized, is a sum of rotational spring potentials, one spring over each crease. The minimal value of a spring potential is found when a fold is completed. The constraints are formulated in order to conserve a physically correct representation of the polyhedron, which means conserving the area and avoiding any self-intersections of the faces of the polyhedron.

The crease pattern over a polyhedron induces a subdivision of polygons called patches. In addition, the patches are triangulated, and the interior of the polyhedron is meshed with tetrahedra. Let the nodes of the mesh be $\{x^i\}_{i=1}^{n}$, and let the indices of the surface nodes be I_S. Let the tetrahedra be $\{K_i\}_{i=1}^{n_K}$ and set $I_K = \{1,\ldots,n_K\}$. Let the four indices of the nodes of tetrahedron k be $V_k(i)$, $i = 1,\ldots,4$. The edges of the triangular faces are denoted $\{E_i\}_{i=1}^{n_E}$, and the indices of the two nodes of edge e are $W_e(i)$, $i = 1, 2$.

Denote the creases $\{C_i\}_{i=1}^{n_C}$. The spring potential over each crease C_i is computed using the scalar product of the normals, n_i^1,n_i^2, of the two neighbouring patches. The normals point outward from the polyhedron, and the scalar product is 1 when the two patches are parallel, and -1 when the fold is completed.

The folding process of a polyhedron with n nodes (surface and interior mesh nodes) is formulated as the following nonlinear program with $f : \mathbf{R}^{3n} \to \mathbf{R}$,

$$\min_x f(x)$$

$$\begin{aligned}
f(x) &= f_1(x) + f_2(x) + f_3(x)\\
&= k_m \sum_{k=1}^{n_K} \left(\sum_{i=1}^{4} \sum_{j=i+1}^{4} \|x^{V_k(i)} - x^{V_k(j)}\| - d_{V_k(i),V_k(j)} \right)^2\\
&\quad + \sum_{i=1}^{n_C} n_i^1 \cdot n_i^2 + k_p \sum_{i=1}^{n_E} \left(\|x^{W_i(1)} - x^{W_i(2)}\| - l_{W_i} \right)^2,
\end{aligned}$$

subject to

$$\begin{aligned}
&\mathrm{vol}(K_i) \quad\ \geq \varepsilon_1, \qquad i = 1,\ldots,n_K,\\
&\mathrm{dist}(x^i, K_j) \geq \varepsilon_2, \qquad i \in I_S,\ j \in I_K \setminus p_i,
\end{aligned}$$

where d_{ij} is the original distance between node x^i and x^j, l_i is the original length of edge i and k_m, k_p are penalty parameters. The first constraint function is $\mathrm{vol}(K_i)$ which is the signed volume of the tetrahedron K_i. The second constraint is $\mathrm{dist}(x^i, K_j)$, which is the distance from a surface node x^i to a tetrahedron K_j, and p_i are the tetrahedron indices connected to node x^i. Finally, ε_1 and ε_2 are small positive constants.

The target function f is composed of three parts. f_1 is a penalty function which strives to keep the tetrahedral mesh uniform. f_2 is the virtual spring potential which drives the folding. f_3 is a penalty function which keeps the edges of the triangles stiff. This is used to maintain the shape and surface area of the patches.

4. Numerical Example

In section 2, a theory for computing a crease pattern was discussed. To demonstrate its practical use, and also to demonstrate the folding algorithm, a numerical experiment is presented. From a CAD-drawing, an airbag shaped polyhedron was constructed. The surface area of the approximation differs about 0.5% to the original area. An in-house optimization solver was used to solve the optimization problem in section 3. It is a Fortran 90 implementation of a low-storage Quasi-Newton SQP method [6, 3, 2], that can handle a few thousand variables and constraints.

The crease pattern was generated by slicing off two upper “bumps”, see Figure 5, from the airbag approximation. The crease pattern for these parts were computed separately from the rest of the polyhedron, and the complete crease pattern was formed by joining the parts.

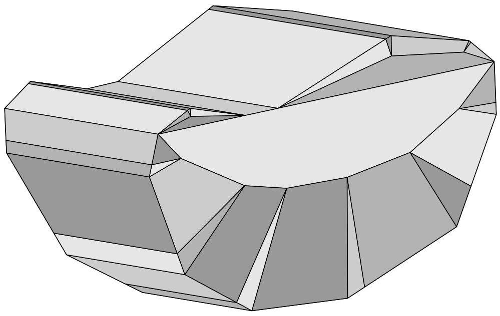

FIGURE 5. Polyhedral approximation of an airbag model together with a computed crease pattern.

The polyhedron approximation with its crease pattern was meshed using `TetGen` [7]. The visual result (solution) from the optimization progress is shown in Figure 6 for different iteration snapshots.

It was found that the surface area of the flat folded polyhedron was within 0.5% of the surface area of the unfolded polyhedron.

Acknowledgement. The authors would like to thank Prof S. Larsson and Dr B. Pipkorn for valuable advice.

References

[1] Arup, *Oasys-primer*, `http://www.arup.com`.
[2] D.P. Bertsekas, *Nonlinear programming*, Athena Scientific, 1999.
[3] R. Byrd, R. Schnabel, and J. Nocedal, *Representations of quasi-newton matrices and their use in limited memory methods*, Mathematical Programming **63** (1994), 129–156.
[4] ESI-group, *Easi-folder*, `http://www.esi-group.com`.
[5] Livermore Software Technology Corp., *Ls-dyna*, `http://www.lstc.com`.
[6] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer-Verlag New York , Inc., 1999.
[7] Hang Si, *Tetgen a quality tetrahedral mesh generator and three dimensional Delaunay triangulator version 1.3 user's manual*, Technical report, Weierstrass-Institut für Angewandte Analysis und Stochastik, 2004.

Department of Mathematical Sciences, Chalmers University of Technology, SE–412 96 Göteborg, Sweden

Department of Mathematical Sciences, Göteborg University, SE–412 96 Göteborg, Sweden
E-mail address: `christoffer.cromvik@chalmers.se`

Department of Mathematical Sciences, Chalmers University of Technology, SE–412 96 Göteborg, Sweden

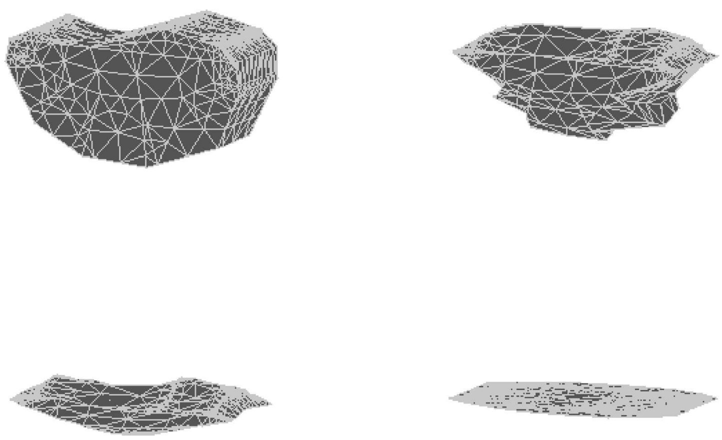

FIGURE 6. The figures show iteration snapshots from the folding of the polyhedron approximation from Figure 5. The upper left shows the unfolded polyhedron, the upper right: 40 iterations, the lower left: 60 iterations, and the lower right: 200 iterations.

DEPARTMENT OF MATHEMATICAL SCIENCES, GÖTEBORG UNIVERSITY, SE–412 96 GÖTEBORG, SWEDEN
E-mail address: kenneth.eriksson@hv.se

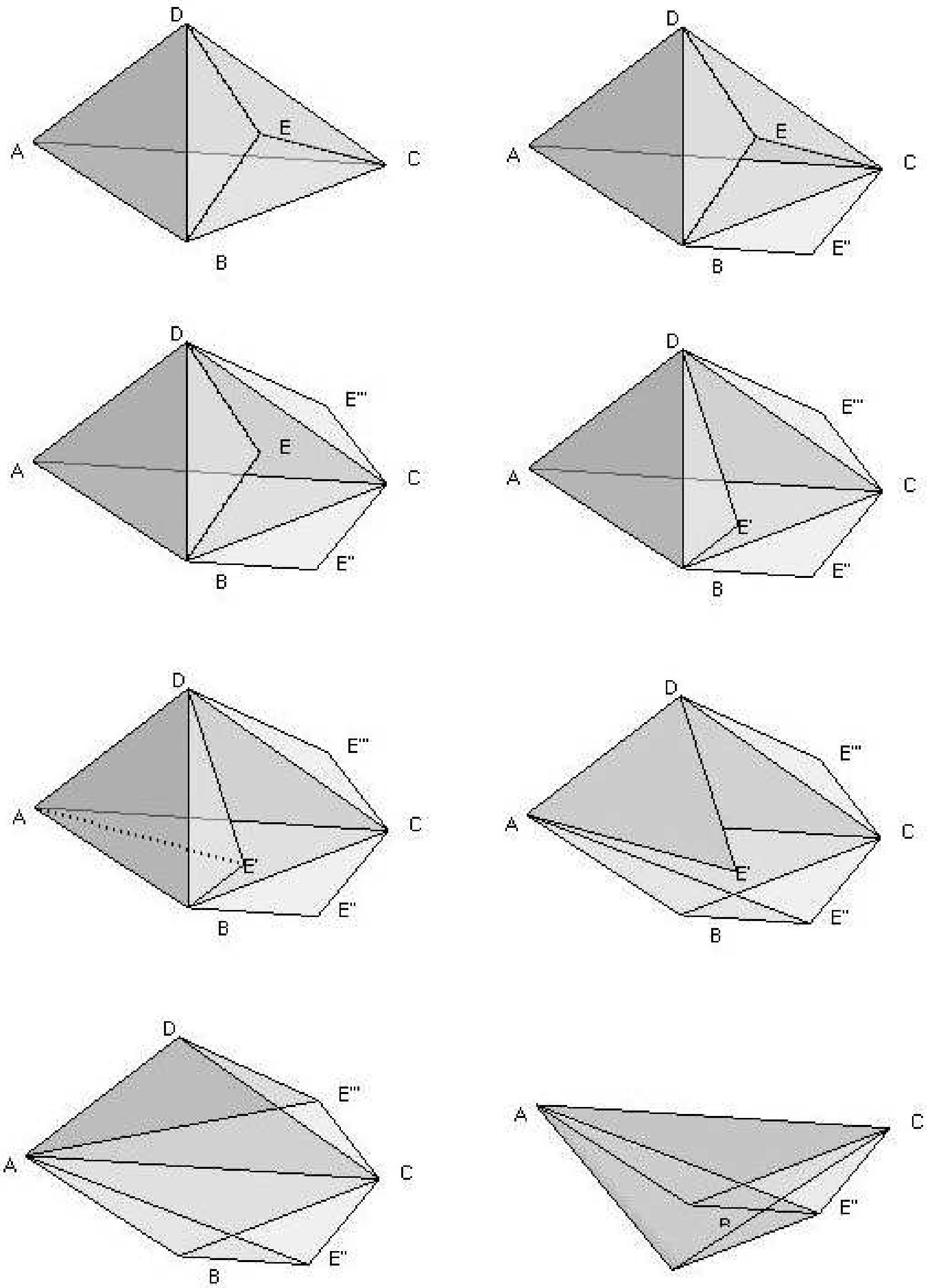

FIGURE 7. Supporting figure for the proof of Theorem 2.1. The proof follows the figures from left to right beginning at the top.