Overcoming Superstrictness in Line Drawing Interpretation

Lluís Ros and Federico Thomas

Abstract—This paper presents a new algorithm for correcting incorrect line drawings—incorrect projections of a polyhedral scene. Such incorrect drawings arise, e.g., when an image of a polyhedral world is taken, the edges and vertices are extracted, and a drawing is synthesized. Along the way, the true positions of the vertices in the 2D projection are perturbed due to digitization errors and the preprocessing. As most available algorithms for interpreting line drawings are "superstrict," they judge these noisy inputs as incorrect and fail to reconstruct a three-dimensional scene from them. The presented method overcomes this problem by moving the positions of all vertices until a very close correct drawing is found. The closeness criterion is to minimize the sum of squared distances from each vertex in the input drawing to its corrected position. With this tool, any superstrict method for line drawing interpretation is now practical, as it can be applied to the corrected version of the input drawing.

Index Terms—Line drawing interpretation, superstrictness, scene understanding, correction algorithms.

1 Introduction

Consider the pictures of the plane-faced alarm devices in Fig. 1a and the line drawings extracted from them in Fig. 1d. How can we tell whether these drawings represent correct projections of the spatial objects they come from? Even more, how can we *reconstruct* the 3D objects they represent? Answering these questions and producing an algorithm able to reconstruct a plane-faced object from its line drawing, with similar results as a human gets, has been one of the goals of Computer Vision and Artificial Intelligence since the early 1970s.

Along the years, several methods have been proposed to test the correctness of line drawings and give their possible reconstructions [2], [3], [4], [5], [6], [7]. These tests succeed in judging as incorrect such impossible figures as those in Fig. 2. However, even when the drawings come from a picture of a real scene, the tests usually judge them as incorrect, failing to derive a spatial reconstruction. To see why, consider the examples in Fig. 1d, showing the projections of two truncated pyramids. These drawings can only be correct when the three edge lines l, m, and n meet at a common point, as this holds in any spatial reconstruction of the drawing (Fig. 1e), and such conditions are preserved after projection. This is a general characteristic of all line drawings: they are only correct for very specific positions of the vertices, satisfying a number of *concurrence conditions* on triplets of lines [6], [8]. Hence, as many geometric relationships between the vertices are lost due to digitization errors and the image processing, most existing tests will judge a drawing as incorrect when it just deviates slightly from a correct and reconstructible configuration. This is why these tests are called *superstrict* [5, chapter 7].

Our contribution is a new procedure to search for the closest correct drawing to a given incorrect one. Such a procedure allows overcoming the superstrictness problem in an easy way: to apply a superstrict method on an incorrect drawing $\mathcal{D}^{inc}$, first compute the closest correct drawing $\mathcal{D}^{cor}$ to it. If the vertices of $\mathcal{D}^{cor}$ are too far from those of $\mathcal{D}^{inc}$ (according to a well-defined distance and a given tolerance), $\mathcal{D}^{inc}$ is judged as incorrect, otherwise, we accept it as "practically correct" and we can start the reconstruction process from $\mathcal{D}^{cor}$.

The next section shortly reviews three classic correctness tests and shows, through an example, why they are superstrict. Then, in Section 3, we compare our approach to two previous methods for overcoming the superstrictness issue. The key point of our correction method is a rational parameterization of the class of correct drawings for a given polyhedron (Sections 5 and 6), which allows us to write down the correction problem as an *unconstrained* minimization of a rational function (Section 4). This minimization can be tackled using a conjugate gradient method and the results of an implementation, together with several experimental results are shown in Section 7. To prevent the minimization from falling into local minima, a good starting point for the search is needed and Section 8 provides one. Finally, Section 9 shows how we can deal with more complicated scenes of opaque polyhedra and the paper concludes in Section 10 summarizing points for further attention.

2 Superstrict Classic Methods

We begin with a few definitions and assumptions used along the paper. To simplify, we will deal with drawings produced by orthogonally projecting *a single* spherical polyhedron, onto the XY plane, showing all edges (even

---

• *The authors are with the Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Llorens Artigas 4-6, 2 planta, 08028 Barcelona, Catalonia Spain. E-mail: {llros, fthomas}@iri.upc.es.*

Manuscript received 4 May 2001; revised 8 Aug. 2001; accepted 28 Aug. 2001.
Recommended for acceptance by L. Vincent.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 114095.

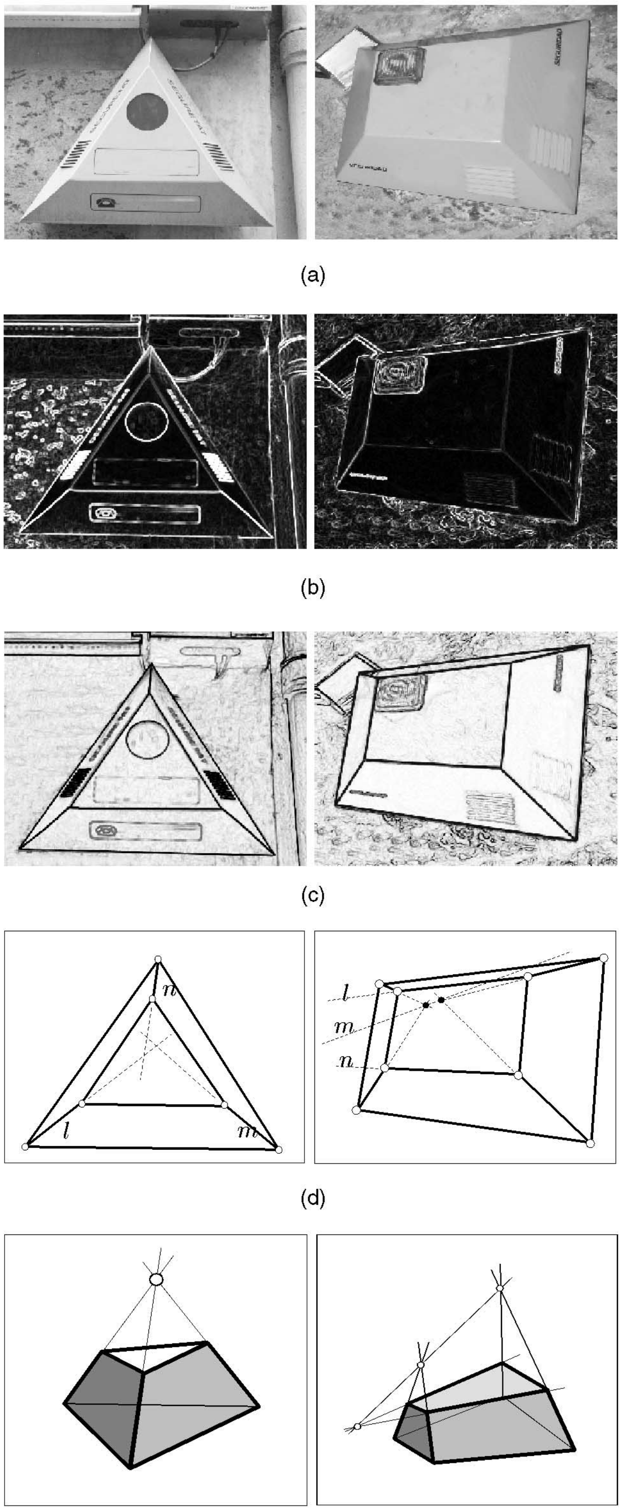


Fig. 1. A picture of an alarm device (a) can be processed to detect the sharp edges (b), extract the straight lines (c), and derive a polyhedral projection (d) which is usually incorrect since it must verify a number of concurrence conditions (e).

the hidden ones). By *spherical*, we mean here that it is homeomorphic to a sphere. This is not too restrictive and Section 9 explains how to extend the results to drawings of more complicated scenes, without hidden edges, several

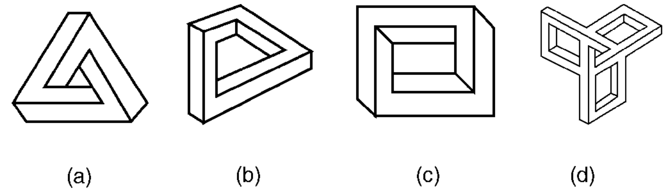


Fig. 2. Some impossible figures: (a) is adapted from Penrose and Penrose [9], (b) from Draper [10], (c) from Huffman [11], and (d) from Ernst [12].

objects, and possible occlusions between them. So, we say that a drawing is *correct*, or *reconstructible*, if there exists a spherical polyhedron that projects onto it, with distinct planes for adjacent faces. Such a polyhedron is called an *interpretation* or a *reconstruction* of the drawing. The *vertices*, *edges*, and *faces* of a line drawing are the projections of their spatial counterparts on the reconstruction.

We will also assume that the drawing is given along with its *incidence structure*. The incidence structure tells the combinatorial structure of the spatial interpretation—basically, which vertices will be incident to which faces. More formally, it is a triple $I = (V, F, R)$, where V is the set of vertices of the drawing and F is the set of its faces. We put a face in F for every subset of vertices that must be kept coplanar in the spatial reconstruction. $R \subseteq V \times F$ is the *incidence set*: there is an *incidence pair* (v, f) in R if vertex v must lie on face f in space. The incidence structure can be computed by applying the method in [5, p. 45], after a consistent labeling of its edges has been obtained. Although finding a consistent labeling is an NP-complete problem [13], several efficient techniques exist to this end. See, for example, the works by Huffman [11], Waltz [14], Hancock and Kittler [15], Myers and Hancock [16], Parodi and Torre [17], or Parodi et al. [18].

In his book [5], Sugihara gives an algebraic test for correctness that, roughly speaking, consists of telling whether a system of linear equalities and inequalities has a solution, which is solvable via linear programming. This system contains an equation of the form

$$(v_x, v_y, v_z, 1) \cdot (A_f, B_f, 1, D_f)^T = 0 \tag{1}$$

for every incidence pair $(v, f) \in R$, to express the constraint that, in any interpretation, vertex v must lie on the plane of face f: $A_f x + B_f y + z + D_f = 0$. To have a set of necessary and sufficient conditons for correctness, Sugihara also adds other depth relations, but, for simplicity, these are omitted here.

One can easily see that, after collecting all the equations (1) for the drawing in Fig. 3a, this linear system has a solution space of dimension four, corresponding to the heights of four vertices that one must fix to get a spatial interpretation. However, the reader can easily check the superstrictness of this test: after moving slightly v_6, the dimension of the solution space drops to three, meaning that the only spatial interpretation is a flat object, with *all* vertices coplanar, and the drawing is judged as incorrect.

In Whiteley's cross-section test [6], a drawing of a spherical polyhedron is correct if, and only if, it is possible to draw a *compatible cross-section* of it. The cross-section is a

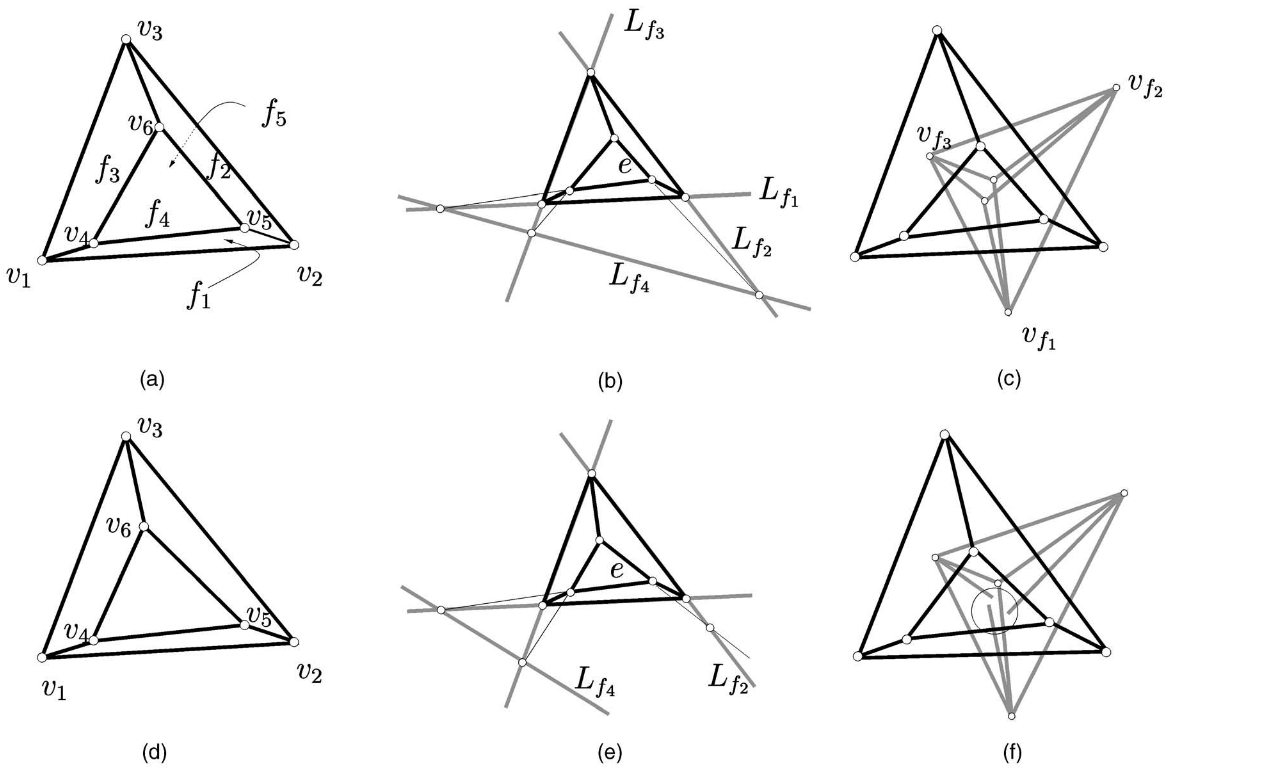


Fig. 3. A correct truncated tetrahedron (a), with two compatible diagrams: the cross-section (b), and the gradient-space diagram (c). If v_6 is slightly moved (d), the diagrams are not compatible anymore ((e) and (f)).

diagram showing the lines of intersection of all faces with one selected face of the polyhedron. Fig. 3b shows (in bold gray) a cross-section of a correct truncated tetrahedron: every line L_f is the intersection of a face plane f with the background face f_5. The cross-section is *compatible* if the line of any edge between a pair of faces contains the point of intersection of the cross-section lines of these faces.

This is also a superstrict test. For example, after slightly moving v_6 (Fig. 3e), the cross-section is not compatible anymore: Edge e does not meet the intersection of lines L_{f_4} and L_{f_2}.

Huffmann [19] and Mackworth [2] use the so-called *dual diagram* for the same purpose.[1] For a drawing to be correct, it must have a *compatible* dual diagram. In this diagram, there is a dual vertex v_f for every face f in the drawing and there is a dual edge joining two dual vertices if their corresponding faces share an edge in the drawing. The dual diagram is compatible if every edge in the drawing is perpendicular to its dual edge in the diagram. Hence, it is possible to generate a dual diagram for a correct truncated tetrahedron, but not for an incorrect one (Figs. 3c and 3f), and "almost correct" drawings are judged as incorrect.

3 Related Work

To the authors' knowledge, the literature offers two approaches to overcome the superstrictness issue: a drawing correction strategy due to Sugihara [5, chapter 7] and an explicit handling of uncertainty, due to Ponce and Shimshoni [21], [22].

Roughly speaking, Sugihara's correction method works as follows: Think about the truncated tetrahedron in Fig. 4a. We see that fixing the heights of v_1, v_2, v_3, and v_4 is enough to determine the heights of the others, as we can use the

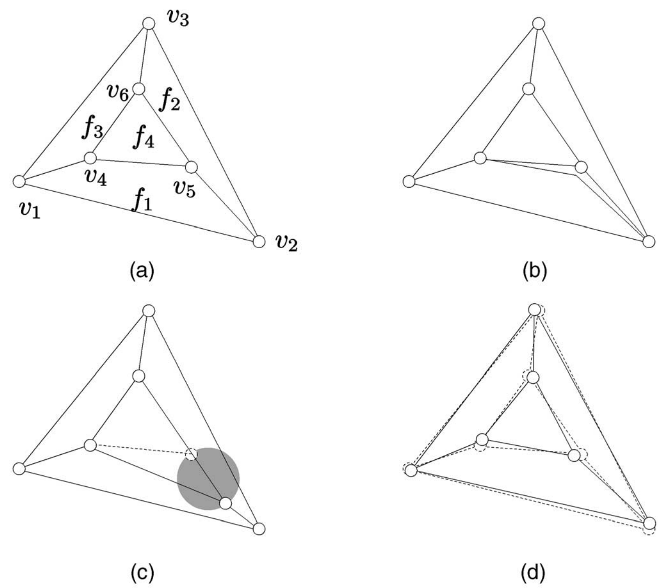


Fig. 4. Sugihara's method corrects (a) by fixing v_1, v_2, v_3, and v_4 then removing the incidence pair (v_5, f_1) (b) and, finally, computing planes for all faces to derive a spatial position for v_5 which is projected back to the plane (c). Nevertheless, better corrections are possible by slightly moving all vertices (d).

1. Actually, this diagram was already discovered in the last century by Maxwell. See, for example, [20].

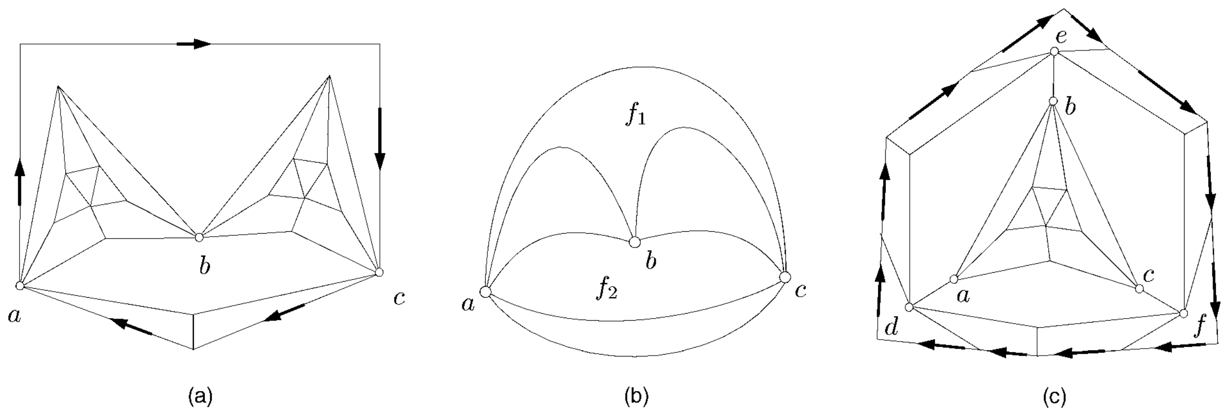


Fig. 5. Sugihara's correction strategy can not be applied on these examples (adapted from [24]). The arrows indicate contour edges where the adjacent faces are actually separated in 3-space.

coplanarity constraint of each face to derive them. However, the height of v_5 is overconstrained, as it can be deduced from *both* the coplanarity of f_1 and f_2. Only when the projection is correct, this height will be identical when computed from both faces. As Fig. 4a is incorrect, a possibility is to drop out the constraint that v_5 must lie on f_1 (Fig. 4b), fix the heights of v_1, v_2, v_3, v_4, compute the resulting planes for the faces, intersect f_1, f_2, f_4 to get a corrected position for v_5 in 3-space, and project v_5 onto the XY plane to get a corrected position for it. In general, the steps are:

1. Take $\mathcal{D}^{inc}$ with its incidence structure (V, F, R) and remove some incidence pairs (v, f) from R, until no vertex has an overconstrained height.
2. Mark all vertices in V that are involved in some of the removed incidence pairs.
3. Lift the new drawing to 3-space, computing the spatial planes of all faces.
4. Derive new spatial positions for the vertices marked in step 2, by computing the intersections of the planes around them. Project these vertices onto the XY plane.
5. Output the incidence structure of the original drawing, but replace the original vertex coordinates by the corrected ones.

For step 1, Sugihara provides a remarkable combinatorial criterion to detect when a line drawing does not have overconstrained heights. Such a line drawing is said to be *generically realizable* to emphasize that it will be correctly reconstructible as long as the vertices occupy generic positions in the plane. Namely, he found this happens if and only if the incidence structure (V, F, R) verifies

$$|V(X)| + 3|F(X)| \geq |X| + 4, \tag{2}$$

for any subset $X \subseteq R$ such that $|F(X)| \geq 2$. Here, $F(X)$ and $V(X)$ are, respectively, the set of all faces and vertices involved in the incidence pairs of X. Sugihara proved this result for incidence structures of trihedral or convex polyhedra [3], and, in 1984, Whiteley extended its validity to arbitrary incidence structures [23].

Although from this result it seems that deciding whether a drawing is generically realizable takes $O(2^{|R|})$ time, Sugihara gave an efficient graph-flow procedure that checks the conditions in $O(|R|^2)$ time (see [5, chapter 8]), which also permits a fast algorithm for step 1, that removes the least possible number of incidence pairs.

However, as already noted by Sugihara, this correction process is only possible when the removed vertices lie on at most three nontriangular faces, because if a vertex lies on more than three nontriangular faces, the intersection of their planes is not a single point in general and step 4 above cannot be performed. Unfortunately, as Whiteley notes in [24], one can find drawings where this does not happen. For example, the drawing in Fig. 5a is not generically realizable ($|V(X)| + 3|F(X)| < |X| + 4$ for the subset X in Fig. 5b) and it can only be corrected by removing one of the vertices a, b, or c, but these are incident with four nontriangular faces. Fig. 5c shows a second "bad" example: The corrected drawing must make the lines ad, be, and cf concurrent, but all six vertices are incident with at least four nontriangular faces.

Also, another drawback of Sugihara's technique is that the corrected drawing may deviate substantially from some original vertices, when moving all of them just a bit, one can find drawings that fall in a smaller neighborhood (compare Figs. 4c and 4d). The correction scheme presented in this paper overcomes these two inconveniencies: All drawings of opaque objects will be correctable and we will allow the movement of all vertices to get correct drawings closer to the input incorrect one.

On a different approach, Ponce and Shimshoni explicitly consider that a vertex true position (x_i, y_i) is unknown in the drawing, but that must fall in a square of side 2ε around the measured position $(\tilde{x}_i, \tilde{y}_i)$. Then, they take Sugihara's system of linear equation (1) and do the change of variables $x_i = \tilde{x}_i + \mu_i,\ y_i = \tilde{y}_i + \nu_i$ for every vertex (x_i, y_i) of the drawing and add the constraints $|\mu_i| \leq \varepsilon$, $|\nu_i| \leq \varepsilon$. This leads to a system of *nonlinear* equalities and inequalities that, after a clever addition of gradient-space constraints, and some algebraic manipulation, they are able to linearize again. The linearization, however, is gained at the cost of the sufficiency of the test and, as they note, the resulting constraints are only necessary for a drawing to be correct. The approach we present does not suffer from this problem: as we avoid adapting to a specific correctness test, any one offering necessary and sufficient conditions can be applied to the corrected version of the drawing.

4 The Overall Algorithm

Given an incorrect drawing $\mathcal{D}^{inc}$, our goal is to obtain a correct one $\mathcal{D}^{cor}$ with the same incidence structure as $\mathcal{D}^{inc}$ which is as close to $\mathcal{D}^{inc}$ as possible. As a function measuring the distance between the two drawings, we have chosen the sum of the squared Euclidean distances between pairs of corresponding vertices in $\mathcal{D}^{inc}$ and $\mathcal{D}^{cor}$.

The problem can be stated as follows: If v_i^{inc} and v_i^{cor} denote, respectively, the 2D coordinates of the ith vertex of $\mathcal{D}^{inc}$ and $\mathcal{D}^{cor}$, we want to minimize

$$\sum_{i=1}^{n} \|v_i^{inc} - v_i^{cor}\|^2,$$

subject to the constraint that the vertices v_i^{cor} define a correct drawing $\mathcal{D}^{cor}$ with the same incidence structure as $\mathcal{D}^{inc}$.

However, we will show that it is possible to parameterize the 2D coordinates of the vertices of all correct drawings with a given incidence structure. More precisely, given an incidence structure $I = (V, F, R)$, it is possible to write the coordinates (x_i^{cor}, y_i^{cor}) of every vertex $v_i^{cor} \in V$ as functions

$$x_i^{cor} = \chi_i(p_1, p_2, \dots, p_n),$$
$$y_i^{cor} = \psi_i(p_1, p_2, \dots, p_n),$$

in such a way that any tuple of parameters $(p_1, p_2, \dots, p_n)$, $p_i \in \mathbb{R}^n - \mathcal{F}\ \forall i$, where $\mathcal{F}$ is a zero-measure subset of $\mathbb{R}^n$, fixes a correct drawing of I. χ_i and ψ_i are rational functions and, thus, everything reduces to the unconstrained minimization of the rational function

$$\sum_{i=1}^{n} \|v_i^{inc} - (\chi_i(p_1, p_2, \dots, p_n), \psi_i(p_1, p_2, \dots, p_n))\|^2,$$

which can be computationally solved by starting a gradient search at an initial correct drawing that estimates the final solution. The next section presents the *resolvable sequence*, the key concept that leads to this parameterization.

5 Resolvable Sequences

Is there a set of independent choices that can be made to construct a polyhedron in a consistent manner? To illustrate this question, let us focus on the simple example of Fig. 6a, a pyramid with a quadrilateral base. The shape of this polyhedron can be fixed by, for example, giving the coordinates of all its vertices or the face plane coefficients of all its faces. But care must be taken in any of the two ways. If we arbitrarily fix all planes, then f_1, f_2, f_3, and f_4 will not probably have a common point of intersection and vertex v_1 will be inconsistently defined. On the contrary, if we arbitrarily fix all vertices, v_2, v_3, v_4, and v_5 need not be coplanar and face f_5 might be inconsistently defined.

In general, we say that a polyhedron is *resolvable* if it is possible to list its vertices and faces in a sequence $\mathcal{S} = (\dots, v_i, \dots, f_j, \dots)$ in such a way that

(C1) when a vertex occurs in $\mathcal{S}$, it is incident to at most three previous faces;

(C2) when a face occurs in $\mathcal{S}$, it is incident to at most three previous vertices;

(C3) when two faces f and f' share three or more vertices (Fig. 6b), f and f' appear earlier in $\mathcal{S}$ than the third of the common vertices;

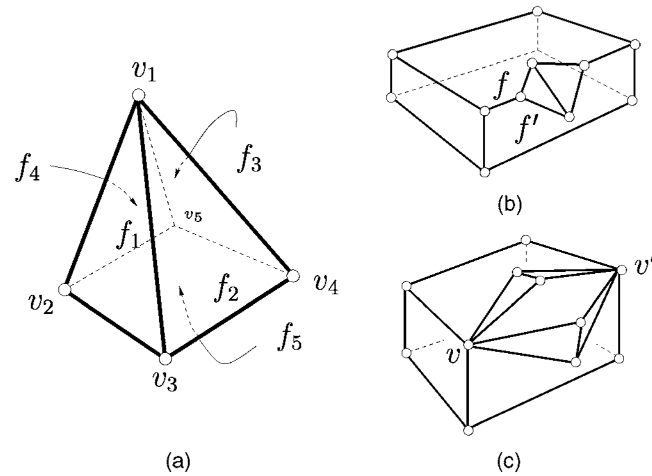


Fig. 6. (a) A pyramid with a quadrangular base. (b) Two faces sharing more than two vertices. (c) Two vertices sharing more than two faces.

(C4) when two vertices v and v' are incident to three or more common faces (Fig. 6c), both v and v' appear earlier than the third of the common faces.

$\mathcal{S}$ is called a *resolvable sequence* for the polyhedron. Note that if such a sequence exists, then we can construct the polyhedron in a consistent way. We just need to fix its vertices and faces, one by one, following the order in $\mathcal{S}$. Along the way, when an element is underconstrained by previous choices, additional choices can be taken arbitrarily.

In 1934, Steinitz proved that all polyhedra whose graph of vertices and edges is planar and vertex 3-connected are resolvable [25]. Actually, for these polyhedra, it suffices to find a sequence satisfying conditions **(C1)** and **(C2)** above as their 3-connectedness ensures they have no face sharing more than two vertices nor any pair of vertices sharing more than two faces.

It is well-known that a graph G is the graph of a spherical polyhedron if and only if G is a vertex 2-connected and edge 3-connected planar graph [26, proposition 2.8]. Hence, Steinitz's result only applies to a subclass of spherical polyhedra. However, very recently, Sugihara has extended Steinitz's results, finding that actually all spherical polyhedra are resolvable [25], a result which has definitely permitted the correction method we present, valid for drawings with the incidence structure of a spherical polyhedron, and other topologies described in Section 9. Moreover, [25] shows that the resolvable sequence is not unique, in general, and that it only depends on the combinatorial structure of the polyhedron at hand.

6 Parameterizing Correct Projections

The resolvable sequence induces a parameterization of all polyhedra with a given incidence structure. For example, a trivial resolvable sequence for the truncated tetrahedron in Fig. 3a is to first list all faces and then all vertices: $\mathcal{S} = (f_1, \dots, f_5, v_1, \dots, v_6)$. (This is clearly valid for any *trihedral polyhedron*, one where every vertex has three incident faces.) Thus, here, the coordinates (x_i, y_i, z_i) of every vertex v_i can be written as functions of the coefficients of its three incident planes by, e.g., solving for x_i, y_i, and z_i,

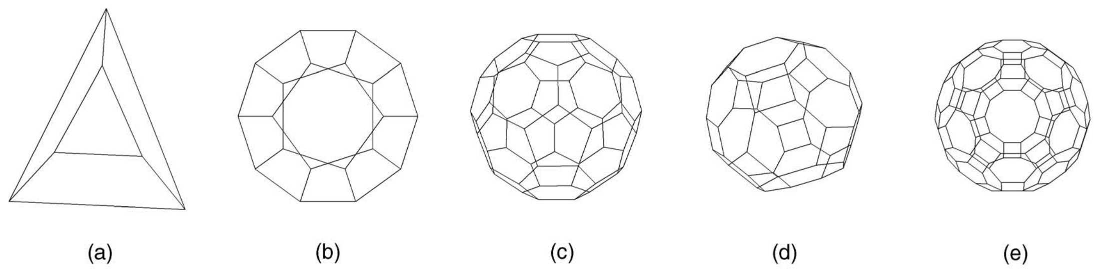


Fig. 7. Line drawings used as a testbed for the correction algorithm: (a) a truncated tetrahedron, (b) a dodecahedron, (c) a truncated icosahedron, (d) a rhombitruncated cubeoctahedron, (e) and a rhombitruncated icosidodecahedron.

and varying these coefficients we get different *realizations* of the truncated tetrahedron.

Note that the resolvable sequence also induces a parameterization of all correct drawings with the given incidence structure, as we only need to project the (parameterized) spatial polyhedron onto the XY plane, keeping the parameterization for the X and Y coordinates of every vertex.

In the general case, we can construct a parameterization of all polyhedra with a given incidence structure I as follows: First, we compute a resolvable sequence $\mathcal{S}$ for I. Then, we visit every element of $\mathcal{S}$, following the order of the sequence. If the element is a face, then:

- If it is not incident to any previous vertex, there is total freedom in choosing its position and the four coefficients of its plane are free parameters.
- If it is incident to three previous vertices, the parameters of the plane are totally fixed and no new parameter is introduced.
- If it is incident to two previous vertices, say p and q, we must select one of all the planes meeting the segment pq. Such a plane can be written as:

$$\begin{vmatrix} p_x & q_x & r_x & x \\ p_y & q_y & r_y & y \\ p_z & q_z & r_z & z \\ 1 & 1 & 1 & 1 \end{vmatrix} = 0,$$

  and the three coordinates of a third point $r = (r_x, r_y, r_z)$ are introduced as new parameters.
- If a face is incident to one previous vertex p, its plane can be expressed as

$$(n_x, n_y, n_z) \cdot ((x, y, z) - (p_x, p_y, p_z)) = 0,$$

  and the three coordinates of the normal vector (n_x, n_y, n_z) are chosen as parameters.

If the element is a vertex v, then:

- If v is incident to no previous face, there is total freedom in choosing its position and its three coordinates are taken as free parameters.
- If v is incident to three previous faces, the vertex is totally fixed and can be found computing the intersection of the three planes.
- If v is incident to two previous faces, say f_i and f_j, we can write two equations:

$$A_i v_x + B_i v_y + C_i v_z + D_i = 0,$$
$$A_j v_x + B_j v_y + C_j v_z + D_j = 0,$$

  and solve for v_x and v_y in terms of v_z which is introduced as a new parameter.
- Finally, if v is incident to one previous face, say f, we can freely choose v_x and v_y and get v_z from the equation of f's plane.

Note that this parameterization is rational as, at each step of its construction, we can write a vertex or face coordinate as a quotient of polynomials in the parameters. Although for certain choices of the parameters it may fail to provide a polyhedron (e.g., there is an indetermination when a vertex is incident to three previous faces, and the chosen planes for them are not all distinct), this only happens for a zero-measure subset of the parameter space, posing no problem to the minimization, as the next section explains.

7 Implementation and Results

The correction algorithm has been implemented in C for drawings of trihedral polyhedra, as these have the advantage that every vertex position is easily parameterized by the 12 coefficients of its three incident planes.

For the minimization, we use TNPACK, a freely available package specially suited for large-scale problems with possibly thousands of variables [27]. To minimize a function $F(X)$, $X \in \mathbb{R}^n$, TNPACK implements the iterative truncated Newton method, based on minimizing a local quadratic approximation of F at every step. For efficiency, an approximated (truncated) solution of this local minimization is allowed which is computed through a preconditioned conjugate gradient algorithm.

The user must essentially supply three routines, returning F, its gradient, and the Hessian matrix, evaluated at a given point $X \in \mathbb{R}^n$. For the gradient, we directly provide the symbolic expressions, as they are easy to derive. For the Hessian matrix, we rely on an (optional) internal TNPACK routine that uses finite differences of the gradient to compute it. To prevent the minimization from falling in a point X of parameter space yielding indetermination (see Section 6), the routine computing $F(X)$ is implemented to return a very high value for these configurations.

We have tested the correction process on several drawings of spherical polyhedra: a truncated tetrahedron, a dodecahedron, a truncated icosahedron, a rhombitruncated-cubeoctahedron, and a rhombitruncated-icosidodecahedron (Fig. 7). The number of optimization variables involved in these examples is 20, 48, 104, 128, and 248, respectively—that is, four times the number of faces.

For each of these drawings, the following experiment has been done. First, a correct drawing $\mathcal{D}^{ini}$ is generated by

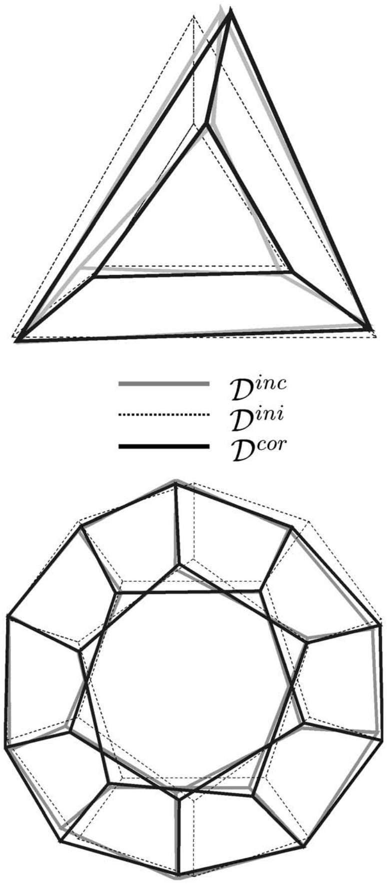


Fig. 8. Correction of a truncated tetrahedron (top) and of a dodecahedron (bottom).

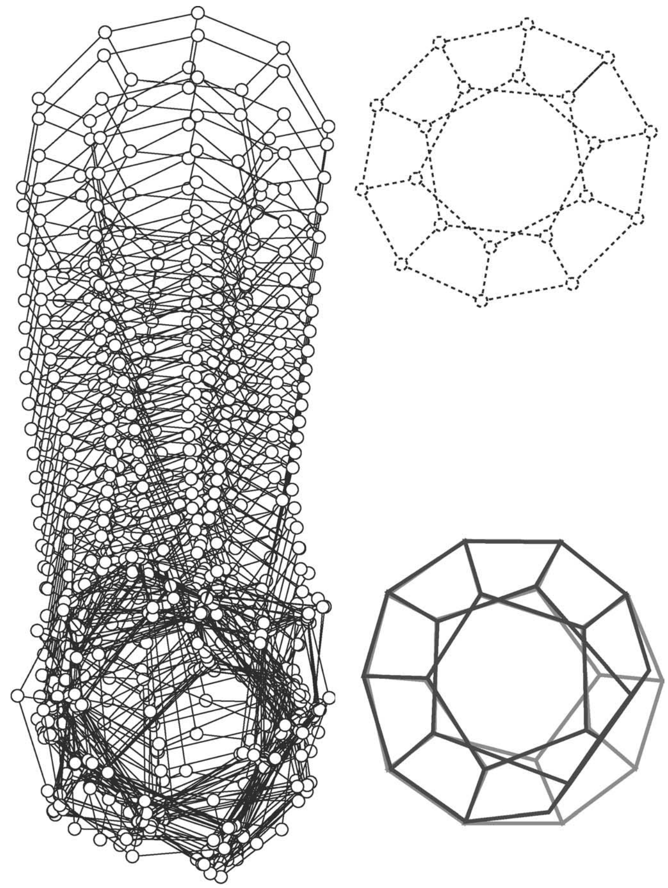


Fig. 9. A correction sequence of a dodecahedron using a bad starting point (left). The incorrect drawing (in gray lines), the initial estimation (dashed) and the final correction (black) have been singled out to the right. The final correction falls on a bad local minimum because the initial estimation is actually translated and rotated with respect to the input drawing.

projecting the spatial polyhedron to a plane. Then, the vertices of $\mathcal{D}^{ini}$ are randomly perturbed to get an incorrect drawing $\mathcal{D}^{inc}$ with the same incidence structure. Finally, the correction algorithm is applied to $\mathcal{D}^{inc}$, by starting a gradient search from $\mathcal{D}^{ini}$. The resulting corrected drawing $\mathcal{D}^{cor}$ is shown in Fig. 8 for the truncated tetrahedron and the dodecahedron.

The gradient search has taken four seconds of CPU time in the toughest case of Fig. 7, using a SUN Ultra-80. This running time does not include that of computing the incidence structure, which is assumed to be given, as already mentioned in Section 2. We note that, although simple, these drawings are far more complex than those one can find in the literature [5], [22]. Extensive tests have been done in all cases, starting the gradient search at different initial drawings and the minimization always converged rapidly to a small neighborhood of the incorrect drawing.

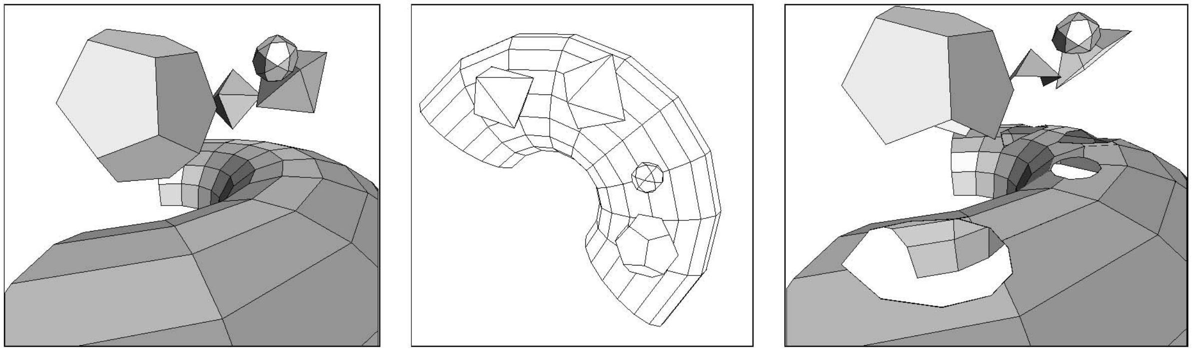
However, the used objective function certainly has local minima and one can find initial drawings $\mathcal{D}^{ini}$ from which the algorithm gets stuck on them. Fig. 9 gives an example of this undesired behavior. Here, $\mathcal{D}^{ini}$ is generated by projecting the dodecahedron onto the XY plane. Also, a copy of $\mathcal{D}^{ini}$ is generated, then translated downwards, rotated 180 degrees about its center and randomly perturbed to finally obtain $\mathcal{D}^{inc}$. The sequence to the left shows the path followed by the gradient method. Notice how the vertices follow crossing trajectories, from their origin to the destination, to undo this 180° rotation. As we see, the final correction is a local minimum compared to that of Fig. 8 bottom, while the input drawing $\mathcal{D}^{inc}$ is exactly the same in both cases. Clearly, there is a need for feeding the search with a good starting point and next section proposes a method to this end.

8 A Good Starting Point

A reasonably good starting drawing can be easily computed using again the resolvable sequence. The idea is to properly place every vertex and face of the sequence, so that the 2D projection is locally close enough to $\mathcal{D}^{inc}$. Let us see this in detail. We distinguish several situations, depending on whether we are fixing a vertex or a face.

Assume first that we are fixing a vertex v, whose 2D position in the incorrect drawing is v^{inc}. v can be incident to zero, one, two, or three previously-fixed faces. In the first case, there is total freedom in choosing v's spatial position but, to be compliant with the drawing, we choose it to lie in the vertical line over v^{inc}, at any height. If v is incident to just one previous face, then we choose it over this face's plane, in the vertical line at v^{inc}. If v is incident to two faces with planes α and β (respectively), we fix v on the line of intersection of α and β at the place where its 2D projection is at a minimum distance from v^{inc}. Finally, if v is incident with three previous faces, we fix it in the intersection of their respective planes.

On the other hand, if we are fixing a face f, it can be incident to three, two, one, or zero previously-fixed vertices.



(a) (b) (c)

Fig. 10. When a drawing (b) is a projection of an opaque scene (a), all we can reconstruct is a collection of polyhedral surfaces, each one topologically equivalent to a disk with possible holes (c).

In the first case, there is no choice for the plane of f as it is fully determined by the three vertices. If the face is incident to two fixed vertices, say p and q, we can choose among all the planes meeting the line pq. But which one? If some of the neighboring faces of f have already been fixed, say faces $f_{i_1}, f_{i_2}, \dots$, then we would like that the lines of intersection of these faces with f lie reasonably close to the corresponding edges in the incorrect drawing. If we label the common vertices between f and $f_{i_1}, f_{i_2}, \dots$ as $w_1, \dots, w_m$, and $z(w_i)$ denotes the height of vertex w_i as computed on the plane of its already-fixed face, over its position in the drawing, then a reasonable way to achieve this is to fix f to the plane α that meets the line pq and minimizes the sum of squared residuals:

$$\sum_{i=1}^{m}(z(w_i) - z_\alpha(w_i))^2,$$

where $z_\alpha(w_i)$ denotes the z-coordinate of vertex w_i as given by the plane α. Obviously, if no adjacent face of f was previously fixed, we simply fix f at *any* plane meeting pq.

The remaining cases, when f is incident to *one* or to *no* previous vertex are analogous, the only difference being that we choose among all the planes meeting a fixed point in the first case, and among all possible planes of 3-space in the second.

Of course, following this strategy, the resulting correct drawing may deviate substantially from $\mathcal{D}^{inc}$ at some vertices, specially if the drawing is large enough. However, from the good convergence behavior seen in the experiments above, we judge this approximation as good enough to avoid local minima.

9 Correction of Other Topologies

Real scenes of polyhedra differ substantially from the model assumed in Section 2. Hidden edges are not visible when the objects are opaque and if several objects are present, they may occlude one another (Fig. 10a). When this happens, the incidence structure of the line drawing is not that of a spherical polyhedron. This can be seen with the help of Fig. 10: although the drawing in the middle is a projection of the scene to the left, all we can reconstruct is a collection of objects as those to the right since only the topmost portions will be visible. The topology of these objects is clearly not spherical. Each of them is actually homeomorphic to a disk possibly containing any number of holes in it. Thus, in practice, the required tool is an algorithm able to correct drawings whose incidence structure is that of a *polyhedral disk*, maybe with polygonal holes in it. Hereafter, this plane faced object will be called *polydisk* for short, and its vertices and edges will be referred to as *boundary* or *interior* according to whether they correspond or not to the boundaries of the spatial objects, as seen from the center of projection. Fortunately, the drawings of these objects are also correctable, as the results in [28] imply that any surface composed of polygons that is homeomorphic to a disk with possible holes is resolvable.

Moreover, note that when several polydisks are present in the drawing, we can treat each one separately in the same way. For this, we only need to have each one of them identified, which can be done by collecting all regions delimited by boundary edges after an edge-labeling algorithm has been applied. However, an issue may arise here. If we correct each polydisk separately, the final boundaries of neighboring polydisks may not coincide as they originally did. Depending on the application this disparity might be irrelevant. For example, if all we want is an approximate reconstruction of the objects in the scene, these small errors may be acceptable. On the contrary, if they are not, we propose the following strategy to make the boundaries coincident again. It will only be valid for trihedral scenes, but we note that this is the case that arises when all faces lie on planes in general position.

First, observe that, if the objects are trihedral, only one of the following three situations occurs (Figs. 11a, 11b, and 11c, respectively): a boundary vertex either has 1) no incident interior edge, or 2) only one interior edge on one side, or 3) one interior edge on both sides. The correction algorithm will separate the boundaries as depicted in Figs. 11d, 11e, and 11f, yielding two copies of the original vertex, say v_1 and v_2. In the first two cases, the boundaries can be made coincident again by moving v_2, the vertex with no incident interior edge, to the position of v_1. Note that this will not alter the correctness of the polydisk of v_2 (Figs. 11g and 11h). In the third case, we can move v_1 and v_2 to the point of intersection of their interior edges without altering the correctness of their respective polydisks (Fig. 11i).

This strategy has been implemented and Fig. 12 shows the results on a synthetic polyhedral scene. In this example, the scenes to the left have been projected to yield the

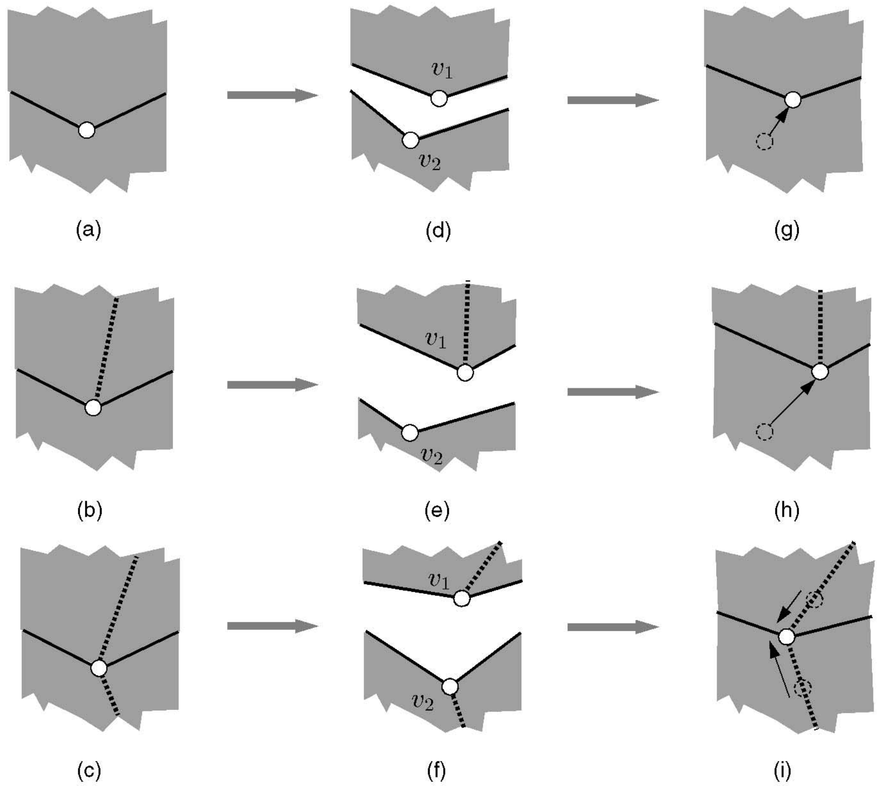


Fig. 11. Independent correction of neighboring polydisks (left) will separate their boundaries (center) but these can be made coincident by properly moving the vertices again (right) while preserving the overall correctness. Interior and boundary edges are indicated in dashed and solid lines, respectively.

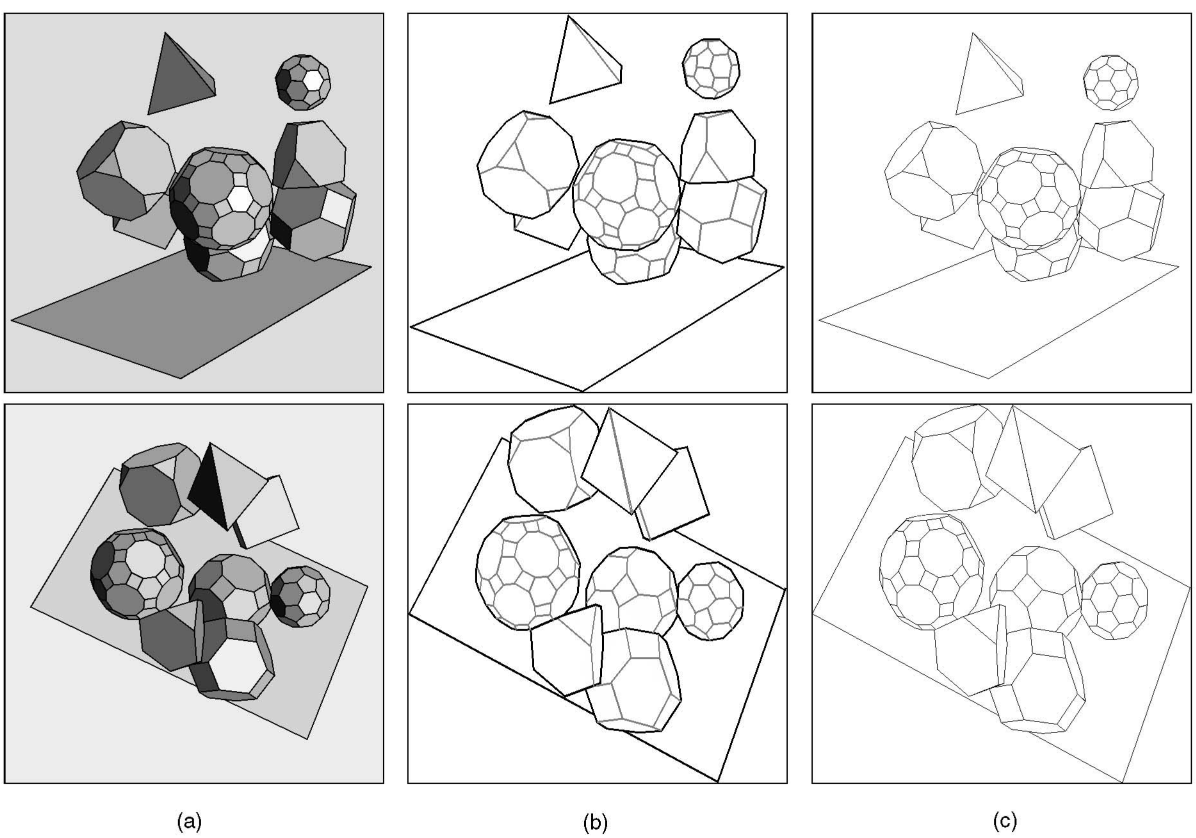


Fig. 12. Two views of a same polyhedral scene (a) together with two incorrect drawings of them, with boundary edges labelled in black (b), and the final corrections (c).

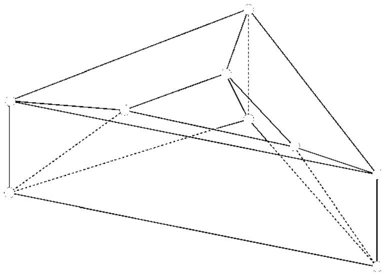


Fig. 13. An unresolvable torus.

drawings in the center, whose vertices have been randomly perturbed to yield the shown incorrect configurations. Boundary edges delimiting each polydisk are marked in thick black lines while interior edges are marked in gray. The resulting correct polydisks are shown in Fig. 12c.

Unfortunately, there are topological structures that cannot be corrected. Namely, polyhedra with a genus equal or greater than one do not have a resolvable sequence in general. A simple counterexample is given by the torus in Fig. 13. Since all vertices are incident to exactly four faces, condition (**C1**) in Section 5 will be necessarily violated at some vertex in any sequence. Although we note that this type of incidence structures will never arise if the projected objects are opaque, we find this is an interesting open problem for future consideration.

10 Conclusions and Future Work

This paper has presented a new approach to correct incorrect projections of polyhedra and has discussed its contributions with respect to the previous method by Sugihara. Surprisingly, our improvements have only been possible thanks to Sugihara's latest finding of the resolvable sequence, offering an unexpected new application of this result. To conclude, it is worth to mention two points deserving further attention.

Although the initial drawing we propose to start the gradient search seems a fairly good approximation of the result, the minimization is still not guaranteed to converge to the global minimum. To mend this up, one can always start the search at several different initial estimations, each derived from a different resolvable sequence of the same incidence structure, and select the best corrected drawing.

Another possibility could be to derive a polynomial (rather than rational) parameterization, by working in projective instead of affine coordinates, and attempt to find the global minimum through interval arithmetic [29] or Bézier-clipping techniques [30].

Acknowledgments

The authors were introduced to "polyhedral determination" by Walter Whiteley while L. Ros was staying with him in 1997. Also, Kokichi Sugihara informed the authors about the general resolvable sequence during a stay of L. Ros with him in 1998. The authors are very grateful to them, as many of the ideas presented here developed from the conversations they held. This paper is an extended and updated version of [1], which was presented in the ICRA 2001 conference. This research has been partially funded by the Spanish CICYT under contract TAP-99-1086-C03-01.

References

[1] L. Ros and F. Thomas, "Correcting Polyhedral Projections for Scene Reconstruction," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. II, pp. 2126-2133, May 2001.

[2] A.K. Mackworth, "Interpreting Pictures of Polyhedral Scenes," *Artificial Intelligence,* vol. 4, pp. 121-137, 1973.

[3] K. Sugihara, "Mathematical Structures of Line Drawings of polyhedrons—Towards Man-Machine Communication by Means of Line Drawings," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 4, pp. 458-469, 1982.

[4] K. Sugihara, "A Necessary and Sufficient Condition for a Picture to Represent a Polyhedral Scene," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 6, pp. 578-586, 1984.

[5] K. Sugihara, *Machine Interpretation of Line Drawings.* The MIT Press, 1986.

[6] W. Whiteley, "Weavings, Sections and Projections of Spherical Polyhedra," *Discrete Applied Math.,* vol. 32, pp. 275-294, 1991.

[7] L. Ros, "A Kinematic-Geometric Approach to Spatial Interpretation of Line Drawings," PhD thesis, Polytechnic Univ. of Catalonia, May 2000, Available at http://www-iri.upc.es/people/ros.

[8] L. Ros and F. Thomas, "Analysing Spatial Realizability of Line Drawings through Edge-Concurrence Tests," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. IV, pp. 3559-3566, May 1998.

[9] L.S. Penrose and R. Penrose, "Impossible Objects: A Special Type of Visual Illusion," *British J. of Psychology,* vol. 49, pp. 31-33, 1958.

[10] S.W. Draper, "The Penrose Triangle and a Family of Related Figures," *Perception,* vol. 7, pp. 283-296, 1978.

[11] D.A. Huffman, "Impossible Objects as Nonsense Sentences," *Machine Intelligence,* vol. 6, pp. 295-323, 1971.

[12] B. Ernst, *Adventures with Impossible Figures.* Parkwest Publications, May 1987.

[13] L.M. Kirousis and C.H. Papadimitriou, "The Complexity of Recognizing Polyhedral Scenes," *J. Computer and System Sciences,* vol. 37, pp. 14-38, 1988.

[14] D. Waltz, "Understanding Line Drawings of Scenes with Shadows," *The Psychology of Computer Vision,* P.H. Winston, ed., pp. 19-91, McGraw Hill, 1975.

[15] E.R. Hancock and J. Kittler, "Edge Labelling Using Dictionary-Based Relaxation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 12, pp. 165-181, 1990.

[16] R. Myers and E.R. Hancock, "Genetic Algorithms for Ambiguous Labelling Problems," *Pattern Recognition,* vol. 33, pp. 685-704, 2000.

[17] P. Parodi and V. Torre, "On the Complexity of Labelling Perspective Projections of Polyhedral Scenes," *Artificial Intelligence,* vol. 70, no. 1-2, pp. 239-276, Oct. 1994.

[18] P. Parodi, R. Lancewicki, A. Vijh, and J.K. Tsotsos, "Empirically-Derived Estimates of the Complexity of Labelling Line Drawings of Polyhedral Scenes," *Artificial Intelligence,* vol. 105, no. 1-2, pp. 47-75, 1998.

[19] D.A. Huffman, "Realizable Configurations of Lines in Pictures of Polyhedra," *Machine Intelligence,* vol. 8, 1977.

[20] J.C. Maxwell, "On Reciprocal Figures and Diagrams of Forces," *Philosophical Magazine,* series 4, vol. 27, pp. 250-261, 1864.

[21] J. Ponce and I. Shimshoni, "An Algebraic Approach to Line-Drawing Analysis in the Presence of Uncertainty," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 1786-1791, May 1992.

[22] I. Shimshoni, "Interpreting Images of Polyhedral Objects in the Presence of Uncertainty," PhD thesis, Univ. of Illinois at Urbana-Champaign, 1995.

[23] W. Whiteley, "Some Matroids on Hypergraphs with Applications to Scene Analysis and Geometry," *Discrete and Computational Geometry,* vol. 4, pp. 75-95, 1988.

[24] W. Whiteley, "From a Line Drawing to a Polyhedron," *J. Math. Psychology,* vol. 31, pp. 441-448, 1987.

[25] W. Whiteley, "How to Design or Describe a Polyhedron," *J. Intelligent and Robotic Systems,* vol. 11, pp. 135-160, 1994.

[26] H. Crapo and W. Whiteley, "Plane Self Stresses and Projected Polyhedra I: The Basic Pattern," *Structural Topology,* vol. 20, pp. 55-78, 1993.

[27] T. Schlik and A. Fogelson, "TNPACK-A Truncated Newton Minimization Package for Large-Scale Problems: I. Algorithm and Usage," *ACM Trans. Math. Software,* vol. 18, no. 1, pp. 46-70, Mar. 1992.

[28] K. Sugihara, "Resolvable Representation of Polyhedra," *Discrete and Computational Geometry,* vol. 21, no. 2, pp. 243-255, 1999.

[29] E. Hansen, "Pure and Applied Mathematics," *Global Optimization Using Interval Analysis.* New York: Marcel Decker, Inc., 1992.
[30] C. Bombín, L. Ros, and F. Thomas, "A Concise Bézier Clipping Technique for Solving Inverse Kinematics Problems," *Advances in Robot Kinematics,* J. Lenarcic and M.M. Stanisic, eds., pp. 53-60, Kluwer Academic, 2000.

Lluís Ros received the mechanical engineering degree in 1992 and the PhD degree (with honors) in industrial engineering in 2000, both from the Polytechnic University of Catalonia. From 1993 to 1996, he worked with the Control of Resources Group of the Cybernetics Institute in Barcelona, involved in the application of constraint logic programming to the control of electric and water networks. Since August 2000, he has been a research scientist at the Institut de Robòtica i Informàtica Industrial of the Spanish High Council for Scientific Research. His current research interests are in geometry and kinematics, with applications to robotics, computer graphics, and machine vision. His Web page is http://www-iri.upc.es/people/ros.

Federico Thomas received the BS degree in telecommunications engineering in 1984 and the PhD degree (with honors) in computer science in 1988, both from the Polytechnic University of Catalonia. Since March 1990, he has been a research scientist at the Institut de Robòtica i Informàtica Industrial of the Spanish High Council for Scientific Research. His research interests are in geometry and kinematics, with applications to robotics, computer graphics, and machine vision. He has published more than 40 research papers in international journals and conferences. His Web page is http://www-iri.upc.es/people/thomas.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** http://computer.org/publications/dlib.