3D Shape Reconstruction by Using Vanishing Points

Pietro Parodi and Giulia Piccioli

Abstract—This paper investigates the quantitative reconstruction of the 3D structure of a scene from a line drawing, by using the geometrical constraints provided by the location of vanishing points. The additional information on vanishing points allows the design of an algorithm which has several advantages with respect to the usual approach based on a reduction to Linear Programming (Sugihara, 1982). These advantages range from a lower computational complexity to error tolerance and exact reconstruction of the 3D-geometry of the objects. These features make the algorithm a useful tool for the quantitative analysis of real-world images, which is useful for several tasks from scene understanding to automatic vehicle guidance.

Index Terms—Reconstruction of 3D shape, vanishing point, line drawing, incidence structure, spatial structure, labeling, realizability.

1 Introduction

The automatic recovery of the 3D structure of a scene from a single view is one of the major goals of computer vision. This goal has often been pursued by the analysis of the line drawing associated with an image, that is the sketchy version of the image in which only the information on the visible edges of the scene is kept.

The key step for the interpretation of line drawings is thought to be the consistent labeling of its segments as the projection of convex ("+"), concave ("–"), or contour ("→") edges [1], [2], [3]. Labelability is a necessary condition for a line drawing to be realizable as the projection of a polyhedral scene. Sugihara [4] found a necessary and sufficient condition for a *labeled* line drawing to be realizable as the orthographic/perspective projection of a polyhedral scene, and presented an algorithm which actually found a 3D reconstruction (if possible). The method was by reduction to Linear Programming, and it therefore took polynomial time.

The classical approach based on a reduction to Linear Programming has, however, several drawbacks:

1) the reconstruction of the scene is not unique;
2) the slightest error in the line drawing can make the reconstruction impossible, and a procedure to correct the location of junctions is necessary;
3) Linear Programming is $\mathcal{P}$-complete, which means that it is presumably impossible to find efficient parallel algorithms that solve it [5], [6].

This paper studies an interesting special case of the reconstruction problem, in which the location of the vanishing points is known. It is an extension to perspective projections of general scenes with planar surfaces and to an arbitrary number of admissible directions of a previous result by Kirousis and Papadimitriou [7] on the realizability of Manhattan (or Legoland) scenes, that is scenes where edges have only three admissible orthogonal directions.

The case treated in this paper seems relevant in practice, as the vanishing points can often be obtained from the image itself by standard techniques (see [8], [9], [10], [11], [12], [13]).[1] This analysis is further motivated by the fact that in [14] it was recently shown that once the location of the vanishing points of the line drawing (of a trihedral, solid scene) is known, the labeling problem has polynomial complexity, and it is important to exploit the constraints provided by the vanishing points in the quantitative reconstruction stage as well.

The main result of this paper is the design of an algorithm for recovering the 3D shape of a scene of planar panels from a labeled line drawing. The algorithm has the following good features:

1) the 3D-geometry of all the objects which appear in the scene is reconstructed in a unique way, apart from a scale factor;
2) the reconstruction tolerates uncertainty on junction position, so that no correction mechanism is necessary;
3) the complexity of the algorithm is lower than the Linear Programming method by Sugihara, and the algorithm is conceptually simpler;
4) the algorithm can be easily parallelized;
5) in the case of trihedral scenes, a labeling can be obtained in polynomial time [14].

The algorithm has been successfully tested on line drawings extracted from real images according to several existing techniques, thus showing its usefulness as a general tool for 3D quantitative reconstruction.

2 Preliminaries

The object under study is a labeled **line drawing** (Fig. 1), which is here to be interpreted as the perspective projection of the visible edges of a 3D scene made up of planar surfaces. The reader is referred to any of the papers [1], [2], [3], [14] for the labeling step.

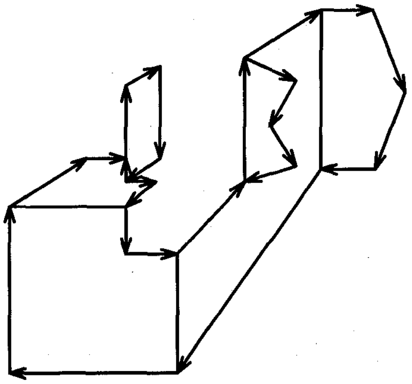

Fig. 1. An example of a labeled line drawing.

The 3D elements of the scene will be called *vertices*, *edges*, and *planar faces*, whereas the corresponding 2D elements of the line drawing will be called *junctions*, *segments*, and *polygons*. Junctions can have an arbitrary number of incident segments.

The *scene* is assumed to be composed by a finite number of planar panels, which are to be regarded, as in [4], as opaque 2D

[1] The fastest general algorithms for the extraction of vanishing points are of complexity $O(N^2)$, where N is the number of segments. If some special assumption about the scene are satisfied, $O(N)$ methods can be used [8]. In this work, we have used—whenever it was possible—a combination of the algorithm [8] with a method [14] which allows to find all the vanishing points of the line drawing from a small set of known vanishing points. The resulting algorithm takes time $O(N)$.

• *P. Parodi and G. Piccioli are with the Department of Physics of the University of Genova, Via Dodecaneso 33, 16146 Genova, Italy.*
E-mail: {parodip,picciolig}@ge.infn.it.

Manuscript received June 14, 1994; revised June 11, 1995. Recommended for acceptance by M.D. Levine.
For information on obtaining reprints of this article, please send e-mail to: transactions@computer.org, and reference IEEECS Log Number P95154.

polygonal areas with possible polygonal holes. This assumption includes, as a special case, polyhedral scenes (in that they can always be *simulated* as scenes of planar panels.) It is assumed that no planar panel lies on a planar surface containing the optical center. It is also assumed that every visible planar panel (except for the background) has at least two nonparallel visible edges, which is usually verified in nonpathological cases.

Following [1], [2], [15], segments are assigned a label describing some 3D physical properties. A collapsed label is here adopted: both "+" and "–" are indicated as an *unlabeled segment*, so that every *connecting segment* (that is, a segment which is the projection of a dihedral angle) is unlabeled; a label "→" is assigned to segments (called *occluding segments*) which are the projection of convex edges such that only one face is visible (the one at the right side of the arrow). Other examples of collapsed labelings can be found in Mackworth (PhD thesis) and [16].[2]

The labeled line drawing already contains much information on the incidence and depth relations between the geometrical elements of the scene (vertices, edges, panels). Such relations are made explicit in the incidence structure and in the spatial structure associated with the line drawing. The *incidence structure* I, which can be defined as a triple $I = (V, F, R)$, where V is the set of vertices of the scene (including points where an edge disappears behind a planar panel), F is the set of planar panels, and R is a set of incidence pairs (v, f) such that $v \in V, f \in F$ and v lies on f, and the *spatial structure* can be defined as[3] the pair $S = (I, T)$ where I is the incidence structure and T is a set of depth relations of the form (v, v', δ) such that $v, v' \in V$ are projected into the same point of the image plane, and δ is taken from the set $\{F, PF\}$. (v, v', F) means "v has the same depth as v' or is in front of v'," and (v, v', PF) means "v is properly in front of v'." A detailed treatment of the construction of the spatial structure can be found in [4]. The general idea is given in Section 4 of this paper.

The concept of *components* of the incidence structure can be introduced. Each component can be seen as a different object (or a set of physically connected objects).[4] The **incidence graph** $G(I)$ associated with the incidence structure I is first defined.

DEFINITION 1. *The incidence graph $G(I)$ associated with $I = (V, F, R)$ is the bipartite graph whose nodes are the elements of V (vertices) and the elements of F (panels) and s.t. there is an arc between node $v \in V$ and node $f \in F$ if-f $(v, f) \in R$ (that is, if v lies on f).*

The components of the incidence structure $I = (V, F, R)$ can now be formally defined as the subsets $I_j = (V_j, F_j, R_j)$ ($j = 1, \ldots, p$) of I associated with the connected components (in the graph-theoretic sense) of the graph $G(I)$. Observe that the sets $V_1, \ldots V_p$ (respectively: $F_1, \ldots F_p$; $R_1, \ldots R_p$) are, by construction, pairwise disjoint.

The extraction of the components of I, which is possible in the general case, becomes useful once the information on vanishing points is provided, as shown in the following section.

3 Using Vanishing Points to Propagate the 3D Information

This section shows that, in case the vanishing points are given, and the 3D location of a vertex v is known, the 3D location of any other vertex belonging to the same component as v can be determined in linear time.

Some well known properties involving perspective projection are first recalled.

Assume that a system of reference $(O; X, Y, Z)$ solid with the imaging device is given, so that the origin of the reference system O coincides with the optical center and that the image plane is defined by the equation $Z = f$.

There is a one-to-one correspondence between possible orientations and the associated vanishing points: 3D edges oriented according to the unit vector $\vec{e} = (\alpha, \beta, \gamma)$ are projected into 2D segments converging on the vanishing point $VP(\vec{e})$: $VP(\vec{e}) = f\left(\frac{\alpha}{\gamma}, \frac{\beta}{\gamma}, 1\right)$ if $\vec{e}$ is not parallel to the image plane, otherwise $VP(\vec{e})$ lies at infinity and can be represented by its projective coordinates $(\alpha, \beta, 0)$.

The line through two distinct vanishing points $VP(\vec{e}_1)$ and $VP(\vec{e}_2)$ is called *horizon line* and represents the intersection of the image plane with the plane through O whose normal vector is proportional to $\vec{e}_1 \wedge \vec{e}_2$. All vanishing points of edges which lie on a plane parallel to such plane must belong to this horizon line.

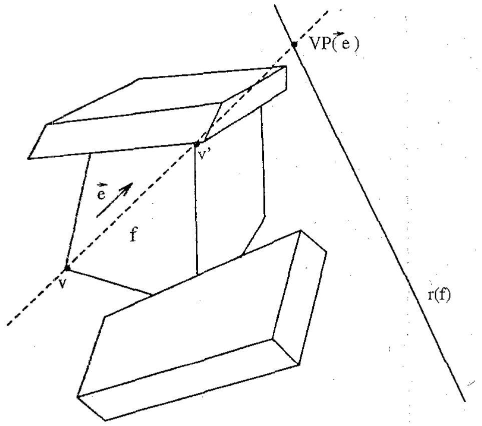

Fig. 2. A pictorial description of the method for propagating the 3D information between two vertices belonging to the same planar panel.

The reconstruction algorithm is based on this observation: if two vertices v_0 and v_n belong to the same component, there is always a path $v_0, f_0, v_1, f_1, \ldots, f_{n-1}, v_n$ in the graph $G(I)$ leading from v_0 to v_n (or *vice versa*); in such a path, two consecutive vertices always share a planar panel, so that the problem of determining the 3D location of v_n from the 3D location of v_0 is reduced to the problem of determining the 3D location of a vertex in a planar panel from the 3D location of another vertex belonging to the same planar panel, which can be solved as follows. If v and v' belong to the same planar panel f, then

$$v' = v \pm \frac{v_z \,||\pi(v) - \pi(v')||}{f \,||VP - \pi(v')||}(x, y, f) \tag{1}$$

where v_z is the z-component of v, $||\pi(v) - \pi(v')||$ is the 2D distance

[2] The collapsed labeling used here is the *minimal* labeling which keeps the information on which edges lie on which faces, and this is exactly what we need for our reconstruction task. The problem of labeling a line drawing of a trihedral scene (the case examined in [7]) can be shown to be $\mathcal{NP}$-complete even for this kind of labeling.

[3] The following definition slightly differs from the one given in [4].

[4] This means no abuse, as if a line drawing $\mathcal{L}$ is realizable as the perspective projection of a polyhedral scene, there is always a realization of $\mathcal{L}$ in which all the components of the incidence structure correspond to physically distinct objects (or sets of touching objects).

between the projections $\pi(v)$ and $\pi(v')$ of v and v' in the image plane, $||VP - \pi(v')||$ is the 2D distance in the image plane between $\pi(v)$ and the point $VP = (x, y)$, which is the intersection of the line through $\pi(v)$ and $\pi(v')$ with the horizon line $r(f)$ of f (Fig. 2).[5] Observe that VP is the vanishing point associated with the direction of the 3D line through v and v'.

The sign in (1) is chosen so that vertex v' actually lies on the line through the center of projection O and $\pi(v')$.

Observe that the location of junctions and vanishing points is always affected by an error, which is at least the discretization step. Such error is propagated by (1).

An iterative application of (1) allows to determine the 3D location of all vertices in the same component by the knowledge of a single vertex, or, which is the same, to express the 3D location of all vertices in the same component in terms of a single scale factor.

Otherwise speaking, a scale factor t_j can be associated with every component I_j ($j = 1, \ldots, p$), such that any vertex $v_i^{(j)} \in V_j$ can be written as $v_i^{(j)} = t_j(\alpha_i^j, \beta_i^j, \gamma_i^j)$ where $\alpha_i^j, \beta_i^j, \gamma_i^j$ do not depend upon t_j.

4 The 3D Reconstruction Algorithm

This section provides a detailed description of the algorithm that checks the realizability and finds a realization of a line drawing as the perspective projection of a scene of planar panels.

The input of the algorithm is a labeled line drawing with N segments, along with a set of n vanishing points, and its output is a set of vertices, each with its 3D location in space. The main steps are the following.

4.1 Extracting the Spatial Structure and Finding the Components

The spatial structure is extracted similarly to [4], the main difference being that the information on vanishing points allows to put all depth relations as relations between vertices (and not between vertices and panels). The general idea is the following.

A panel f is created for every polygon $\pi(f)$, except for the background. Since every planar panel has at least two nonparallel visible edges, the horizon line $r(f)$ associated with panel f can be determined for all fs.

A number of vertices is created for every junction, roughly according to the principle that every incident occluding segment s individuates a (possible) depth discontinuity between the planar surfaces f_1, f_2 at the two sides of s, and two (possibly coincident) vertices v_1, v_2 such that $(v_1, v_2, F) \in T$ are created to represent the discontinuity. If the edge projected into s is parallel to the occluded panel, the relation must be strengthened to $(v_1, v_2, PF) \in T$ (Fig. 3).

The next step is finding the components of the scene, that is the connected components of the incidence graph (see Definition 1). This can be done in time $O(N)$ by means of a depth-first search.

4.2 Propagating the 3D Information to All the Vertices of the Same Component

As shown in Section 3, an iterative application of (1) allows to locate all vertices belonging to the same component modulo a scale factor in time $O(N)$.

In practice, the propagation of the 3D information is driven by a depth-first search on the incidence graph $G(I)$. A starting vertex-node is elected at random and its 3D location is assumed to be coincident with its 2D projection; every time a new vertex-node is visited, (1) is applied until all vertex-nodes in the same connected component of $G(I)$ have been reached.[6] Another starting node is then elected which belongs to a new component, and the propagation process starts again. This goes on until all the components have been explored.

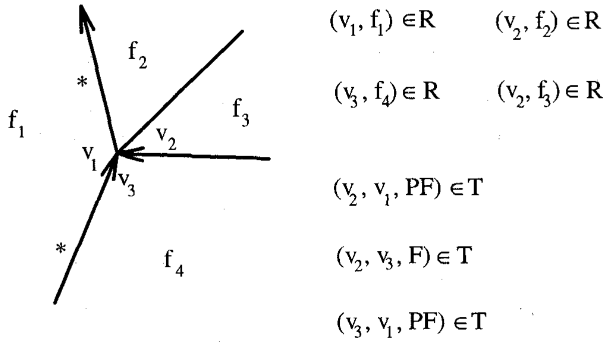

Fig. 3. The construction of the spatial structure is junction-based. In the case above, three vertices must be created (one for each occluding segment). The edges projected in segments marked as "*" are assumed to be parallel to f_1.

4.3 Checking Realizability

Step 2 has assigned a 3D location to all vertices modulo a scale factor t_i ($i = 1, \ldots, p$), but the assigned 3D locations may not be consistent with the geometrical constraints implied by the incidence and spatial structure associated with the labeled line drawing. If so, the current labeling must be rejected and replaced with another labeling. It is therefore important to check whether the line drawing, with the given labeling, is realizable as the perspective projection of a planar-surfaces scene, that is whether a p-tuple $(t_1, \ldots t_p)$ of scale factors (one for each component) can be found such that the resulting 3D locations of all vertices simultaneously satisfy all the incidence and depth relations specified in the sets R and T of the spatial structure [4].

The realizability check can be subdivided into the three following independent tests, which are to be performed in sequence. If a line drawing passes all the three tests it is realizable.

4.3.1 Checking the Incidence Relations

The consistency of the incidence relations in R can be checked in linear time as follows. An arbitrary scale factor is assigned to every component (the components of the spatial structure can be checked independently), and the quantity $C(v_i, f_j) = a_j x_i + b_j y_i + c_j z_i$ ((a_j, b_j, c_j) is the unit vector of f_j and x_i, y_i, z_i are the 3D coordinates of v_i) is computed for all pairs $(v_i, f_j) \in R$. All $C(\cdot, f_j)$s associated with f_j must coincide *within the experimental error*.

4.3.2 Checking the Depth Relations Inside Each Component

Depth relations involving vertices belonging to the same component do not depend on the value of the scale factors and can be checked in linear time as follows. An arbitrary scale factor is assigned to every component, and for all pairs (v, v', F) (respectively, (v, v', PF)) it is checked whether $Z(v) \le Z(v')$ (respectively, $Z(v) < Z(v')$), where $Z(v)$ and $Z(v')$ are the Z-

[5] The extension to the case in which $r(f)$ or VP (or both) are at infinity is straightforward.

[6] Observe that the error by which every vertex is affected depends in general on the way the vertex is visited by the depth-first search. We have not investigated the possibility of minimizing this error by a suitable planning of the propagation.

coordinates of v and v'.[7]

4.3.3 Checking the Depth Relations Between Different Components

All depth relations between vertices of components i and j can be translated into inequalities about the scale factors t_i and t_j associated with the components. At most two inequalities (the strictest ones) are needed (for simplicity's sake, all inequalities are written as "≥," but some ">" may occur): $t_j \geq M(j, i)\ t_i$ and $t_i \geq M(i, j)\ t_j$ where $M(j, i)$ and $M(i, j)$ are independent of the scale factors.[8]

By introducing new variables $u_i = \log t_i$ and constants $m(i, j) = \log M(i, j)$, the above inequalities can be written as:

$$u_j \geq u_i + m(j, i) \text{ and } u_i \geq u_j + m(i, j) \qquad (2)$$

The set of inequalities thus obtained is a so-called *system of difference constraints*, whose solvability can be checked by constructing a weighted directed graph G, called the *occlusion graph*, which has the variables u_i as nodes, and an arc from u_j to u_i whenever an inequality (2) occurs. The line drawing is then realizable if-f the system of difference constraints is solvable, which is true if-f all cycles in the occlusion graph G have negative total weight (possibly containing some strict in-equalities) or have weight zero (without containing, however, any strict inequality). This can be checked in time $O(p^3)$ where p is the number of components.

4.4 Finding a 3D Realization

If the line drawing is realizable, a solution can be obtained as follows. A node u^* is added to the constraint graph G, and for all nodes u_i ($i = 1, \ldots p$) an arc of weight 0 leaving u^* is added to G (u^* acts therefore as a single source). No cycles are added in this way, in that there are no arcs entering u^*. If all inequalities were weak, a solution to the system is represented by the shortest-path weights from u^* to u_i for all $i = 1, \ldots p$. If there are some strict inequalities, they can be turned into weak inequalities by adding a quantity ε to all strict inequalities, small enough to keep negative all negative cycles. In any case, a solution can be found within the time bound $O(pq)$, where p is the number of components and q is the number of inequalities ($q = O(p^2)$ in general).

The following proposition has therefore been proved:

PROPOSITION 1. *Once the vanishing points are given, the realizability of a line drawing as the perspective projection of a planar-surfaces scene can be checked in time $O(N + p^3)$ where N is the number of segments of the line drawing and p is the number of components; a 3D realization, if it exists, can be found within the same time bounds.*

It is often the case that for all the components represented by the line drawing the 3D coordinates of one of its vertices, or the 3D length of one of its edges, is known a priori. This allows us to determine the scale factor of all components, with no need to guess a possible p-tuple by the above algorithm. In this case, the following corollary holds true.

Corollary 1. *Let $\mathcal{L}$ be a labeled line drawing. Assume that the location of the vanishing points in the image plane is given. Assume that a scale factor is given for all components. It is then possible to give a unique and consistent (if any) reconstruction of the 3D scene represented by the line drawing in time $O(N)$, where N is the number of segments of the line drawing.*

In the midway case when only *some* of the scale factors are known, one has to apply Proposition 1 in order to get proper values for the unknown ones; in that case the number of components p in the proposition must be replaced with the number of unknown scale factors.

5 Experimental Results

In the previous section an algorithm for the 3D reconstruction of a scene from a line drawing has been outlined. Such an algorithm can be used as a general tool for 3D quantitative reconstruction, and can be applied in practice whenever it is possible to obtain a line drawing from a real image, independently of the way the line drawing has been obtained, provided that the vanishing points are given or can be extracted.

The problem of obtaining a faithful line drawing under general circumstances from a real image is an open problem in Computer Vision. There are, however, several cases in which line drawings can actually be obtained, e.g., by exploiting some additional knowledge on the scene. Although this paper is only concerned with the reconstruction of the 3D information from a line drawing which is assumed to be already extracted and labeled, we always have indicated the techniques we have used to extract and label the line drawing; and more than that, we have classified the experimental results according to the very techniques used for the extraction stage: this has been done both to make explicit the range of application of our algorithm and to explain what specific problems may arise for any of these paradigmatic situations.

5.1 Line Drawings Extracted in a Bottom-Up Fashion

For well-contrasted indoor scenes, it is sometimes possible automatically to extract a line drawing in a purely bottom-up way; such a line drawing, however, is likely to suffer from the presence of missing edges, spurious segments and ill-detected junctions. It is useful to introduce an additional label: Along with plain (connecting) and arrow (occluding) segments, *dashed* segments are considered, which may correspond to cracks, shadows, drawings on the panel, or simply spurious segments.[9]

Once such a labeling is obtained, the reconstruction task can be carried out as explained in Section 4, ignoring dashed segments in the first stage. It is, however, trivial to find the 3D location of possible cracks, shadows, or drawings corresponding to these segments, *after* the 3D recontruction has been carried out and the planar panels have been located in space.

Two examples of the application of our algorithm to line drawings automatically extracted are shown in Figs. 4 and 5.

Fig. 4A depicts a rather complicated Legoland scene (that is, a scene with only three vanishing points corresponding to orthogonal 3D directions). After some low-level processing of the image[10] the vanishing points were extracted as described in [20]. Junctions were recovered by exploiting the information on vanishing points as in [8], thus obtaining the line drawing of Fig. 4B. The line

[7] The same as in orthographic projection: Whatever the viewpoint, being closer means having lower Z-coordinate.

[8] The $M(i, j)$s are of course given with some error. The correct choice is taking the strictest of the inequalities, each of which must, however, be the weakest in relation to the experimental error: $M(i, j) = \max_k\{M^{(k)} - \Delta M^{(k)}\}$.

[9] A reasonable labeling of segments as plain, arrow, and dashed can be achieved by Constraint Satisfaction methods [17], along with some heuristic rules for pruning the set of possible labelings. It is not possible to apply linear time algorithms as the one described in [14], and this is one of the reasons for which the resort to model-based reconstruction as in Section 5.2 is more profitable.

[10] This and all the other real-world images shown in this paper have undergone the following elaborations: edge extraction by a Canny [18] edge detector; chain following; polygonal approximation by Pavlidis' algorithm [19]. The different high-level elaborations are specified in the text.

drawing has many missing and spurious segments; yet, the labeling shown in Fig. 4B allows correctly to recover the main features of the 3D scene. There is only one component in the labeled line drawing, apart from the 'window' through which the scene is viewed, which is irrelevant. It is therefore sufficient to determine a single scale factor in order to have a complete 3D reconstruction. The scale factor was in our case determined by the knowledge that the width of the SUN server is 0.56 ± 0.01 *m*, from which we can assign a unique 3D location to all vertices in Fig. 4B. As a double check, the height of the closet on the right as determined from our algorithm, 1.75 ± 0.30 *m*, was compared to its measured height, 1.95 ± 0.02 *m*.

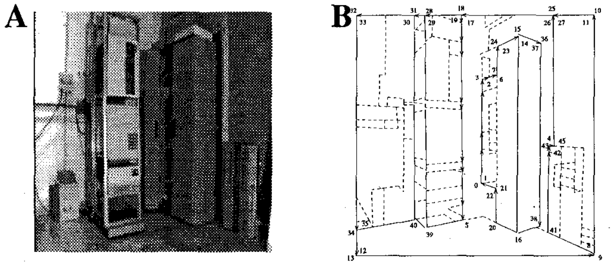

Fig. 4. (A) A picture of a laboratory. (B) The labeled line drawing extracted by the algorithm in [8], along with the 3D vertices produced by the reconstruction algorithm.

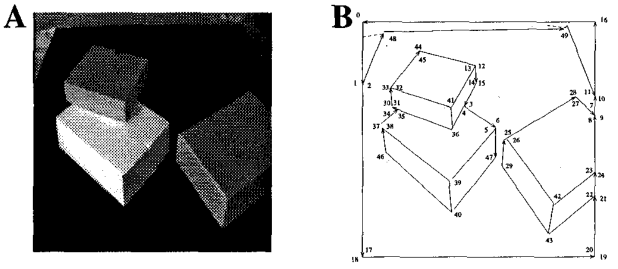

Fig. 5. (A) A photograph of a set of boxes on a table. (B) The line drawing extracted by the algorithm in [21], labeled, and the 3D vertices extracted by the reconstruction algorithm.

Fig. 5A is the picture of a set of boxes laid upon a table. It is not a Legoland scene; in these conditions the extraction of the line drawing can be best performed by a general algorithm as the one described in [21]. The result is shown in Fig 5B. Observe that this line drawing is rather faithful, in that no segment is missing and there are only two spurious segments, close to the two visible table vertices. After labeling it as shown in the same figure, the line drawing was given as an input to our algorithm, along with the vanishing points (manually computed as the intersection of those lines which are known to be parallel in the scene), in order to recover the 3D positions of all vertices, marked by their identification numbers in Fig. 5B. There is only one component; as a scale factor for this component we have chosen the length of the visible side of the table, which has been measured to be 1.00 ± 0.01 *m*. Table 1 reports the size of the three boxes as computed by the location of the 3D vertices.

In both examples, the focal length was unknown and has been computed according to the calibration algorithm shown in [22] which gave two focal lengths, f = 918 ± 20 *pxl* and f = 481 ± 10 *pxl* respectively for the images of Figs. 5 and 4; the error on junction locations has been considered to be equal to the threshold under which the end-points of two segments are considered to belong to the same junction (8 *pxl* in the case of Fig. 4, 10 *pxl* in the case of Fig. 5) [8], [21].

5.2 Line Drawings Extracted by Model-Based Techniques

In most images of outdoor and indoor scenes, it is not possible to extract a line drawing in a bottom-up automatic way. There are, however, methods by which is possible to extract a line drawing by model-based feature extraction schemes [23], [24]: The extraction is guided by prior knowledge on the environment. This seems the most promising field of application of our algorithm.

TABLE 1
THE THREE DIMENSIONS (WIDTH x HEIGHT x LENGTH, IN *CM*) OF THE THREE BOXES OF FIG. 5A AS COMPUTED FROM THE 3D COORDINATES OF ALL THE VERTICES SHOWN IN FIG. 5B, WHICH ARE THE TRUE OUTPUT OF THE ALGORITHM.

BOXES	DIMENSIONS
TOP-LEFT	24 ± 4 x 9 ± 3 x 27 ± 6 *cm*
BOTTOM-LEFT	35 ± 3 x 14 ± 3 x 29 ± 4 *cm*
RIGHT	34 ± 4 x 16 ± 5 x 41 ± 4 *cm*

The computed size of the boxes is consistent with their real size.

It is useful to spend a few words about these schemes. In [23] a context constrained matching of hierarchical CAD based models for outdoor scene interpretation is proposed. The system can deal with outdoor scenes with artificial objects which may have various shapes and sizes, and these are not in finite number but vary according to a number of parameters. Context information (e.g., a car always lies on the ground plane) is used to constrain the search space and reduce the number of parameters. A similar approach is adopted in [24], where the main features of the scene (road, cars, buildings, trees...) are identified by independent, ad hoc criteria and their parameters (width, length, height...) are computed after the identification process.

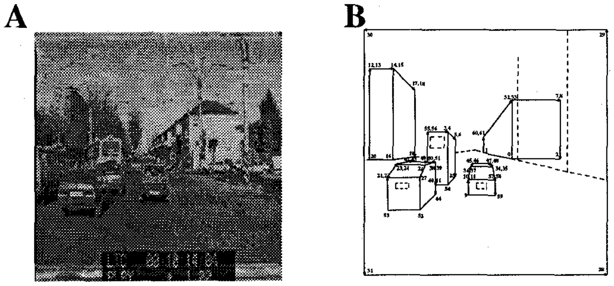

Fig. 6. (A) A frame from a sequence acquired by a camera set on a moving vehicle. (B) The labeled line drawing extracted as described in [24], and the 3D vertices extracted by the reconstruction algorithm.

As an example, Fig. 6A shows an image of a typical road scene. The image was grabbed from a sequence acquired by means of a Sony Betacam camera set on a traveling vehicle, at an height from the ground of 1.70 ± 0.05 *m* and shifted to the right of the vehicle axis of 0.21 ± 0.02 *cm*. The line drawing in Fig. 6B was obtained from the output of the low-level algorithms as described in [24], and then labeled by the algorithm described in [14] (same figure).

A calibration sequence has been used to calibrate the camera, thus obtaining a focal length of 595 ± 5 *pxl*. The vanishing points were automatically extracted from the image as described in [20], [14]. As an output of the application of the metrical reconstruction algorithm, the 3D positions of all 61 vertices, marked in Fig. 6B by their identification number, have been obtained.

The labeled line drawing only has one meaningful component (not caring about the viewing window), all the objects being connected by the roadbed. The scale factor for this component of the line drawing has been fixed by imposing that the width of the car on the left of the image be 150 ± 5 *cm*.

TABLE 2

THE THREE DIMENSIONS (HEIGHT × WIDTH × LENGTH) OF ALL THE OBJECTS OF FIG. 2A, AS COMPUTED FROM THE 3D COORDINATES OF THE VERTICES SHOWN IN FIG. 2B.

OBJECT	DIMENSIONS
LEFT BUILDING	2370 ± 420 × 880 ± 80 × 3850 ± 1400 *cm*
RIGHT BUILDING	1150 ± 180× 1165 ± 250× 3760 ± 3540 *cm*
LEFT CAR	130 ± 20× 150 ± 5 × 480 ± 80 *cm*
CENTRAL CAR	110 ± 30 × 165 ± 30 × 760 ± 220 *cm*
BUS	270 ± 80 × 150 ± 50 × 870 ± 270 *cm*

The errors are often large, due to the many sources of error (focal length, estimate of the scale factor, location of junctions). Observe that the error on the length of the building (or block of buildings) is huge: This depends on the fact that the end of the building is very close to the central vanishing point, and it is inescapable. The computed size of the objects is always consistent with their real size, except when the process of extracting the line drawing has failed to capture the true shape of the object, due to occlusion or noise.

The size of all objects in the scene has thus been found (see Table 2) and successfully compared to their real size.

5.3 Line Drawings Extracted in a Semi-Automatic Fashion

Even when no automatic way of extracting a line drawing works, the reconstruction algorithm can be applied to a line drawing obtained semi-automatically, which means by manually adjusting the output of the low-level elaboration algorithms, specifically the polygonal approximation. Adjusting the output basically means inserting the lacking segments and removing the spurious ones.

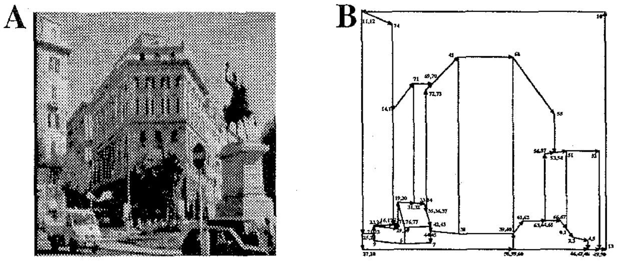

Fig. 7. (A) An image of a typical urban scene. (B) The labeled line drawing, extracted in a semi-automatic fashion, and the 3D vertices extracted by the reconstruction algorithm. There are four meaningful components. First: van, left building, central building; second: right car; third: statue; fourth: rear building.

An example is given in Fig. 7A shows a typical urban scene, and (B) shows the line drawing extracted from (A) as explained above. The labeling shown in (B) corresponds to an interpretation of (A) as an Origami scene (the rear part of the van is viewed as a hollow box) [15], [25], [26].

Since the line drawing depicted in Fig 7A has four meaningful components (five, if considering the 'window' through which the scene is viewed), four scale factors need to be fixed in order to have an absolute 3D reconstruction of the scene. However, a reasonable estimate of the scale factor was possible only for the component including the roadbed (which could be fixed by imposing the height of the van to be 2.00 ± 0.05 *m*) and the component which coincides with the car on the right (which could be fixed by imposing its length to be 3.20 ± 0.10 *m*). There are no clues on how to obtain the other two scale factors, so one must resort to the automatic search for a consistent set of scale factors as explained in Section 4 (Step 4 of the algorithm); this allows us to assign an arbitrary size to the unknown objects which is consistent with the remainder of the scene. The dimensions of the objects belonging to the component comprising the roadbed (the other results being trivial—the car on the right—or partially arbitrary—the rear building and the pedestal) are reported in Table 3.

TABLE 3

THE DIMENSIONS (HEIGHT × WIDTH × LENGTH) OF THE OBJECTS OF FIG. 7A, WHICH BELONG TO THE THIRD COMPONENT OF THE LABELED LINE DRAWING OF FIG. 7B, *I.E.* THE COMPONENT INCLUDING THE GROUND AND ALL THE OBJECTS WHICH ARE PHYSICALLY CONNECTED TO IT, AS COMPUTED FROM THE VERTICES SHOWN IN FIG. 7B.

OBJECT	DIMENSIONS
VAN	200 ± 5 × 160 ± 10 × 410 ± 50 *cm*
LEFT BUILDING	2350 ± 60 × – × - *cm*
CENTRAL BUILDING	1890 ± 60 × 625 ± 40 × 1135 ± 70 *cm*

Observe that the width of the central building means the width of its central wall and its length is the length of the oblique side wall.

6 CONCLUSION

The a priori information on vanishing points allows to solve some of the problems of the usual approach to the reconstruction problem [4]. Let us take a closer look at the advantages of this approach..

6.1 Error Tolerance

According to [4], the incidence structure $I = (V, F, R)$ is said to be position-free if realizability of the line drawing is not affected by slightly moving any of its junctions. The line drawing of Fig. 8A is position-free, whereas the line drawing of Fig. 8B is not, in that (B) is realizable iff the three segments s_1, s_2, s_3 meet at a common point when extended. Algebraically, this means that the system of linear equations associated, as in [4], with (B), has redundant equations. The straightforward application of Linear Programming to (B) is therefore meaningless. The solution proposed in [4] is the introduction of a computationally costly mechanism of junction-position correction.

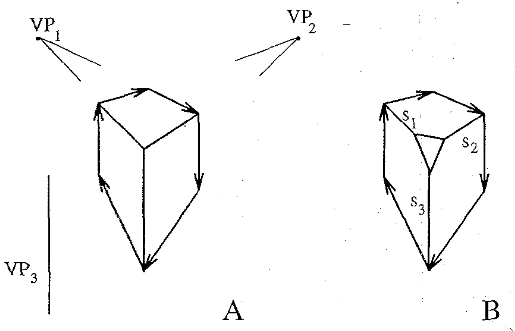

Fig. 8. Line drawing (A) is position-free, as it is possible slightly to move any of its junctions and the line drawing remains realizable. (B) is not position-free, in that it is realizable iff segments s_1, s_2, s_3 exactly meet at a common point.

The approach of this paper is different. Once 3D coordinates are assigned to all vertices of the line drawing, a consistency check (Step 3.I) is carried out on the incidence relations between vertices and panels, such that if some 2D junction is too much misplaced (that is, misplaced beyond the admissible junction location error), the corresponding 3D vertex fails to belong to some of the planar panels to which it is incident. In the case of Fig. 8B, the realizability of the line drawing simply depends on how much the junctions are misplaced with respect to the correct situation in which s_1, s_2, and s_3 meet at a common point, as compared to the admissible error on junctions. Realizability can therefore be a matter of degree, in that a line drawing can be realizable or not depending on the tolerance with which junctions are assumed to be known.

6.2 Unique Reconstruction of 3D Geometry

There are in general many possible 3D realizations associated with the same labeled line drawing. This is in contrast with the fact that human observers usually understand a picture in only one way, which is considered somehow more natural [27]. The labeled line drawing of Fig. 8A is usually perceived by the brain as a cube, but by applying Sugihara's procedure a three-dimensional parameter family of possible interpretations can be obtained by slightly varying the orientation of the planar faces (there are three degrees of freedom apart from the absolute size). The knowledge of vanishing points, however, drastically reduces the number of degrees of freedom of the line drawing. In the specific case, if the vanishing points are as shown in Fig. 8A (observe that VP_3 is at infinity and has therefore been indicated as a direction), the line drawing is correctly recognized to be a cube, and the only degree of freedom is its size.

6.3 Complexity

Linear Programming problems can be solved by Karmarkar's algorithm in time $O(N^{3.5})$. This is to be compared with the performances of the algorithm described in this paper: $O(N + p^3)$ time, where $p << N$ in most 'natural' line drawings (as those which are extracted from real images: see Section 5). If a scale factor is available for every component, the computational complexity becomes $O(N)$.

6.4 Parallelizability

Linear Programming is among the most difficult problems in P (the set of polynomially solvable problems), in the sense that it is P-complete [28]. This means that, presumably, it is not efficiently parallelizable (a problem of input size n is said to be efficiently parallelizable if it can be solved in $O(\log^{O(1)} n)$ steps by $O(n^{O(1)})$ processors). On the other hand, it has been proved [29] that the algorithm presented in this paper is efficiently parallelizable: it takes $O(\log^2 N)$ steps with $O(N^3)$ processors.

ACKNOWLEDGMENTS

Prof. V. Torre suggested that we should analyze the problem of 3D reconstruction by using the information on vanishing points. Prof. A. Bertossi and Dr. E. De Micheli helped us with useful discussion and suggestions. Dr. M. Campani gave us technical advice. This research was supported by Progetto Finalizzato Robotica (CNR), Agenzia Spaziale Italiana, Esprit Basic Research Action (Insight-2), Esprit 6961 Smart Sensory Systems. Giulia Piccioli was supported by a fellowship of Elsag-Bailey (Finmeccanica SpA). Laura Giovanelli checked the English.

REFERENCES

[1] D.A. Huffman, "Impossible objects as nonsense sentences," *Machine Intelligence*, vol. 6, pp. 295–323, 1971.

[2] M.B. Clowes, "On seeing things," *Artificial Intelligence*, vol. 2, pp. 79–116, 1971.

[3] D. Waltz, "Understanding line-drawings of scenes with shadows," *Artificial Intelligence*, vol. 2, pp. 79–116, 1971.

[4] K. Sugihara, "A necessary and sufficient condition for a picture to represent a polyhedral scene," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 5, pp. 578–586, 1984.

[5] J. JàJà, *An Introduction to Parallel Algorithms*. Reading, Mass.: Addison-Wesley, 1992.

[6] R.M. Karp and V. Ramachandran, "Parallel algorithms for shared-memory machines," J. Van Leeuwen, ed., *Handbook of Theoretical Computer Science: Algorithms and Complexity*, chapter 17, pp. 869–942. New York: Elsevier, 1990.

[7] L.M. Kirousis and C.H. Papadimitriou, "The complexity of recognizing polyhedral scenes," *J. Computer and System Sciences*, vol. 37, pp. 14–38, 1988.

[8] M. Straforini, C. Coelho, M. Campani, and V. Torre, "The recovery and understanding of a line drawing from indoor scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 298–303, 1992.

[9] B. Brillault and O'Mahony, "New method for vanishing points detection," *CVGIP: Image Understanding*, vol. 54, no. 2, pp. 289–300, 1991.

[10] H. Nakatani and T. Kitahashi, "Determination of vanishing point in outdoor scene," *Trans. IEICE*, vol. 64, no. 5, pp. 357–358, 1981.

[11] S.T. Barnard, "Interpreting prospective images," *Artificial Intelligence J.*, vol. 21, no. 4, pp. 435–462, 1983.

[12] L. Quan and R. Mohr, "Determining perspective structures using hierarchical Hough transformation," *Pattern Recognition Letters*, vol. 9, pp. 279–286, 1989.

[13] M.J. Magee and J.K. Aggarwal, "Determining vanishing points form perspective images," *Computer Vision, Graphics, and Image Processing*, vol. 26, pp. 256–267, 1984.

[14] P. Parodi and V. Torre, "On the complexity of labeling line drawings of polyhedral scenes," *Artificial Intelligence*, vol. 70, pp. 239–276, 1994.

[15] T. Kanade, "A theory of origami world," *Artificial Intelligence*, vol. 13, pp. 279–311, 1980.

[16] J. Malik, "Interpreting line drawings of curved objects," *Int'l J. Computer Vision*, vol. 1, pp. 73–103, 1987.

[17] A.K. Mackworth, "Interpreting pictures of polyhedral scenes," *Artificial Intelligence*, vol. 4, pp. 121–137, 1984.

[18] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 199–213, 1986.

[19] T. Pavlidis, *Algorithms for Graphics and Image Processing*. Rockville, Md.: Computer Science Press, 1982.

[20] M. Straforini, C. Coelho, and M. Campani, "Extraction of vanishing points of images of indoor and outdoor scenes," *Image Vision Computing*, vol. 11, no. 2, pp. 91–100, 1993.

[21] M. Gatti, P. Olivieri, M. Straforini, and V. Torre, "Robust recovery of the 3D structure of indoor scenes," *Second Int'l Workshop on Robust Computer Vision*, pp. 401–412, Bonn, Germany, 1992.

[22] B. Caprile and V. Torre, "Using vanishing points for TV camera calibration," *Int'l J. Computer Vision*, 1990.

[23] K. Kadono, M. Asada, and Y. Shirai, "Context-constrained matching of hierarchical cad-based models for outdoor scene interpretation," *Proc. IEEE Workshop on Directions in Automated CAD-Based Vision*, pp. 186–195, Hawaii, 1991.

[24] M. Campani, M. Cappello, G. Piccioli, E. Reggio, M. Straforini, and V. Torre, "Visual routines for outdoor navigation," *Proc. Intelligent Vehicles Symp.*, pp. 107–112, Tokyo, 1993.

[25] P. Parodi, "Labeling perspective projections of origami scenes," *Asian Conf. Computer Vision, ACCV '93*, pp. 446–449, Osaka, Japan, 1993.

[26] P. Parodi, "The complexity of understanding of origami scenes," *Int'l J. Computer Vision*, in press.

[27] T. Kanade, "From a real chair to a negative chair," *Artificial Intelligence*, vol. 59, no. 1–2, pp. 95–101, 1993.

[28] D. Dobkin, R.J. Lipton, and S. Reiss, "Linear programming is log-space hard for p. *Inf. Process. Lett.*, vol. 9, pp. 96–97, 1979.

[29] N. Dendris and P. Parodi, "Parallel algorithms for the interpretation of line drawings," 1994. Unpublished.