Pergamon

0952-1976(95)00057-7

Engng Applic. Artif. Intell. Vol. 9, No. 1, pp. 53–64, 1996

0952-1976/96 $15.00 + 0.00

Contributed Paper

A Scheme for 3D Object Reconstruction from Dimensioned Orthographic Views

DOV DORI
Israel Institute of Technology, Israel

MIRI WEISS
Israel Institute of Technology, Israel

(Received December 1994; in revised form August 1995)

3D object reconstruction from 2D orthographic views has been a major research issue during the past two decades. Existing algorithms assume coordinate-based, noise- and error-free input without dimensioning annotation. The approach presented here assumes that the original input is a real-life engineering drawing, in which the 2D geometry of each orthographic view is annotated with dimensioning. Detected dimensions are translated into sets of constraints, one for each view, from which proper dimensioning is validated and 2D minimal graphs are obtained. The method combines elements from variational geometry, matrix algebra and graph theory to construct a composite network describing the structural and topological relations among the various entities that combine the 3D object. This graph provides the basis for a complete 3D object reconstruction. The paper describes the details of the method, and demonstrates it on a comprehensive example.

Keywords: 3D reconstruction, engineering drawings understanding, document analysis and recognition, orthographic views.

1. INTRODUCTION AND MOTIVATION

Engineering drawings have been traditionally used for designing and communicating object information among designers, customers, subcontractors and quality assurance professionals. Using CAD systems, the process of design is interactive. Both geometric data and annotations are stored in the CAD system database and can be accessed and used for future design, or as input for CAM systems. Prior to the introduction of CAD/CAM systems, paper drawing models were the major means of design. Numerous mechanical engineering drawings of parts still exist in this form. Significant progress in scanning devices and storage technology has made the reconstruction of 3D objects from paper engineering drawings a viable research issue. A number of studies have provided partial solutions for restricted cases.

This paper presents a new approach for reconstruction of 3D objects from engineering drawings that extends the work described in Ref. 1. The approach combines the dual nature of engineering drawings with elements of variational geometry, matrix algebra and graph theory. Section 2 surveys approaches for reconstructing 3D objects from orthographic views, introduces criteria for classifying the methods, and discusses their merits and shortcomings. In Section 3 the new approach to 3D reconstruction is introduced and discussed. The description of the variational geometry rule base and the graph theoretic elements, as well as a comprehensive example are provided in Sections 4, 5 and 6, respectively.

2. 3D RECONSTRUCTION ALGORITHMS

A number of approaches and algorithms have been developed over the past two decades to automatically interpret user-supplied orthographic views for the purpose of 3D object reconstruction.

The two main reconstruction approaches are the *wireframe-oriented bottom-up* approach and the *volume-oriented* approach. Existing algorithms within each approach are categorized into those handling only planar surfaces and those which manage more general surfaces, including cylindrical, canonical, spherical and toroidal. The brief survey in this section covers algor-

Correspondence should be sent to: Dr Dov Dori, Faculty of Industrial Engineering and Management, Technion, Israel Institute of Technology, Haifa 32000, Israel.

ORIENTATION —> SURFACE TYPE	Wireframe	Solid
Planar	Wireframe-Planar Algorithms	Solid-Planar Algorithms
Complex	Wireframe-Complex Algorithms	Solid-Complex Algorithms

Fig. 1. Classification of 3D reconstruction algorithms

ithms in both approaches. Within each approach a distinction is made between algorithms that handle planar surfaces only and those which handle more general surfaces.

Figure 1 depicts the classification of 3D reconstruction algorithms. Each algorithm has two attributes: orientation and surface-type. Orientation has two values (instances): "wireframe" and "solid", while surface-type has the two values: "planar" and "complex". The Cartesian product of the two attribute ranges yields four algorithm types. Each type features multiple inheritance: it inherits one orientation value and one surface-type value, giving rise to four families: wireframe-planar algorithms, solid-planar algorithms, wireframe-complex algorithms and solid-complex algorithms.

2.1. The wireframe-oriented bottom-up approach

The use of wireframe-oriented bottom-up algorithms is widespread. It employs fundamental topological ideas, and features four major steps:

(1) Transformation of 2D vertices extracted from orthographic views into 3D vertices.
(2) Generation of 3D edges from the 3D vertices.
(3) Construction of faces from the 3D edges.
(4) Formation of 3D objects from the 3D faces.

Idesawa[2] has proposed the following definitions for the terms used in the above scheme.

A *solid* is a body occupying a range in 3D space or a partial space enclosed by several surfaces in 3D space.

A *face* is a segment of a surface which constitutes a boundary between the solid and the exterior space. A segment of the surface which has different properties (e.g. curvature, position, direction, etc.) is regarded as a different face.

An *edge* is an intersection of two faces.

A *vertex* is an intersecting point of three edges or more.

Wireframe planar algorithms

One of the earliest wireframe reconstruction algorithms of planar surface objects was formalized by Idesawa.[2] It automatically generates a solid object from a three-view engineering drawing. Lafue[3] presented an algorithm which accepts as input two or three orthographic projections and is constrained to face-wise projection drawings, collections of faces, rather than edges and vertices. Preiss[4] suggested an approach in which the input data is not restricted to representation in a face-wise manner, and also handles solid and dashed lines. In addition to the basic vertex, edge and face entities, the definitions of visible and hidden nodes are added and dashed lines are also treated. Haralick and Queeney[5] showed that understanding engineering drawings is basically a labeling problem. Markowsky and Wesley[6] presented an algorithm for fleshing out an object described as a wireframe to a 3D object. The algorithm handles cases with multiple solutions, and deals even with pathological cases. Fleshing out projections[7] is an extension of the work in Ref. 6. The algorithm finds polyhedral solid objects from 2D projections. In Ref. 8, Markowsky and Wesley survey methods for automatically transforming 2D projections into 3D models. They provide basic definitions for Constructive Solid Geometry (CSG), Boundary Representation (B-Rep) and Volume representations of 3D objects. Approaches to volume reconstruction are described, and difficulties are pointed out. The wireframe and projection fleshing-out algorithm is presented as a partial solution to automatic transformations. Considering the list of unsolved problems on the way to achieving complete 3D reconstruction, they propose an interactive approach that would allow progress in the translation and interpretation of geometric data, and would enable a wider variety of surfaces, to be handled.

Wireframe complex algorithms

Preiss[9] presented the 3D reconstruction process as a problem of a heuristic search for bodies with planar and cylindrical faces. The approach is based on determining vertices, edges and faces by a set of constraints. Sakurai and Gossard[10] extended Wesley and Markowsky's algorithm[7] by adding to the basic elements (face, edge, vertex) the silhouette edges (lines and arcs) and the tangent edges (lines and arcs). Lequette[11] presented an algorithm for constructing 3D objects from orthographic views, that distinguishes between the general case and a special case, in which only two views are needed. This method is similar to that of Sakurai and Gossard,[10] but differs in the way in which it generates tangent edges and solids.

2.2. Solid approach algorithms

Solid (volume-oriented) algorithms are based on constructing 3D primitive subparts through translation

sweep operations, and combining them to generate the complete object. Aldefeld[12] suggested viewing a complex part as being composed of elementary objects belonging to a set of predefined classes. Recognition of these objects is done by making use of knowledge about the class-dependent pattern of their 2D representation. The term "primitive" denotes a basic element of 2D representation, such as arc, line, or circle. An object for a 3D representation is termed a "composed part". In further work,[13] Aldefeld and Richter suggested interpreting a 3D body from its 2D projections by a user-guided process, while using automatic procedures where they can be relied upon. Nagasami *et al.*[14] extended and implemented Aldefeld's work[12] by defining two classes of objects that represent a wide variety of objects. One class is obtained by translation sweep operations with a planar base, an arbitrary contour in one view and a uniform thickness in the direction perpendicular to the base. The other class consists of objects having an axis of symmetry. The algorithm is implemented by a knowledge-based system which interprets the orthographic views and represents the reconstructed object. Chen *et al.*[15] introduced a method for reconstructing 3D objects which may be polyhedrons, cylinders, partial cylinders and their composites. The algorithm consists of three stages: decomposition, in which the input drawing is decomposed into several predefined types of subviews, reconstruction, in which translational sweep and cutting operations are used to reconstruct the corresponding subpart of each set of subviews, and composition, in which volume enclosure relations among various subparts are utilized to compose the complete object, which is represented by a CSG tree.

2.3. Problems with current solutions

The methods described are aimed at reconstructing a 3D object from orthographic views. As formalized by Requicha,[16] the objects may be represented either by CSG or by B-rep. None of the representations provides a complete automatic procedure for the range of elements which are found in engineering drawings. The following is a list of factors that prevent current methods from being capable of converting objects described by conventional mechanical engineering drawings into 3D CAD objects.

Surface type

The surface types handled by current methods are planar, cylindrical, conical and toroidal, respectively. Most works (Wesley and Markowsky,[8,9] Preiss,[4] Idesawa[2] and Haralick and Queeney[5]) deal with polyhedral objects only. This is a major drawback, because most engineering drawings contain bars (straight-line segments), arcs and circles, denoting 3D objects which are planar, cylindrical or toroidal. Only Aldefeld,[13] Lequette[11] and Sakurai and Gossard[12] handle nonplanar surfaces.

Computational complexity

Only Wesley and Markowsky[8] and Nagasami *et al.*[14] have treated the computational complexity involved in the algorithms and in their implementation. Wesley and Markowsky[8] indicate that their "Fleshing-out wireframes and projections" algorithms[6,7] are computationally very complex and sometimes unmanageable. To reduce the computational load, they suggest an interactive approach.

Assumptions about input style and quality

A major drawback, pointed out by Dori[17] and Dori and Tombre,[18] is that all the methods surveyed above assume a noise-free, idealized set of projections, uncluttered by annotations or any other "irrelevant" graphic entities, and free of any uncertainties about the exact dimensions. This may be possible when the starting point is a set of views generated with a computer, available as formatted data structures. However, when the reconstruction process has to start from a scanned paper drawing, be it a manually prepared or a CAD-based drawing, such idealized assumptions are unrealistic. The combination of "image" and "annotation" requires a level of human-like intelligence for understanding this dual nature, which current methods do not support. In addition, many methods assume that the input is provided in a specific mode or order. Lafue,[3] for example, assumes that the different drawing views are a collection of faces, drawn counterclockwise or clockwise. Aldefeld[13] assumes that creating the input is based on a user-guided procedure.

3. SCHEME OF A PROPOSED APPROACH FOR 3D RECONSTRUCTION

The proposed approach for reconstructing 3D objects from 2D engineering drawing is based on the following three stages:

(1) high-level 2D orthographic view understanding;
(2) analysis of topological relations and dimensioning schemes in each 2D view using variational geometry elements; and
(3) 3D construction by merging the dimensioning scheme of each 2D view into a common dimensioning scheme for the entire object, using graph-theoretic methods.

The top-level view of the approach is described in the object-process diagram (OPD) of Fig. 2. An OPD[19] is a graphic representation of the objects and processes in the system with their structural and procedural relations. As the legend of Fig. 2 shows, objects and processes are represented as rectangles and ellipses, respectively. Effect links (arrows) lead from an affected object to a process in which it takes part, or from a

process to a resulting object. An instrument link (a line ending with blank circle) leads from an enabling object—the instrument—to the enabled process.

4. UNDERSTANDING 2D ORTHOGRAPHIC VIEWS

For high-level information of an engineering drawing to be extracted, it ought to be viewed as a dual, two-faceted entity. One facet is that of the object's geometry, i.e. the graphics describing the contours of the 2D projections of the object on the drawing plane. The other facet is the annotation—dimensioning and other manufacturing directions—which complements the information provided by the geometry. Normally, the geometry itself cannot be accurate enough for defining and manufacturing the object without the additional information provided by the annotation. Each facet can be regarded as a separate layer of the drawing, such that the two layers together provide a complete product definition.

In most current CAD-conversion systems, the separation into layers is done on the basis of 'graphics' vs 'text'. In the approach described here, however, the two layers are the image, or geometry layer, which includes projections of the object, and the language, or annotation layer, which includes dimensioning and other functional symbols. Dori and Tombre[18] divide the understanding process into three phases: lexical—early vision, syntactic—intermediate vision and semantic—high-level vision. The lexical phase is preceded by scanning, noise removal, enhancing, thresholding and other preprocessing operations. Lexical analysis includes recognition of primitives—basic elements found in most engineering drawing: bars (straight-line segments with non-zero width), circular arcs (with non-zero width as well), text, and arrowheads.[20] At the syntactic phase, the primitives are aggregated into groups, such as dimension-sets, that obey the rules of the grammar of dimensioning in engineering drawings.[17] The semantic phase involves recovering for each 2D view of the object described in the drawing each one of the two major layers—geometry and annotation—and verifying that the dimensioning is proper,[17] i.e. that there are neither missing nor redundant dimension-sets.

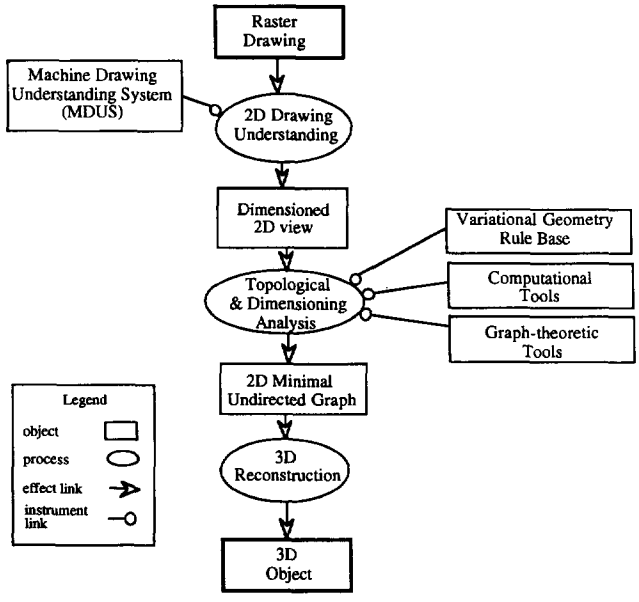

Fig. 2. A top-level object-process diagram (OPD) of the proposed approach for 3D object reconstruction from engineering drawings.

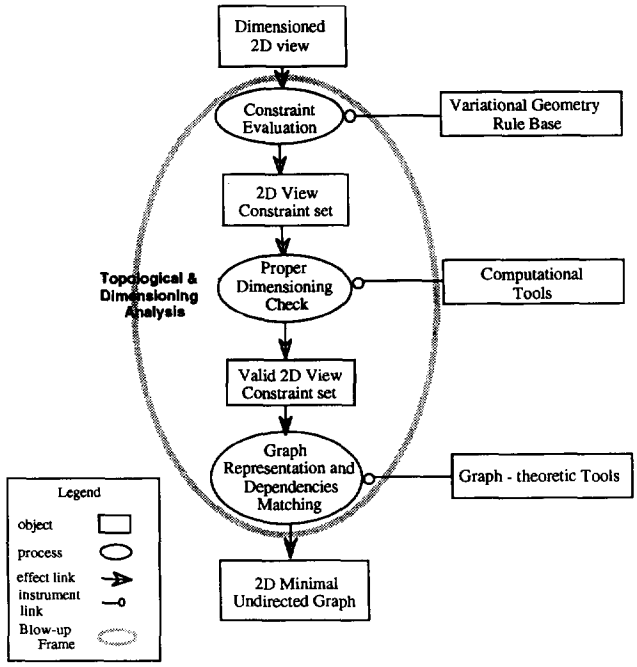

Fig. 3. The topological and dimensioning analysis process of Fig. 2 (enlarged).

The Machine Drawing Understanding System (MDUS), an experimental system designed to automate the process of converting mechanical engineering drawings into CAD format,[20] is currently used for the lexical analysis phase.

5. TOPOLOGICAL AND DIMENSIONING ANALYSIS

Following 2D understanding of engineering drawings, each of the projections is analyzed by means of geometrical and topological relations. Given an orthographic view, geometric constraints, which result from detected dimensions, are extracted.

In Fig. 3, the process "Topological and Dimensional Analysis" of Fig. 2 is enlarged, exposing three main processes: "Constraint Evaluation", "Proper Dimensioning Check" and "Graph Representation and Dependencies Matching". Note that the input and the output of the blown-up process—Dimensioned 2D view and 2D directed graph, respectively—are consistent with those depicted in Fig. 2. This consistency is a basic feature of the scaling of Object-Process Diagrams.[19]

Mechanical CAD systems can be classified into *con-*

ventional and *parametric* systems. In most conventional mechanical CAD systems, the geometry determines the dimension values. The user must accurately define the geometric positioning of each entity in the drawing, and changes to the design are made directly in the graphics. Dimensions should then be modified separately by the user to match the described geometry. In parametric, constrained-based systems dimensions define the geometry and are used to specify the position of the topological entities.[21] Changing one or more numerical values of the dimensions in the drawing yield a new geometry, consistent with the new values of the dimensions. The dimensions are treated as constraints describing relations among entities. The main use of the parametric approach is to provide the user with a flexible CAD system, in which design and modifications are made simple and intuitive. Approaches to constraint management in parametric design, which are discussed briefly below, are classified into two main types: algebraic and artificial intelligence (AI)-oriented.

The variational geometry algebraic approach is based on translating each of the constraints to an algebraic equation. Hilliard and Braid[22] laid down the basic concepts of variational geometry. Light and Gossard[21] presented a procedure for 2D CAD, based on the idea that dimensions in a mechanical drawing describe the object's geometry and topology, and can be treated as constraints. Lin *et al.*[23] utilized variational geometry to design and modify constraints of 3D objects. The method handles planar, cylindrical and canonical faces, and uses the Jacobian matrix to determine the sensitivity matrix.

An artificial intelligence-oriented approach is described by Aldefeld[24] as a combination of geometric elements and formulae defining constraints. The 2D geometry is based on three types of primitives: points, tracks, and line segments. Constraints are categorized into two types: metric (dimensional) constraints, which involve *midpoint*, *parallel tangent*, etc. Constraints are translated to equations and tested for degrees of freedom. Constraint-propagation rules are constructed from atomic formulae, and forward inferencing is used as a control strategy. Extending the work of Aldefeld,[24] Verroust *et al.*[25] focus on evaluating the main rules in 2D models. The method is based on an expert-system shell, which contains constraints and points. The system evaluates rules, produces geometric locations, and detects inconsistencies. Bruederlin[26] presents a method of modeling operations using constraints and group hierarchy. The system is an interactive 3D geometric modeler, in which an object can be created by a combination of transformation operations.

5.1. Constraint definition and proper dimensioning check

The basic primitives comprising a 2D orthographic view of an engineering drawing are bars, arcs and circles. Each dimension and topological relation is formulated as a constraint which defines the positioning of each point.[21] The variational geometry rule base defined and used in this work is described in detail in Ref. 1. Dimensions define geometric constraints, such as the distance between two points, the distance between a point and a bar, and an angle between two bars. Similarly, topological relations involve primarily connectivity. Tangency is an example of a topological relation, since it requires that an arc (usually representing a cylindrical face in 3D) be adjacent to a bar (usually representing a planar face in 3D). Parallelism and perpendicularity involve angular measures, and are therefore considered as metric constraints.

A sample of three rules from the variational geometry rule base is given in equations (1)–(3).

- Euclidean distance: the distance between two points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ is given by

$$(x_2-x_1)^2+(y_2-y_1)^2-D^2=0. \quad (1)$$

- Perpendicularity: two bars P_1P_2 and P_3P_4 are perpendicular if the scalar product of their associated vectors is zero

$$(x_2-x_1)(x_4-x_3)+(y_2-y_1)(y_4-y_3)=0. \quad (2)$$

- Line Parallelism: two bars P_1P_2 and P_3P_4 are parallel if their vector product is zero

$$(x_2-x_1)(y_4-y_3)+(y_2-y_1)(x_4-x_3)=0. \quad (3)$$

These constraints provide the basis of constructing the Jacobian constraint matrix. For the dimensioning to be proper, the Jacobian constraint matrix should meet the following two requirements:

(1) The number of constraints must be equal to twice the number of vertices in the drawing.
(2) The rank of the Jacobian matrix must be equal to the number of constraints.

These requirements are valid because each vertex is represented by a pair of numbers, the x and y coordinates, such that ultimately, if all the constraints are satisfied, the value of each one of these numbers is determined unambiguously and without redundant or contradicting constraints. This remains true even though some of the constraints may not refer directly to vertex coordinates, because propagating the constraints eventually causes the coordinates of each vertex to be fixed in the plane. If at least one of the requirements is not fulfilled, the dimensioning is improper, i.e. there is at least one extra or one missing constraint.

5.2. Graph representation and dependency matching of 2D view constraint-set

Following the constraint definitions and proper dimensioning check for each 2D orthographic view, the constraints are combined into a 2D undirected graph.

The constraints extracted from each 2D view repre-

sent relations among explicit and implicit dimension-sets.[17] An explicit dimension-set is a dimension-set which appears explicitly in the drawing. An implicit dimension-set is a measure which is implied from conventions of draftsmanship. For example, if two lines in the drawing appear to be perpendicular (parallel) to each other and there is no indication that contradicts this, then the angle between these two lines is indeed 90° (0°).

The resulting expressions are mostly nonlinear equations, and represent equalities. The graph representation of a single constraint expresses the relationships and connections among the parameters in one corresponding, metric (dimensional) or structural (topological) relation. A graph representation of a constraint-set expresses the relations of parameters among more than one constraint.

Figure 4 is an enlargement-up of the process "Graph Representation and Dependency Matching" depicted as a single ellipse in the OPD of Fig. 3. It describes in detail the procedures of converting the constraint set into a minimal undirected graph. The motivation of this operation is to find the minimal set of relations that fully represent the 2D view. The minimal graph representation provides a compact quantitative and qualitative relation mapping of connections among the parameters.

Detailed terminology of graph theory can be found in Refs 27 and 28. The terms used in this paper are described below.

Definition 1: A ***graph*** $\mathbf{G}=(\mathbf{V}, \mathbf{E})$ is a structure which consists of a set **V** of *nodes (points, vertices)* and a set **E** of *edges (arcs, pointers)*, such that each edge e is

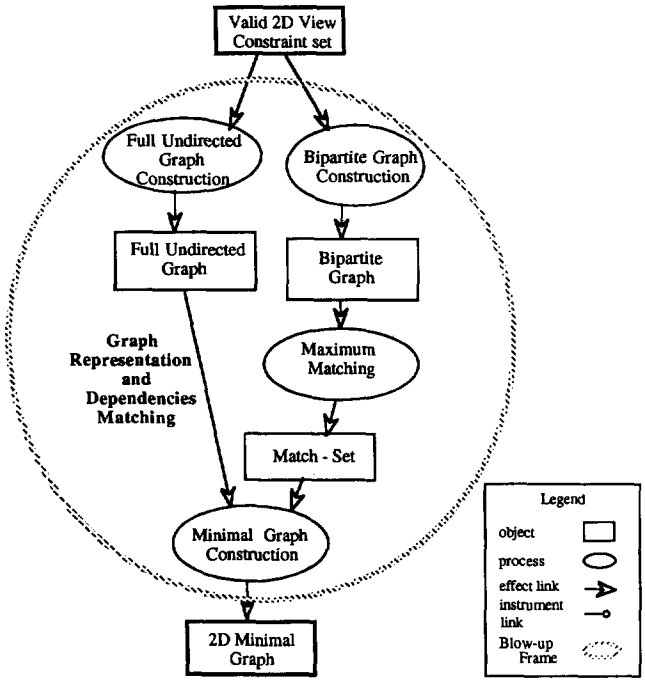

Fig. 4. The graph representation and dependencies matching process of Fig. 3 (enlarged).

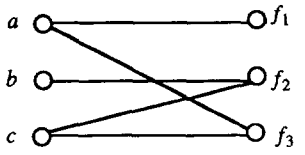

Fig. 5. An example of a bipartite graph.

incident to the elements of an unordered pair of nodes (u, v), where u and v are not necessarily distinct.

Definition 2: a ***bipartite graph*** $\mathbf{BG}=(\mathbf{V}, \mathbf{E})$ is a graph in which the set of nodes **V** consists of two disjoint subsets **B** and **C** such that every member of **E** has one element in **B** and another in **C**.

Figure 5 shows an example of a bipartite graph, where $\mathbf{B}=\{a, b, c\}$ represents the set of parameters and the set $\mathbf{C}=\{f_1, f_2, f_3\}$ represents the constraints.

The arcs are represented by $\mathbf{E}=\{(a,f_1), (a,f_3), (b,f_2), (c,f_2), (c,f_3)\}$.

Definition 3: Let $\mathrm{BG}=(\mathrm{V}, \mathrm{E})$ be a bipartite graph and $\mathrm{B}, \mathrm{C} \subset \mathrm{V}$. ***Maximum matching***, also known as the "marriage algorithm", is a method in which the largest subset of **E** (edges) is found in BG, such that no two edges share a common node in the same set **B** or **C**.

Definition 4: A ***complete matching*** is a match set **M** where the number of matching edges $|\mathbf{M}|$ is equal to the number of nodes in set **B** and in set **C**, i.e. $|\mathbf{B}|=|\mathbf{C}|$. Complete matching is achieved by implementing a network flow algorithm.[27]

Although the matching process is not unique and can yield different match sets for the same bipartite graph, this non-uniqueness does not affect the accuracy of the solution, because there is no preference for a specific match set.

It is important to understand the relationship among parameters in the explicit and implicit dimensions represented via the constraint-set, as described by Serrano.[29] The advantages of using graphs for representing the constraint set are twofold. Graphs allow both qualitative and quantitative operations to be carried out and enable a large number of algorithms from graph theory to be applied for a variety of purposes.

As described in Fig. 4, graph representation and dependency matching for the 2D view constraint set consists of four steps:

(1) representing the constraint set by a complete graph;
(2) representing the constraint set as a bipartite graph;
(3) finding a maximum matching for the bipartite graph; and
(4) transforming the complete graph into a minimal graph using dependencies from the maximum matching found in step 3.

5.3. A simple example

The minimal graph representation process can be demonstrated by a simple example concerning the

position of a point constrained to lie on a line and a circle, represented by the following two constraints:

$$x^2+y^2-D_1^2=0 \quad (f_1) \qquad (4)$$

$$x+y-D_2=0 \quad (f_2).$$

A distinction is made between two types of parameters: "known" and "unknown". The known parameters are explicit dimensions detected in the 2D view, while the unknowns are point locations. For the constraint set of equation (4), the set of known parameters is $\{D_1, D_2\}$, while the unknowns are $\{x, y\}$. In this example there are two solutions to the pair $\{x, y\}$, but this is irrelevant, because the underlying geometry here does not represent what is normally found in engineering drawings. The four steps of the process follow:

Step 1: A representation of the constraints as a full graph is shown in Fig. 7. In Fig. 6, each constraint yields a separate partial graph, where each node represents a parameter, and an arc represents the existence of the corresponding constraint (f_1 or f_2) between the parameters. The arcs are labeled according to the constraint.

In Fig. 7, the two graphs are combined into a complete graph, where the parameters x and y are common and therefore appear only once.

Step 2: The bipartite graph, shown in Fig. 8, is obtained from the constraint set in equation (4) as follows. Only the "unknown" parameters are involved in the bipartite graph representation. The two sets $\mathbf{B}=\{x, y\}$ and $\mathbf{C}=\{f_1, f_2\}$, and the set of edges $\mathbf{E}=\{(x,f_1), (y,f_1), (x,f_2), (y,f_2)\}$ are defined. These four edges are indicated in Fig. 8 by solid and dashed lines.

Step 3: Applying the maximum matching process yields the match set $\mathbf{M}=\{(x,f_1), (y,f_2)\}$, which is indicated by the solid lines in Fig. 8. The dashed lines represent the relations which were not chosen in the matching process.

Step 4: Using the match set, the full undirected graph is transformed into a minimal undirected graph. This transformation provides a graph with the minimal set of edges needed to represent the connections and relations among the parameters. This is done by applying the following rules: for each pair (v, f) in the match set, the full undirected graph is modified such that all arcs labeled f and incident on v are kept. For example, x was matched with f_1; therefore only arcs that are labeled f_1 and are incident on x are kept. The resulting minimal undirected graph is shown in Fig. 9.

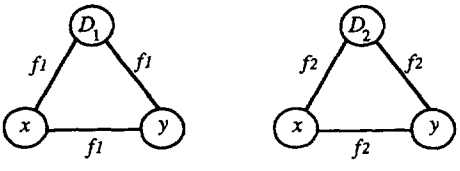

Fig. 6. Two partial graphs for the constraints f_1 (left) and f_2 (right).

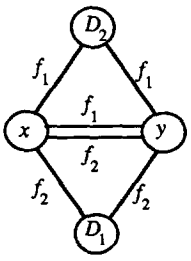

Fig. 7. A complete graph for f_1 and f_2.

6. 3D RECONSTRUCTION

Having obtained the 2D minimal undirected graph for each 2D orthographic view, the 3D object is reconstructed. The 3D reconstruction process, depicted within the blow-up frame in Fig. 10, is an up-scaling of the 3D reconstruction process shown in Fig. 2. The detail within the blow-up frame indicates that the 3D reconstruction process consists of three major steps:

(1) composite network generation;
(2) 3D dimensioning labeling; and
(3) network-to-object conversion.

6.1. Composite network generation

The process of composite network generation uses the set of separate 2D minimal undirected graphs constructed in the previous stage. These graphs are analyzed for matching and merging conditions. A schematic illustration is shown in Figs 11 and 12. Figure 11 illustrates the starting point, in which each 2D view has an independent minimal graph representation. Figure 12 illustrates the composite network generated by adding new edges (the dashed lines) to connect the 2D view graphs.

The composite network-generation process consists of two sub-processes:

(1) Initial matching

Matching the first 3D vertex is based on examining candidate vertices in each one of the 2D representation graphs to be matched as belonging to the same 3D point. This can be done, for example, by matching the extreme three points, one in each view, where "extreme" is in the sense that each point is the closest to the origin of the 2D coordinate system dividing the plane into quadrants, such that each view is in a different quadrant.

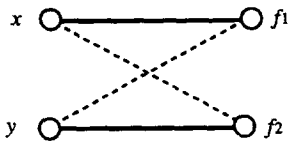

Fig. 8. Bipartite graph and maximum matching.

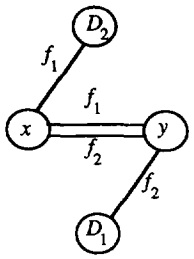

Fig. 9. Constraint set minimal graph.

(2) Complete matching

The initial 3D vertex serves as a starting point to the completion of the matching process. Each 3D link represents both the connecting related parameters and the nature of the link (parallelism, dimension, etc.)

6.2. 3D dimensioning labeling and network-to-object conversion

In the 2D constraint set graphs, names of parameters and constraint labels are unique. After generating a composite network, the graph has to be re-labeled to maintain the consistency of the relations among the constraints and the parameters.

The network-to-object conversion is a translation of the dimensional and topological relations, represented by the composite network graph, into the geometrical 3D world. The exact procedure is a topic for further research. However, at this point it is already clear that since the composite network represents the 3D relationships among vertices and higher-level geometric entities, the final 3D coordinates of the vertices are gradually solved for. The traversal will be guided by topological considerations, such as moving along an edge from one vertex to the other or finding closed loops around faces.

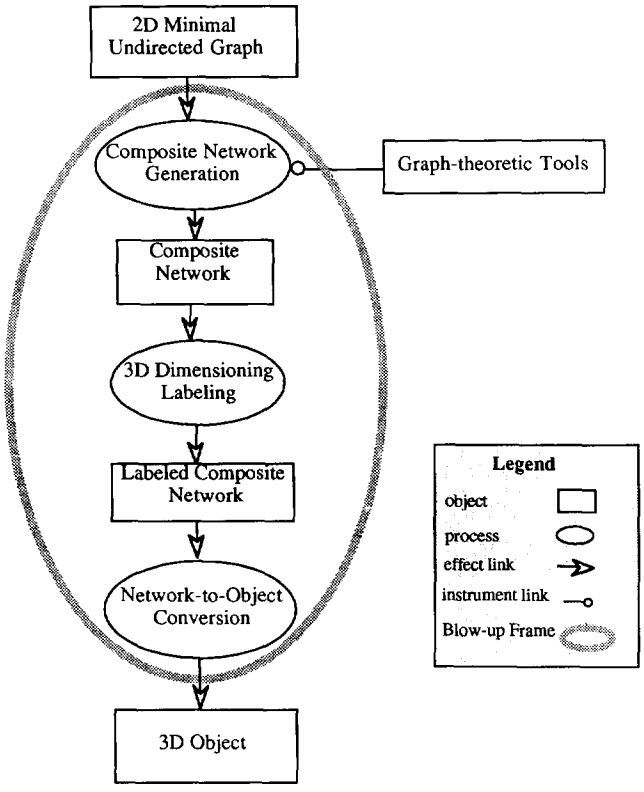

Fig. 10. 3D reconstruction process (enlarged).

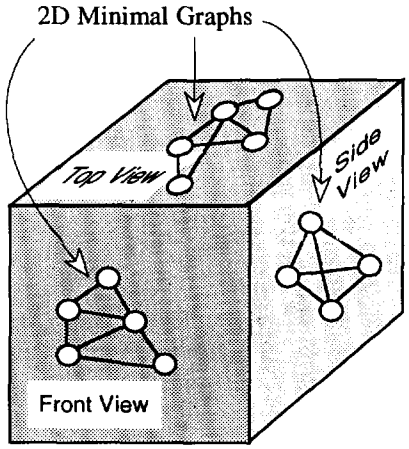

Fig. 11. 2D minimal graphs for each orthographic view.

7. A COMPREHENSIVE EXAMPLE

The following example illustrates the process of constructing a minimal undirected graph from two 2D dimensioned views of an engineering drawing. A triangular prism is described in Fig. 13 by two views: front view (F) and side view (S).

7.1. Constraint evaluation and proper dimensioning check

The following parameters are used for the points in the two 2D views of Fig. 13. For the side view there are the four points $P_1(y_1, z_1)$, $P_2(y_2, z_2)$, $P_3(y_3, z_3)$, and $P_4(y_4, z_4)$, and for the front view, $P'_1(x_1, y_1)$, $P'_2(x_2, y_2)$, and $P'_3(x_3, y_3)$.

Two rules from the variational geometry rule base described in Section 5.1 are used: Rule No. 1—Euclidean distance between two points, and Rule No.

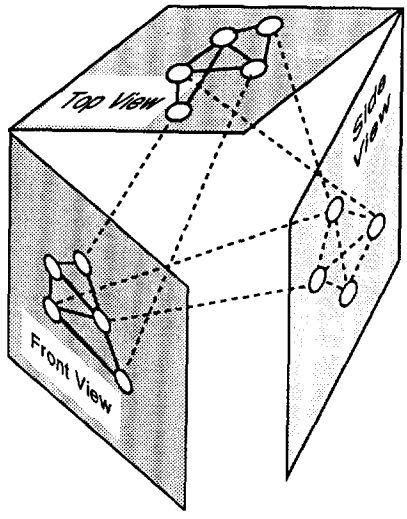

Fig. 12. Composite network of orthographic views.

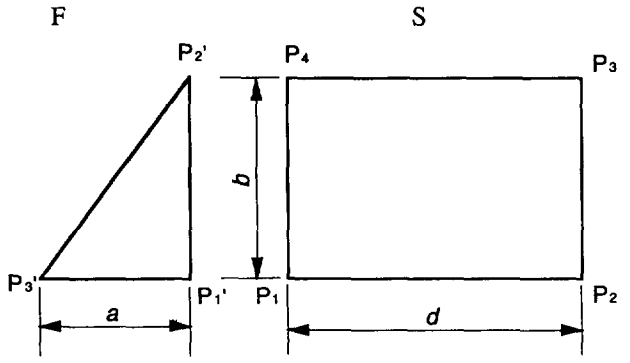

Fig. 13. Front view (F) and side view (S) of an engineering drawing of a triangular prism.

2—Perpendicularity. In addition, to prevent solid body translation,a selected point, called the *anchor point*, is fixed as the origin (0,0). All the points are defined relative to this anchor point. Likewise, to fix the orientation (i.e. to prevent solid body rotation), a particular bar[15] is chosen to be horizontal, i.e. parallel to the horizontal axis. Thus, for the front view of Fig. 13 the following six constraints are formulated:

Rule No.	Source	Constraint	
1	b	f_1: $(x_2-x_1)^2+(y_2-y_1)^2-b^2=0$	(5)
1	a	f_2: $(x_2-x_1)^2+(y_3-y_1)^2-a^2=0$	(6)
2	$P_1P_2 \perp P_1P_3$	f_3: $(x_1-x_2)(x_3-x_1)+(y_1+y_2)(y_3-y_1)=0$	(7)
	x anchor	f_4: $x_1=d=0$	(8)
	y anchor	f_5: $y_1=e=0$	(9)
	orientation	f_6: $y_2-y_1=0$.	(10)

From equations (5)–(10) a 6×6 Jacobian matrix was obtained. The rank of the matrix was computed using the Matlab package. The dimensioning in this case was found to be proper, because, as required, the number of constraints is twice the number of points, i.e. 6, and the rank of the Jacobian matrix is also 6.

For the side view, eight other constraints are similarly formulated as follows:

Rule No.	Source	Constraint	
1	d	f_1: $(y_2-y_1)^2+(z_2-z_1)^2-d^2=0$	(11)
1	b	f_2: $(y_2-y_1)^2+(z_3-z_1)^2-b^2=0$	(12)
1	d	f_3: $(y_4-y_3)^2+(z_4-z_3)^2-d^2=0$	(13)
1	b	f_4: $(y_4-y_2)^2+(z_4-z_2)^2-b^2=0$	(14)
1	$P_1P_2 \perp P_1P_3$	f_5: $(y_1-y_2)(y_3-y_1)+(z_1-y_1)+(z_1-z_2)\times(z_3-z_1)=0$	(15)
	x anchor	f_6: $y_1=e=0$	(16)
	y anchor	f_7: $z_1=a=0$	(17)
	orientation	f_8: $z_2-z_1=0$.	(18)

As before, the 8×8 Jacobian matrix resulting from equations (11)–(18) was computed. The rank of this matrix was found to be 8, indicating proper dimensioning and constraint definition of the side view as well.

7.2. Graph representation and dependencies matching

Figure 14 describes the full undirected graph constructed from the set of constraints in equations (5)–(10). The graph was constructed by the method de-

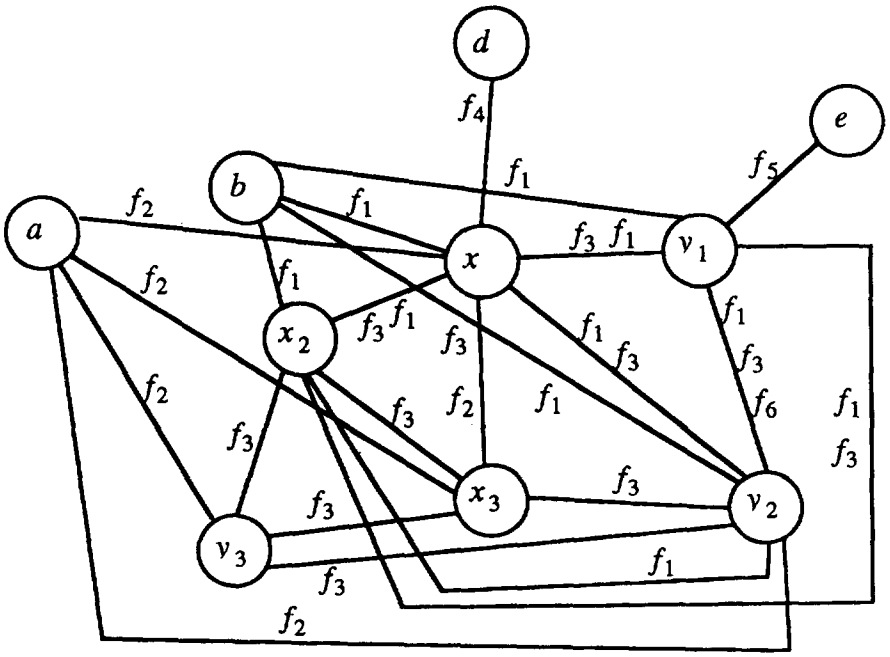

Fig. 14. Full undirected graph for the front view constraint set of Fig. 13.

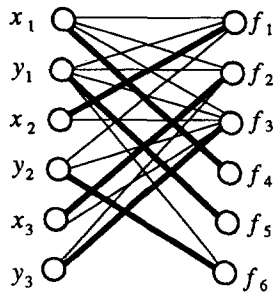

Fig. 15. Bipartite graph for the front view.

scribed in Section 6.2. For example, the edge between x_2 and y_3 is labeled f_3, because the parameters are connected through the formulation of constraint f_3 [equation (7)].

Another example is the edge connecting x_1 and y_2, which is labeled f_1 and f_3, indicating that these parameters are connected by the two constraints, f_1 [equation (5)] and f_3 [equation (7)]. The nodes d and e stem from the constraints (8) and (9), corresponding to the x and y anchors, respectively.

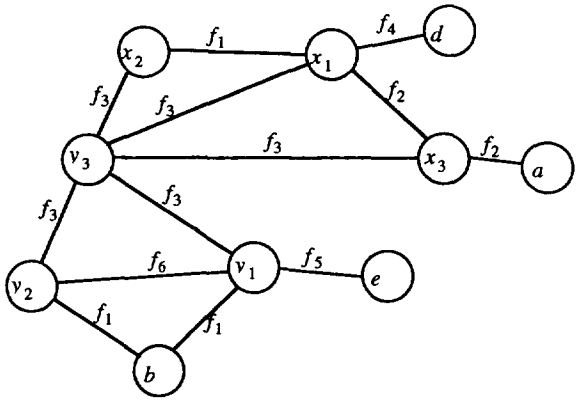

Fig. 16. Minimal graph for the front view.

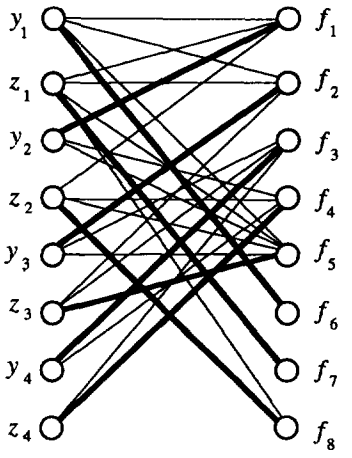

Fig. 17. Bipartite graph for the side view.

Figure 15 is a bipartite graph, which was also obtained from the set of constraints in equations (5)–(10). The match set $\mathbf{M}=\{(x_1,f_4), (y_1,f_5), (x_2,f_1), (y_2,f_6), (x_3,f_2), (y_3,f_3)\}$, found from the maximum matching procedure, is indicated by the thick lines connecting the parameters $(x_1, \ldots, y_3)$ and the constraints $(f_1, \ldots, f_6)$, as explained in Section 5.2.

Figure 16 represents the minimal undirected graph for the front view, obtained from the undirected graph of Fig. 14 and the corresponding bipartite graph of Fig. 15.

Figures 17 and 18 describe the corresponding bipartite and minimal graphs for the side view of Fig. 13.

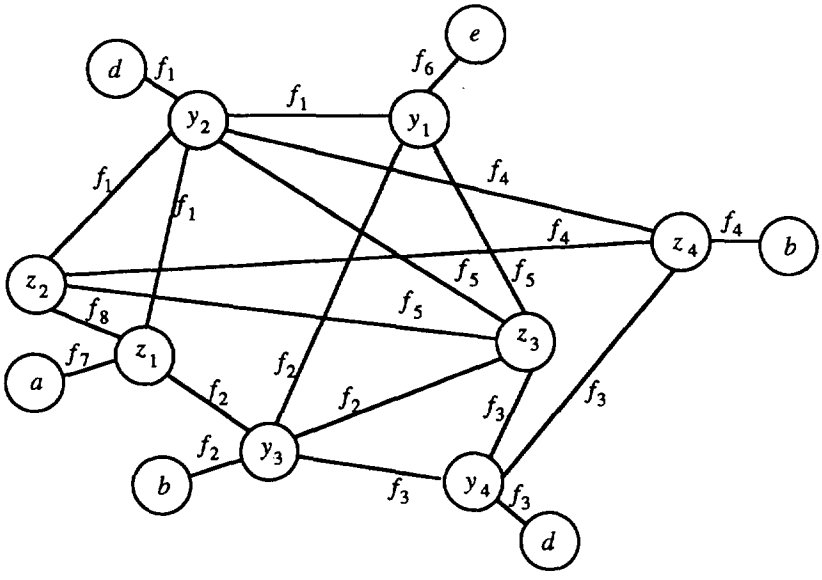

Fig. 18. Minimal graph for the side view.

8. SUMMARY AND FUTURE WORK

The approach to solving the problem of 3D object reconstruction from 2D views proposed in this work combines a variational geometry rule base and a set of graph-theoretic tools. These two elements are amenable to automation, and the complex procedure described in this work can therefore potentially serve as a means for reliable and accurate 3D reconstruction from engineering drawings.

Problems that remain to be solved for a complete implementation of the methodology include:

(1) obtaining the composite network;
(2) translating the composite network to a 3D object;
(3) devising an automated process for the (currently manual) input of constraint information; and
(4) investigating the complexity of the process.

Automating the process of constraint information input can be guided by a set of rules of the type "For each longitudinal explicit dimension in the 2D view insert a constraint relating to the distance between two points" for explicit dimensions. An example of a constraint stemming from an implicit dimension is "If two bars are about perpendicular to each other and there is no angular dimension-set indicating the angle between these two bars, then insert a perpendicularity constraint". A set of heuristics will probably have to be devised to reduce the complexity of the graph operations.

Acknowledgements—The authors wish to thank the two anonymous referees for their valuable comments. This research was supported in part by the Technion VPR Fund.

REFERENCES

1. Weiss M. and Dori D. Variational geometry as a tool for dimensioning validation of recognized 2D views in engineering drawing. In *Shape, Structure and Pattern Recognition* (Edited by Dori D. and Bruckstein A.), pp. 416–431. World Scientific (1995).
2. Idesawa M. A system to generate a solid figure from three view. *Bull. JSME* **16**, No. 92, 216–225 (1973).
3. Lafue G. Recognition of three-dimensional objects from orthographic views. *Proc. 3rd Annual Conf. Computer Graphics, Interactive Techniques and Image Processing ACM SIGGRAPH*, pp. 103–105 (1976).
4. Preiss K. Algorithms for automatic conversion of the 3-view drawing of a plane faced part to the 3-D representation. *Computers Ind.* **3**, 133–139 (1981).
5. Haralick R. M. and Queeney D. Understanding engineering drawings. *Computer Graph. Image Process.* **20**, 244–258 (1982).
6. Markowsky G. and Wesley M. A., Fleshing out wireframes. *IBM J. Res. Devel.* **24**, 582–597 (1980).
7. Wesley M. A. and Markowsky G. Fleshing out projections. *IBM J. Res. Devel.* **25**, 934–953 (1981).
8. Markowsky G. and Wesley M. A. Generation of solid models from two dimensional and three dimensional data. In Picket M. S. and Boyse J. M. (Eds), *Solid Modeling by Computer: From Theory to Application*, pp. 23–51.
9. Preiss K. Constructing solid representation from engineering projections. *Computers Graph.* **8**, 381–389 (1984).
10. Sakurai H. and Gossard D. C. Solid model input through orthographic views. *Computer Graph. (SIGGRAPH)* **17**, No. 3 (1983).
11. Lequette R. Automatic construction of curvilinear solids from wireframe views. *CAD* **20**, 171–180 (1988).
12. Aldefeld B. On automatic recognition of 3D structures from 2D representations. *CAD* **15**, No. 2 (1983).
13. Aldefeld B. and Richter H. Semiautomatic three dimensional interpretation of line drawings. *Computers Graph.* **8**, 371–380 (1984).
14. Nagasami V. and Noshir A. L. Reconstructing of three-dimensional object using a knowledge-based environment. *Engng Computers* **7**, 23–35, (1991).
15. Chen Z. and Perng D. Automatic reconstruction of 3D solid objects from 2D orthographic views. *Pattern Recognition* **21**, 439–449 (1988).
16. Requicha A. A. G. Representation for rigid solids: theory, methods, and systems. *Computer Surv.* **12**, 437–464 (1980).
17. Dori D. Dimensioning analysis: toward automatic understanding of engineering drawing. *Commun. ACM* **35**, 92–103 (1992).
18. Dori D. and Tombre K. From engineering drawings to 3-D CAD models—are we ready now? *Computer Aided Design* **27**, 243–254 (1995).
19. Dori D. Object-process analysis: maintaining the balance between system structure and behaviour. *J. Logic Computat.* **5**, 227–249 (1995).
20. Dori D., Liang Y., Dowell J. and Chai I. Sparse pixel recognition of primitives in engineering drawings. *Machine Vision Applic.* **6**, 69–82 (1993).
21. Light R. A. and Gossard D. C. Modification of geometric models through variational geometry, *CAD* **14**, No. 4, 209–214 (1982).
22. Hilliard R. A. and Braid I. C. Analysis of dimensions and tolerances in computer-aided mechanical design. *CAD* **10**, 161–166 (1978).
23. Lin, V. C., Light, R. A. and Gossard, D. C., Variational geometry in computer aided design, *Computer Graph.* **15**, 171–177 (1981).
24. Aldefeld B. Variation of geometries based on a geometric reasoning method. *CAD* **20**, 117–126 (1988).
25. Verroust A., Schonek F. and Roller D. Rule-oriented method for parameterized computer-aided design. *CAD* **24**, 531–540 (1992).
26. Bruederlin B. Constructing three dimensional geometric objects defined by constraints. *Proc. 1986 Workshop of Interactive 3D Graphics*, Chapel Hill, NC, USA, pp. 111–129 (1986).
27. Even S. *Graph Algorithms*. Computer Science Press, Rockville, MD (1979).
28. Berztiss A. T. *Data Structures: Theory and Practice*, 2nd edition. Academic Press, New York (1975).
29. Serrano D. Constraint management in conceptual design. D.Sc. dissertation, Massachusetts Institute of Technology (1987).

AUTHORS' BIOGRAPHIES

Dov Dori has been a senior lecturer in the Faculty of Industrial Engineering and Management, Technion, Israel Institute of Technology since 1991. Since 1995 he has been head of the Information Systems Engineering area. He received his B.Sc. in Industrial Engineering and Management from the Technion in 1975, an M.Sc. in Operations Research from Tel Aviv University in 1981, and a Ph.D. in Computer Science from the Weizmann Institute of Science, Rehovot, Israel, in 1988.

Between 1975 and 1984 he served as an officer in the Israeli Defence Force. Between 1988 and 1990 he was an Assistant Professor in the Department of Computer Science, University of Kansas. His research interests include document analysis, understanding engineering drawings and machine vision, and he has developed the object-process analysis and design methodology.

Miri Weiss is a Ph.D. candidate in the Faculty of Industrial Engineering and Management, Technion, Israel Institute of Technology. She received her B.Sc. in Industrial Engineering and Management—Information Systems—in 1985, and her M.Sc. in Mechanical Engineering—Computer Aided Design—in 1992, both from the Technion. Between 1985 and 1989 she worked for IBM in information systems development. Her research interests include computer-aided design, variational geometry and the 3D reconstruction of engineering drawings.