ELSEVIER

Pattern Recognition Letters 19 (1998) 265–277

Pattern Recognition Letters

Entity-based aspect graphs: Making viewer centered representations more efficient

Christopher C. Yang [1], Michael M. Marefat [*], Erik J. Johnson

Department of Electrical and Computer Engineering, The University of Arizona, Tucson, AZ 85721, USA

Received 16 May 1997; revised 4 December 1997

Abstract

The aspect graph, a graphical representation of an object's characteristic views has been widely developed by several researchers. However, researchers have stated that aspect graphs are limited due to their high complexity and computational cost. A simple non-convex object, such as a step, has 71 distinct characteristic views (nodes in the aspect graph); more complicated objects could have thousands of characteristic views (nodes). Many characteristic views of an aspect graph are not necessary for many applications. In this paper, a new entity-based aspect graph, EAG, is proposed based on the observation that, for most applications, the visibility of only some of the entities on the object is important. The objects of interest are polyhedral solids. We present algorithms for constructing new entity-based aspect graphs based on the faces, edges and vertices of the object, and for combining and contracting previously constructed EAGs in a database to generate EAGs for new objects. The computation time for construction is reduced, yet sufficient information is provided to the application. © 1998 Elsevier Science B.V. All rights reserved.

Keywords: Aspect graph; Entity-based aspect graph; Characteristic views; Viewer-center representation; Sensor planning; Object recognition

1. Introduction

The aspect graph, AG, is a viewer-centered representation of an object based on concepts generally attributed to Koenderink and van Doorn (1979). Each node of an AG corresponds to a distinct characteristic view domain, and each arc between a pair of nodes represents the adjacency between the corresponding characteristic view domains. Each characteristic view domain has a ‘‘topologically distinct’’ view of the object. Algorithms have been derived to create AGs for three-dimensional objects by following specific rules to partition the three-dimensional space. Detail literature reviews of aspect graph can be found in (Bowyer and Dyer, 1990).

The problem with the AG is its complexity and its limited practical utility. A simple object may have more than a hundred nodes in its aspect graph. The worst-case node complexity is $O(N^9)$ for an N-faced polyhedron. In the 1991 *IEEE Workshop on Directions in Automated CAD-Based Vision* (Bowyer,

[*] Corresponding author. E-mail: marefat@ece.arizona.edu.
[1] Current address: Department of Computer Science and Information Systems, The University of Hong Kong, Hong Kong. E-mail: yang@cs.hku.hk.

0167-8655/98/$19.00 © 1998 Elsevier Science B.V. All rights reserved.
PII S0167-8655(98)00006-3

1991), researchers have discussed why aspect graphs are not (yet) practical for computer vision in a panel. Therefore, an efficient viewer-centered representation of objects, which satisfies the requirements of its intended application, is desired.

In most applications of the aspect graph, only a small fraction of the characteristic views are necessary. Most applications focus on a small fraction of the object of interest. For visual inspection, only the entities that are important for the function of the part are inspected or dimensioned. For example, the length, width and height of a pocket on an object are the entities that are important for computing the volume (function) of the pocket. The observability of other entities on the object is not necessary in this situation. For object recognition, many objects are given, some of which may have common features. However, only the observability of the entities that are related to the unique features on the objects is important for identifying them. In these applications, a reduced aspect graph, which focuses on only the important portion of an object would be of much better practical importance.

1.1. Related aspect graph work

Several researchers have modified aspect graphs in order to reduce their complexity. These modified aspect graphs are based on the properties that are observed in their applications. For example, some characteristic view domains may be so small that they are hardly useful and some characteristic views for which the observable edges correspond to the same line drawings are considered redundant for object recognition.

Weinshall and Werman (1997) define view likelihood and view stability, which are used to identify characteristic views and generic views, respectively. View likelihood measures the probability that a certain view of a given 3D object is observed and view stability measures how little the image changes as the viewpoint is slightly perturbed.

Eggert et al. (1993) introduces the scale space aspect graph, which incorporates a quantitative measure of scale into the qualitative representation of the AG used. The method based on scale assumes that some characteristic views will never be useful because of the size of the domains or the limited visibility of the entities in the object. Using this approach reduces the large set of theoretical aspects to a smaller set of the more significant aspects.

Shimshoni and Ponce (1997) introduce the finite-resolution aspect graphs for polyhedral objects. The finite-resolution aspect graphs are developed based on the ideas that (i) an edge is observable when its visible portion projects onto a segment of length greater than a threshold, (ii) two vertices are distinct when the distance between their projections in the image is greater than a threshold. The threshold is determined based on the resolution of camera. As a result, adjacent regions in classical aspect graph with identical finite-resolution views aspects are merged to produce the finite-resolution aspect graph. However, orthographic camera is used in the finite-resolution aspect graphs. Orthographic model is more limited than the full-perspective model used in the scale-space aspect graph developed by Eggert et al. (1993).

Laurentini (1995) introduced the reduced aspect graph. For topological identification of polyhedral objects from multiple views, nodes in the aspect graph, which represent the same line drawing are redundant. Therefore, any pair of nodes in the aspect graph is merged if their aspects are given the same labels and they are topologically equivalent.

The scale space aspect graph, the finite-resolution aspect graph, and the reduced aspect graph approaches reduce the size of the aspect graph. However, for the application of sensor placement (Yang et al., 1994; Yang and Marefat, 1995; Tarabanis et al., 1995a,b; Trucco et al., 1997) in visual inspection and for object recognition, some unnecessary information is retained and some useful information is lost. A new entity-based aspect graph, which shows only the characteristic views of the entities of interests, is desired. The scale space aspect graph (Eggert et al., 1993) has eliminated characteristic views in which the visibility of the entities is low or the size of the characteristic domains is too small. The finite-resolution aspect graph (Shimshoni and Ponce, 1997) has merged adjacent regions that have identical finite-resolution view aspects. However, the scale space aspect graph and the finite-resolution aspect graph still include many characteristic views representing unnecessary entities in the graph. Thus for application purposes, they can still be further reduced to a smaller size. The reduced aspect graph (Laurentini, 1995) merges the characteristic views (nodes) that have the same line drawings, however, the lines in the line drawings could represent different entities (edge segments) on the object. This results in the identities of the entities in the characteristic views being lost. Consequently, this reduced aspect graph is not appropriate for determining the sensor parameters for inspecting a desired geometric entity. In order to provide sufficient information and reduce the complexity, an aspect graph, which provides all the distinct characteristic views of the object for a set of geometric entities of interests is desired.

1.2. Our contributions

In this paper, we propose the entity-based aspect graph, EAG, in each of which the nodes show only the distinct characteristic views for a set of desired entities. AGs are used for recognizing objects and determining the sensor placement for observing object's entities. Only the observability for the entities of interest is important for these applications, so any characteristic views, which do not differ in their views of the desired entities will not be necessary. In this paper, the entities of interest are selected by the users instead of automatically computed. EAGs reduce the complexity of creating the aspect graph, yet provide enough information for applications.

The contributions of this paper can be summarized as:

- Introducing a new entity-based aspect graph based on the assumption that only the observability of some entities of interest is useful for applications.
- Algorithm for creating an EAG from the faces, edges and vertices of an object is presented. Algorithms are also provided for reusing previously constructed EAGs by contraction and combination. Constructing an aspect graph by using previously built aspect graphs has not previously been proposed. Such a technique allows aspect graphs to be formed more efficiently while allowing the earlier-built aspect graphs to be reused.
- An analysis of the complexity of computing the EAG is presented. It is shown that complexity is decreased, especially significantly when the entities of interest in the EAG is small compared to all the entities of the original AG.

2. Aspect graph and entity-based aspect graph

The AG is used in application to recognize objects and determine sensor placements that allow observation of certain entities of an object without occlusion. For example, given the model of an object, Yang et al. (1994) and Yang and Marefat (1995) utilized the AG for determining the sensor location and orientation that can observe a set of desired entities of the object. In most applications, given the entities of interest, only some of the characteristic views in the complete aspect graph are important. Hence, one can construct the characteristic view domains for these entities of interest and form an aspect graph such that each node of the graph is concerned with only the visibility of these entities. Using this approach, fewer characteristic view domains are formed and less computation time is needed, yet sufficient information is provided for the purpose of the application. The proposed *entity-based aspect graph* is based on the assumption that only some entities or features of the object are used in an application.

The AG is a pair (C,A) where C is the set of the characteristic view domains represented as nodes, $\{C_1,C_2,C_3,\ldots\}$, and A is the set of the adjacent pairs of characteristic view domains represented as arcs, $\{\ldots,(C_i,C_j),\ldots\}$. The EAG is a quadruple (E,V,O,A). E is a set of entities of interest for the object, $\{\ldots,e_i,\ldots,e_j,\ldots\}$, such that the EAG only provides the observability of these entities. V is a set of viewing domains, $\{V_1,V_2,V_3,\ldots\}$, in the three-dimensional space. O is the set of observable lists of entities for each element in V, $\{O_{v_1},O_{v_2},O_{v_3},\ldots\}$, where O_{v_i} is the list of observable entities of entity viewing domain V_i. An entity is observable in an entity viewing domain only if no portion of this entity is occluded. Similar to the AG, A is the set of adjacent pairs of entity viewing domain, $\{\ldots,(V_i,V_j),\ldots\}$. In this paper, the domain of considered objects are restricted to planar-faced solids.

Freeman (1990) and Bowyer et al. (1988, 1993) developed algorithms to construct AG for planar-faced solids. Basically, there are three types of partitioning rules for determining the characteristic view domains. Bowyer has described two visual events, the edge–vertex (EV) event and the edge–edge–edge (EEE) event. These visual events are similar to the partitioning types (type A, type B, type C), developed by Freeman. The EV events contain two categories. One involves edge–vertex pairs on the same face and the other involves pairs on separate faces. The first and second category of the EV event corresponds to the type-A partition surfaces and type-B partition barriers of Freeman's method, respectively. The EEE event is the general case where three edges share no common vertex and it corresponds to the type-C quadric surfaces.

The partition rules introduced by Freeman and Bowyer divided the three-dimensional space into regions such that each region (characteristic view domain) has a distinct view of the object. Each partition plane or surface is constructed based on the edges and/or vertices of the object to divide the space into two regions. The vantage points from one region are able to observe the corresponding edges or vertices but the vantage points from the other region are unable to do so. To view a face of a convex planar-faced solid, the vantage point is located at the positive side of the face. The positive side of the face is defined as the half space exterior to the object. Type-A partition surface developed by Freeman (1990) is obtained by expanding the object faces. This partitioning rule is the only rule required for convex planar-faced solids. For non-convex solids, two additional partitioning rules, type-B partition barriers and type-C partition quadric surfaces are needed. Type-B partition barriers are used to separate the vantage points, which have the same visible set of faces but different visible set of vertices. The barrier is a bounded planar region formed by the vertices and edges of the non-convex part of the object. The visibility status of the vertices and the edges are determined by which side of the bounded barrier the vantage point is located. Type-C partition quadric surfaces are used to separate the vantage points, which have the same visible set of faces and vertices but different structural features. For planar-faced solids, a T-junction on the image is a virtual junction that is not a real vertex of the object. The visibility of a T-junction formed by two non-adjacent edges is determined by type-C partition quadric surfaces. Given three edge lines of the model, e_1, e_2 and e_3, a straight line may pass through all the edges with points p_1, p_2 and p_3 as the intersection points of the straight line with the edges e_1, e_2 and e_3, respectively. The partitioning surfaces are composed of infinite number of these straight lines that intersect with the edges e_1, e_2 and e_3. The T-junction is visible from one half space and invisible from the other half space.

Given a set of entities of interest (EOI), one can use all the partition planes and surfaces of the object and eliminate those that are not necessary for constructing the EAG of the object. These partition planes or surfaces formed by the vertices and edges, which are not elements of the set of EOI are eliminated, because the formed planes and surfaces partition only the regions having different observability with respect to the entities that are not in the EOI. The domain formed is possibly the combination of several characteristic-view domains of the AG. Although each domain may contain more than one characteristic view of the object; each of these characteristic views will observe the same entities in the set E of the EAG.

Algorithm for constructing EAG with EOI $= E$
Input: The boundary representation of the object (vertices, edges and faces).
Output: An EAG with set of viewing domains, V, set of lists of observable entities, O, one list for each element of V, and set of adjacent pairs of viewing domain, A, EAG (E,V,O,A).

1. Find all the partition planes and surfaces (using type-A, type-B and type-C partition planes and surfaces described by Freeman (1990)) and eliminate those generated by entities that are not elements of E, $\{p_1, p_2, \ldots, p_n\}$.
2. Construct all possible n-tuples, $L = \{\{p_1^+, p_2^+, \ldots, p_n^+\}, \{p_1^-, p_2^+, \ldots, p_n^+\}, \ldots, \{p_1^-, p_2^-, \ldots, p_n^-\}\}$, where p_i^+, p_i^- are inequalities denoting the different half spaces of the partition plane p_i.
3. For each n-tuple,
   Determine the feasibility of the three-dimensional region described by the n-tuple. (If a point cannot belong to the corresponding set of n inequalities, the region is infeasible.)
   If such an n-tuple is feasible,

```
        Determine the exact boundaries of such a three-dimensional region. (If all the intersections of an element in $n$-tuple (a half-space) with the other half spaces in the $n$-tuple are not on the boundary of the region represented by the $n$-tuple, then that element (half space) can be removed from the $n$-tuple.)
        Save the reduced tuple in $V$.
        Save the corresponding list of observable entities in $O$. (The list of observable entities is the union of entities observable from each $p_i^+$ in the $n$-tuple.)
4. For each pair of elements in $V$,
    If these elements share the same boundary in the three-dimensional space,
        The corresponding regions are adjacent.
        Save the pair in $A$.
```

Fig. 1(b) shows the AG of an object, the step shown in Fig. 1(a) and Fig. 1(c) shows the EAG with $E=\{e_1,e_2,e_3,e_4,e_5,e_6,e_7,v_1,v_2,v_3,v_4,v_5,v_6\}$ (see also Table 1). The label of the entities is given in Fig. 1(a). The constructed AG has 71 nodes which is also the number reported by Freeman and Bowyer, while the EAG has only 20 nodes. To construct the EAG in Fig. 1(b) with the set of entities of interest labeled, 14 out of 16 possible partition planes are used. However, the number of nodes the EAG contains is only 28% of the number in the complete aspect graph. The complexity in the EAG is significantly reduced. For constructing the aspect graph for the step (Fig. 1(a)), there are eight type-A and eight type-B partition planes. The type-B partition planes are formed by eight edge–vertex pairs, $\{e_1,v_6\}$, $\{e_2,v_2\}$, $\{e_3,v_5\}$, $\{e_3,v_6\}$, $\{e_5,v_1\}$, $\{e_5,v_2\}$, $\{e_6,v_5\}$ and $\{e_7,v_1\}$. Each of these edge–vertex pairs contains the elements of the entities of interest, therefore, none of

Table 1
The list of observable entities in each of the viewing domains of the entity-based aspect graph shown in Fig. 1(c)

Nodes	Visibility
O_1	e_1,e_2,v_1,v_3,v_5
O_2	e_6,e_7,v_2,v_4,v_6
O_3	$e_1,e_2,e_3,e_4,e_5,e_6,e_7,v_1,v_2,v_3,v_4,v_5,v_6$
O_4	e_3,e_5,v_1,v_2,v_5,v_6
O_5	e_3,e_5,v_1,v_2,v_5,v_6
O_6	$e_1,e_2,e_5,v_1,v_3,v_5,v_6$
O_7	$e_1,e_2,e_5,v_1,v_3,v_5,v_6$
O_8	$e_1,e_2,e_3,v_1,v_2,v_3,v_5$
O_9	$e_1,e_2,e_3,v_1,v_2,v_3,v_5$
O_{10}	$e_5,e_6,e_7,v_2,v_4,v_5,v_6$
O_{11}	$e_5,e_6,e_7,v_2,v_4,v_5,v_6$
O_{12}	$e_3,e_6,e_7,v_1,v_2,v_4,v_6$
O_{13}	$e_3,e_6,e_7,v_1,v_2,v_4,v_6$
O_{14}	$e_1,e_2,e_3,e_5,v_1,v_2,v_3,v_5,v_6$
O_{15}	$e_1,e_2,e_3,e_5,v_1,v_2,v_3,v_5,v_6$
O_{16}	$e_3,e_5,e_6,e_7,v_1,v_2,v_4,v_5,v_6$
O_{17}	$e_3,e_5,e_6,e_7,v_1,v_2,v_4,v_5,v_6$
O_{18}	e_3,v_1,v_2
O_{19}	e_5,v_5,v_6
O_{20}	none

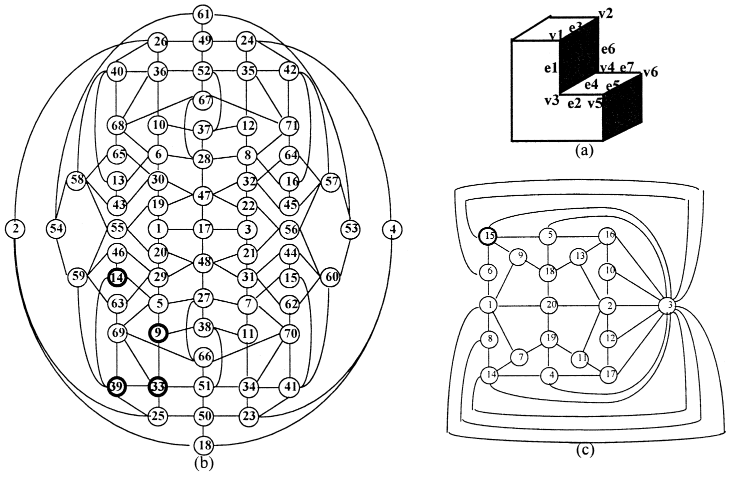

Fig. 1. (a) An object with a blind step. (b) AG of object in (a). (c) EAG with $E=\{e_1,e_2,e_3,e_4,e_5,e_6,e_7,v_1,v_2,v_3,v_4,v_5,v_6,v_7\}$ as labeled in (a).

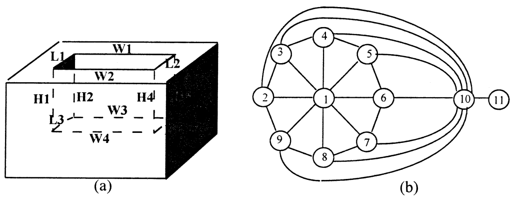

Fig. 2. (a) An object with a pocket on the top face. (b) The entity-based aspect graph of the object in (a) with the set of entities of interest as $\{L_1,L_2,L_3,L_4,W_1,W_2,W_3,W_4,H_1,H_2,H_3,H_4\}$.

Table 2
The sets of observable entities in each viewing domain of the entity-based aspect graph of the object in Fig. 2(a)

Node	Visibility
O_1	$L_1,L_2,L_3,L_4,W_1,W_2,W_3,W_4,H_1,H_2,H_3,H_4$
O_2	$L_1,L_2,L_4,W_1,W_2,H_3,H_4$
O_3	L_1,L_2,W_1,W_2,H_4
O_4	$L_1,L_2,W_1,W_2,W_4,H_1,H_4$
O_5	L_1,L_2,W_1,W_2,H_1
O_6	$L_1,L_2,L_3,W_1,W_2,H_1,H_2$
O_7	L_1,L_2,W_1,W_2,H_2
O_8	$L_1,L_2,W_1,W_2,W_3,H_2,H_3$
O_9	L_1,L_2,W_1,W_2,H_3
O_{10}	L_1,L_2,W_1,W_2
O_{11}	none

them can be eliminated in constructing the EAG. The eight type-A partition planes actually correspond to the planes that are extended by the eight faces of the object. Six of these faces have some entities of interest along their boundaries, which means that the corresponding type-A partition planes are formed by the edge–vertex pairs, with the entities of interest as their elements. Therefore, only the other two type-A partition planes can be eliminated in constructing the EAG.

Fig. 2(a) shows an object with a pocket on the top face with all pocket edges labeled. If the entire pocket was a shape feature of interest, the 12 edges on the pocket are the elements of the set of EOI. In order to determine the possible characteristic views of these entities, an EAG with the set of entities, $E=\{L_1,L_2,L_3,L_4,W_1,W_2,W_3,W_4,H_1,H_2,H_3,H_4\}$, is constructed as shown in Fig. 2(b). Similarly, Fig. 8(a) in shows another object with a blind step. The corresponding entity-based aspect graph is constructed for the entities, $E=\{L_1,L_2,L_3,W_1,W_2,W_3,H_1,H_2,H_3\}$, as given in Fig. 8(b). The lists of observable entities in the viewing domains for EAGs in Fig. 2(b) and Fig. 8(b) are given in Tables 2 and 7, respectively.

Using an EAG, the complexity of the graph is reduced, while all important characteristic views of the object are captured. In object identification, for example, using an EAG allows identification of an object to be performed with a reduced number of matching, and unnecessary information is eliminated.

3. Construction of entity-based aspect graph by contraction and combination of previously computed EAGs

One can form an entity-based aspect graph by contraction or combination of the EAGs that previously have been constructed and saved in a database. Using these techniques, the burden of determining the characteristic viewing domains from the partitioning planes or surfaces is reduced and the previously built information is reused. The contraction of an entity-based aspect graph is accomplished by identifying adjacent nodes, which have the same observability of the desired entities and deleting the arcs between them. The combination of entity-based aspect graphs is achieved by checking the intersec-

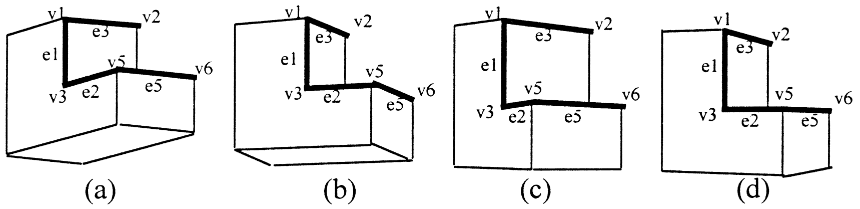

Fig. 3. The characteristic views of four nodes in the aspect graph in Fig. 1(b). (a) Characteristic view of node 9. (b) Characteristic view of node 14. (c) Characteristic view of node 33. (d) Characteristic view of node 39.

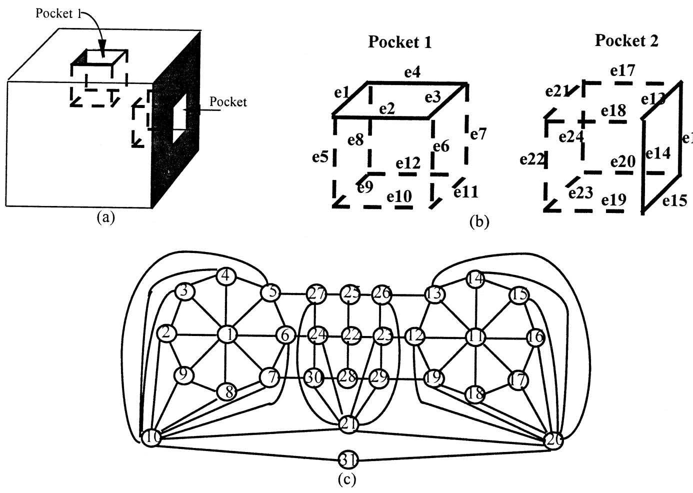

Fig. 4. (a) An object with two pockets, Pocket 1 and Pocket 2, (b) the labels for the entities of Pocket 1 and entities of Pocket 2, (c) EAG with $E=\{e_1,e_2,e_3,e_4,e_5,e_6,e_7,e_8,e_9,e_{10},e_{11},e_{12},e_{13},e_{14},e_{15},e_{16},e_{17},e_{18},e_{19},e_{20},e_{21},e_{22},e_{23},e_{24}\}$.

tions of the entity viewing domains from different EAGs, constructing new partition planes or surfaces as necessary, and creating new nodes and arcs.

3.1. Contraction

If the EAG with the entities set, E, for an object is saved in the database, one can easily construct another EAG with the entities set, E' which is a subset of E, by contracting the existing EAG. Using the contraction technique, the process of determining the characteristic viewing domains from the partitioning planes or surfaces is no longer necessary. The new EAG is constructed by manipulating the EAG in the database.

The contraction of a graph, $G\backslash a$, is obtained by removing an arc a and identifying the nodes, n_1 and n_2, which are incident to the arc a. The resulting node, n_3, is then incident to those arcs, which were originally incident to n_1 and n_2. Assume $E^1=\{\dots,e_i,e_j,e_k,\dots\}$ and $E^2=\{e_i,e_j,e_k\}$, and E^2 is a subset of E^1 $(E^2\subset E^1)$, therefore, EAG^2 can be constructed by contracting EAG^1. Let us assume that EAG^1 $=\{E^1,V^1,O^1,A^1\}$ and EAG^2 $=\{E^2,V^2,O^2,A^2\}$. $V^1=\{v_1^1,v_2^1,\dots,v_i^1,\dots,v_j^1,\dots,v_n^1\}$; $O^1=\{O_{v_1}^1,O_{v_2}^1,\dots,O_{v_i}^1,\dots,O_{v_j}^1,\dots,O_{v_n}^1\}$; $A^1=\{(v_i^1,v_j^1),\dots,(v_k^1,v_l^1),\dots\}$. If $(O_{v_i}^1\cap E^2)=(O_{v_j}^1\cap E^2)$ and (v_i^1,v_j^1) is an element of A^1, v_i^1 and v_j^1 are contracted to produce a new node v_i^2, $O_{v_i}^2$ is equal to $(O_{v_i}^1\cap E^2)$. (v_i^1,v_j^1) is deleted from the set of adjacencies. v_i^2 is substituted for v_i^1 and v_j^1 if any elements of the set of adjacencies contain v_i^1 or v_j^1. For example, the EAG in Fig. 1(c) is a contraction of the AG in Fig. 1(b).

```
Algorithm for contraction of entity-based aspect graph EAG$^1$ to construct EAG$^2$ (where $E^2\subseteq E^1$) CONTRACT_EAG(EAG$^1$)
Input: EAG$^1$($E^1$,$V^1$,$O^1$,$A^1$), $E^2$
Output: EAG$^2$($E^2$,$V^2$,$O^2$,$A^2$)
1. Initialize EAG$^2$
     Set $V^2=V^1$, $O^2=O^1$, $A^2=A^1$.
2. Determine the visibility of the entities in $E^2$ for each element in $V^2$
     For every element of $O^2$,
         Update the list of observable entities to be the intersection of the original list of observable entities and $E^2$. ($O_{v_i}^2=O_{v_i}^1\cap E^2$)
3. Contract the adjacent viewing domains if their observable entities are the same
     For every element of $A^2$, ($V_i^2$,$V_j^2$),
         If the updated lists of observable entities for the two adjacent entity viewing domains, $V_i^2$ and $V_j^2$, are the same, ($O_{v_i}^2=O_{v_j}^2$)
         Contract $V_i^2$ and $V_j^2$
         Replace $V_i^2$ and $V_j^2$ by a new viewing domain, $V_k^2$
         Set the corresponding list of observable entities $O_{v_k}^2$ as the updated observable entities, ($O_{v_k}^2=O_{v_i}^2$)
```

```
        Remove $O^2_{v_i}$ and $O^2_{v_j}$ from $O^2$
        Remove the current arc of EAG$^2$, $(V_i^2, V_j^2)$, from $A^2$
        For every element of $A^2$, substitute $V_i^2$ and $V_j^2$ by $V_k^2$
```

Fig. 3 shows four characteristic views from the aspect graph in Fig. 1(b) which are contracted into one characteristic view for the EAG in Fig. 1(c). The intersection of the observable entities in these characteristic views and the set of entities of interest of the EAG are the same. The entities of interest as shown in Fig. 1(a) are $E = \{e_1, e_2, e_3, e_5, v_1, v_2, v_3, v_5, v_6\}$. Since their corresponding nodes are adjacent to one another, they are contracted into one node, node 15, in Fig. 1(c). All arcs that are incident to the four nodes become incident to the contracted node.

3.2. Combination

If the EAGs of higher level generic shape features, such as slots, pockets, steps, etc. are stored in a database, one can easily construct the EAG of an object with an aggregation of such geometric features by combining the appropriate EAGs available in the database. Using the combination techniques, the EAGs of the more complicated objects with multiple features can be generated more efficiently.

Let us assume that EAG$^1 = \{E^1, V^1, O^1, A^1\}$ and EAG$^2 = \{E^2, V^2, O^2, A^2\}$ are available. One can form EAG$^3 = \{E^3, V^3, O^3, A^3\}$, such that the entities of interest for this new EAG, $E^3 = E^1 \cup E^2$, by combining EAG^1 and EAG^2. Given $V^1 = \{v_1^1, v_2^1, \ldots, v_i^1, \ldots, v_j^1, \ldots, v_n^1\}$, $V^2 = \{v_1^2, v_2^2, \ldots, v_i^2, \ldots, v_j^2, \ldots, v_m^2\}$, if the intersection of the entity viewing domains, v_i^1 and v_j^2, is not an empty space ($v_i^1 \cap v_j^2 \neq \emptyset$), new viewing domains are formed based on their intersection in three-dimensional space. New viewing domains can also be formed by new partition planes or surfaces created by the entities in E^1 and E^2. The new partition planes or surfaces intersect with the existing partition planes and surfaces to create new viewing domains. V^3 is the union of all v_i^1 and v_j^2 that do not intersect with the new viewing domains, and the newly formed viewing domains. The adjacencies in A^3 are reconstructed based on the common surfaces shared with the new viewing domains. The algorithm to determine the intersection of two three-dimensional regions is described by Mäntyla (1988) in detail.

```
    **Algorithm for combination of entity-based aspect graphs** EAG$^1$ **and** EAG$^2$ **to construct** EAG$^3$
    COMBINE_EAG(EAG$^1$,EAG$^2$)
    *Input*: EAG$^1$$(E^1, V^1, O^1, A^1)$,
    EAG$^2$$(E^2, V^2, O^2, A^2)$, $E^3 = E^1 \cup E^2$
    *Output*: EAG$^3$$(E^3, V^3, O^3, A^3)$
1. **Initialize** EAG$^3$
        Set $V^3 = \{\}$, $O^3 = \{\}$, $A^3 = \{\}$.
2. **Combine** EAG$^1$ **and** EAG$^2$
        For every element in $V^1$, $V_i^1$, and every element in $V^2$, $V_j^2$,
```

Table 3
The visibility of the geometric entities of pocket 1 and pocket 2 on the object in Fig. 4(a) in each of the EAG viewing domains

Nodes	Visibility
1	$e_1,e_2,e_3,e_4,e_5,e_6,e_7,e_8,e_9,e_{10},e_{11},e_{12}$
2	$e_1,e_2,e_3,e_4,e_6,e_7,e_{11}$
3	e_1,e_2,e_3,e_4,e_6
4	$e_1,e_2,e_3,e_4,e_5,e_6,e_{10}$
5	e_1,e_2,e_3,e_4,e_5
6	$e_1,e_2,e_3,e_4,e_5,e_8,e_9$
7	e_1,e_2,e_3,e_4,e_8
8	$e_1,e_2,e_3,e_4,e_7,e_8,e_{12}$
9	e_1,e_2,e_3,e_4,e_7
10	e_1,e_2,e_3,e_4
11	$e_{13},e_{14},e_{15},e_{16},e_{17},e_{18},e_{19},e_{20},e_{21},e_{22},e_{23},e_{24}$
12	$e_{13},e_{14},e_{15},e_{16},e_{19},e_{20},e_{23}$
13	$e_{13},e_{14},e_{15},e_{16},e_{19}$
14	$e_{13},e_{14},e_{15},e_{16},e_{18},e_{19},e_{22}$
15	$e_{13},e_{14},e_{15},e_{16},e_{18}$
16	$e_{13},e_{14},e_{15},e_{16},e_{17},e_{18},e_{21}$
17	$e_{13},e_{14},e_{15},e_{16},e_{17}$
18	$e_{13},e_{14},e_{15},e_{16},e_{17},e_{20},e_{24}$
19	$e_{13},e_{14},e_{15},e_{16},e_{20}$
20	$e_{13},e_{14},e_{15},e_{16}$
21	$e_1,e_2,e_3,e_4,e_{13},e_{14},e_{15},e_{16}$
22	$e_1,e_2,e_3,e_4,e_5,e_8,e_9,e_{13},e_{14},e_{15},e_{16},e_{19},e_{20},e_{23}$
23	$e_1,e_2,e_3,e_4,e_{13},e_{14},e_{15},e_{16},e_{19},e_{20},e_{23}$
24	$e_1,e_2,e_3,e_4,e_5,e_8,e_9,e_{13},e_{14},e_{15},e_{16}$
25	$e_1,e_2,e_3,e_4,e_5,e_{13},e_{14},e_{15},e_{16},e_{19}$
26	$e_1,e_2,e_3,e_4,e_{13},e_{14},e_{15},e_{16},e_{19}$
27	$e_1,e_2,e_3,e_4,e_5,e_{13},e_{14},e_{15},e_{16}$
28	$e_1,e_2,e_3,e_4,e_8,e_{13},e_{14},e_{15},e_{16},e_{20}$
29	$e_1,e_2,e_3,e_4,e_{13},e_{14},e_{15},e_{16},e_{20}$
30	$e_1,e_2,e_3,e_4,e_8,e_{13},e_{14},e_{15},e_{16}$
31	none

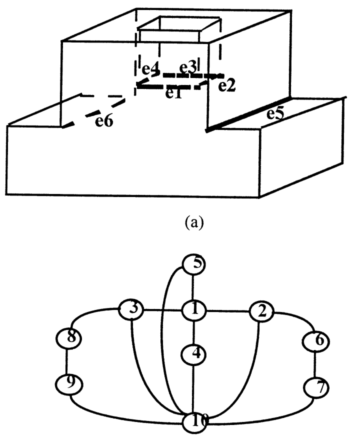

Fig. 5. (a) An object with two steps and a pocket. (b) The entity-based aspect graph with $E=\{e_1,e_2,e_3,e_4,e_5,e_6\}$, in (a).

If V_i^1 intersects with V_j^2 in the three-dimensional space,
If $V_i^1 \subseteq V_j^2$,
Add $V_k = V_i^1 \cap V_j^2$ and $V_k'' = V_j^2 - (V_i^1 \cap V_j^2)$ to V^3.
Else if $V_j^2 \subseteq V_i^1$,
Add $V_k = V_i^1 \cap V_j^2$ and $V_k' = V_i^1 - V_i^1 \cap V_j^2$ from to V^3.
Else
Add $V_k = V_i^1 \cap V_j^2$, $V_k' = V_i^1 - V_i^1 \cap V_j^2$ and $V_k'' = V_j^2 - V_i^1 \cap V_j^2$ to V^3.
The viewing domain formed by $V_k = V_i^1 \cap V_j^2$ has a list of observable entities $O_{V_k} = O^1_{V_i} \cap O^2_{V_j}$.
The viewing domain formed by $V_k' = V_i^1 - V_i^1 \cap V_j^2$ has a list of observable entities $O_{V_k'} = O^1_{V_i}$.
The viewing domain formed by $V_k'' = V_j^2 - V_i^1 \cap V_j^2$ has a list of observable entities $O_{V_k''} = O^2_{V_j}$.

3. If the entities of E^1 and E^2 form any new partition planes,
   Construct the new partition planes. Find and add to V^3 the new viewing domains generated by these partitions.
   (The new viewing domains are formed by the new partition planes and the existing partition planes using the technique described in Section 2.)
4. For every pair of viewing domains in V^3, (V_i^1, V_j^2),
   If V_i^1 and V_j^2 share a common surface on the boundaries, add (V_i^1, V_j^2) to A^3.

Fig. 4 shows an object with two pockets. Pocket 1 is on the top face and Pocket 2 is on one of the side faces. Each of the EAGs, $\mathrm{EAG}^{\mathrm{Pocket1}}$ and $\mathrm{EAG}^{\mathrm{Pocket2}}$, based on the entities of Pocket1, $\{L_1,L_2,L_3,L_4,W_1,W_2,W_3,W_4,H_1,H_2,H_3,H_4\}$, and the entities of Pocket2, $\{L_5,L_6,L_7,L_8,W_5,W_6,W_7,W_8,H_5,H_6,H_7,H_8\}$, is isomorphic to the EAG which is shown in Fig. 2(b). Each, $\mathrm{EAG}^{\mathrm{Pocket1}}$ and $\mathrm{EAG}^{\mathrm{Pocket2}}$, provides a viewer-centered representation in which the only entities of interest are those of one of these pockets along with the viewing domains for an object having one such pocket. However, the description of the viewing domains and the visibility of entities for an object having both of such pocket features, both Pocket1 and Pocket2, is not provided in $\mathrm{EAG}^{\mathrm{Pocket1}}$ or $\mathrm{EAG}^{\mathrm{Pocket2}}$. In order to obtain the viewing domain for an object having both such pockets, a new EAG, $\mathrm{EAG}^{\mathrm{Pocket1} \cup \mathrm{Pocket2}}$, is necessary. $\mathrm{EAG}^{\mathrm{Pocket1} \cup \mathrm{Pocket2}}$ can be constructed by combining $\mathrm{EAG}^{\mathrm{Pocket1}}$ and $\mathrm{EAG}^{\mathrm{Pocket2}}$. Fig. 4(c) shows the resulting $\mathrm{EAG}^{\mathrm{Pocket1} \cup \mathrm{Pocket2}}$, which is obtained from application of the combination algorithm described. The visibility of the entities for the nodes (viewing domains) in Fig. 4(c) is given in Table 3.

The combination algorithm is restricted to combine EAGs where the higher level features such as pockets, or steps, are not interacting with each other

Table 4
The list of observable entities in the viewing domains of the EAG in Fig. 5(b)

Node	Visibility
O_1	e_1,e_2,e_3,e_4
O_2	e_4
O_3	e_2
O_4	e_3
O_5	e_1
O_6	e_4,e_5
O_7	e_5
O_8	e_2,e_6
O_9	e_6
O_{10}	none

Table 5
Computation time (in seconds) for generating AG and EAGs for Object 1 and Object 2

	AG (20 entities)	EAG (13 entities)	EAG (9 entities)	EAG (7 entities)	EAG (5 entities)
Object 1	22.441	7.428	2.346	0.682	0.133
Object 2	28.729	8.337	2.694	0.810	0.229

on the new object. For example, if the EAG for a step is available and a new object has two steps intersecting with one another, combining two EAGs the available step does not create the EAG for the new object. The reason for this is because new edge segments (new entities) are generated when two steps intersect one another. The presented algorithm should be used only to combines EAGs when the higher level features do not intersect on the new object.

4. Experimental results and discussion

In this section, two experiments are described to illustrate the space and time savings.

4.1. Reducing characteristic views

The number of characteristic views using an EAG for the object, as shown in Fig. 5(a) with two steps on two opposite sides and a pocket on the top face, is greatly reduced. Recall that in the aspect graph for a step as shown in Fig. 1, there are 71 characteristic views. Because of larger number of faces, edges, vertices and higher complexity, the aspect graph of the object in Fig. 5(a) could have hundreds of nodes (or more). Constructing such an aspect graph is both complicated and time consuming. If the entities of interest are only e_1, e_2, e_3, e_4, e_5 and e_6 as labeled in Fig. 5(a), using the described techniques, an entity-based aspect graph with $E=\{e_1,e_2,e_3,e_4,e_5,e_6\}$ can be constructed as shown in Fig. 5(b) (see also Table 4). There are only 10 nodes in this entity-based aspect graph. Much unnecessary information is automatically removed.

4.2. Reducing computation

The computation time for generating EAGs diminishes as the number of entities of interest decreases. In this experiment, three objects were investigated. The experiments were run on a Sun SPARCstation IPC using SmallTalk (an object oriented programming language from Xerox PARC). Object 1, Object 2 and Object 3 are shown in Fig. 7. Complete aspect graphs were built and the EAGs with different number of entities of interest were also generated. Object 1 and Object 2 each has 20 entities, and generating the AGs for these objects took from 22 to 29 seconds. However, when the number of entities of interest was reduced to 13, it took only one third of that time to generate the corresponding EAGs. When the number of entities of interest was reduced to five (one fourth of all the entities on the objects) the computation time was decreased to 0.133 seconds. A significant amount of time was saved using entity-based aspect graph. A third object with 13 entities, Object 3, was also tested. Similar results were obtained. A significant portion of computation time was saved when the EAGs were generated. The computation times for generating AGs and EAGs for Object 1, Object 2 and Object 3 are shown in Tables 5 and 6 and Fig. 6.

5. Complexity analysis

The construction procedures are very tedious and time consuming, and some aspect graph generation algorithms, only find the different characteristic views of the object and not the spatial boundaries of each viewing region. Therefore, each characteristic

Table 6
Computation time (in seconds) for generating AG and EAGs for Object 3

	AG (13 entities)	EAG (10 entities)	EAG (6 entities)	EAG (4 entities)
Object 3	9.473	2.722	0.738	0.125

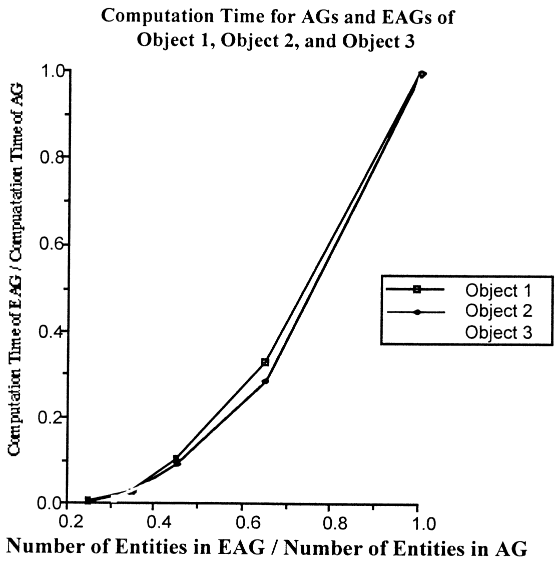

Fig. 6. The experimental computation time for generating the aspect graph and the entity-based aspect graphs for Object 1, Object 2 and Object 3.

view is represented as one of the potential vantage points that represent such view. However, for many applications such as sensor planning, it is desirable to have the exact boundaries of the spatial region from which all the vantage points with such a view are obtained. The cost of obtaining such additional information is high, so an effective and efficient algorithm is necessary.

Given that n partition planes or surfaces are to be used in constructing an aspect graph, the time complexity of determining a characteristic view for the aspect graph is $O(2^n n^3)$. Let us assume that the n partition planes or surfaces are $\{p_1, p_2, \ldots, p_n\}$. Considering the different half spaces of each of these partitions (denoted p_i^+, p_i^-), an n-tuple, such as $\{p_1^+, p_2^-, \ldots, p_n^+\}$, can be constructed. There are in total 2^n of such n-tuples. If the corresponding partition boundaries of the n-tuple provide a feasible region in the three-dimensional space, each such n-tuple represents a characteristic viewing domain. Given an n-tuple, which represents a feasible characteristic viewing domain, the exact boundaries of the corresponding region are obtained by eliminating those partition planes or surfaces that do not lie on the boundaries of the region. To determine the feasibility and to obtain the exact boundaries for an n-tuple, the time complexity is $O(n^3)$. Hence, the total time complexity of generating the aspect graph is $O(2^n n^3)$.

The set of entities of interest, E, of an EAG is a subset of all the entities of the object, E^T. E^c is an entity set such that $E \cup E^c = E^T$ and $E \cap E^c = \emptyset$. A partitioning plane or surface in generation of an aspect graph can be formed by (a) only the entities in E, (b) the entities in E and the entities in E^c, or (c) only the entities in E^c. Let N_E be the number of partition planes or surfaces that are formed by the entities in cases (a) and (b) above, and let $N_E c$ be the number of partition planes or surfaces that are formed by the entities in case (c). Then, $N = N_E + N_{E^c}$. Since the characteristic views of the AG are formed by N partition planes or surfaces, its time complexity is $O(2^{(N_E+N_{E^c})}(N_E + N_{E^c})^3)$. However, the EAG's characteristic views are partitioned by N_E partition planes and surfaces. Therefore, the time complexity is $O(2^{N_E} N_E^3)$. The time complexity of the

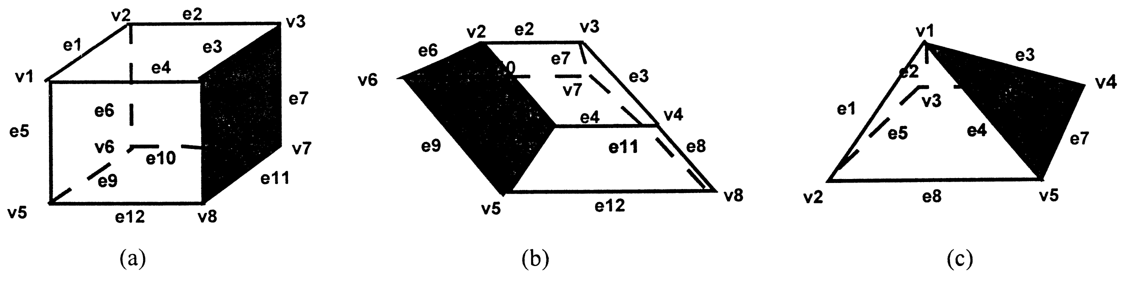

Fig. 7. (a) Object 1, (b) Object 2 and (c) Object 3.

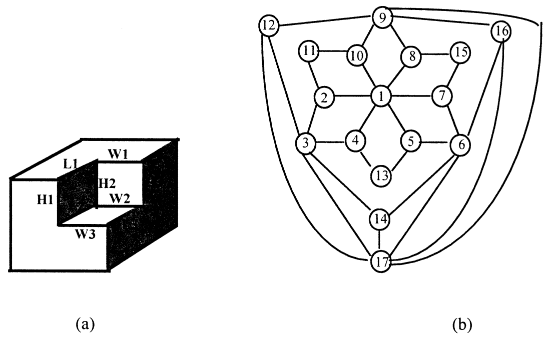

Fig. 8. (a) An object with a blind step. (b) The entity-based aspect graph of the object in Fig. 3(a) with the set of entities of interests as $\{L_1,L_2,L_3,W_1,W_2,W_3,H_1,H_2,H_3\}$.

Table 7
The sets of observable entities in each entities viewing domain of the object in Fig. 8

Noe	Visibility
O_1	$L_1,L_2,L_3,H_1,H_2,H_3,W_1,W_2,W_3$
O_2	L_3,H_1,H_3,W_3
O_3	H_1,W_3
O_4	L_1,H_1,W_1,W_3
O_5	L_1,L_3,H_3,W_1
O_6	L_3,H_3
O_7	L_3,H_1,H_3,W_3
O_8	L_1,H_1,W_1,W_3
O_9	L_1,W_1
O_{10}	L_1,L_3,H_3,W_1
O_{11}	L_1,L_3,H_1,H_3,W_1,W_3
O_{12}	L_1,H_1,W_1,W_3
O_{13}	L_1,L_3,H_1,H_3,W_1,W_3
O_{14}	L_3,H_1,H_3,W_3
O_{15}	L_1,L_3,H_1,H_3,W_1,W_3
O_{16}	L_1,L_3,H_3,W_1
O_{17}	none

entity-based aspect graph will significantly decrease when N_{E^c} is large.

6. Conclusion

The aspect graph, a viewer-centered representation of an object, is useful tool for object recognition, for sensor planning, and several other applications, but it has an important limitation: its complexity. Many research works (Bowyer, 1991, Eggert et al., 1993, Laurentini, 1995) have discussed its practical utility and some have recently proposed aspect graphs having lower complexity based on particular assumptions. In this work, we have described entity-based aspect graphs, which include only the viewing domains that embody distinct visibility characteristics for a particular set of entities of interest belonging to an object. Usually the number of entities of interest is fewer than the total number of entities on the object, and the total number of nodes on the graph is greatly reduced. Using the entity-based aspect graph approach, we still have enough information to plan sensor settings or to recognize the object.

The contributions of this paper can be summarized as follows:

- A new entity-based aspect graph is introduced, based on the assumption that only the observability of some entities of interests are useful for applications such as object recognition and sensor placement.
- Algorithms for creating an EAG from the faces, edges and vertices of an object are presented. Also algorithms for contracting and combining previously constructed EAGs are presented. By employing the presented algorithms, the burden of determining the characteristic views from the partition planes or surfaces can be reduced and the previously built EAGs can be reused.
- The complexity of computing an EAG is analyzed. This analysis shows that complexity is decreased significantly. The computational savings are produced by reducing the number of partition planes or surfaces formed by the involved geometric entities.

Fig. 8 and Table 7 show an object with a blind step.

For further reading, see (Charkravarty and Freeman, 1982; Edelman and Weinshall, 1991; Kriegman and Ponce, 1990; Petitjean et al., 1992; Ponce et al., 1992; Stark et al., 1988; Wang and Freeman, 1990; Watts, 1988; Yang, 1997).

References

Bowyer, K., 1991. Why aspect graphs are not (yet) practical for computer vision. In: IEEE Workshop on Directions in Automated CAD-based Vision, Maui, HI, 2–3 June, pp. 97–104.

Bowyer, K.W., Dyer, C.R., 1990. Aspect graphs: An introduction and survey of recent results. Internat. J. Imaging Systems and Technology 2, 315–328.

Bowyer, K., Stewman, J., Stark, L., Eggert, D., 1988. A 3-D object recognition system using aspect graph. In: 9th Internat. Conf. on Pattern Recognition, Rome, Italy.

Bowyer, K., Sallam, M.Y., Eggert, D.W., Stewman, J.S., 1993. Computing the generalized aspect graph for objects with moving parts. IEEE Trans. Pattern Anal. Machine Intell. 15 (6), 605–610.

Charkravarty, I., Freeman, H., 1982. Characteristic views as a basis for three-dimensional object recognition. SPIE Robot Vision 336, 37–45.

Edelman, S., Weinshall, D., 1991. A self-organizing multiple-view representation of 3D objects. Biological Cybernet. 64 (3), 209–219.

Eggert, D.W., Bowyer, K.W., Dyer, C.R., Christensen, H.I., Goldgof, D.B., 1993. The scale space aspect graph. IEEE Trans. Pattern Anal. Machine Intell. 15 (11), 1114–1130.

Freeman, H., 1990. The Use of Characteristic-View Classes for 3D Object Recognition. Machine Vision for Three-Dimensional Scenes. Academic Press, New York, pp. 109–163.

Koenderink, J.J., van Doorn, A.J., 1979. The internal representation of solid shape with respect to vision. Biological Cybernet. 32.

Kriegman, D.J., Ponce, J., 1990. Computing exact aspect graphs of curved objects: solids of revolution. Internat. J. Comput. Vision 5 (2), 119–135.

Laurentini, A., 1995. Introducing the reduced aspect graph. Pattern Recognition Letters 16, 43–48.

Mäntyla, M., 1988. An Introduction to Solid Modeling. Computer Science Press, Rockville, MD.

Petitjean, S., Ponce, J., Kriegman, D.J., 1992. Computing exact aspect graphs of curved objects: algebraic surfaces. Internat. J. Comput. Vision 9 (3), 231–255.

Ponce, J., Petitjean, S., Kriegman, D.J., 1992. Computing exact aspect graph of curved objects: Algebraic surfaces. In: Proc. 2nd European Conf. on Computer Vision, Santa Margherita Ligure, Italy, 18–23 May, pp. 599–614.

Shimshoni, I., Ponce, J., 1997. Finite-resolution aspect graphs of polyhedral objects. IEEE Trans. Pattern Anal. Machine Intell. 19 (4), 315–327.

Stark, L., Eggert, D., Bowyer, K., 1988. Aspect graphs and nonlinear optimization in 3-D object recognition. In: Proc. IEEE 2nd Internat. Conf. on Computer Vision, Tampa, FL, pp. 501–507.

Tarabanis, K.A., Tsai, R.Y., Allen, P.K., 1995a. The MVP sensor planning system for robotic vision tasks. IEEE Trans. Robotics Automation 11 (1), 72–85.

Tarabanis, K.A., Allen, P.K., Tsai, R.Y., 1995b. A survey of sensor planning in computer vision. IEEE Trans. Robotics Automation 11 (1), 86–104.

Trucco, E., Umasuthan, M., Wallace, A.M., Roberto, V., 1997. Model-based planning of optimal sensor placements for inspection. IEEE Trans. Robotics Automation 13 (2), 182–194.

Wang, R., Freeman, H., 1990. Object recognition based on characteristic view classes. In; Proc. 10th Internat. Conf. on Pattern Recognition, pp. 8–12.

Watts, N.A., 1988. Calculating the principal views of a polyhedron. In: Proc. 9th Internat. Conf. on Pattern Recognition, Rome, Italy.

Weinshall, D., Werman, M., 1997. On view likelihood and stability. IEEE Trans. Pattern Anal. Machine Intell. 19 (2), 97–108.

Yang, C.C., 1997. Active vision inspection: Planning, error analysis, and tolerance design. Chapter 2: Entity-based aspect graphs. Ph.D Dissertation, Dept. of Electrical Computer Engineering, The University of Arizona.

Yang, C.C., Marefat, M.M., 1995. Object oriented concepts and mechanisms for feature-based computer integrated inspection. Adv. in Engineering Software 20 (2–3), 157–179.

Yang, C.C., Marefat, M.M., Kashyap, R.L., 1994. Active visual inspection based on CAD models. In: Proc. IEEE Internat. Conf. on Robotics and Automation, San Diego, CA, pp. 1120–1125.