Optimal Common Subgraph Isomorphism for Model Synthesis

Bruce A. McArthur[1] Andrew K.C. Wong
Department of Systems Design Engineering
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1

Abstract

Matching image features to partial models is a key step in the automated synthesis of geometric models of 3-D objects. This paper introduces an algorithm for optimal common subgraph isomorphism which has been applied to the problem of model-image matching using random graph-based model representations. In the first stage of the algorithm, a set of feasible matches between image and model graph vertices is determined based on the statistical consistency of the vertex attribute probability density functions. The second stage of the algorithm involves a search for the optimal common subgraph isomorphism between the model and image graphs, using the set of feasible vertex matches and a cost function based on the increment of entropy of the model graph structural probability.

1 Introduction

One approach to automatically acquiring geometric models of 3-D environments is to integrate image data obtained from multiple viewpoints using one or more cameras [4, 11, 7]. A key step in this process involves identifying previously observed model features. The partial 3-D model is projected onto the image, using an estimated model-to-image transformation, and projected model features are matched against features in the image. In order for the matching process to be robust, it must take into account the incomplete and uncertain nature of real-world data [2, 4]. Incompleteness arises because, first, only a subset of the model features will be visible from any given viewing location, and second, image data can be guaranteed to contain features which do not correspond to the model object. Uncertainty is a result of the complex transformation (involving geometry, lighting, imaging, and feature extraction) which takes place between the 3-D model and the 2-D image. This guarantees that the attribute values of corresponding features will not match exactly.

In [8, 9] a new class of object representation, based on the random graph, was introduced for application to the incremental synthesis of 3-D object models. The random graph is a probabilistic representation of an ensemble of attributed graphs which can describe variations in both the structure and attribute values of structural patterns. Consequently, the random graph model representation has been shown to be effective in dealing with uncertain and incomplete data. In the random graph object model, vertices represent geometric features, such as points, edges, and planar surfaces, and arcs represent topological relations. Uncertainty in geometric feature attributes and in model structure is described by attaching probability distributions to vertex and arc attribute values of the model.

This paper introduces an algorithm for matching the random graph-based object models to image data, termed *Optimal Common Subgraph Isomorphism for Model Synthesis* (OCSIMS). In the first stage in the OCSIMS algorithm, a set of feasible matches between image and model graph vertices is determined based on the statistical consistency of the vertex attribute probability density functions. Using the set of feasible vertex matches, the second stage of the algorithm involves a search for the optimal common subgraph isomorphism between the model and image graphs. The increment of entropy of the structural probability of the model graph is used as a cost function.

2 Random Graph Representations for 3-D Object Models

Here, we briefly introduce the terminology and concepts for the random graph which are necessary background to the OCSIMS algorithm. More formal definitions are available in [8, 15].

The *random graph* [13, 14, 15] is a probabilistic description of an ensemble of attributed graphs. Let $R = (W, B)$ be a random graph with a set of random vertices $W = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ and a set of random arcs $B = (\beta_1, \beta_2, \ldots, \beta_m)$. Let the attributed graph G be an outcome of R as defined by a monomorphism μ. For each random vertex α_i and random arc β_j, the corresponding attributed vertices and arcs of G are given by $a_i = \mu(\alpha_i)|i = 1, 2, \ldots, n$ and $b_j = \mu(\beta_j)|j = 1, 2, \ldots, m$, respectively. The range of each random vertex and arc includes the *null attribute value* $\emptyset$. This attribute is used to signify a missing vertex or arc without having to modify the underlying graph structure.

[1] Current address: Defence Research Establishment Atlantic, P.O. Box 1012, Dartmouth, NS, B2Y 3Z7

For *first-order* random graphs, the probability of an outcome G according to μ is defined as

$$\begin{aligned} P(G,\mu) &= \prod_{i=1}^{n} Pr(\alpha_i = a_i) \\ &\quad \cdot \prod_{j=1}^{m} Pr(\beta_j = b_j | \sigma(\beta_j) = a_{j1}, \tau(\beta_j) = a_{j2}) \\ &= \prod_{i=1}^{n} P_i(a_i) \prod_{j=i}^{m} P_j(b_j | a_{j1}, a_{j2}), \end{aligned} \quad (1)$$

where $\sigma(\beta_j) = a_{j1}$ and $\tau(\beta_j) = a_{j2}$ are the values assigned to the head and tail of β_j. The first-order random graph is based on the following three assumptions: the random vertices $\{\alpha_i\}$ are mutually independent; given values for the random vertices $\{\alpha_i\}$, the random arcs $\{\beta_j\}$ are independent; and, a random arc β is independent of any random vertex other than its endpoints, $\sigma(\beta)$ and $\tau(\beta)$. In [8, 9] it is demonstrated that the outcome probabilities for first-order random graphs can be expressed as the product of *structural* and *contextual* probabilities, where the structural probability expresses the likelihood of observing a pattern with a specified topology of primitives and relations and the contextual probability expresses the likelihood that the observed structural pattern has a specified set of vertex and arc attribute values.

The random graph object model has been demonstrated using a simple class of wireframe model termed the *point feature* model. Random graph vertices represent point features and random graph edges represent edge feature relations. Point features, defined to be corners and points formed by the intersection of extended edges, are described by 3-D location $\mathbf{x} = (x, y, z)^T$. The uncertainty in point feature $\mathbf{x}_i$ is described by a 3-D Gaussian probability density function with mean $\hat{\mathbf{x}}_i$ and covariance Σ_i. Edge feature relations describe the presence or absence of straight or curved line segments connecting point features. In the case of the point feature model, the structural probability $P_s(G,\mu)$ and contextual probability $p_c(G,\mu)$ take the form

$$\begin{aligned} P_s(G,\mu) &= \prod_{i:\mu(\alpha_i)=\emptyset} P_i(\emptyset) \prod_{i:\mu(\alpha_i)\neq\emptyset} P_i(\neg\emptyset) \\ &\quad \cdot \prod_{j:\mu(\beta_j)=\emptyset} P_j(\emptyset | \mathbf{x}_{j1}, \mathbf{x}_{j2}) \\ &\quad \cdot \prod_{j:\mu(\beta_j)\neq\emptyset} P_j(\neg\emptyset | \mathbf{x}_{j1}, \mathbf{x}_{j2}), \end{aligned} \quad (2)$$

$$\begin{aligned} p_c(G,\mu) &= \prod_{i:\mu(\alpha_i)\neq\emptyset} p_i(\mathbf{x}_i | \neg\emptyset) \\ &= \prod_{i:\mu(\alpha_i)\neq\emptyset} \frac{e^{-1/2(\mathbf{x}_i - \hat{\mathbf{x}}_i)^T \Sigma_i^{-1} (\mathbf{x}_i - \hat{\mathbf{x}}_i)}}{(2\pi)^{3/2} |\Sigma_i|^{1/2}}, \end{aligned} \quad (3)$$

where

$$P_i(\neg\emptyset) = Pr(\alpha_i \neq \emptyset)$$

and

$$P_j(\neg\emptyset | a_1, a_2) = Pr(\beta_j \neq \emptyset | \sigma(\beta_j) \neq \emptyset, \tau(\beta_j) \neq \emptyset)$$

express the probability that a random vertex or arc has a non-null outcome.

The process of matching the projected model data to the image is formulated using two additional first-order random graph representations, which describe the projected 3-D model and the image data. The *projected model graph*, R_p, is derived from the point feature model by projecting the 3-D coordinates of model graph point features onto an image plane. The contextual probability density functions for point features are transformed by the projection process to produce 2-D Gaussian distributions. The structural probability distribution of projected model graph is identical to that of the model graph. Image data is also represented using a random graph model. An *image graph*, R_i, is a restricted form of random graph in which the structural probability is deterministic (since an image represents a single sample). The uncertainty in the location of point features in the image is represented by 2-D Gaussian distributions.

3 Optimal Common Subgraph Isomorphism for Model Synthesis

In the first stage in the OCSIMS algorithm, a set of feasible matches between image and projected model graph vertices is determined based on the statistical consistency of the vertex attribute probability density functions. For each mapping between image graph and projected model graph vertices, we test a null hypothesis that the attribute values are consistent. Consider an image vertex $\alpha_s, \alpha_s \in W(R_i)$, with point location mean $\hat{\mathbf{x}}_s$ and covariance Σ_s; and a projected model vertex $\alpha_u, \alpha_u \in W(R_p)$, with point location mean $\hat{\mathbf{x}}_u$ and covariance Σ_u. Define the null hypothesis

$$H_0 : \alpha_s \equiv \alpha_u$$

$$H_1 : \alpha_s \not\equiv \alpha_u$$

The hypothesis is tested based on the Mahalanobois distance, $d(\alpha_u, \alpha_s)$, given by

$$d(\alpha_u, \alpha_s) = (\hat{\mathbf{x}}_u - \hat{\mathbf{x}}_s)^T (\Sigma_u + \Sigma_s)^{-1} (\hat{\mathbf{x}}_u - \hat{\mathbf{x}}_s), \quad (4)$$

which has a chi-squared distribution with 2 degrees of freedom. The null hypothesis is rejected if $d(\alpha_u, \alpha_s)$ exceeds a threshold t_γ fixed at a confidence level γ. This process is termed *geometric screening* because a subset of the possible matches are screened out based on purely geometric criteria.

Consistent image vertices are then defined to be those vertices for which the null hypothesis is not rejected. The *feasible match set*,

$$F_u = \{\alpha_s | \alpha_s \in W(R_i), d(\alpha_u, \alpha_s) \leq t_\gamma\},$$

is defined to be the set of image vertices which are consistent with respect to the projected model vertex α_u.

The feasible match set for each projected model vertex is augmented with the null match which describes the possibility that there are no image points which match the projected model vertex, i.e. $F'_u = F_u \cup \{\emptyset\}$. Thus, each feasible match set has a minimum order of one. The use of the geometric screening procedure greatly reduces the number of candidate image point matches to each projected model vertex. Experimental tests using real image data show that, on average, there are less than two consistent matches per projected model vertex (not including null matches), and from 1/3 to 1/2 of the projected model vertices will have a single candidate match.

Using the feasible match sets, the second stage of the OCSIMS algorithm involves a search for the optimal common subgraph isomorphism between the projected model graph and image graph, using the increment of entropy of the projected model graph structural probability as a cost function. Consider an image graph, R_i, of order m and a projected model graph, R_p, of order n. Let $\mathcal{U}$ be the set of all complete subgraphs of R_i of order less than or equal to n. In other words, if $\mathcal{U}_k$ is the set of all complete subgraphs of R_i of order k, then $\mathcal{U} = \{\mathcal{U}_k | k = 1, 2, \ldots, n\}$. Given $U \in \mathcal{U}$, let $U' = f(U)$ be the extension of U, with null arcs and vertices, to the order of R_p. Then define $\mathcal{U}' = \{U'_i | U'_i = f(U_i) \forall U_i \in \mathcal{U}\}$. For each $U'_i \in \mathcal{U}'$, we define a set of isomorphisms $M = \{\mu_{ij}\}$ between U'_i and R_p. Let $\mathcal{S} = \{(U'_i, \mu_{ij})\}$ be the set of all possible image subgraph-isomorphism pairs. For each pair (U'_i, μ_{ij}), we compute a distance measure $d(U'_i, \mu_{ij})$. The *optimal common subgraph isomorphism* between R_i and R_p among all possible morphisms μ_{ij} is defined to be the morphism μ^* for which the distance measure $d(U'_i, \mu_{ij})$ is a minimum.

The increment of entropy has been used previously as a distance measure for random graph synthesis [14, 3]. The *entropy* of a random graph R is defined as

$$\begin{aligned} H(R) &= -\sum_{G,\mu} Pr(G,\mu) \log Pr(G,\mu) \\ &= \sum_{i=1}^{n} H(\alpha_i) + \sum_{j=1}^{m} H(\beta_j) \end{aligned} \quad (5)$$

where $H(\alpha_i)$ and $H(\beta_j)$ are the entropies of the random vertices and arcs, respectively. $H(R)$ is a measure of the structural and attribute value variation of the attributed graphs in an ensemble. A low value of $H(R)$ indicates greater similarity among the random graph outcomes.

The increment of entropy of a random element γ, resulting from the synthesis of random elements γ_1 and γ_2 in random graphs R_1 and R_2, is defined as

$$H'(\gamma) = H(\gamma) - q_1 H(\gamma_1) - q_2 H(\gamma_2) \quad (6)$$

where q_1 and q_2 are the relative frequencies of graphs associated with R_1 and R_2 in the combined sample. The increment of entropy of the random graph R, resulting from the synthesis of random graphs R_1 and R_2, is equal to the sum of the increments of entropy calculated over all of the random elements in R as follows

$$H'(R) = \sum_{\gamma \in R} H'(\gamma) = \sum_{i=1}^{n} H'(\alpha_i) + \sum_{j=1}^{m} H'(\beta_j). \quad (7)$$

Expressions for the vertex and arc entropies for structural probability distributions are given by

$$\begin{aligned} H(\alpha_i) &= -P_i(\neg\emptyset) \log P_i(\neg\emptyset) \\ &\quad - (1 - P_i(\neg\emptyset)) \log(1 - P_i(\neg\emptyset)), \end{aligned} \quad (8)$$

$$\begin{aligned} H(\beta_j) &= -P_{j1,j2}(\neg\emptyset) \\ &\quad \cdot \{P_j(\neg\emptyset | a_{j1}, a_{j2}) \log P_j(\neg\emptyset | a_{j1}, a_{j2}) + \\ &\quad (1 - P_j(\neg\emptyset | a_{j1}, a_{j2})) \\ &\quad \cdot \log(1 - P_j(\neg\emptyset | a_{j1}, a_{j2}))\}. \end{aligned} \quad (9)$$

If random vertices are considered to be independent, as is the case for the first-order random graph, then (9) can be rewritten as

$$\begin{aligned} H(\beta_j) &= -P_{j1}(\neg\emptyset) P_{j2}(\neg\emptyset) \\ &\quad \cdot \{P_j(\neg\emptyset | a_{j1}, a_{j2}) \log P_j(\neg\emptyset | a_{j1}, a_{j2}) + \\ &\quad (1 - P_j(\neg\emptyset | a_{j1}, a_{j2})) \\ &\quad \cdot \log(1 - P_j(\neg\emptyset | a_{j1}, a_{j2}))\}. \end{aligned} \quad (10)$$

In the case of incremental synthesis, the increment of entropy resulting from the synthesis of a random graph R_1 (the structural component of the projected model graph) and an attributed graph R_2 (the structural component of the image graph) has a particularly simple form as the entropy of an attributed graph is zero. Given that R_1 is the synthesis of f_1 samples, (6) then simplifies to

$$H'(\gamma) = H(\gamma) - \frac{f_1}{f_1 + 1} H(\gamma_1). \quad (11)$$

Note also that given an edge β in the model graph, if either one of its endpoints are not observed in an outcome graph, then $H'(\beta) = 0$.

4 Algorithm Implementation

The OCSIMS algorithm is implemented using a tree search of similar form to those used by Grimson [5] or Boyer and Kak [1]. Consider the optimal common subgraph isomorphism between a projected model graph of order n and an image graph of order m. Let each level i in the search tree correspond to the index of the vertex in the projected model graph. Each node in the tree is labelled with the ordered pair (i, q_i) which represents the mapping of a vertex i in the projected model graph to a vertex q_i in the image graph (or the "null" mapping $(i, \emptyset)$ if there is no matching vertex in the image graph). The path from the root to any node in the tree defines a partial mapping between the projected model graph and the image graph. For example $\{(1, q_1), (2, q_2), \ldots, (k, q_k)\}$ denotes the sequence

of nodes in the path from the root to the node (k, q_k) at level k. Each leaf node then represents a different subgraph isomorphism defined by the path from the root to the leaf. When null mappings are included, each leaf node represents a common subgraph isomorphism. In the general search tree with null matches, there will be $m - p + r$ sons for each node at level $p = 0, 1, \ldots, m-1$, where r is the number of null matches in the path from the root to the node at level $p (r \leq p)$. The number of possible common subgraph mappings between graphs of order n and m is given by the expression [1]

$$\sum_{\rho=0}^{s} \frac{n!m!}{\rho!(n-\rho)!(m-\rho)!} \tag{12}$$

where ρ is the order of the mapping and $s = \min(m, n)$. However, using the geometric screening procedure, the set of sons for a node at level p is restricted to being a subset of the feasible match set $F'_{\alpha_{p+1}}$, which typically has from one to three members.

The cost, $C(k, q_k)$, of assigning a match (k, q_k) at level k in the search tree is given by

$$C(k, q_k) = c_a(k, q_k) + \sum_{i=1}^{k-1} c_e((i, k), (q_i, q_k)) \tag{13}$$

where $c_a(k, q_k)$ is the increment of entropy $H'(\alpha)$ resulting from the mapping between a vertex k in the projected model graph and a vertex q_k in the image graph and $c_e((i, k), (q_i, q_k))$ is the increment of entropy $H'(\beta)$ resulting from the mapping between the edge (i, k) in the projected model graph and the edge (q_i, q_k) in the image graph. The cost of the partial mapping $\{(1, q_1), (2, q_2), \ldots, (k, q_k)\}$ is given by

$$C_k = \sum_{i=1}^{k} C(i, q_i) \tag{14}$$

The search for the optimal (i.e. minimum cost) mapping is implemented using a uniform-cost branch and bound search, with search tree pre-pruning and pre-ordering. There are two components to the algorithm. First, in the branch and bound technique [12], the lowest cost solution found so far is used as an upper bound on the cost of the optimal solution. Any partial solution with cost exceeding the lowest cost solution is discarded. The corresponding search path is then said to have been *pruned* from the tree. We modify the basic branch and bound pruning technique by also incorporating the number of null matches. Given two solutions with equal costs but differing numbers of null matches, the solution having the fewest null matches is preferred. Thus if a partial solution has a cost equal to the lowest cost solution, it is pruned if its number of null matches exceeds the number of null matches for the current lowest cost solution.

The second component is a strategy for determining which node in the search tree is to be investigated next. This is done using an evaluation function, $f'(N) = g(N) + h'(N)$, which places a lower bound estimate on the cost of any solution path which includes a given node N in the search tree. We can decompose $f'(N)$ into two components: $g(N)$, the actual cost of the path from the root to N (equal to C_k in (14)), and $h'(N)$, the estimated cost of the path from N to a solution node, which is also known as the *heuristic function*.[2] In the *uniform-cost* search [10], $h'(N) = 0$; that is, node selection is based only on the cost of the partial solution. This strategy is used in a modified form by selecting the minimum cost node and, in the case of a tie, by selecting the match with the maximum depth in the search tree. This strategy combines the two goals of first maximizing the pruning effect of a good first solution and of obtaining an initial solution as quickly as possible.

Although the branch and bound search technique can discard many of the false solutions, it still may result in a combinatorial explosion for matches of moderate size. Considerable improvements in search efficiency can be achieved by exploiting the unique characteristics of the search tree which result from the geometric screening procedure and the increment of entropy cost function. These improvements are made by pre-pruning the search tree and ordering the projected model graph vertices. Both methods utilize the upper and lower bounds on the cost of mapping between a projected model graph vertex and an image graph vertex.

4.1 Search Tree Pre-pruning

The first step in pre-pruning is to construct the superset, $\mathcal{F} = \{F'_u | \alpha_u \in W(R_p)\}$ of all feasible match sets. Next, we consider the different conflicts which can occur between feasible vertex mappings. Conflicts may be classified into two types: within a feasible match set (intraset conflicts) and between feasible match sets (interset conflicts). Three types of simple conflicts are of special interest for pre-pruning. A Type I conflict occurs when a feasible match set has no interset or intraset conflicts. In this case, the feasible match set contains only the null match. Type II conflicts occur when a given feasible match set has only intraset conflicts. Type III conflicts occur when interset conflicts occur only between a single pair of feasible match sets, each of which may have multiple intraset conflicts. For the purpose of pre-pruning, we will consider only the simplest class of Type III conflicts; those for which each feasible match set has only a single intraset conflict (i.e. each set has one null match and one non-null match and the two non-null matches conflict).

Feasible matches with Type I conflicts(i.e. null matches) can be assigned immediately. Note that the incremental cost of a null match will only involve a vertex cost. As this is a fixed cost, null matches need not actually be expanded in the tree search procedure.

Pre-pruning of Type II and III conflicts involves calculation of upper and lower bounds on the incremental cost of each feasible match. The lower bound

[2] The primed and unprimed variables denote estimated and known quantities, respectively.

on the cost of any feasible match (k, q_k) is given by $C_l(k, q_k) = c_a(k, q_k)$, whereas the upper bound, $C_u(k, q_k)$ is given by

$$C_u(k, q_k) = c_a(k, q_k) + \sum_{\substack{(i,q_i)\in F \\ i\neq k}} c_e((i,k),(q_i,q_k)) \quad (15)$$

For Type II conflicts, given the feasible match set $F'_u = \{s_i, i = 1, \ldots, n\}$, we prune any feasible match (u, s_j) for which $C_u(u, s_j) < C_l(u, s_i)$ for all $i \neq j$. Pruning of Type II conflicts occurs most frequently for feasible match sets which have two members: i.e., we must choose between a null and a non-null match. In these cases a match can be immediately assigned. Note that for a null match the upper and lower bounds are equal: i.e., $C_u(k, q_k) = c_a(k, q_k)$.

For Type III conflicts, consider the pair of feasible match sets, $F'_u = \{s, \emptyset\}$ and $F'_v = \{s, \emptyset\}$. Matches can be assigned immediately under the following conditions:

1. $\{(u, s), (v, \emptyset)\}$ is assigned if $C_u(u, s) + c_a(v, \emptyset) < c_a(u, \emptyset) + C_l(v, s)$ and $C_u(u, s) < c_a(u, \emptyset)$,
2. $\{(u, \emptyset), (v, s)\}$ is assigned if $c_a(u, \emptyset) + C_u(v, s) < C_l(u, s) + c_a(v, \emptyset)$ and $C_u(v, s) < c_a(v, \emptyset)$,
3. $\{(u, \emptyset), (v, \emptyset)\}$ is assigned if $c_a(u, \emptyset) < C_l(u, s)$ and $c_a(v, \emptyset) < C_l(v, s)$.

Pre-pruning is quite effective because the feasible match sets resulting from geometric screening have from one to three members. Thus a significant number of conflicts fall into one of the three types outlined above.

4.2 Search Tree Pre-ordering

The idea behind pre-ordering the search tree is to reduce the complexity of the search space by placing nodes with the fewest branches at the top of the tree [6]. To this end, we use the following strategy to order the vertices in the projected model graph (corresponding to depth in the search tree).

1. Type I conflicts are assigned and removed from the search tree.
2. First choose projected model graph vertices with assigned matches resulting from Types II and III conflict pre-pruning.
3. Order the remaining vertices by increasing size of feasible match sets, with the restriction that pairs of feasible match sets corresponding to unresolved Type III conflicts are kept together.

Note that the size of feasible match sets may be reduced by pre-pruning.

4.3 Results

The application of pre-pruning and ordering techniques is illustrated using the following example. Consider a set of 14 feasible vertex mappings, between 11 projected model graph vertices and 11 image graph vertices, resulting in the 11 feasible vertex sets shown in Table 1. Costs for mapping arcs and vertices are given in Table 2. Figure 1(a) shows the search tree obtained using the uniform-cost branch and bound search algorithm without pre-pruning or pre-ordering. Levels in the search tree are ordered according to the projected model vertex index. Nineteen nodes are expanded to obtain the minimum cost solution, with 14 of the nodes used to expand the first five levels in the search tree.

Model vertex	Feasible match set
0	26, $\emptyset$
1	23, $\emptyset$
2	27, 28, $\emptyset$
3	27, 28, $\emptyset$
4	22, 29, $\emptyset$
5	23, $\emptyset$
6	24, $\emptyset$
7	25, $\emptyset$
8	20, $\emptyset$
9	21, $\emptyset$
10	16, $\emptyset$

Table 1: Pre-pruning and pre-ordering example: Feasible match sets

Match set	Lower bound	Upper bound
$\{(1,23),(5,\emptyset)\}$	0.93	1.74
$\{(1,\emptyset),(5,23)\}$	0.31	0.31
$\{(1,\emptyset),(5,\emptyset)\}$	1.12	1.12

Table 3: Pre-pruning and pre-ordering example: Analysis of a Type III conflict

Figure 1(b) shows an equivalent partial search tree obtained by applying the pre-pruning and pre-ordering procedure prior to using the branch and bound algorithm. Six of the 11 feasible vertex sets are in the Type II conflict category: i.e., there is only a conflict between the null and non-null match. Of these, five matches can be assigned based on the pre-pruning criteria. Note that in the case of $F_8 = \{20, \emptyset\}$, we choose (8,20) over $(8, \emptyset)$ even though $C_u(8, 20) = C_l(8, \emptyset)$ because this will result in fewer null mappings. The pair of feasible match sets F_1 and F_5 form a Type III conflict. The upper and lower bound costs of each of three possible mappings are shown in Table 3. The pair of mappings $\{(1, \emptyset), (5, 23)\}$ satisfy the pre-pruning criteria for Type III conflicts and, thus, are also immediately assigned before the tree search is initiated.

The results of pre-pruning operations are used to reorder the feasible match sets for searching. Those sets which are assigned (i.e. now have only a single feasible match) through pre-pruning are placed at the top of the search tree. The remaining feasible match sets are ordered according to strategy 3 for pre-ordering. Feasible match set F_9 is placed next because it is of order

	0:26	1:23	2:27	2:28	3:27	3:28	4:22	4:29	5:23	6:24	7:25	8:20	9:21	10:16
0:26	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1:23		0.12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.81	0.00	0.00
2:27			0.00	0.00	0.00	0.00	0.81	0.00	0.00	0.00	0.00	0.00	0.81	0.00
2:28				0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3:27					0.00	0.00	0.00	0.81	0.00	0.00	0.00	0.00	0.00	0.00
3:28						0.00	0.81	0.81	0.00	0.00	0.00	0.00	0.81	0.00
4:22							0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4:29								0.00	0.00	0.00	0.00	0.00	0.00	0.00
5:23									0.00	0.00	0.00	0.00	0.00	0.00
6:24										0.12	0.00	0.00	0.00	0.00
7:25											0.00	0.00	0.00	0.00
8:20												0.00	0.00	0.00
9:21													0.00	0.00
10:16														0.00

Table 2: Pre-pruning and pre-ordering example: Vertex and arc mapping costs. The label $a : b$ indicates a mapping between the projected model graph vertex a and image graph vertex b. The entry defined by the intersection of labels $a : b$ and $c : d$ is the cost of mapping the projected model graph arc (a, c) onto the image graph arc (b, d). Vertex mapping costs are shown along the diagonal entries enclosed in boxes.

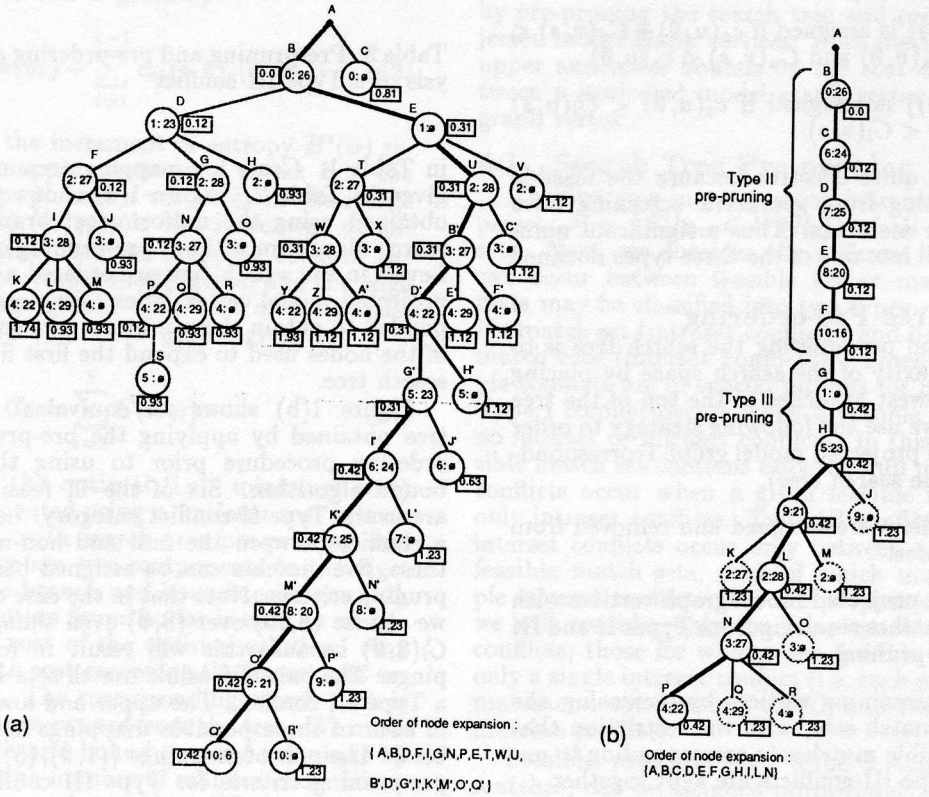

Figure 1: Branch and bound search tree: (a) without pre-pruning or pre-ordering, (b) with pre-pruning and pre-ordering. Search tree nodes are indicated by circles. Each node is labelled with the following information: an upper-case letter which indicates the order in which the nodes were created; a label of the form $a : b$ located inside the circle which indicates a mapping between the projected model graph vertex a and image graph vertex b; and a number inside a box which indicates the total cost of the partial mapping from the root to the node. The optimal mapping is indicated by a darkened path through the search tree. Nodes having having a grey outline indicate mappings which have been eliminated using the branch and bound pruning criteria.

Test #	n	m	Expanded nodes	
			With PO/P	No PO/P
1	13	4	41	781
2	10	4	30	171
3	10	3	10	42
4	11	3	11	49
5	11	3	30	98
6	14	3	168	413
7	12	3	31	93
8	12	3	18	108
9	14	2	16	416
10	11	2	20	132
11	12	4	21	99
12	14	3	49	624
13	15	4	64	1289
14	16	3	28	1519
15	11	5	79	3339

Notes:

1. n = Number of feasible match sets.
2. m = Maximum order of feasible match set (includes null matches).
3. PO/P = Pre-ordering/Pruning.

Table 4: Comparison of number of expanded nodes with and without pre-pruning and pre-ordering.

2. Feasible match sets F_2 and F_3, which together form an unresolved Type III conflict, are kept together.

Several important observations can be made by comparing the search trees in Figures 1(a) and (b). By placing higher-order feasible match sets lower in the search tree, where greater numbers of constraints are applied to each possible mapping in the form of edge costs, one is more likely to be able to distinguish the correct mapping. This is demonstrated by considering feasible match set F_2. With no pre-ordering F_2 is placed at the third level in the search tree. Here, the incremental costs of assigning the matches (2,27) and (2,28) are both zero. When pre-ordering is applied, F_2 is moved to the ninth level in the search tree. Now the edge cost between mappings (2,27) and (9,21) can be applied and the incremental costs of assigning (2,27) is higher than that for (2,28).

A direct comparison of the effect of using pre-pruning and pre-ordering on search efficiency is given in Table 4. Tests were conducted with data from 15 images obtained using our random graph-based incremental model synthesis system [8]. The model and image data used in test #1 are shown in Figure 2. In each test we compare the number of expanded nodes required to reach the identical optimal solution with and without the use of pre-pruning and pre-ordering. The use of pre-pruning and pre-ordering always results in fewer nodes being expanded with a maximum reduction of a factor of 54 being achieved in test #14. Given no pre-pruning or pre-ordering there is a general trend for the number of expanded nodes to increase sharply as either the number or the order of the feasible match sets increases. In contrast, when pre-

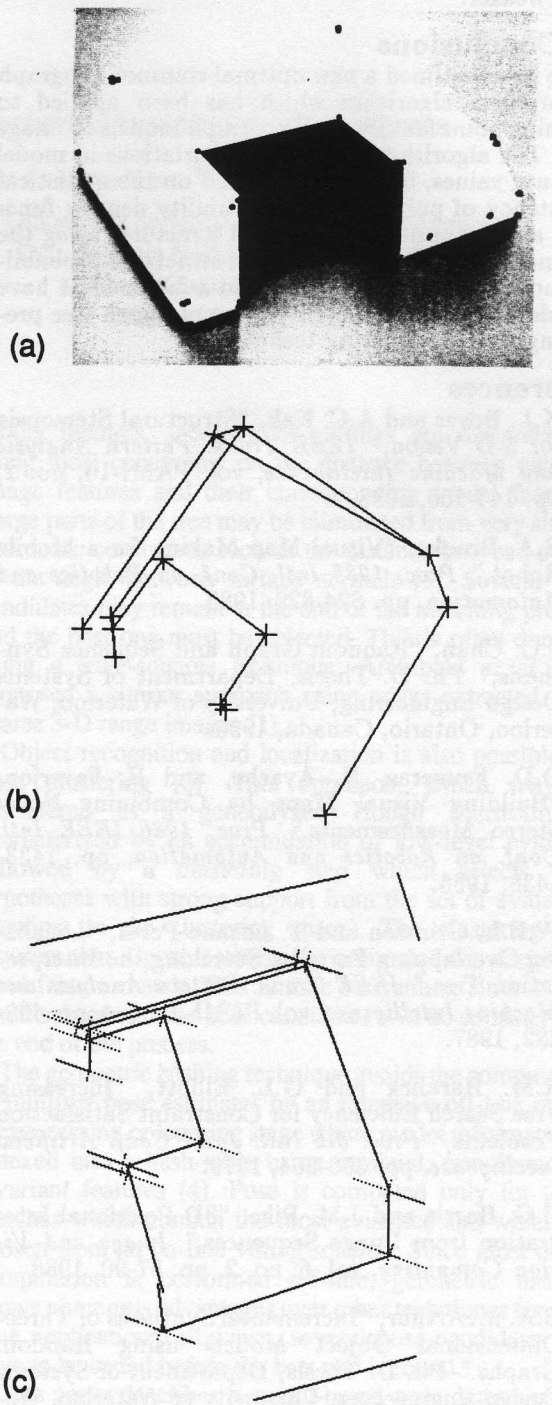

Figure 2: Model and image data for test #1. (a) Metal cover used for tests. (b) Projected model graph of metal cover. Crosses denote major and minor axes of the unit standard deviation (USD) contours for model point feature locations. (c) Image data overlaid by USD axes of projected model vertices.