Structural Indexing Extended to Fuzzy Graphs of 2D Parts

Guillaume-Alexandre Bilodeau and Robert Bergevin
Laboratoire de vision et systèmes numériques, Pavillon Adrien-Pouliot, Université Laval, Sainte-Foy (QC), Canada, G1K 7P4
bilodeau@gel.ulaval.ca, bergevin@gel.ulaval.ca

Abstract

Robust object recognition and image retrieval often lead to the construction of attributed graphs. When dealing with images of real scenes, these graphs must include fuzzy attributes to account for interpretation uncertainties and describe properly the observed object or scene. This paper first presents a 3D object model that requires the use of a graph with fuzzy attributes. This model is computed from a single 2D image of an object. Then, our method to match these graphs is introduced. This method is an extension of structural indexing. Being based on structural indexing, our method can match graphs with fuzzy attributes in large databases. A first experiment demonstrates the validity of our matching method, and a second one compares the performance of the combination of our model and our matching method with humans performing a similar task..

1. Introduction

In image querying applications, graphs are often used to model scenes or objects in the image. Since creating a model involves interpretation phases, there is always uncertainty. Hence realistically, graphs with fuzzy attributes (GFAs) must be used. Taking our application as an example [1], the fuzzy attributes may be all the possible volumetric primitives that can be inferred from a simplified part. Hence, one node (a simplified part) of a graph can have many fuzzy attributes (each possible volumetric primitive and a fuzzy ranking value). The edges of the graph can also have many fuzzy attributes (e.g. the different ways parts may be connected). How can such graphs be compared?

A significant research effort ([2],[3],[4]) has been devoted to inexact matching of attributed graphs. However, in general, these studies do not address the problem of matching inexact graphs, i.e. GFAs. An exception is the research of Christmas et al. [4]. Their method can match graphs where the attributes are fuzzy. However, because they are using probabilistic relaxation, their method needs a convergence criterion on the probability. A method with no convergence criterion would be more appropriate to image retrieval applications as the level of similarity attainable is not known a priori. In addition, their method is more adapted to find a similar graph, than to rank graphs based on a query graph. Chan and Cheung [5] have studied matching of a GFA and an attributed graph. Their method does not apply to matching pairs of GFAs. Perchant et al. [6] and Medasani et al. [7] have addressed this specific topic. Their approaches imply sequential matching of pairs of graphs using relaxation algorithms or genetic algorithms. Since our application of image querying requires fast matching and similarity ranking of many graphs, an approach using indexing is more appropriate as it is less computationally expensive [8]. To compare graphs quickly, structural indexing is a good choice as it compares graphs by matching their subgraphs independently. Structural indexing has been studied by Stein and Medioni [8], and by Nishida [9]. Structural indexing as defined and used in these researches cannot be applied to GFAs.

In this paper, a model for representing 2D parts by volumetric primitives is presented and structural indexing is extended to compare GFAs. The extension of structural indexing to GFAs is the main contribution of this paper. This extension is not trivial as it is stressed in Section 4.3, yet essential in computer vision applications.

The paper is organized as follows. Our interest in matching GFAs is explained by presenting our application (named PLASTIQUE) in Section 2. Section 3 describes the representation model. Section 4 details the fuzzy matching problem and describes the matching method. Section 5 gives results of experiments aimed at validating our structural indexing method and its applicability in matching graphs of volumetric primitives. Section 6 concludes the paper.

2. Overview of PLASTIQUE

The application under development, named PLASTIQUE (Parts, Links, and ASsociated Templates Image QUery Engine), aims at querying an image database of manufactured objects, which are interpretable as arrangements of simple volumetric

primitives. Images in the database are from real scenes with one main object in the foreground that must be detectable in the image (see [10]).

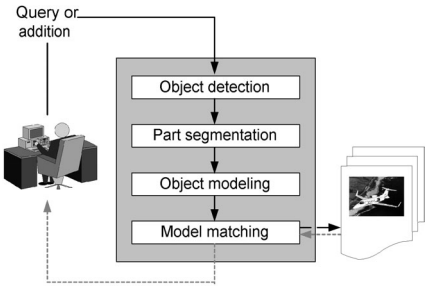


Figure 1. PLASTIQUE overview.

Figure 1 shows an overview of the image database query engine under development. The shaded region represents the four modules required to add an image or query the database. The database is composed of various 2D images of 3D objects, and their associated models. To query or add an image in the database, the user gives as input an example 2D image or a sketch of the 3D object. The image is first processed to obtain contours of linked local intensity edges that are segmented to produce a map of constant curvature primitives (CCPs) [11]. An initial grouping of the CCPs produces the outline of the object (Object detection) [10]. The CCP map is then processed further to obtain parts using the extracted outline (Part segmentation) [12]. These parts are labelled based on the possible volumetric primitives that may project onto them (Object modeling) [13]. Parts are interpreted as volumetric primitives (VPs) since the aspect of a projected 3D object may change significantly for different viewpoints. The object modeling module also computes the spatial relationships between parts. Finally, the constructed model is compared with the models in the database (Model matching). If similar models are in the database, the corresponding 2D images are shown to the user. If not, the newly built model and its corresponding image may be added in the database.

3. Construction of graphs from simplified parts

As discussed briefly in the previous section, to increase the overall matching performance of simplified parts found in an image, they are not matched directly. A direct matching of the simplified parts would not account for viewpoint variations. For this reason, volumetric primitive hypotheses are inferred from the parts. The goal of the 3D inference is to establish which volumetric primitives, when projected in an image, may look like the simplified parts found. For more information on obtaining simplified parts, see ([10],[13]). Simplified parts for a lamp are shown in Figure 2A.

Graphs are constructed from the spatial relationships of the simplified parts and from hypothesized volumetric primitives. Volumetric primitive hypothesizing is performed using a rule-based classifier. The rules are based on measures on the boundary and the shape of each simplified parts. These measures verify the convexity and the symmetry of the simplified part, the type and arrangements of the constant curvature primitives (circular arc or straight line segment) making the boundary. When a simplified part verifies a rule, one or more ranked volumetric primitive hypotheses are generated. Since simplified parts in an image each corresponds to a node of the graph under construction, each hypothesized volumetric primitive corresponds to a node attribute. Specifically, nodes attributes are the labels of each hypothesis associated with a fuzzy value (Figure 2B). The constructed graph will have nodes with one or more attributes that have values between 0 and 1 reflecting the ranking between the hypotheses. Edges of the graph correspond to the spatial proximity relationships between the simplified parts. Edges have attributes with values between 0 and 1 that describe the connections between the hypothesized volumetric primitives. The edges attributes consist of a label that describes a connection type and a fuzzy value (Figure 2B). Edges are undirected.

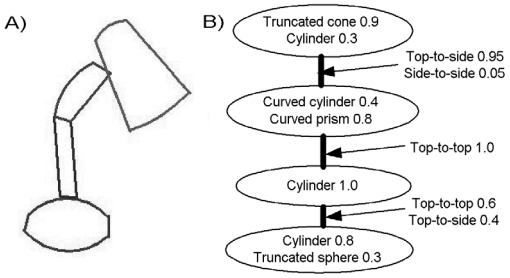


Figure 2. Example of simplified parts for a lamp and its associated graph.

4. Our approach to matching

Our approach to matching these graphs is to use structural indexing. Instead of attempting to compare graphs pairwise, their level of similarity is established by a voting mechanism and by comparing subgraph structures.

4.1. Details on the matching problem

The constructed graphs have nodes with 18 fuzzy attributes and edges with 4 fuzzy attributes. Each attribute has a fuzzy value between 0 and 1. The attributes for the nodes are labels identifying the 18 volumetric primitives (see [13]) that can be hypothesized for each simplified part. The attributes for the edges are labels that identify how the volumetric primitives are connected. Since objects have parts that interconnect in complex fashions, nodes can form cycles, and because of processing errors, some nodes may not be connected.

These graphs must be matched in the context of an image database query engine. Thus, it has to be done quickly, and matching cannot rely on perfect models of objects, because the same processing modules model both query and database images.

4.2. Structural indexing

Structural indexing (without fuzzy attributes) consists of comparing graphs by comparing their subgraphs (structural matching) using an index structure. The graphs to be compared are first decomposed into subgraphs of a chosen number of nodes. Then, to avoid a sequential search of subgraphs during matching, they are entered into an index. Typically, the index is a table where each row corresponds to a list of labels of graphs containing a given subgraph. There are as many rows as there are possibilities of sub-graphs. Hence, the choice of the dimension of the sub-graphs is important. Large subgraphs require a very large index because of combinatory explosion, whereas small subgraphs have little discriminatory power (i.e. more graphs will be considered as being similar).

To find the most similar graphs, each subgraph of the query graph is used to query the index. Each time a graph is indexed via its label, it receives a vote. The graphs that obtain the most votes are the best matches.

Figure 3 shows an example of the use of structural indexing to find the graph (between Obj1 and Obj2) that is more similar to graph Q. In the figure, shapes represent the attributes of the graphs. A graph receives votes if it has similar node attributes and similar duos of attributes for groups of two nodes.

For graphs with attributes that are not fuzzy, the use of structural indexing is simple. However, attributes with fuzzy values complicate matters significantly. The next section explains why.

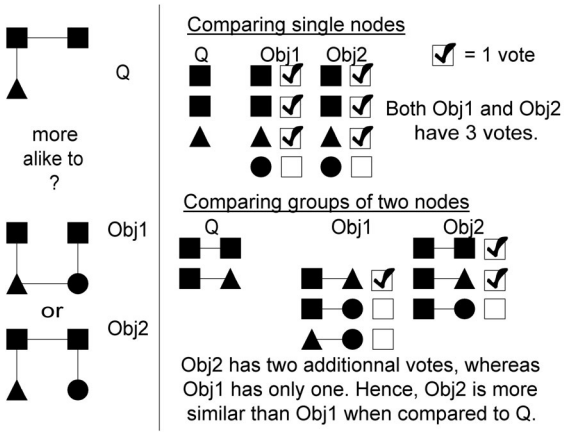


Figure 3. Example of structural indexing.

4.3. Difficulties with matching GFAs

To illustrate the difficulties with matching graphs with fuzzy attributes, consider again Figure 3. Let's suppose that for graph Q, the two dark squares have fuzzy values of 0.9 and 0.5 (i.e. these nodes cannot be represented by perfect dark squares). Let's also suppose that the two dark squares of Obj1 have fuzzy values of 0.6 and 0.2. If a square of Obj1 and Q are

matched, a vote of value one cannot be attributed, since these nodes are not perfect squares and they do not belong to the square type with the same strength. A weighted vote value proportional to these uncertainties and differences must be attributed. For example, the minimum fuzzy value of the two squares can be chosen. In any case, the weighted vote attributed to a match will depend on the fuzzy values of the node matched. Hence, the order of the matches will influence the total match score for a pair of graphs.

In our case, there is also more than one attribute per node. In theory, they should all be compared at once. However, in practice this would create a very large index to account for all possibilities (all possible combinations of 18 attributes, only to compare graphs node by node!).

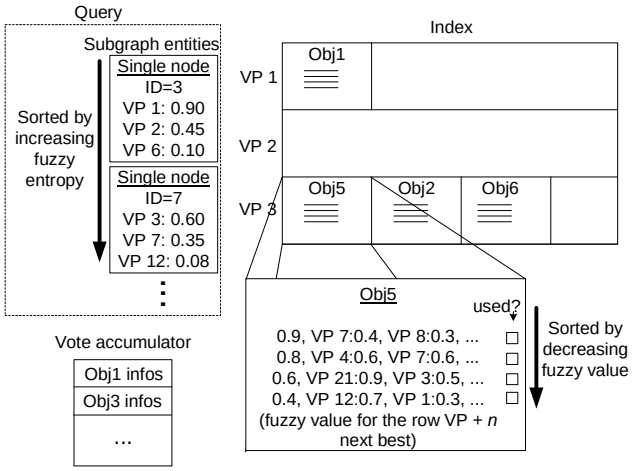


Figure 4. Data structure for matching GFAs.
1) For the query, there is a list of nodes sorted by increasing entropy. 2) There is an index for the graphs in the database. 3) There is a vote accumulator to register votes for each graph in the database.

4.4. Our method for matching GFAs

Our method handles the two difficulties mentioned above using the data structure shown in Figure 4. The query graph is decomposed into subgraph entities (SGEs). The SGEs are single nodes, groups of two nodes and groups of three nodes. Recall that nodes have 18 attributes and edges have 4 attributes. Therefore 94625 (18 + 18*4*18 + 18*4*18*4*18) index rows are required to cover all combinations of volumetric primitives and connection types. For simplicity, groups of two and three nodes are left out of the following explanations. Note though, that they are processed and used in exactly the same way as single nodes, as combined attributes are associated to subgraphs of two or three nodes. The combined attributes are obtained by concatenating the attributes of the nodes and the edges for each possible combination of the hypotheses associated to nodes and edges for a given subgraph. For example, the two upper nodes and the edge between them of Figure 2B, would give a two nodes subgraph with 8 different combined attributes (Truncated cone-Top-to-side-Curved cylinder 0.4, Truncated cone-Side-to-side-Curved cylinder 0.05, and so on).

After the decomposition, SGEs are processed in two different ways depending if they are to be used as queries or to be added to the database.

If they are used as queries, SGEs are sorted by increasing fuzzy entropy. In our case, the fuzzy entropy is calculated by:

$$Entropy = 1 - \frac{FVal_{\max}}{\sum FVal}, \tag{1}$$

where $FVal_{max}$ is the maximum fuzzy value for any attribute of the SGE and $\sum FVal$ is the sum of all the fuzzy values of the attributes of the SGE. Our measure is called entropy because it estimates how ambiguous the volumetric primitive hypotheses are. This definition of entropy is different from the usual definition used in fuzzy logic, because the fuzzy value of all attributes (volumetric primitives) in a node do not sum to 1. It is based on the fact that the way volumetric primitives are generated, a SGE is less ambiguous when its best hypothesis has a fuzzy value much larger then the other hypotheses. The sorting is done to ensure that the matching order does not influence results. Therefore, parts that have been hypothesized with less ambiguity are matched first.

If the SGEs are to be added to the database, they are processed as follows. Before being added in the database, the nodes can be truncated to the first *n* best hypotheses. The number of attributes of a node to include in the database depends on how

much memory and disk space is available for the index. In our current implementation we have chosen to truncate to the first forty best hypotheses. The chosen truncation does not significantly affect matching results because the fuzzy values of the truncated hypotheses are usually very low. For 195 graphs, the index size is of about twelve megabytes.

Coming back to Figure 4, each row of the index is a list of entries each associated with the graph of one object that has one or more copies of the SGE corresponding to the row. For single node SGEs, the rows correspond to the 18 volumetric primitives (VPs) that can be hypothesized. In each entry, there is the identity of the graph or object and a sorted list of its SGEs of the row type. For each SGE in an entry, there is a field to indicate if the SGE has been used during the current query.

To allow access to a SGE in the database not only from its best volumetric primitive hypotheses, entries for the *n* best are inserted in the index. Hence, nodes with different best volumetric hypotheses can be matched. An example appears in the next section.

The vote accumulator is a simple structure that records the number of votes each graph obtains. It also records how many SGEs have been used for matching, and their identifier to avoid multiple matches with the same SGEs.

4.5. Matching process

Let us now assume that three one-part objects (one node graphs) are to be matched and the SGE (composed uniquely of volumetric primitive hypotheses in this case, since it is a one node SGE) for each object are the ones shown in Figure 5. The numerical values reflect the ranks of the volumetric primitive hypotheses. Therefore these fuzzy values are called ranking values. The SGE identified by Q is hypothesized from the query object. The SGEs identified by OBJ1 and OBJ2 are hypothesized for objects (OBJ1 and OBJ2) added to the database. SGE of object OBJ1 is indexed at the cylinder row and at the cone row, whereas SGE of object OBJ2 is indexed at the cone row and at the pyramid row. To find the best match for query Q, a cylinder is first searched in the database. At the cylinder row, OBJ1 is found. The match value is calculated as followed. First, the minimum value is calculated for the ranking of the cylinders. The minimum of 0.65 and 0.8 is 0.65. Next, the other hypotheses are verified to complete the similarity comparison. SGEs Q and OBJ1 both have another hypothesis, which is a cone. Taking the minimum, this adds 0.4 to the matching score. The matching for the two SGEs is then 1.05. However, this score has to be normalized so that a perfect SGE match gives 1. The normalization factor is the sum of the rank values for all the volumetric hypotheses of the current query SGE. In the present case, it is 0.65+0.55=1.2. Therefore, the match score for Q and OBJ1 is 1.05/1.2=0.88.

Formally, the matching score of two SGEs is:

$$MS = \frac{\sum_{i \approx j} \min(RV_Q(i), RV_D(j))}{\sum_{i \in SGE_Q} RV_Q(i)} \tag{2}$$

where SGE_Q is the query SGE, and $RV_Q(i)$ and $RV_D(j)$ are the ranking values for the ith or jth volumetric primitives hypotheses of the query and of the database SGE respectively. $i \approx j$ means that the sum is computed only for matching hypotheses in the two SGEs. The total score for two objects or graphs (for all the SGEs of both graphs) is given by:

$$TS = a\sum_i \left(\frac{SGEQ_i}{SGEDB_i} \right) + \sum_i \left(\frac{b_i \sum MS_i}{\min(SGEQ_i, SGEDB_i)} \right) \tag{3}$$

where $SGEQ_i$ and $SGEDB_i$ are respectively the number of SGEs with i nodes the query and the graph in the database has, MS_i is a matching score for a SGE of i nodes, a and b_i are weighting coefficients. The weighting coefficients are adjusted dynamically. The smallest is the term it multiplies, the highest is the coefficient value. This focuses the comparison on the differences between two graphs. The sum of the weighting coefficients is 1, and in the current implementation of our matching method, the coefficients are selected, from sets of fixed values, based on the value of each term of *TS*. The value for *TS* varies between 0 and 1, where 1 is a perfect match between two graphs.

To maximize matching scores, matching is performed in consecutive passes. The first pass indexes the best volumetric primitive hypothesis in each SGE of the query. The second pass indexes the second best volumetric primitive and so on. This is done to consider cases where an object or part of an object is seen from another viewpoint, and where noise or errors slightly deform simplified parts. Therefore, a partial SGE match must be considered. Indexing of the SGEs of the query must be done in a constant and ordered fashion to ensure consistency in the value of *TS*. A change in the order of indexing change the value of *TS* for two matched graphs. This why a fixed consecutive passes procedure if used to index the database.

Let's now consider OBJ2. After the first pass, OBJ2 does not match with Q. In fact, it does match partially because both Q and OBJ2 have cone hypotheses. The cone is then used to index the database at the second pass and OBJ2 is found. OBJ1 is

ignored because this SGE has already been matched. The match score for Q and OBJ2 is 0.55/1.2=0.46. For this example, *TS* is equal the match score calculated, because the graphs have only one SGE.

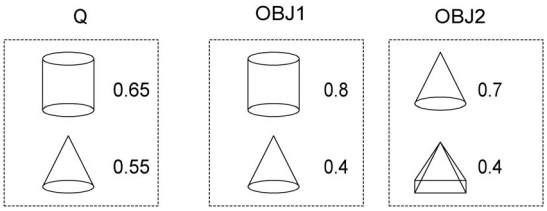


Figure 5. SGEs for illustrating the matching process.

5. Experiments

In this section, the matching performance of our method is analyzed by two experiments. The first experiment investigates if identical graphs have the highest total matching score when queried in the index. The second experiment studies the ability of the combination of the volumetric primitive model and the matching method to group simplified parts from images of six objects. It is compared to human abilities.

5.1. Validation of the matching method

This experiment aims at verifying if the matching method matches graphs as it was designed to do. To verify this aspect, 195 graphs built from simplified parts of objects in images have been used. Each image contains only one object in the foreground. The graphs have been entered in an index and each graph in the index has been queried. If our matching method performs adequately, the graph in the index that will get the highest total matching score will be the same as the query. The results of this experiment are shown in Figure 6. For 192 graphs out of 195, the graph identical to the query obtained the highest total matching score. However, it has not been the case for three graphs which have ranked second. It is because in the database, SGEs are sorted based on the index row SGE hypotheses, whereas for the graph use to index, the SGEs are sorted by entropy. Hence, matched SGEs for the graph used to index and its copy in the database are not necessarily the same because they are sorted differently. Therefore, when compared, the matching score cannot always be equal to the perfect mark. For the case of these three graphs, another very similar graph has benefited from this difference in the sorting methods and sneaked in at first rank. Since our method is an inexact matching method, this was to be expected. Note that the query SGEs and the SGEs of the graphs in the database cannot be sorted in the same way, because entropy cannot be used in the index as it would not maximize globally matching scores, and because the sorting used in the index cannot be used for the query graph as it is linked to the index structure.

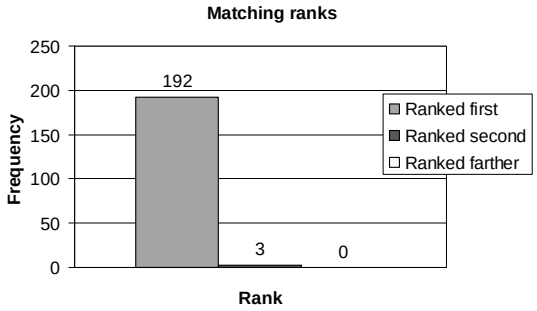


Figure 6. Histogram of matching ranks for indexed graphs queries.

5.2. Grouping performance compared to human cognitive abilities

This experiment has been designed to assess the abilities of our volumetric primitive model and matching method to group simplified parts obtained for six objects viewed in 195 images (about the same number of images for each object). Figure 7 gives two examples for each object, of images and associated simplified parts used for this experiments. This experiment will

allow us to verify how the 3D model and the matching method allow PLASTIQUE to abstract viewpoint and imperfections. To do so, PLASTIQUE is compared with six humans that have performed a similar clustering task. Among the six humans, four were researchers in the field.

The experiment with human subjects has been conducted as follows. Printouts of the simplified parts of the 195 images have been given to the human subject. They have been asked to classify the printouts in groups they thought corresponded to the same object. It has been mentioned to the human subjects, that the objects could be seen from different viewpoints and deformed. The number of groups to form was not specified.

The same clustering task has been performed with PLASTIQUE. Images of simplified parts were grouped together if the total matching score, with an image of simplified parts given as a query, was within a threshold. Then, the grouped images of simplified parts were removed from the index, and another query (an image of simplified parts still in the index) was used. This process has been done until the index was empty. This task has been performed with two thresholds. With a 60% and a 70% threshold. That is, the matching score had to be higher than 0.6 and 0.7 (a perfect matching score being equal to 1). Threshold values larger than 70% give very small groups. Hence, these values are not of interest. Furthermore, threshold values smaller than 60% give poor grouping performance as a lot of graphs are considered alike.

The graph of Figure 8 shows the precision obtained for each object. The precisions have been computed as follows. The number of images or printouts in each group where an instance of a particular object was found has been summed. The number of images of the particular object in the database was divided by this value. This precision value reflects how many images or printouts of simplified parts had to be considered so the human subjects or PLASTIQUE could find all the images of the particular object. The smaller the value, the larger the number of images considered before finding all the images of an object.

Except for the watering can, the human subjects outperform PLASTIQUE on average. Considering that human subjects have a priori knowledge about the object grouped in the experiment, PLASTIQUE suffers from a major handicap. PLASTIQUE does not have a priori models of objects. It simply compares the simplified parts by hypothesizing and comparing 3D shapes and verifying their relationships. The human subjects on the other hand can guess the identity of objects by the area or the shape the simplified parts are forming altogether. In any case, PLASTIQUE using a 70% threshold approaches the lowest precision level achieved by human subjects. However, this part of our system can be improved to reduce the gap between PLASTIQUE and human subjects. For example, the use of global shape criteria on the area formed by all the simplified parts may significantly increase the precision of PLASTIQUE. Note that human subjects have had a lot of difficulties recognizing objects such as the stool, the coffee cup and the watering can. This is because the simplified parts extracted do not always adequately represent the object in the images. Discussion on this issue will be the subject of another paper.

It can be noted from Figure 8 that some complex objects are grouped with more precision then simple objects, even by humans. It is the case for airplanes and coffee cups. The cause of this is not related to the subject of this paper. This is explained by the simplified parts obtained for these objects. The simplified parts obtained for the airplanes capture more their shapes than those for the coffee cups, because our part segmentation algorithms are not aware of holes in objects [12] (see simplified parts of stools and coffee cups in Figure 7). Therefore, the handle is not well segmented from the body of the coffee cup.

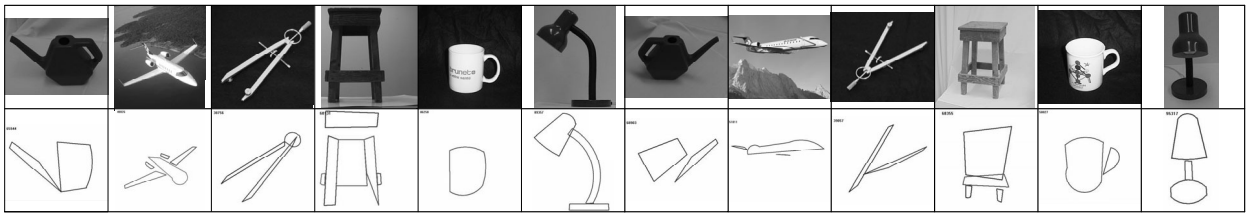


Figure 7. Example of images used for the grouping performance

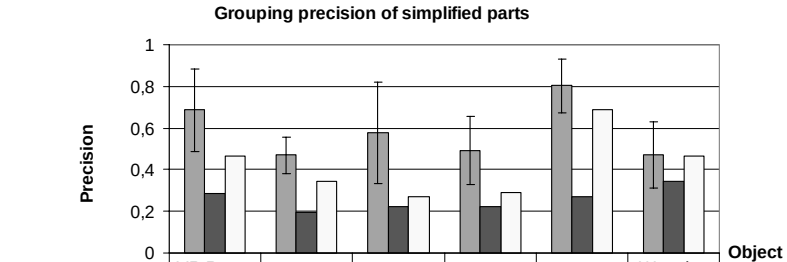
Conclusion

This paper has presented a novel method for inexact matching of graphs with fuzzy attributes (GFAs) and a 3D model of representation for 2D parts. We first stated that matching GFAs is essential to many computer vision applications since interpretation phases are always uncertain as it is the case for our volumetric primitive inference method. These uncertainties can be accounted for by using GFAs.

Our matching method adapts structural indexing to GFAs. The use of an entropy measure and adapted data structures allows our method to handle the fuzzy nature of the graphs. Our method ranks graphs by similarity based on a query graph. Matching is done quickly as the models are not compared sequentially. For 195 graphs, matching and displaying the results take less then one second on a Athlon 1.2Ghz. The complexity of our matching method is $O(nm)$, where n is the number of SGEs of the query and m is the number of graphs in the database. This is the worst-case complexity where each graph in the

database has at least an SGE at each row of the index. Furthermore, the size of the index can be adjusted to fit one need of matching accuracy and computer memory utilization. For 195 objects and SGEs truncated to 40 hypotheses, the memory requirement is about twelve megabytes.

The results obtained thus far show that our method can match GFAs adequately. When an image in the index is queried, it ranks first in general for the matching score, except in a few cases where the sorting methods used have lowered the matching score below the score of another image. Grouping simplified parts using our method gives results that are still inferior to human subjects. However, this experiment allowed us to gain valuable knowledge and discover new ways to improve our method. These improvements will be the subject of future work.



	VR Room Lamp	Coffee cup	Airplane	Stool	Compass	Watering can
Human average	0,6854469	0,4683908	0,5770591	0,4943092	0,8024038	0,4706508
PLASTIQUE 60%	0,2866242	0,1979167	0,2222222	0,2231405	0,2721088	0,3434343
PLASTIQUE 70%	0,4639175	0,3454545	0,2678571	0,2934783	0,6896552	0,4657534

Figure 8. Grouping precision of simplified parts.

References

[1] Bilodeau, G.A., Bergevin, R., PLASTIQUE: Image Retrieval Based on Cognitive Theories, in Vision Interface 2003, (2003). Halifax, Canada.

[2] Messmer, B.T., Bunke H., A New Algorithm for Error-Tolerant Subgraph Isomorphism Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, (1998) 20(5), pp. 493-504.

[3] Tirthapura, S., et al, Indexing Based on Edit-Distance Matching of Shape Graphs, in Proc. SPIE Multimedia Storage and Archiving Systems II, (1998), pp. 25-36.

[4] Christmas, W.J., Kittler J., Petrou M., Structural Matching in Computer Vision Using Probabilistic Relaxation. IEEE Transactions on Pattern Analysis and Machine Intelligence, (1995) 17(8), pp. 749-764.

[5] Chan K.P., and Y.S. Cheung. Correction to "Fuzzy-Attribute Graph with application to Chinese Character Recognition", IEEE Trans. Syst., Mans, Cybern., 1992, 22(2), pp. 402-410.

[6] Perchant, A., et al. Model-based Scene Recognition using graph Fuzzy Homomorphism Solved by Genetic Algorithms. in Graph Based Representation, (1999), Austria.

[7] Medasani, S., Krishnapuram R., Choi Y.S., Graph Matching by Relaxation of Fuzzy Assignments, IEEE Trans. on Fuzzy Systems, (2001) 9(1), pp. 173-182.

[8] Stein, F., Medioni G., Structural Indexing: Efficient 2-D Object Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, (1992) 14(12): pp. 1198-1204.

[9] Nishida, H. Shape Retrieval from Image Database through Structural Feature Indexing. in Vision Interface'99. (1999), Trois-Rivières, Canada, pp. 328-335.

[10] Bilodeau, G.A., Bergevin R., Generic Modeling of 3D Objects from Single 2D Images, in International Conference on Pattern Recognition, (2000), Barcelona, Spain, pp. 770-773

[11] Bergevin, R., Mokhtari M., Multiscale Contour Segmentation and Approximation: An Algorithm Based on the Geometry of Regular Inscribed Polygons, Computer Vision and Image Understanding, (1998) (71(1)), pp. 55-75.

[12] Bilodeau, G.A., Bergevin R., Part segmentation of objects in real images, Pattern Recognition, vol. 35, November 2002, pp. 387-400.

[13] Bilodeau, G.A., Bergevin R., Modeling of 2D Parts Applied to Database Query, in Vision interface 2001, (2001). Ottawa, Canada, pp. 228-235.