PROFI

Project number:	FP6-511572
Project acronym:	PROFI
Title:	Perceptually-relevant Retrieval Of Figurative Images

Deliverable No: D5.2: Implementation of partial region matching

Short description:
We implemented the matching algorithm based on image primitives described in Deliverable 5.1 for finding partial similarity between two shapes that represent figurative images. Shapes are modeled by sets of polygonal regions. A figurative image is first decomposed into a small set of salient components and the components and relations between them are weighted. For given two images the matching algorithm finds the best correspondence between the components. For the task of complete-partial matching the unmatched or badly matched components of the image to be matched complexity are penalized in the total similarity value, and those of the second image are not. For the task of partial-partial matching no penalty is introduced for unmatched parts.

Due month:	M33
Delivery month:	M33
Lead partner:	Freie Universität Berlin
Partners contributed:	Freie Universität Berlin
Classification:	RE

Project funded by the European Community under the "Information Society Technologies" Programme

1 Introduction

In this workpackage we develop and implement algorithms for *matching* parts of two planar shapes that represent figurative images. We assume that shapes are modeled by sets of polygonal regions or curves. We address the problem of *complete-partial matching* (CPM), i.e., matching shape B completely as good as possible to some part of shape A, and the problem of *partial-partial matching* (PPM), i.e., matching some part of B as good as possible to some part of A. Clearly, the partial-partial matching problem is not uniquely specified since there is a tradeoff between the quality of the match and the size of the matched parts.

In this report we describe the usage of our implementation to perform the task of partial matching. Since complete and partial matching algorithms share a large part of implementation details, such as shape representation, input/output operations and some algorithmic structures, we developed and deliver both implementations as a single library. The detailed documentation of classes is attached to the Deliverable 4.2.

2 Algorithms as described in Deliverable D5.1

2.1 Probabilistic matching

For complete-complete matching of sets of regions we developed a probabilistic algorithm that finds a transformation minimizing the area of the symmetric difference of the two given sets if the allowed transformations are translations or rigid motions. The algorithm is described and analyzed in the deliverable report 4.1, here we shortly summarize the idea of the algorithm: We take a random sample of points of suitable size in each shape and record a "vote" for the transformation that maps the sample of shape A to the sample of shape B. The size of the sample depends on the class of the allowed transformations. If this experiment is repeated many times, we get a certain distribution of votes in transformation space. We showed that the density function of this distribution is proportional to the area of overlap of the two shapes induced by the corresponding transformation. The transformation with the largest number of votes in its neighborhood then approximately maximizes the area of overlap. For translations and rigid motions maximizing the area of overlap is equivalent to minimizing the area of the symmetric difference.

The area of the symmetric difference as a measure of dissimilarity of two shapes A and B indicates how much of the shape A is unmatched in B plus how much of B is unmatched in A. A natural extension of this idea to a complete-partial matching, if we want to match the complete shape A to some part of shape B, is to measure how much of the shape A is unmatched in B and to ignore the unmatched parts of B. That means for the complete-partial match we want to minimize the area of $A \setminus B$.

If the transformation does not allow scaling, which is the case with translations and rigid motions, minimizing the area of the difference $A \setminus B$ means to maximize the area of overlap of A and B. Therefore, the algorithm we described in deliverable 4.1 realizes complete-partial matching in a sense that it finds a transformation that approximately maximizes the area of overlap of two given sets of regions and, thus, minimizes the area of $A \setminus B$. For the cases of similarity maps and more general affine transformations analysis of the algorithm turns out to be quite complex and is part of our ongoing work. For the shapes that are similar in the sense that they contain parts with large possible areas of overlap, see for an example Figure 1, our algorithm finds the best transformation for the partial matching.

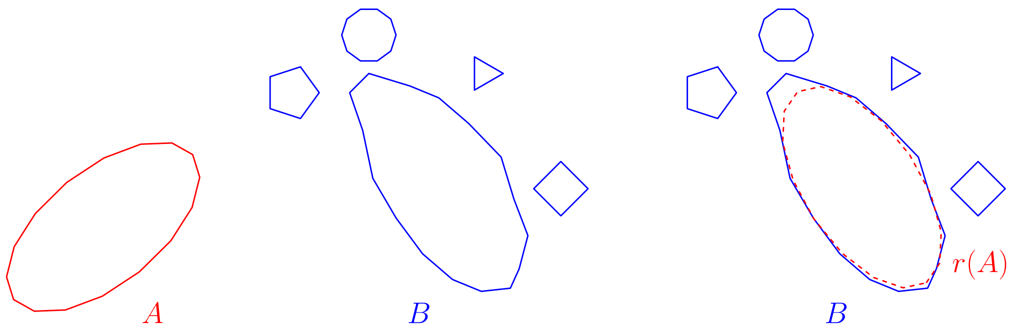

Figure 1: Partial matching of two shapes with large similar parts.

On the other hand, if the shape A is a small figure and the shape B has large parts where A can be completely fit into, see Figure 2 top row for an example, then any position of A within the large part of B is equally good. Furthermore, even if B contains a region congruent to shape A (Figure 2, bottom row) the position of A on top of that region is as good as any position of A within the large part of B. The distance from A to B as measured by the area of $A \setminus B$ is zero in all described positions. Nevertheless, we would probably not consider A

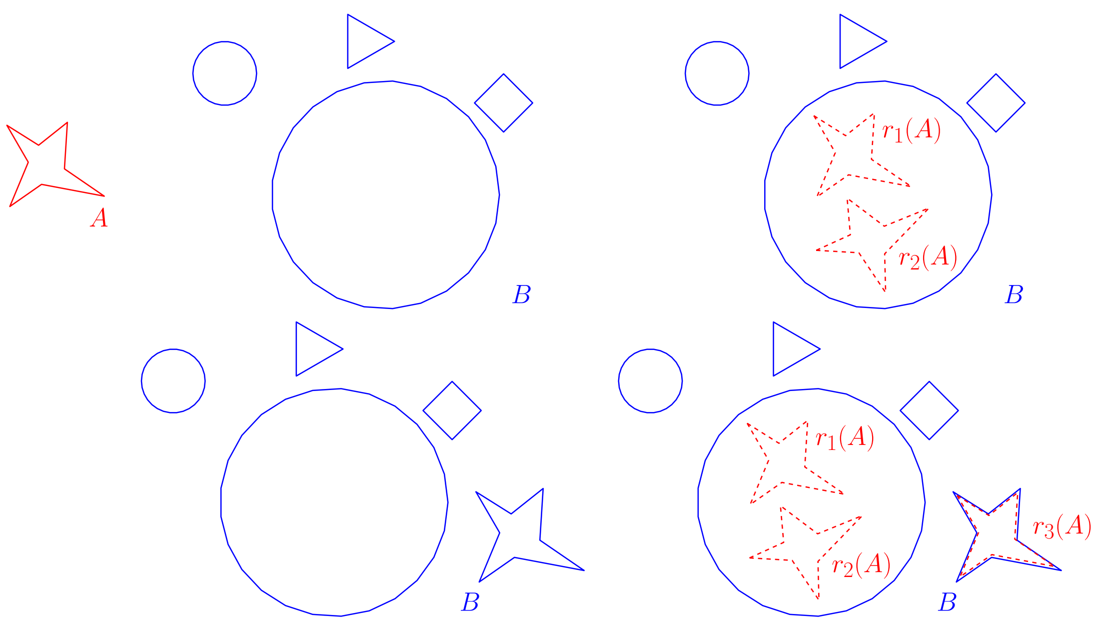

Figure 2: Partial matching of a small shape A to a shape B containing large parts.

as similar to the large part of B, but rather to the smaller nearly congruent part as match $r_3(A)$ indicates in the bottom row of Figure 2. This, with respect to the area of regions, natural notion of partial distance seems to be less suitable for perceptually relevant partial matching of shapes. It still remains interesting for other applications, for example, cutting stock problems.

2.2 Matching based on Image Primitives

Definition of partial similarity

For the partial-partial-matching of figurative images based on sets of (polygonal) curves we developed a criterion for rating the similarity based on the similarity of the matched parts and on their size relative to the whole images. An analogous technique for the ppm problem based on regions is not applicable, because most of the visual information of a shape is coded in the boundary and not in the interior. When comparing a star and a circle, they do not appear to be similar just because parts of the star's interior do match well with parts of the circle's interior. However if the images are composed of several distinctive shapes, the similarity of these shapes can be rated and — together with the salience of these shapes — be a basis for the evaluation of partial similarity.

Matching Algorithm

In deliverable 4.1 we described an algorithm for the perceptually relevant comparison of figurative images (as given by sets of shapes). The idea in our approach is that an image is divided into a set P of (not necessarily spatially independent) parts — preferably simple and salient geometric figures. These parts are classified, weighted, and the set R of "relations" between the parts are identified. The relation are weighted as well based on the weights of the figures they connect. Comparing two images is accomplished by searching for subsets of the parts and their relations that match well.
The comparison of the parts is done independently, leaving aside their relative sizes and positions. It can be done using a similarity measure that works well for shapes whose parts lie close together whereas the resulting measure can handle arbitrary composed shapes.
For the comparison of two images I^1 and I^2 the relevance w_P of the figures and the relevance w_R of the relations is preset such that $w_P + w_R = 1$ — for images consisting only of one type of figures, e.g., only squares, the relations between these figures are of greater importance than for images consisting of totally different figures. The figures $p \in P$ and relations $r \in R$ get weights $w(p)$ and $w(r)$ such that for each image all weights sum up to 1, namely: $\sum_{p \in P} w(p) = w_P$ and $\sum_{r \in R} w(r) = w_R$.

For every pair $(p^1_i, p^2_k) \in P^1 \times P^2$ of figures and every pair $(r^1_{i,j}, r^2_{k,l}) \in R^1 \times R^2$ of relations, where $r_{i,j}$ denotes a relation between figures p_i and p_j, a value of similarity $s \in [0, 1]$ is computed, using simple measures of similarity.

Let $\mathcal{M}$ be the set of all one-to-one matchings between figures of image I^1 and image I^2. The value of similarity of the two images is then defined as the weighted sum of the similarities of the matched figures, plus the weighted sum of the similarities of the (implicitly) matched relations:

$$s(I^1, I^2) = \max_{M \in \mathcal{M}} \left\{ \sum_{(p^1,p^2)\in M} s(p^1, p^2) \cdot \frac{w(p^1) + w(p^2)}{2} + \sum_{\substack{(p_i^1,p_k^2)\in M \\ (p_j^1,p_l^2)\in M}} s(r^1_{i,j}, r^2_{k,l}) \cdot \frac{w(r^1_{i,j}) + w(r^2_{k,l})}{2} \right\}$$

Finding images I^2 that contain a part that is similar to the query image I^1 is referred to as *partial-complete-matching*. This can easily be accomplished using our approach, by ignoring the weights of I^2. If *partial* refers to the matching between single figures, the similarity $s(p^1, p^2)$ of two figures p^1, p^2 has to be replaced by a partial-complete similarity measure $s_{\rightarrow}(p^1, p^2)$.

3 Results

As for the complete-complete matching, the similarity evaluation is based on a preprocessed description of the shapes in the images. This preprocessing may directly use the shapes as given by the segmentation (as is desired with the high level York-segmentation) or it may rearrange and group the shapes given by the segmentation (as is necessary with the low level FU-segmentation).

3.1 Implementation

The Algorithms for the comparison based on image primitives as described in deliverables D5.1 and D4.1 have been implemented in C++ as well as in Java. Input and output is via filesystem. In addition to the similarity based on image primitives the curve based similarity is also computed (it yields better results than region based similarity (see D4.2) and its theoretical basis is more solid (see D5.1).

Java The Java archive *Comparator.jar* contains a stand alone application that allows using the algorithms via a command-line interface. It offers the following options:

segment <inputFile> <outputFile> For performing a low level segmentation of an image (png/gif) and storing the closed polygonal chains of the shapes.

prepare-fu <inputFile> <outputFile> For reading in segmented shapes from an xml-file and generating the image description with rearrangement and grouping of polylines.

prepare-york <inputFile> <outputFile> For reading in segmented shapes from an xml-file and generating the image description without rearrangement and grouping of polylines.

segmentandprepare <inputFile> <outputFile> For performing a low level segmentation of an image (png/gif) and generating the image description with rearrangement and

grouping of polylines. The result of a call
`segmentandprepare <inputFile> <outputFile>` equals the two consecutive calls
`segment <inputFile> <intermediateFile>` and
`prepare-fu <intermediateFile> <outputFile>`.

compare <file1> <file2> <outputFile> [<ccm>] [<cpm>] [<ppm>] For comparing two image descriptions. The optional parameters 'ccm', 'cpm', 'ppm' specify whether a complete-complete-matching (ccm), whether a complete-partial-matching (cpm), and whether a partial-partial-matching is to be computed. They may be `true`, `false` or `1`, `0` respectively. The default is `true true false`.

C++ The C++ library *libshapemlib.so* contains the algorithms for comparing shapes and images. Via the class *FigurativeMatcher* it provides direct access to the comparison methods to other applications.

static void setSeed(int seed) To have reproducible behavior of the probabilistic parts of the algorithms, the pseudo-randon-generator may be seeded.

static void compare(char* inputFile1, char* inputFile2, char* outputFile, ...) To compare two images based on their descriptions.

static bool validate(char* inputFile, char* dtd) To check the validity of an image description file. The input file is checked regarding the validity of the xml format according to the given dtd, and regarding the consistency of the data. This part is separated from the comparison of images to have the possibility of having collections of valid files that can be compared without extra costs of re-checking.

For both implementations the comparison routine offers two ways of partial-matching to satisfy different needs.

cpm (complete-partial-matching) The similarity of a complete image to the best fitting parts of a second image is computed and returned as a single value of similarity ignoring the ratio of matched parts in the second image. In addition information about the parts that contribute to that resemblance is provided. For the matching based on image primitives, the corresponding figures together with their similarity and with their weights are listed. For the probabilistic matching, the polygon parts that are matched and the transformation under which they are matched are listed.

ppm (partial-partial-matching) The partial-partial matching on the other hand does not result in a single value. The similarities of all single polygons in the first image to all single polygons in the second image are computed using the probabilistic matching. This option is offered because of a request of our partners to have the possibility to use it as basis for higher application layers. A so called trademark logic layer may use the information to either deduce overall similarity or to search for special occurrences of shape patterns etc. For all the polygons the similarity as well as the transformation under which they are matched is provided.

4 Deviations from Plan

There have been no deviations from plan.