

# Elastic Image Matching is NP-Complete

Daniel Keysers<sup>a,\*</sup>

<sup>a</sup>*Lehrstuhl für Informatik VI, Computer Science Department  
RWTH Aachen – University of Technology, D-52056 Aachen, Germany*

Walter Unger<sup>b,1</sup>

<sup>b</sup>*Lehrstuhl für Informatik I, Computer Science Department  
RWTH Aachen – University of Technology, D-52056 Aachen, Germany*

---

## Abstract

One fundamental problem in image recognition is to establish the resemblance of two images. This can be done by searching the best pixel to pixel mapping taking into account monotonicity and continuity constraints. We show that this problem is NP-complete by reduction from 3-SAT, thus giving evidence that the known exponential time algorithms are justified, but approximation algorithms or simplifications are necessary.

*Key words:* elastic image matching, 2-dimensional warping, NP-completeness

---

## 1 Introduction

In image recognition, a common problem is to match two given images, e.g. when comparing an observed image to given references. In that process, elastic image matching, 2D-warping (Uchida and Sakoe, 1998) or similar types of invariant methods (Keysers et al., 2000) can be used. For this purpose, we can define cost functions depending on the distortion introduced in the matching and search for the best matching with respect to a given cost function. In this

---

\* Corresponding author

*Email addresses:* [keysers@informatik.rwth-aachen.de](mailto:keysers@informatik.rwth-aachen.de) (Daniel Keysers),  
[unger@informatik.rwth-aachen.de](mailto:unger@informatik.rwth-aachen.de) (Walter Unger).

*URLs:* [www-i6.informatik.rwth-aachen.de/~keysers](http://www-i6.informatik.rwth-aachen.de/~keysers) (Daniel Keysers),  
[www-i1.informatik.rwth-aachen.de/quax](http://www-i1.informatik.rwth-aachen.de/quax) (Walter Unger).

<sup>1</sup> This work was partially supported by DFG-grant HR 14/5-1.

paper, we show that it is an algorithmically hard problem to decide whether a matching between two images exists with costs below a given threshold. We show that the Problem IMAGE MATCHING is  $NP$ -complete by means of a reduction from 3-SAT, which is a common method of demonstrating a problem to be intrinsically hard (Garey and Johnson, 1979). This result shows the inherent computational difficulties in this type of image comparison, while interestingly the same problem is solvable for one-dimensional sequences in polynomial time, e.g. the dynamic time warping problem in speech recognition (see e.g. Ney et al., 1992). This has the following implications: researchers who are interested in an exact solution to this problem cannot hope to find a polynomial time algorithm, unless  $P = NP$ . Furthermore, one can conclude that exponential time algorithms as presented and extended by Uchida and Sakoe (1998, 1999a,b, 2000a,b) may be justified for some image matching applications. On the other hand this shows that those interested in faster algorithms – e.g. for pattern recognition purposes – are right in searching for sub-optimal solutions. One method to do this is the restriction to local optimizations or linear approximations of global transformations as presented in (Keysers et al., 2000). Another possibility is to use heuristic approaches like simulated annealing or genetic algorithms to find an approximate solution. Furthermore, methods like beam search are promising candidates, as these are used successfully in speech recognition, although linguistic decoding is also an  $NP$ -complete problem (Casacuberta and de la Higuera, 1999).

## 2 Image matching

Among the varieties of matching algorithms, we choose the one presented by Uchida and Sakoe (1998) as a starting point to formalize the Problem IMAGE MATCHING. Let the images be given as (without loss of generality) square grids of size  $M \times M$  with grayvalues (resp. node labels) from a finite alphabet  $\mathcal{G} = \{1, \dots, G\}$ . To define the problem, two distance functions are needed, one acting on grayvalues  $d_g : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{N}$ , measuring the match in grayvalues, and one acting on displacement differences  $d_d : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{N}$ , measuring the distortion introduced by the matching. For these distance functions we assume that they are monotonous functions (computable in polynomial time) of the commonly used squared Euclidean distance, i.e.  $d_g(g_1, g_2) = f_1(\|g_1 - g_2\|^2)$  and  $d_d(z) = f_2(\|z\|^2)$  with  $f_1, f_2$  monotonously increasing. Now we call the following optimization problem the IMAGE MATCHING problem (let  $\mathcal{M} = \{1, \dots, M\}$ ).

**Instance:** The pair  $(A, B)$  of two images  $A$  and  $B$  of size  $M \times M$ .

**Solution:** A mapping function  $f : \mathcal{M} \times \mathcal{M} \rightarrow \mathcal{M} \times \mathcal{M}$ .

**Measure:**

$$\begin{aligned}
c(A, B, f) = & \sum_{(i,j) \in \mathcal{M} \times \mathcal{M}} d_g(A_{ij}, B_{f(i,j)}) \\
& + \sum_{(i,j) \in \{1, \dots, M-1\} \times \mathcal{M}} d_d\left(f\left((i, j) + (1, 0)\right) - \left(f(i, j) + (1, 0)\right)\right) \\
& + \sum_{(i,j) \in \mathcal{M} \times \{1, \dots, M-1\}} d_d\left(f\left((i, j) + (0, 1)\right) - \left(f(i, j) + (0, 1)\right)\right)
\end{aligned}$$

**Goal:**  $\min_f c(A, B, f)$

In other words, the problem is to find the mapping from  $A$  onto  $B$  that minimizes the distance between the mapped grayvalues together with a measure for the distortion introduced by the mapping. Here, the distortion is measured by the deviation from the identity mapping in the two dimensions. The identity mapping fulfills  $f(i, j) = (i, j)$  and therefore  $f\left((i, j) + (x, y)\right) = f(i, j) + (x, y)$ .

The corresponding *decision problem* is fixed by the following

**Question:** Given an instance of IMAGE MATCHING and a cost  $c'$ , does there exist a mapping  $f$  such that  $c(A, B, f) \leq c'$ ?

In the definition of the problem some care must be taken concerning the distance functions. For example, if either one of the distance functions is a constant function, the problem is clearly in  $P$  (for  $d_g$  constant, the minimum is given by the identity mapping and for  $d_d$  constant, the minimum can be determined by sorting all possible matchings for each pixel by grayvalue cost and mapping to one of the pixels with minimum cost). But these special cases are not those we are concerned with in image matching in general.

We choose the matching problem of Uchida and Sakoe (1998) to complete the definition of the problem. Here, the mapping functions are restricted by continuity and monotonicity constraints: the deviations from the identity mapping may locally be at most one pixel (i.e. limited to the eight-neighborhood with squared Euclidean distance less than or equal to 2). This can be formalized in this approach by choosing the functions  $f_1, f_2$  as e.g.

$$f_1 = id, \quad f_2(x) = \text{step}(x) := \begin{cases} 0 & , \quad x \leq 2 \\ (10 \cdot G)^{M \cdot M} & , \quad x > 2 \end{cases}$$

### 3 Reduction from 3-SAT

3-SAT is a very well-known *NP*-complete problem (Garey and Johnson, 1979, p.259), where 3-SAT is defined as follows:

**Instance:** Collection of clauses  $C = \{c_1, \dots, c_K\}$  on a set of variables  $X = \{x_1, \dots, x_L\}$  such that each  $c_k$  consists of 3 literals for  $k = 1, \dots, K$ . Each literal is a variable or the negation of a variable.

**Question:** Is there a truth assignment for  $X$  which satisfies each clause  $c_k$ ,  $k = 1, \dots, K$ ?

The dependency graph  $D(\Phi)$  corresponding to an instance  $\Phi$  of 3-SAT is defined to be the bipartite graph whose independent sets are formed by the set of clauses  $C$  and the set of variables  $X$ . Two vertices  $c_k$  and  $x_l$  are adjacent iff  $c_k$  involves  $x_l$  or  $\bar{x}_l$ .

Given any 3-SAT formula  $\Phi$ , we show how to construct in polynomial time an equivalent IMAGE MATCHING problem  $\mathfrak{I}(\Phi) = (A(\Phi), B(\Phi))$ . The two images of  $\mathfrak{I}(\Phi)$  are similar according to the cost function (i.e.  $\exists f : c(A(\Phi), B(\Phi), f) \leq 0$ ) iff the formula  $\Phi$  is satisfiable. We perform the reduction from 3-SAT using the following steps:

- From the formula  $\Phi$  we construct the dependency graph  $D(\Phi)$ .
- The dependency graph  $D(\Phi)$  is drawn in the plane.
- The drawing of  $D(\Phi)$  is refined to depict the logical behaviour of  $\Phi$ , yielding two images  $(A(\Phi), B(\Phi))$ .

For this, we use three types of components: one component to represent variables of  $\Phi$ , one component to represent clauses of  $\Phi$ , and components which act as interfaces between the former two types. Before we give the formal reduction, we introduce these components.

#### 3.1 Basic components

For the reduction from 3-SAT we need five components from which we will construct the instances for IMAGE MATCHING, given a Boolean formula in 3-DNF, resp. its graph. The five components are the building blocks needed for the graph drawing and will be introduced in the following, namely the representations of connectors, crossings, variables and clauses. The connectors represent the edges and have two varieties, straight connectors and corner connectors. Each of the components consists of two parts, one for image  $A$  and one for image  $B$ , where blank pixels are considered to be of the ‘background’

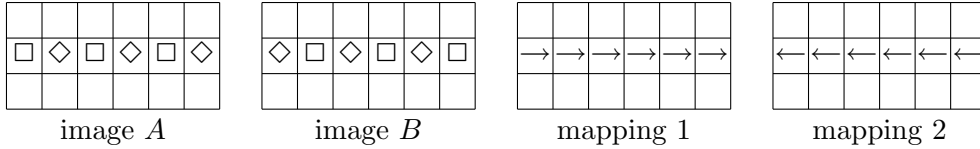
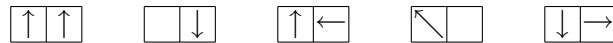


Fig. 1. The straight connector component with two possible zero cost mappings color.

We will depict possible mappings in the following using arrows indicating the direction of displacement (where displacements within the eight-neighborhood of a pixel are the only cases considered). Blank squares represent mapping to the respective counterpart in the second image. For example, the following displacements of neighboring pixels can be used with zero cost:



On the other hand, the following displacements result in costs greater than zero:



Figure 1 shows the first component, the straight connector component, which consists of a line of two different interchanging colors, here denoted by the two symbols  $\diamond$  and  $\square$ . Given that the outside pixels are mapped to their respective counterparts and the connector is continued infinitely, there are two possible ways in which the colored pixels can be mapped, namely to the left (i.e.  $f(2, j) = (2, j - 1)$ ) or to the right (i.e.  $f(2, j) = (2, j + 1)$ ), where the background pixels have different possibilities for the mapping, not influencing the main property of the connector. This property, which justifies the name ‘connector’, is the following: It is not possible to find a mapping, which yields zero cost where the relative displacements of the connector pixels are not equal, i.e. one always has  $f(2, j) - (2, j) = f(2, j') - (2, j')$ , which can easily be observed by induction over  $j'$ . That is, given an initial displacement of one pixel (which will be  $\pm 1$  in this context), the remaining end of the connector has the same displacement if overall costs of the mapping are zero. Given this property and the direction of a connector, which we define to be directed from variable to clause, we can define the state of the connector as carrying the ‘true’ truth value, if the displacement is 1 pixel in the direction of the connector and as carrying the ‘false’ truth value, if the displacement is  $-1$  pixel in the direction of the connector. This property then ensures that the truth value transmitted by the connector cannot change at mappings of zero cost.

For drawing of arbitrary graphs, clearly one also needs corners, which are represented in Figure 2. By considering all possible displacements which guarantee overall cost zero, one can observe, that the corner component also ensures the

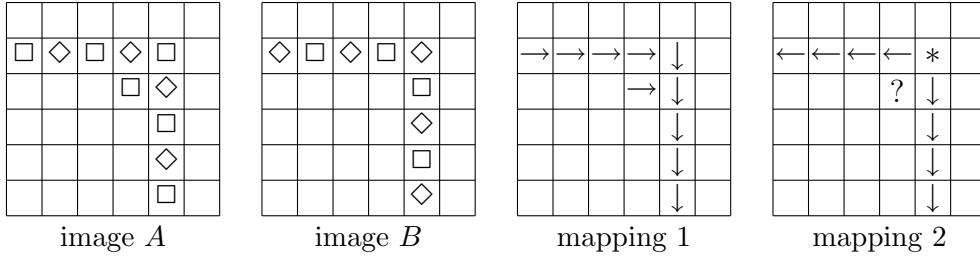


Fig. 2. The corner connector component and two example mappings

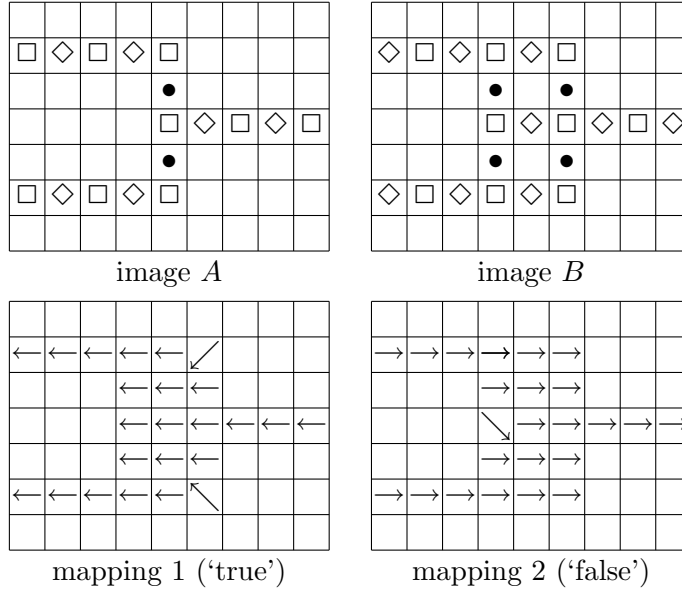


Fig. 3. The variable component with two positive and one negated output and two possible mappings (for true and false truth value)

basic connector property. For example, consider the first depicted mapping, which has zero cost. On the other hand, the second mapping shows, that it is not possible to construct a zero cost mapping with both connectors ‘leaving’ the component. In that case, the pixel at the position marked ‘?’ either has a conflict (that is, introduces a cost greater than zero in the criterion function because of mapping mismatch) with the pixel above or to the right of it, if the same color is to be met and otherwise, a cost in the grayvalue mismatch term is introduced.

Figure 3 shows the variable component, in this case with two positive (to the left) and one negated output (to the right) leaving the component as connectors. Here, a fourth color is used, denoted by  $\bullet$ . This component has two possible mappings for the colored pixels with zero cost, which map the vertical component of the source image to the left or the right vertical component in the target image, respectively. (In both cases the second vertical element in the target image is not a target of the mapping.) This ensures  $\pm 1$  pixel relative displacements at the entry to the connectors. This property again can be deduced by regarding all possible mappings of the two images. The property

that follows (which is necessary for the use as variable) is that all zero cost mappings ensure that all positive connectors carry the same truth value, which is the opposite of the truth value for all the negated connectors. It is easy to see from this example how variable components for arbitrary numbers of positive and negated outputs can be constructed.

Figure 4 shows the most complex of the components, the clause component. The component consists of two parts. The first part is the horizontal connector with a ‘bend’ in it to the right. This part has the property that cost zero mappings are possible for all truth values of  $x$  and  $y$  with the exception of two ‘false’ values. This ‘two input disjunction’ can be extended to a three input disjunction using the part in the lower left. If the  $z$  connector carries a ‘false’ truth value, this part can only be mapped one pixel downwards at zero cost. In that case the junction pixel (the fourth pixel in the third row) cannot be mapped upwards at zero cost and the ‘two input clause’ behaves as described above. On the other hand, if the  $z$  connector carries a ‘true’ truth value, this part can only be mapped one pixel upwards at zero cost, and the junction pixel can be mapped upwards, thus allowing both  $x$  and  $y$  to carry a ‘false’ truth value in a zero cost mapping. Thus there exists a zero cost mapping of the clause component iff at least one of the input connectors carries a ‘true’ truth value.

The described components are already sufficient to prove  $NP$ -completeness by reduction from PLANAR 3-SAT (which is an  $NP$ -complete sub-problem of 3-SAT where the additional constraints on the instances is that the dependency graph is planar), but in order to derive a reduction from 3-SAT, we also include the possibility of crossing connectors.

Figure 5 shows the connector crossing, whose basic property is to allow zero cost mappings iff the truth values are consistently propagated. This is assured by a color change of the vertical connector and a ‘flexible’ middle part, which can be mapped to four different positions depending on the truth value distribution.

### 3.2 Reduction

Using the previously introduced components, we can now perform the reduction from 3-SAT to IMAGE MATCHING.

**Proof** of the claim that IMAGE MATCHING problem is  $NP$ -complete:

Clearly, the IMAGE MATCHING problem is in  $NP$  since, given a mapping  $f$  and two images  $A$  and  $B$ , the computation of  $c(A, B, f)$  can be done in polynomial time. To prove  $NP$ -hardness, we construct a reduction from the

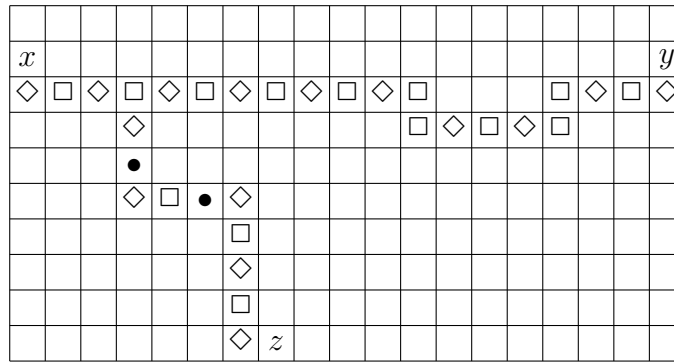


image A

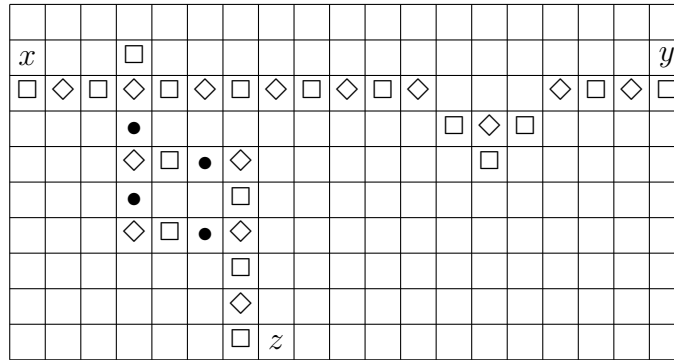
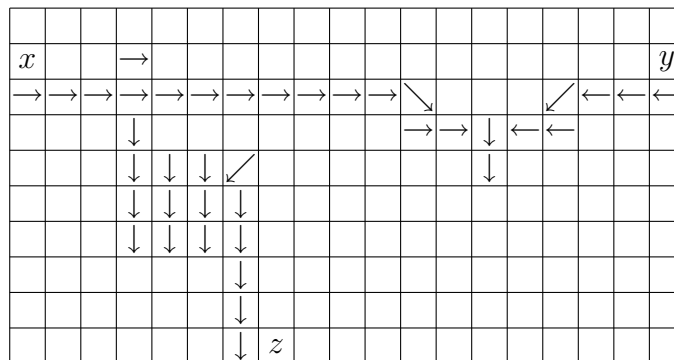
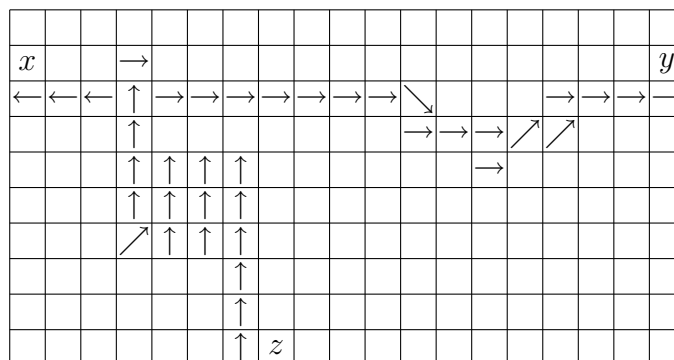


image B



mapping 1 (true,true,false)



mapping 2 (false,false,true)

Fig. 4. The clause component with three incoming connectors  $x, y, z$  and zero cost mappings for the two cases (true, true, false) and (false, false, true)

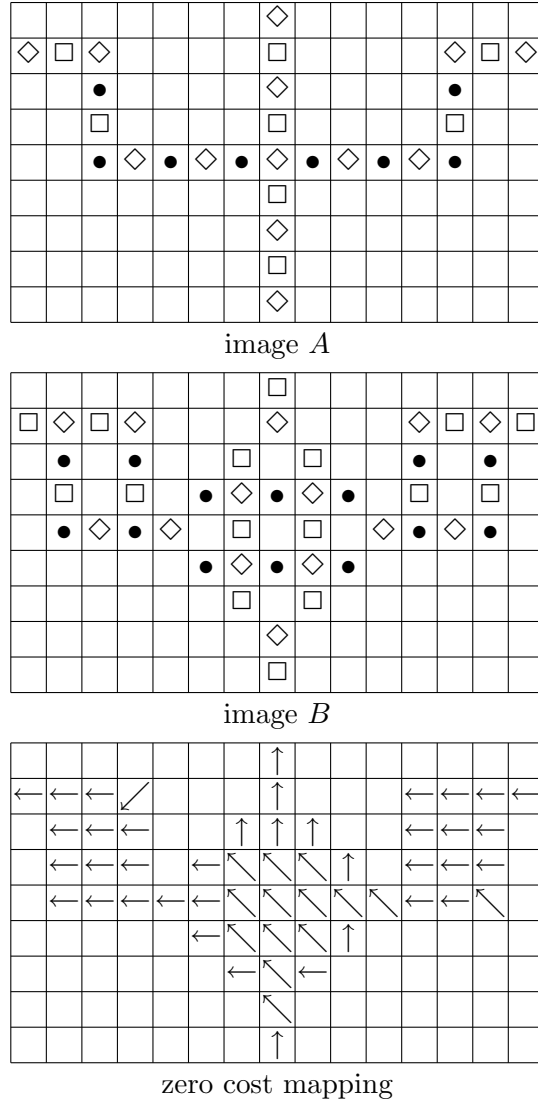


Fig. 5. The connector crossing component and one zero cost mapping

3-SAT problem. Given an instance of 3-SAT we construct two images  $A$  and  $B$ , for which a mapping of cost zero exists iff all the clauses can be satisfied.

Given the dependency graph  $D$ , we construct an embedding of the graph into a two-dimensional pixel grid, placing the vertices on a large enough distance from each other (say  $100(K + L)^2$ ). This can be done using well-known methods from graph drawing (see e.g. di Battista et al., 1999). From this image of the graph  $D$  we construct the two images  $A$  and  $B$ , using the components described above. Each vertex belonging to a variable is replaced with the respective parts of the variable component, having a number of leaving connectors equal to the number of incident edges under consideration of the positive or negative use in the respective clause. Each vertex belonging to a clause is replaced by the respective clause component, and each crossing of edges is replaced by the respective crossing component. Finally, all the edges

are replaced with connectors and corner connectors, and the remaining pixels inside the rectangular hull of the construction are set to the background grayvalue. Clearly, the placement of the components can be done in such a way that all the components are at a large enough distance from each other, where the background pixels act as an ‘insulation’ against mapping of pixels, which do not belong to the same component. It can be easily seen, that the size of the constructed images is polynomial with respect to the number of vertices and edges of  $D$  and thus polynomial in the size of the instance of 3-SAT, at most in the order of  $(K + L)^2$ . Furthermore, it can obviously be constructed in polynomial time, as the corresponding graph drawing algorithms are polynomial.

Let there exist a truth assignment to the variables  $x_1, \dots, x_L$ , which satisfies all the clauses  $c_1, \dots, c_K$ . We construct a mapping  $f$ , that satisfies  $c(f, A, B) = 0$  as follows.

For all pixels  $(i, j)$  belonging to variable component  $l$  with  $A(i, j)$  not of the background color, set  $f(i, j) = (i, j - 1)$  if  $x_l$  is assigned the truth value ‘true’, set  $f(i, j) = (i, j + 1)$ , otherwise. For the remaining pixels of the variable component set  $f(i, j) = (i, j)$  if  $A(i, j) = B(i, j)$ , otherwise choose  $f(i, j)$  from  $\{(i, j + 1), (i + 1, j + 1), (i - 1, j + 1)\}$  for  $x_l$  ‘false’ respectively from  $\{(i, j - 1), (i + 1, j - 1), (i - 1, j - 1)\}$  for  $x_l$  ‘true’, such that  $A(i, j) = B(f(i, j))$ . This assignment is always possible and has zero cost, as can be easily verified.

For the pixels  $(i, j)$  belonging to (corner) connector components, the mapping function can only be extended in one way without the introduction of nonzero cost, starting from the connection with the variable component. This is ensured by the basic connector property. By choosing  $f(i, j) = (i, j)$  for all pixels of background color, we obtain a valid extension for the connectors. For the connector crossing components the extension is straightforward, although here – as in the variable mapping – some care must be taken with the assignment of the background value pixels, but a zero cost assignment is always possible using the same scheme as presented for the variable mapping.

It remains to be shown that the clause components can be mapped at zero cost, if at least one of the input connectors  $x, y, z$  carries a ‘true’ truth value. For a proof we regard all seven possibilities and construct a mapping for each case. In the description of the clause component it was already argued that this is possible, and due to space limitations we omit the formalization of the argument here.

Finally, for all the pixels  $(i, j)$  not belonging to any of the components, we set  $f(i, j) = (i, j)$ , thus arriving at a mapping function which has  $c(f, A, B) = 0$ , as all colors are preserved in the mapping and no distortion of squared Euclidean distance greater than 2 is introduced.

On the other hand, let there exist a mapping  $f$  with  $c(f, A, B) \leq 0$ . This means that  $c(f, A, B) = 0$  since there are only positive cost terms involved in the term for  $c$ . Furthermore, we can conclude, that

$$\begin{aligned} \forall (i, j) \in \mathcal{M} \times \mathcal{M} \quad d_g(A_{ij}, B_{f(i,j)}) &= 0 \\ \forall (i, j) \in \{1, \dots, M-1\} \times \mathcal{M} \quad d_d(f((i, j) + (1, 0)) - (f(i, j) + (1, 0))) &= 0 \\ \forall (i, j) \in \mathcal{M} \times \{1, \dots, M-1\} \quad d_d(f((i, j) + (0, 1)) - (f(i, j) + (0, 1))) &= 0 \end{aligned}$$

This means that  $f$  maps all pixels onto pixels of the same color and that the difference in mapping between neighboring pixels has at most squared Euclidean distance 2.

These facts now ensure a number of basic properties:

- (1) The basic elements of the components and the overall represented graph are mapped to their corresponding parts, because of the monotonicity and continuity assured by the maximum local mapping difference.
- (2) All the variable components are mapped such that the leaving connectors all carry consistent truth values (i.e. all the positive outputs carry the same truth value, and all the negative outputs carry the negated truth value).
- (3) All (corner, crossing, straight) connectors are mapped such that the basic connector property of propagated truth values is fulfilled, since no neighboring pixels are mapped into opposite directions.
- (4) Each clause component has at least one entering connector carrying a ‘true’ truth value, as otherwise a mapping with zero cost is not possible.

Now we construct an assignment of truth values to the variables  $x_1, \dots, x_L$ , which satisfies all the clauses  $c_1, \dots, c_K$ . We set  $x_l$  to ‘true’, if  $f(i, j) = (i, j-1)$  for the pixels of the corresponding variable component, which are not background pixels. We set  $x_l$  to ‘false’, otherwise. By means of the properties stated above we can conclude that this assignment satisfies all the clauses  $c_1, \dots, c_K$ , which concludes the proof.

## 4 Discussion and Conclusion

With the completion of the proof some open questions remain. From the theoretical point of view it is interesting to understand for which distortion measure the problem becomes *NP*-complete, as indicated by the question marks in Table 1 and in which way the result is influenced if we allow interpolation of the discrete pixel array. Another interesting theoretical question is if the

Table 1

Open questions about the complexity with respect to the distortion measure.

distortion measure	$f_2(x) = \text{const.}$	...	$f_2(x) = x$	...	$f_2(x) = \text{step}(x)$
algorithmic complexity	$\in P$	?	?	?	$NP$ -complete

problem remains  $NP$ -complete if we allow only less than the four colors used in the proof here.

In the presented proof, the displacements of the pixels in the mapping are at most one pixel for each zero cost mapping between two constructed images. This may seem to be a restriction with respect to the general matching problem, which allows larger displacements. But obviously the general problem is  $NP$ -complete, if already the class of problems which contains only these restricted versions is  $NP$ -complete.

With the proof that the problem IMAGE MATCHING is  $NP$ -complete, we have arrived at a fundamental result for a problem that has been studied thoroughly in the literature before. The works of Uchida and Sakoe deal with exactly the problem formalized here and they present a variety of exponential algorithms and restrictions to apply to the problem in order to make it tractable. Ronee et al. (2001) state that the computation “is in the exponential order of the image size”, but they do not elaborate on this statement. In the same manner, Levin and Pieraccini (1992) discuss a very similar problem and state that their algorithm is exponential in the general case, but do not go into more detail on this question. They propose to regard the lines of the images independently to keep the problem tractable. This is a similar approach to the use of pseudo 2-dimensional hidden Markov models as presented in (Kuo and Agazzi, 1994), but the drawback is that any correspondence information between the deformation of the lines is disregarded, which may not be advisable in all domains. Samaria (1994) writes for the case of face recognition that “In the general case, the HMM would need to be 2D”, but also restricts the problem to a 1-dimensional one. Moore (1979) already discusses a similar task which is dealt with using a generalization of the 1-dimensional edit- or Levenshtein-distance, which also disregards local dependencies between certain lines. There are also works that view the image-warping problem based on a physical model of the grayvalue-surface that is represented by an image (Moghaddam et al., 1996) and others that use variational approaches that are similar to the computation of optical flow (Schnörr and Weickert, 2000), which is inherently linked with the problem of image registration (Fischer and Modersitzki, 2001).

We may conclude that despite of the fundamental result that the IMAGE MATCHING problem is  $NP$ -complete there is a variety of methods to modify the basic problem such that it becomes tractable or to approximate the best solution. It is an interesting question to find out which of the possibilities is best suited for which practical application.

## References

- Casacuberta, F., de la Higuera, C., 1999. Linguistic decoding is a difficult computational problem. *Pattern Recognition Letters* 20, 813–821.
- di Battista, G., Eades, P., Tamassia, R., 1999. *Graph Drawing*. Prentice Hall, Upper Saddle River, NJ.
- Fischer, B., Modersitzki, J., Mar. 2001. A super fast registration algorithm. In: *Proceedings Bildverarbeitung für die Medizin*. Springer, Lübeck, Germany, pp. 169–173.
- Garey, M. R., Johnson, D. S., 1979. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York.
- Keysers, D., Dahmen, J., Theiner, T., Ney, H., Sep. 2000. Experiments with an extended tangent distance. In: *Proceedings 15th International Conference on Pattern Recognition*. Vol. 2. Barcelona, Spain, pp. 38–42.
- Kuo, S., Agazzi, O., Aug. 1994. Keyword spotting in poorly printed documents using pseudo 2-d hidden markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16 (8), 842–848.
- Levin, E., Pieraccini, R., Mar. 1992. Dynamic planar warping for optical character recognition. In: *International Conference on Acoustics, Speech and Signal Processing*. Vol. 3. San Francisco, CA, pp. 149–152.
- Moghaddam, B., Nastar, C., Pentland, A., Aug. 1996. A bayesian similarity measure for direct image matching. In: *Proceedings 13th International Conference on Pattern Recognition*. Vienna, Austria, pp. 350–358.
- Moore, R. K., Jan. 1979. A dynamic programming algorithm for the distance between two finite areas. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1 (1), 86–88.
- Ney, H., Mergel, D., Noll, A., Paeseler, A., Feb. 1992. Data driven search organization for continuous speech recognition. *IEEE Transactions on Signal Processing* 40 (2), 271–281.
- Ronee, M. A., Uchida, S., Sakoe, H., Sep. 2001. Handwritten character recognition using piecewise linear two-dimensional warping. In: *Proceedings 6th International Conference on Document Analysis and Recognition*. Seattle, WA, pp. 39–43.
- Samaria, F. S., Oct. 1994. *Face recognition using hidden markov models*. PhD thesis, University of Cambridge, Cambridge, UK.
- Schnörr, C., Weickert, J., Sep. 2000. Variational image motion computation: Theoretical framework, problems and perspectives. In: *Proceedings 22. DAGM Symposium Mustererkennung 2000*. Springer, Kiel, Germany, pp. 476–487.
- Uchida, S., Sakoe, H., Aug. 1998. A monotonic and continuous two-dimensional warping based on dynamic programming. In: *Proceedings 14th International Conference on Pattern Recognition*. Vol. 1. Brisbane, Australia, pp. 521–524.
- Uchida, S., Sakoe, H., Mar. 1999a. An efficient two-dimensional warping algorithm. *IEICE Transactions on Information & Systems* E82-D (3), 693–700.

- Uchida, S., Sakoe, H., Sep. 1999b. Handwritten character recognition using monotonic and continuous two-dimensional warping. In: Proceedings 5th International Conference on Document Analysis and Recognition. Bangalore, India, pp. 499–502.
- Uchida, S., Sakoe, H., Jan. 2000a. An approximation algorithm for two-dimensional warping. IEICE Transactions on Information & Systems E83-D (1), 109–111.
- Uchida, S., Sakoe, H., Sep. 2000b. Piecewise linear two-dimensional warping. In: Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain. Vol. 3. pp. 538–541.