Journal of Computer and System Sciences 82 (2016) 1360–1387

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

The exact complexity of projective image matching

Christian Rosenke

Institute of Computer Science, University of Rostock, Albert-Einstein-Straße 22, 18051 Rostock, Germany

ARTICLE INFO

Article history:
Received 19 December 2015
Received in revised form 30 May 2016
Accepted 10 June 2016
Available online 1 July 2016

Keywords:
Digital image matching
Projective transformations
Discretization of function spaces
Computational geometry
Design and analysis of parallel algorithms

ABSTRACT

Projective image matching is an important image processing task. Given digital images A and B, the challenge is to find a projective transformation f for A such that it most closely resembles B. Previous research led to a polynomial time algorithm that elaborately searches the so-called dictionary $\mathcal{D}(A)$ of all projective transformations of A using a preprocessed data structure. This paper shows that projective image matching is even TC_0-complete by applying a much simpler parallel way of browsing $\mathcal{D}(A)$. This reveals further insight into the problem's structural properties and relates it to fundamental computer operations like integer multiplication and division.

1. Introduction

Projective image matching is an important, canonical problem in signal processing. Given two digital images A and B, the objective is to determine a projective transformation f for A such that it most closely resembles B.

The research in image matching is motivated by many practically relevant computational challenges that are related to finding the similarity between objects displayed in digital images. The reason for this is that image matching can be used to compensate geometric distortions of those objects occurring when the image is created or due to some subsequent image processing. MPEG video compression [e.g. [1]], for instance, needs to determine moving objects in successive images of the video stream to reduce their redundant representation. Likewise, a challenge in medical imaging [e.g. [2]] is to correlate images of the same object taken in different times or using different medical image devices. In computer vision, objects like a Latin letter [e.g. [3]] or a human silhouette [e.g. [4]] displayed with unknown location and orientation have to be detected in given images. Image matching can be also applied for robust digital watermarking [e.g. [5,6]] where imperceptible patterns that have been embedded into an image should be detected even after geometric removal attacks or print-and-scan-processes.

Often, projective transformations, denoted in this paper as $\mathcal{F}$, are used because they describe natural geometric basic operations like scaling, rotating, translating and shearing. However, despite its importance and diversity of applications our understanding of the computational complexity of the image matching problem under projective transformations remains limited in important respects. Applying standard image matching approaches for this class requires either exponential running time or guarantees only approximate solutions and, until recently, it was unknown whether the problem is tractable or not. Lately, research has succeeded in solving the projective image matching problem in polynomial time [7]. However, the exact complexity of the problem remains unknown. In particular, it is an open question whether the problem belongs to one of the following complexity classes below polynomial time solvable problems:

E-mail address: christian.rosenke@uni-rostock.de.

http://dx.doi.org/10.1016/j.jcss.2016.06.002
0022-0000/© 2016 Elsevier Inc. All rights reserved.

$$AC_O \subseteq TC_O \subseteq NC_1 \subseteq LOGSPACE \subseteq PTIME.$$

Every class in this hierarchy below *PTIME* provides a theoretical computational advantage against sequential polynomial time computing. More precisely, speeding up computation by parallel architectures is less difficult for lower classes.

The achievement of this paper is a proof for the containment of projective image matching in the surprisingly low complexity class TC_O, which exactly expresses the complexity of a variety of basic problems in computation, such as integer addition, multiplication, comparison and division. To be more precise, we show that projective image matching is a member of the function class U_D-FTC_O that consists of all binary word functions $g : \{0,1\}^* \to \{0,1\}^*$ that can be computed by circuit families having (1) constant depth, (2) a polynomial amount of Boolean gates and threshold gates, and (3) so-called DLOGTIME-uniformity. Although this new result has no immediate impact on practical image matching settings, it represents an ambition to uncover the structural properties of suchlike matching problems. Theoretically, the presented result implies that image matching can be solved in logarithmic time on multi-processor systems. In fact, it can even be solved in constant time, if the parallel processors are assumed to have unbounded fan-in. Moreover, since $TC_O \subseteq LOGSPACE$, the problem can also be solved on sequential machines using only a very limited, logarithmic amount of memory, hence, without the need of preprocessing massive data structures as the previous algorithmic approach in [7].

We subsequently show the completeness of projective image matching in U_D-FTC_O. This implies that it is not possible to solve the problem by even weaker theoretical parallel computing architectures, as for instance represented by the class AC_O. In particular, image matching requires a certain way of efficient parallel counting which is not available below TC_O.

An extended abstract of this paper has been presented at CPM 2010 [8] which outlines the basic ideas presented here for the less sophisticated case of affine image matching, the same problem for the affine subclass of projective transformations.

The paper is organized as follows. In the next section we fix the setting of the projective image matching problem and Section 3 gives an overview of known results. The Sections 4–9 present our contribution. Finally, Section 10 discusses implications of our result.

To maintain a continuous text flow and improve the comprehensibility of this paper, the proofs of all theorems and lemmas have been moved to the Appendix.

2. The image matching problem

The image matching problem as described informally in the introduction, leaves some room for ambiguity. Yet, digital images A and B, transformed versions $f\langle A\rangle$ of a digital image A, and the deviation of $f\langle A\rangle$ from B are concepts that we have not well-defined so far. As image matching occurs in so many different fields, the abstract idea behind the problem has been fine tuned in many different ways. Statements about the computational properties of image matching are clearly desired in a most general fashion in order to cover a maximum number of aspects of the problem's diversity at the same time. However, general statements like this are usually impossible to find and even if they are found, they do not tend to be very sharp. Consequently, this section limits the variability of each aspect as required by the methodology of this paper and explains the reasonability of every decision.

In most applications, digital images are rectangular pixel arrays where every pixel is a 2D index in the array giving a value from a limited set of colors, which are usually encoded by natural numbers. We conform to this interpretation and mathematically describe a digital image A as a mapping

$$A : \mathbb{Z}^2 \to \mathcal{C} \tag{1}$$

that assigns to every pixel $(i,j)^T \in \mathbb{Z}^2$ a color value $A(i,j)$ from $\mathcal{C}$, a finite set of colors usually encoded by C natural numbers, that is, $\mathcal{C} = \{0,\dots,C-1\}$. That digital images are of a limited size $n \in \mathbb{N}$ is reflected in our notation by the so-called n-support

$$z(n) = \left\{(i,j)^T \mid -n \le i,j \le n\right\}$$

and the fact that we let $A(i,j) = 0$ for all pixels $(i,j)^T$ that do not belong to $z(n)$. By this notion we are, however, restricted to square images. Extending the theory of this paper to arbitrary rectangular images is not really a problem but it comes with the cost of a significantly more complicated notation.

We also like to point out that this is not the only possible way to model digital images and that this discrete approach has strengths and weaknesses compared to others. Often, especially in medical imaging [e.g. [9]], a digital image is described as a continuous function $a : \mathbb{R}^2 \to \mathbb{R}$ where non-integer arguments are assigned mixed color values from neighboring pixels. This formulation, called continuous setting in our paper, is sometimes considered more natural, as digital images often depict real objects which are also of a continuous nature. On the other hand, this approach clouds the discrete nature of digital images and its degree of naturalness depends a lot on how the mixed colors are actually generated. Moreover, the mathematical methods to cope with continuous images tend to be more sophisticated.

Even more arguable than the definition of a digital image is the question how to specify the effect of a projective transformation $f : \mathbb{R}^2 \to \mathbb{R}^2$ on a digital image. In the continuous setting, for instance, this is simply achieved by the function $f\langle a\rangle(x,y) = a(f(x,y))$. For our discrete scenario, there are basically two principal ways of defining color values for a transformed image $f\langle A\rangle$ of size n: *forward* and *inverse* mapping. In forward mapping, the color of each pixel $(i,j)^T$