Chapter 7
Data Flow Implementation of Erosion and Dilation

Christina Pacher
Universität Wien, Austria

ABSTRACT

This chapter describes a data flow implementation of the image processing algorithms Erosion and Dilation. Erosion and Dilation are basic image processing algorithms which are used to reduce or increase the size of objects in images, respectively, and which are used in a wide number of image processing applications. The chapter first describes the control flow versions of the algorithms in detail. Subsequently, the translation of these algorithms to the Data Flow paradigm is examined, and the details of the data flow implementation as well as possible optimizations are discussed.

INTRODUCTION

This chapter describes a data flow implementation of the image processing algorithms “erosion” and “dilation”. The purpose of these algorithms is to reduce (for erosion) or increase (for dilation) the size of objects in images by removing or adding pixels around their edges. They are also necessary for the image processing algorithms “opening” and “closing”, which use sequences of erosion and dilation operations to separate or join objects in images (Phillips, 2000).

Typical use cases for erosion and dilation include the de-noising of images as part of some larger image processing application (e.g. road detection as used by Ming et al. (2017)). But there are also applications outside the domain of image processing: Hanjun & Huali (2010), for example, use erosion and dilation in image rendering for the creation of fake soft shadows.

Image processing algorithms are generally good candidates for implementation as data flow algorithms, as they usually require the same set of operations to be executed for each pixel. In fact there actually exist data flow architectures and hybrid data flow architectures that were designed specifically for working with image processing algorithms; they are highly beneficial for the speed of those algorithms and can even be used for real-time image processing (Quénot & Zavidovique, 1992; Sinha et al., 2002).

DOI: 10.4018/978-1-7998-7156-9.ch007

Like most algorithms, erosion and dilation come in many different flavours. Phillips (2000) offers descriptions of two different versions of each of the algorithms; yet another definition is given by Nixon & Aguado (2012). There are also a great many variants and performance-tuned versions of those algorithms, and/or versions that are tailored to special cases like binary images (e.g. Chen & Haralick (1995) and Cheng (2009)).

The data flow implementation described here is based on one of the versions of erosion and dilation described by Phillips (2000).

The remainder of this chapter will first describe the chosen versions of the algorithms in detail. Subsequently, it will be examined how these algorithms can be translated to the data flow paradigm.

CONTROL FLOW ALGORITHMS

In order to be able to successfully transfer the algorithms to data flow, it is necessary to first understand exactly how the control flow implementations work; for this reason, this section is concerned with a description of erosion and dilation in control flow. It is first necessary to specify some assumptions about the input data, before examining the logic behind the algorithms.

Assumptions

The implementation described here uses the same notion of an image that is employed by Phillips (2000), where each pixel corresponds to one integer number; this basically means that the processed image has only one colour channel (e.g. greyscale). However, the algorithms could just as well be applied to images with three colour channels; it would suffice to perform the operations on each of the colour channels separately and then re-combine the results.

It is also assumed that a pixel value of 0, which is the minimum possible value, denotes the background colour; all pixels with values other than 0 belong to objects. The maximum possible value is assumed to be 255. Of course the algorithms would also work if other values were specified as the minimum and maximum, but it is important that the range of possible values has to be known beforehand since this influences the values of constants in the code, as will be shown below in the discussion of the data flow implementation.

Algorithm Description

The basic idea behind both algorithms is very simple: They iterate over all pixels in the image and choose a new value for each pixel based on its own current value and the current values of its neighbours.

As mentioned above, there are two versions of both erosion and dilation described by Phillips (2000). They differ mainly in their approach to determining the new value: One uses a threshold parameter and changes the value of the current pixel depending on whether or not the number of neighbours with a different value exceeds the threshold, and one uses masks to determine the new value of a pixel. This implementation was based on the mask versions because they translate to data flow very naturally, as will be seen in the discussion of the data flow implementation.