1. Introduction

Neighborhood processing is one of the most widely used kinds of parallelism in image processing. The degree of parallelism depends on the size of the neighborhood. Very popular are 3x3-neighborhood processors. Some theoretical and practical reasons for it may be reminded:
- almost all basic algorithms of preprocessing can be implemented within a 3x3-window,
- using methods of image algebra preprocessing within larger windows can be reduced to 3x3-window processing under weak constraints,
- special purpose hardware for typical image size and gray level resolution may be developed using a single board for one processor working at video rate.

In neighborhood processors the single pixels in an image are processed serially, therefore the time is growing linearly with image size. Nevertheless, the processing may be accelerated by building up pipelines of differently programmed processors each of them realizing one processing pass.

Neighborhood processors are built for some classes of algorithms, especially for convolution and morphology. An external pipeline of differently programmed processors is used in the most prominent device of this kind, the Cytocomputer (Sternberg, 1981). The GIPP processor (gray scale image preprocessor) described later on is the basic structure for a very similar architecture. It is an ASIC which by sorting or comparing pixel values in dependence of certain criteria, by mask matching for deriving conditions for application of local operators, preprocesses images within windows of 3x3 pixels.

The GIPP processor also supports the transition from the low-level raster representation to intermediate-level representation by Freeman chains.
2. Principles of GIPP-processing

2.1 Basic computing algorithm

The principles of GIPP-processing can be described in the following three steps:

1. Given is a set of a finite number \( k \) of image processing algorithms.
2. It is possible to map this restricted class of image processing algorithms onto a single basic computing algorithm with \( k \) parameters microprograms or for this restricted class of image processing algorithms. This basic computing algorithm must be well adapted to an efficient implementation.
3. This basic computing algorithm can be efficiently implemented in the hardware-structure GIPP, which is programmable for a number \( k \) of image processing algorithms.

In this sense a GIPP can be considered as a basic structure for image processing leading to the concept of GIPP-implementable image processing functions. At present all work is related to a structure using the raster-scan approach and a 3x3-window with the characteristic movement of the centre-pixel.

In general, the computation of the new centre-pixel value consists in selecting one of the gray values inside of the window. The selection rule is determined by the given processing function. Furthermore, masks can be set. If the mask is matched, the computation of the new centre-pixel value is carried out, if not, the centre-pixel remains at its former value.

Usually the GIPP works in a bit-plane manner. One iteration step is needed for each of the \( M \) bit planes. This means, that every computation of a new centre-pixel value is executed by \( M \) iteration steps:

\[
M = \log_2 NG,
\]

\( M = \) number of necessary iteration steps,
\( = \) number of bit planes,
\( NG = \) number of gray levels.

In each iteration step \( k \) \( (k=1...M) \) the available gray value range \( G^k \) is divided into a lower gray value range \( G^k_l \) and an upper one \( G^k_u \) of equal size:

\[
G^k = \{ g^k_1, g^k_{1+1}, ..., g^k_m \} \quad \text{gray value range,}
\]

\[
G^k_l = \{ g^k_1, g^k_{1+1}, ..., g^k_{m-1} \} \quad \text{lower gray value range,}
\]