Quad-tree Image Compression using reconfigurable free-space optical interconnections and pipelined parallel processors

A. Cassinelli, M. Naruse, and M. Ishikawa

Department of Information Physics and Computing, Graduate School of Information Science and Technology / University of Tokyo, Bunkyo-ku, Tokyo 113-8656, Japan.

Abstract: *We present here a proof-of-principle -though realistic application- for a pipelined multilayered optoelectronic parallel processor with reconfigurable optical interconnects (OCULAR-II), namely quad-tree compression of binary images.*

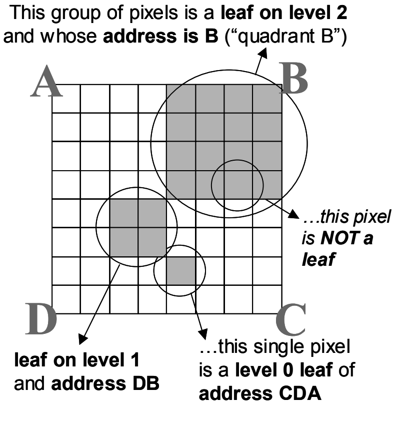
Introduction: This paper describes a possible application for our optoelectronic multilayered parallel processor using reconfigurable optical interconnects, OCULAR-II [1]. The system is based on two-dimensional optoelectronic arrays, connected trough a compact interconnection module based on reconfigurable holograms. Previous work has focused on the design and performances of this module, as well as rather simple tests of the input/output arrays [2]. We want to describe now the first complete demonstration trough a realistic application, namely quad-tree image compression.

Quad-Tree Compression: A quad-tree is a tree-like data structure where each node either terminates on a *leaf* containing useful information, or branches into *four sub-level quad-trees*. In the case of interest, the tree represents a binary image in the following way: the root of the tree contains, when possible, information about the image as a whole (all black/all white) ; otherwise it branches into four sub-trees relative to quadrants of the image (see Fig.1). Recursively, each sub-tree codes each quadrant in the same manner, so that if the three were completely developed, it would end up only on leaves, containing information about individual *pixels*.

Fig.1 : principle of the quad-tree algorithm. The binary 8x8 image is compressed into only three leafs (of a maximum of eight bits each) :

Final Tree = (2 , B) + (1 , DB) + (0 , CDA)

level address


This group of pixels is a **leaf on level 2** and whose **address is B** ("quadrant B")
A
B
…*this pixel is* ***NOT*** *a* ***leaf***
D
C
leaf on level 1 and address **DB**
…this single pixel is a **level 0** leaf of **address CDA**

The goal of the algorithm is to determine all the *maximal leafs* of the image, i.e., leafs that can't be put together to form a higher-level leaf.

System Architecture: The architecture of the system is based on a pipelined configuration of processing arrays, reduced here to only two processors: the « sender » consisting on a 8x8 array of processing elements (PE) attached to an array of vertical-cavity surface-emitting lasers (VCSELs). The « receiver » array is a second array of PEs attached to an equally large array of photodetectors (see Fig.2). Each PE has simple computing capabilities, and a 24-bit memory [3].

A host computer uploads and downloads data from the arrays, performs electronic feedback between the receiving and sender side, and synchronize and gives instructions to both parallel processors in a SIMD fashion.

Processing: Given a 2^Nx2^N pixels large image, the compression algorithm runs N times, from the « finest » grain or *level 0* (corresponding to actual pixels of the image) to the roughest level, or *level N*, corresponding to the image as a whole (here, N=8). The algorithm start by electronically uploading the binary image on bit **leaf** of each PE in sender array – i.e., initially all pixels are considered as leafs on level 0. Additional bits are used to produce computing « masks » depending on the level being processed.

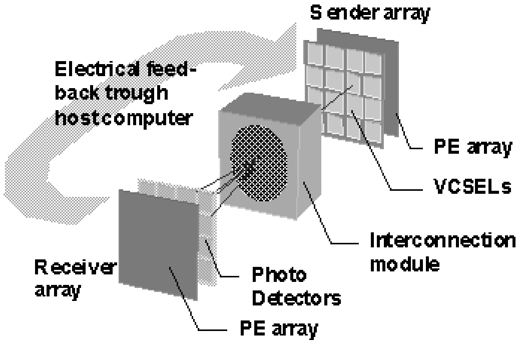


Fig.2 : two layer optically interconnected demonstrator.

For each level, the algorithm can be decomposed in two phases (see fig.3): **(a) detecting and regrouping false leaves** by sequential (optical) broadcast of sub-corner leaf's values to the corresponding upper-left PE on receiver array. These are compared on the receiver array and the result is electronically fed-back to sender array, where update of (upper-left PE) leaf's level is done, *if* sub-corners were "false" leafs. **(b) cutting false leaves** : leaf's values of sub-corner for which upper-left PE is now an upper leaf, must be reset (one optical broadcast, using a *symmetric* hologram). A final electronic feedback is done before going to the next level. For each level, the synthetic hologram (pre-calculated by simulated annealing) is changed. The algorithm ends by downloading data of **PEs that are still leaves.** Fig.4 shows some clichés taken during processing.

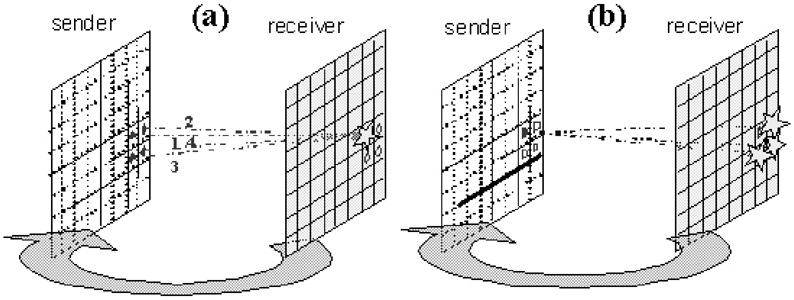


Fig.3 :

(a) : *detecting leafs* : sender PEs sequentially send data to corresponding upper-left PE on receiving array.

(b) : *cutting false leafs* if necessary (the star indicates an active PE).

Results and discussion: A full compression of a 2^Nx2^N pixel large image takes about 6xN clock cycles. Since chip *logic* can operate at more than 100 MHz, that would mean almost **two million 1024x1024 images compressed per second**. Ratio of compression is strongly image-dependent, though; also, one has to remember that our chips are only 8x8 pixel large.

Presently, the LCD reconfiguration time accounts for the strongest device limitation (less than 100Hz [2]); an interesting approach if very high speed reconfiguration is required would be to use an optical disc or any other optical medium to store the images to be projected on the optically addressable PAL-SLM instead of using an electrically controlled LCD device. Also, our present arrays have *either* photodetectors *or* VCSELS: this means that processing is slowed down since data in the receiver array must be *electronically* sent to the sender side. A possible solution to this bottleneck is to use arrays capable of both emission and detection [5].

Anyway, tests were not run in real time because the interconnection module is presently controlled by a second computer which is not synchronized with the main one, and a human operator is needed to launch the operating sequence for each level.