Gradient Pile up for Edge Detection on DSP

Thiago Figueiró, André Soares, Leticia Guimarães, Altamiro Susin
Departamento de Engenharia Elétrica – Universidade Federal do Rio Grande do Sul
Av. Osvaldo Aranha, 103, CEP. 90 035-190, Porto Alegre, RS, Brazil
{figueiro,borin}@eletro.ufrgs.br

Abstract

Edge detection is an important task on image processing. Many edge detectors algorithms have already been proposed, but most of them are based on step edges model and applying smooth filters to minimize the noise and the image derivative or gradient to enhance the edges. However, image sensors have a limited bandwidth, producing ramp edges with the same gradient magnitude as those produced by noise. This work presents a DSP implementation of the Gradient Pile up, an algorithm proposed to edge detection, enhancing the gradient correspondent to ramp edges without amplifying the noisy edges. The experimental results show that the proposed DSP implementation of the pile up process is efficient on enhancement of the gradient of an image and time required for its execution comparing with a PC implementation.

1. Introduction

The estimation of the boundary location of an object is an important subject on image processing, and several techniques for edge enhancement were proposed [Canny 1986, Marr 1980]. Most of the edge detection techniques apply the derivative to enhance the edge of the objects on an image. Usually, the derivative of an image is produced by the convolution with gradient operators [Gonzales 1992], as Sobel, Roberts, Prewitt, for example.

However, there are problems on using the derivative as edge detector, such as false edges detection produced by noise and the not accurate ramp edges location, since the ramp edges produce lower derivative magnitude.

The gradient pile up approach [Guimarães 2004] aims to enhance the gradient value of ramp edges without enhancing noisy gradient. The enhancement occurs over the two component vector of the gradient, the horizontal and vertical gradient map separately. The experiments presented in this paper demonstrate the results of applying the pile up technique using DSP, in order to apply the algorithm in real time embedded systems.

2. Pile up Method

First, a smooth filter is applied to the original image. Second, the horizontal and vertical components of the gradient of an image are calculated. Third, each component of the gradient generates a connected component map, the gradient enhancement map (GEM), that guide the gradient enhancement process. Finally, the horizontal and vertical gradient maps are enhanced by a process similar to a pile up, guided by their respective gradient enhancement map (GEM). A general scheme of the Vertical Pile up process is shown in Fig. 1.

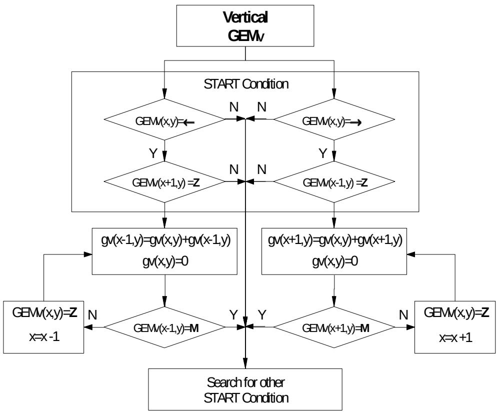

Figure 1. Scheme of the gradient enhancement process (pile up algorithm) for the vertical gradient enhancement.

2.1. Gradient Map Calculation

The gradient is composed by the horizontal and vertical components. The horizontal gradient map gh(x,y) is calculated by the convolution of the image f(x,y) with a horizontal gradient operator, while the vertical gradient map gv(x,y) is calculated using a vertical gradient operator. The gradient magnitude map g(x,y) is the magnitude of the vector composed by two components. An example of horizontal and vertical operators is the Prewitt operators, shown in Figure 2.

$$\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

Figure 2. The vertical and horizontal Prewitt operators, respectively.

2.2. Gradient Enhancement Map

The two gradient maps generated during the gradient map calculation are enhanced, separately, generating two different gradient enhancement maps, the GEMH(x,y) and GEMV(x,y). The GEMs while guides the pile up process.

The GEM is based on connected components map, generating a arrows map pointing to the maximum neighbor in magnitude. Since the direction of the arrows are related to the gradient direction, GEMH is composed only by up and down arrow and GEMV is composed by only left and right arrow.

2.3. Pile up process

The gradient pile up process occurs over the horizontal and vertical gradient maps, gh(x,y) and gv(x,y), guided by the gradient enhancement maps, GEMs. The GEMs indicate the start and the end points of the gradient pile up process. The gradient value correspondent to an arrow on GEM is piled up over the neighbor gradient value where is an arrow on GEM. The pile up process ends at a maximum.

The edge points of the resulted edge map are those which correspond to the maximum at the GEM after the pile up process. However, only piled up maximum is considered as an edge point. More details and a performance evaluation of the method can be obtained in [Guimaraes 2004].

3. Experimental Results

The Pile up algorithm was developed in language C in order to be added to the LaPSI Image Processing Library [Figueiro 2004, lili]. In order to evaluate the performance of the pile up process in embedded systems, we decided to generate a code version for ADSP21XX family of DSP processors. A partial result of using this algorithm is presented in Figure 3, where the original image is the Lena, 256x256, in grayscale.

The original C code suffered several modifications in order to aid the memory management, which is considerably smaller than executed in a commercial PC. The algorithm had to be modified in order to operate with a small block of the image each time. First, the program executes the operations in all the blocks for generating the horizontal maps, and returns a partial result. Then, the vertical maps are processed in all the blocks, and the system generates another partial result. The last step is to merge both partial results in order to generate a final enhance gradient map, indicating the edges of the original image.

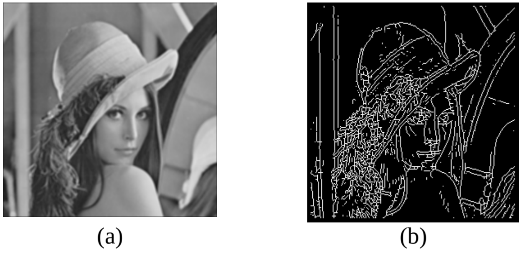

Figure 3: Edge detection on Lena; (a) original image; (b) edge map obtained using the pile-up method architecture

In order to evaluate the process we compared the developed system with two different systems. The first was the original C program in a Pentium III, 550 MHz, and using memory of 384 MB. The Pentium set up was prepared using a freeware C compiler named Dev-C++ and an image processing library [Figueiro 2004, lili]. The second was an architecture developed using VHDL [Soares 2005] and prototyped in a FPGA running at 50MHz with a single communication channel to a PC.

The evaluation was made considering the number of cycles necessary to process the same Lena image (256 x 256, grayscale). The results are shown in Table 1.

Table 1. Comparison Results of the DSP, PC and FPGA Pile Ups.

System	Cycles
ADSP2181, 30 MHz	105,228,666 cycles
Pentium 550 MHz, 384 MB	203,500,000 cycles
FPGA, 50 MHz	986,085 cycles

4. Conclusion

The pile-up method for edge enhancement and detection was presented. An implementation to perform in hardware the Pile up edge detection was developed and tested on a DSP. The software used was based on a C algorithm from the LaPSI Image processing Library (*lili*).

The result presented denotes the viability of using C code in order to implement hardware applications on a DSP. Considering the facility of the implementation, and the low cost of a system using DSP, the loss on time is not so significant in most sort of applications.

5. References

Canny, J., A Computational Approach to Edge Detection, PAMI, V. 8, No. 6, pp. 679-698, 1986.

Figueiro, T., Schuch, N., Socal, F., Guimarães, L.V., Susin, A., "LaPSI Image Processing Library : lili", SIDEE 2004, Porto Alegre, RS.

Gonzalez, R., Woods, R., Digital Image Processing, Addison-Wesley, 1992.

Guimarães,L.V., Soares, A.B., Cordeiro,V., Susin,A., "Gradient Pile Up Algorithm for Edge Enhancement and Detection." ICIAR, 2004

***Lili* - LaPSI Image Processing Library, www.lapsi.eletro.ufrgs.br/lili**

Marr, D. and Hildreth, E. C., Theory of Edge Detection, Proc. of the Royal Society of London B207, pp. 187-217, 1980.

Soares, A.B., Guimarães,L.V., Cordeiro,V., Susin,A., "Gradient Pile Up for Edge Detection in Hardware". ISCAS, 2005.