# Edge Recognition using Image-Processing Hardware


**Alan M. McIvor***

Robotics Research Group, University of Oxford



*This paper discusses the implementation of several edge recognition systems: Sobel; Laplacian of Gaussian; and Canny, on commerically available Datacube MaxVideo image-processing hardware at or near frame rate. Any approximations necessary to operate at these throughput rates are described and a comparison made between the output of these systems and slower implementations on general-purpose machines.*


The recognition of edges in an image is often considered an important first step in image understanding. By edges it is meant the loci of "significant intensity changes". This is usually interpreted as the gradient maxima or the zerocrossings of a second derivative. Edges in an image are important because, in general, the boundaries of objects in the scene being viewed produce edges in the image. Also the edges divide the image into regions of smooth intensity, leading to a compact description of the image.

The process of edge recognition can be divided into three stages:

- Application of an *edge operator* to the image to produce another image in which the edges form easily recognizable structures;
- *Localization* of the edges in this image, since the output of the edge operator often produces multiple responses in the vicinity of the edges;
- Extraction of *edge lists*, that is, converting the data from an image into a set of lists of consecutive points forming each edge segment in the image and a list of connectivity between edge segments.

Many techniques have been proposed for doing each of these three steps. Often the third is considered to be a separate task from the first two. This paper will discuss three different techniques used for the first two steps: Sobel [1], Laplacian of Gaussian [5], and Canny [2,3]. It will be shown how they can be implemented at or close to frame rate on Datacube MaxVideo image-processing hardware [4].

## SOBEL

Early on in computer vision research, it was recognized that detecting edges could be accomplished by forming the gradient of the image. Many operators which approximated the $x$ and $y$ derivatives of an image were proposed. One of the most successful is the Sobel operator [1]. Each operator consists of central difference derivative operators combined by weighted averaging in the perpendicular direction.

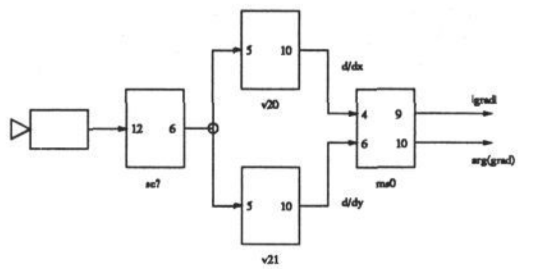



Figure 1: *MAXbus connections for Sobel.*

The two convolution operators have $3 \times 3$ kernels and hence can be applied to an image by any of the Datacube video-FIR filter boards at frame rate. It is usual to convert the gradient from $x$ and $y$ components to magnitude and orientation. This operation can also be performed at frame-rate in the look-up tables on a Max-SP or MaxMux. Thus, with a set of three boards as illustrated in Figure 1, the Sobel operator can be applied to an image stream at frame-rate. Results from using this implementation are shown in Figure 2.

A simple localization scheme is to threshold on the magnitude component of the gradient, as illustrated in Figure 2d,e. This can be performed in a separate look-up table or by modifying the look-up table used to generate the magnitude and orientation images. So thresholded magnitude and orientation images can also be generated at frame rate. The resulting images often contain thick edges, i.e., edges several pixels wide. This may not be inconvenient for late processing stages such as Hough transforms [1] but would upset most edge tracing algorithms. One solution is to use an ad-hoc thinning algorithm [6]. The only restriction in this implementation is that the $x$ and $y$ derivatives must be truncated to 8-bits each to form the look-up table address for calculating the magnitude and orientation using a MaxMux and 6-bits each when using a Max-SP. Neither of these restrictions seriously affects the results.

## LAPLACIAN OF GAUSSIAN

The Laplacian of Gaussian operator detects edges as the zerocrossings of images resulting from the convolution of the input image with a Gaussian smoothing kernel followed by applying the Laplacian [5]. The size of the kernel depends on the Gaussian's parameter $\sigma$ and at what magnitude it is truncated. Since the convolution boards use 8-bit coefficients and the kernel has signed entries, this means that the kernel must be truncated where the magnitude falls below 1/128th of its maximum. This is at a higher level than usual. In the

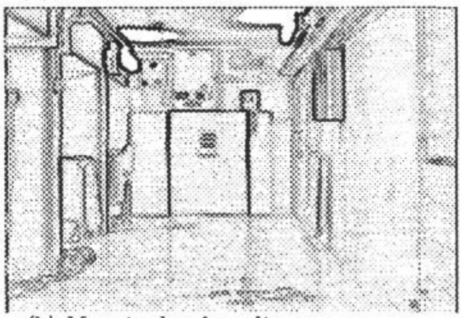

*Current Address: B. P. Research Centre, Chertsey Road, Sunbury on Thames, Middlesex TW16 7LN



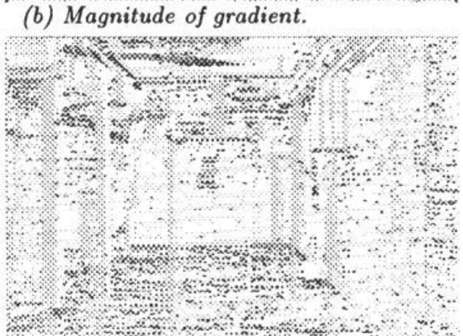


AVC 1989 doi:10.5244/C.3.48

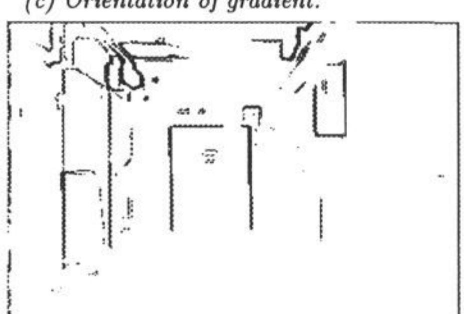
*(a) Original image.*



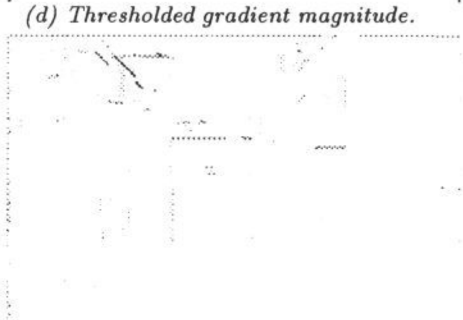
*(b) Magnitude of gradient.*



*(c) Orientation of gradient.*



*(d) Thresholded gradient magnitude.*



*(e) Thresholded gradient orientation.*

Figure 2: *Results from the Sobel operator.*

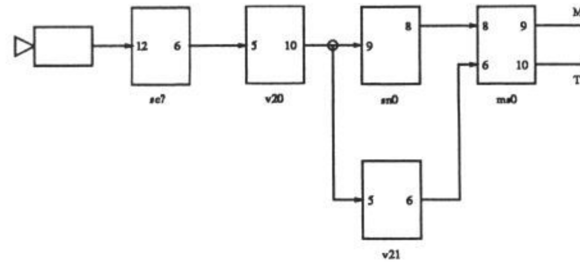



Figure 3: *MAXbus connections for LoG.*

$8 \times 8$ VFIR-II, the maximum usable $\sigma$ is $\sigma = 0.83$.

A pixel is a zerocrossing if it satisfies one of the following two conditions:

- The pixel's value is positive and an 8-neighbour's value is negative.
- The pixel's value is zero, one 8-neighbour's value is positive and the opposite 8-neighbour's value is negative.

The first condition can be tested for in a single pass through a SNAP board. The second cannot be tested for in one pass, although it can be using two SNAPs and a Max-SP. Neglecting zerocrossings which satisfy the second condition results in a slight loss of connectivity. Such a Laplacian of Gaussian can be implemented at frame rate as in Figure 3. Results are shown in Figure 4.

# CANNY

After the initial development of ad-hoc gradient-based operators, the question of optimal detectors was studied and several advanced edge detectors resulted. One of these was the Canny edge detector [2,3]. This involves convolving the input image with a Gaussian smoothing kernel and forming the gradient (or convolving the image with the two components of the gradient of a Gaussian) to form an image in which edges correspond to maxima. The edges in this image are isolated and thinned by non-maximal suppression, that is, all pixels whose gradient magnitude is not a local maxima in the direction of the gradient are not considered to be edgel candidates. Then thresholding with hysteresis is performed to threshold the edgels. This means that two thresholds are used. All pixels whose gradient magnitude is above the higher threhold are labelled edgels. All pixels whose gradient magnitude is above the lower threshold and which are connected to a pixel whose gradient magnitude is above the upper threshold, by a chain of connected pixels with gradient magnitudes above the lower threshold, are also labelled edgels. All pixels whose gradient magnitude is below the lower threshold are not labelled edgels.

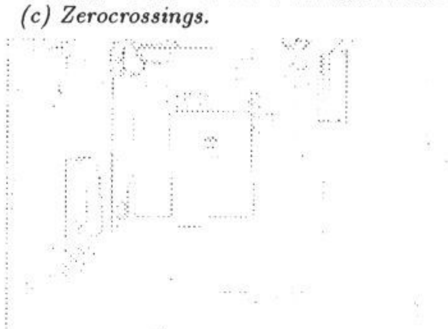
The maximum value of the Gaussian parameter $\sigma$ implementable in a VFIR-II is $\sigma = 1$. The results from this convolution are not significantly different from the output of the convolutions used in the Sobel operator.

Two simplifications are necessary to get this operating on a Datacube MaxVideo system:

*(a) Original image.*

*(b) Laplacian of Gaussian.*

*(c) Zerocrossings.*



*(d) Thresholded zerocrossings.*

Figure 4: *Results from the Laplacian of Gaussian operator.*

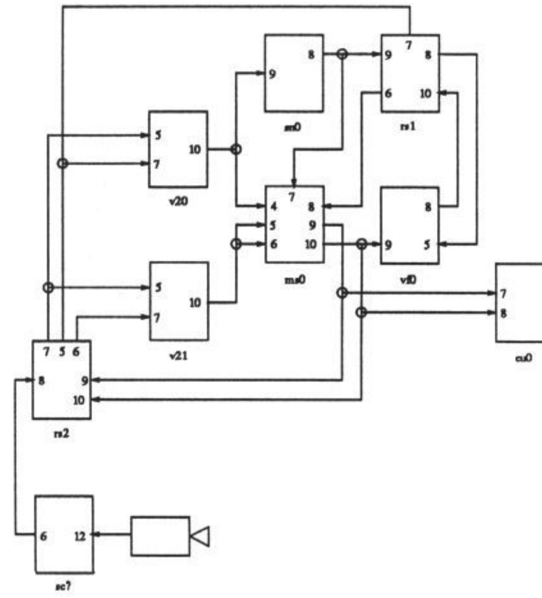


Figure 5: *MAXbus connections for Canny.*

1. In the non-maximal suppression stage, a comparison is made between a pixel's gradient magnitude and the magnitude values in the gradient direction. This requires interpolation between magnitude values at neighbouring pixels. Instead of this interpolation, the comparison is made with the 8-neighbour closest to the gradient direction. This simplification doesn't result in much loss of performance.

2. The thresholding with hysteresis phase can be thought of as following along edges and looking at the gradient magnitudes. This is done in a Datacube MaxVideo system by multiple passes through a SNAP. Each pass looks at a pixel's 8-neighbours and possibly adds it to an edge. That is, each pass only extends edges by one pixel at most. There is no quick way to test if any edges have been extended. Therefore it is necessary to apriori fix the number of passes and hence the maximum possible edge extension.

   This limitation could be overcome by incorporating the thresholding with hysteresis into the edge-listing phase on a board such as Euclid, the host, or a post-processor such as Kiwivision-II [7].

With these two restrictions and enough boards it would be possible to implement Canny at frame-rate. With the 7 boards in the Oxford AGV system, it is possible to process $256 \times 256$ images at 5 fps. This system is illustrated in Figure 5 and some results shown in Figure 6.

# SUMMARY

A brief review of three edge detectors has been made and it is shown that they all, with some minor restrictions, can be implemented on Datacube MaxVideo image-processing systems. As is normal with these systems, there is a trade-off between the size of the system and the throughput achievable. All three – Sobel, Laplacian of Gaussian, Canny – can be implemented at frame-rate.

# References


[1] **D. H. Ballard** and **C. M. Brown**, *Computer Vision*, Prentice-Hall, 1982.

[2] **J. F. Canny**, "Finding Edges and Lines in Images", MIT AI Lab Technical Report No. 720, June 1983.

[3] **J. F. Canny**, "A Computational Approach to Edge Detection", *IEEE Transactions on Pattern Recognition and Machine Intelligence*, Vol. 8, No. 6, November 1986, pp 679–698.

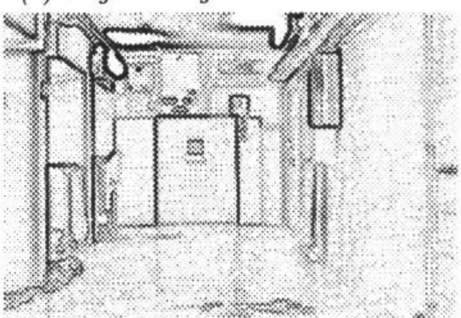
[4] Datacube Inc., Peabody, MA, USA.

[5] **D. Marr**, *Vision*, Freeman, 1982.

[6] **W. K. Pratt**, *Digital Image Processing*, Wiley, 1978.

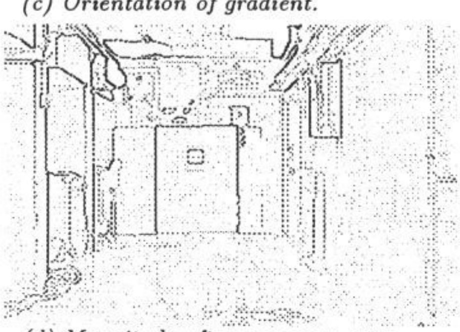
[7] **R. J. Valkenburg** and **C. C. Bowman**, "Kiwivision II – a hybrid pipelined/multitransputer architecture for machine vision", DSIR/AIDD Technical Report, 1989.


*(a) Original image.*

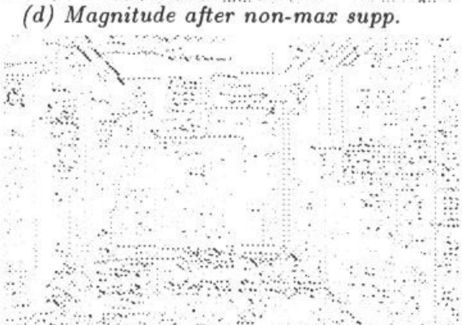


*(b) Magnitude of gradient.*

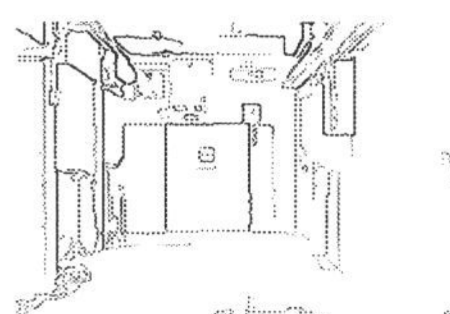
*(c) Orientation of gradient.*



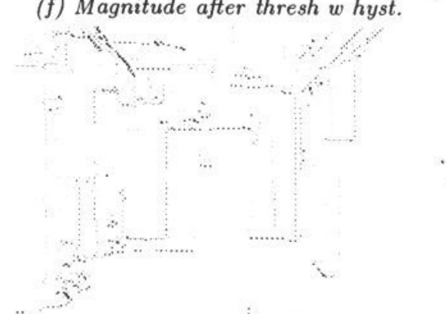
*(d) Magnitude after non-max supp.*



*(e) Orientation after non-max supp.*



*(f) Magnitude after thresh w hyst.*



*(g) Orientation after thresh w hyst.*

Figure 6: *Results from the Canny operator.*