VLSI Architecture for Real-Time Dominant Point Extraction

Stéphane Dallaire, Marc Tremblay and Denis Poussart

Computer Vision and Systems Laboratory, Department of Electrical and Computer Engineering
Laval University, Québec, Canada, G1K 7P4, e-mail: stefdal@gel.ulaval.ca

Abstract

This paper presents a special-purpose VLSI architecture for dominant point extraction along 2-D contours. It is designed to be integrated as part of a machine vision system with real-time edge-extraction and edge-tracking capabilities [1] in order to allow the creation of a high-level database representation of the observed scene. Such dominant points carry useful information for shape analysis and pattern recognition applications since they represent a local shape property and segment object contours into piecewise linear segments and circular arcs. The proposed architecture implements an algorithm based on the Curvature Primal Sketch [2]. It consists of a set of 1-D systolic FIR filters performing a multiresolution analysis of the scene's object contours, a set of finite-state-machines extracting zero-crossings and extrema of the filtered data, and a set of scale-space integration cells combining the accurate locations provided by the finest filters with the noise rejection properties of the coarsest ones in order to reliably extract relevant dominant points with accurate localization. The overall architecture has been successfully simulated using real edge images. Some of these results will be presented and discussed.

1: Introduction

Significant changes in curvature have long been considered important features describing object shapes [3]. Such dominant points have been used extensively in shape analysis and pattern recognition applications. Their extraction can greatly simplify the analysis of images since they drastically reduce the amount of data to process, while at the same time preserving important information about object shapes.

To date, many dominant point extraction algorithms have been published. These can be classified as either grey-level-based [4][5] or boundary-based [2][6][7][8]. Grey-level-based approaches detect dominant points directly from the grey-level intensity image while boundary-based approaches detect dominant points from a list of connected edge elements that have been previously extracted from a grey-level image. Boundary-based algorithms are more widely used than grey-level ones because they are easy to implement and because edge detection is part of most machine vision systems. In order to be reliable for object recognition applications, a dominant point extraction scheme should be invariant to the size and orientation of objects, provide accurate localization, and be insensitive to noise. However, most of the algorithms which aim at meeting these requirements are computationally intensive, which restricts their use in real-time applications. They usually involve low-level, repetitive, highly structured processing, like filtering, which can be efficiently implemented using a special-purpose architecture.

In this paper, we present a VLSI architecture which aims at extracting dominant points in real-time. It will be integrated into a machine vision system with real-time edge-extraction and edge-tracking capabilities that has been previously developed in our laboratory [1]. Dominant points will be used as break points in the computation of a piecewise linear and circular representation of object contours. The use of a boundary-based approach is the preferred option since it will greatly benefit from the embedded edge-extraction and edge-tracking capabilities. The proposed architecture implements an algorithm based on the Curvature Primal Sketch [2]. It consists of a set of 1-D systolic FIR filters performing a multiresolution analysis of the scene's object contours, a set of finite-state-machines extracting zero-crossings and extrema of the filtered data, and a set of scale-space integration cells combining the accurate locations provided by the finest filters with the noise rejection properties of the coarsest ones in order to reliably extract relevant dominant points with accurate localization. To our knowledge, this VLSI architecture is the first hardware implementation of a dominant point extraction algorithm.

The algorithm used to extract dominant points along 2-D contours is presented in Section 2 while its VLSI implementation is extensively discussed in Section 3. The overall VLSI architecture is presented first, followed by a detailed discussion of its main components: multiresolution Gaussian filtering, extrema and zero-crossing detection, and scale-space integration. Some very promising simulation results obtained from real edge images are presented and discussed in Section 4. Finally, Section 5 presents the conclusions.

2: Algorithm

The proposed algorithm to extract dominant points is inspired by the Curvature Primal Sketch (CPS) [2]. In the past, such a representation has provided good results and its process flow, though relatively computationally intensive, is particularly well-suited for hardware implementations. Designed to represent significant curvature changes along object contours, the CPS consists of various symbolic descriptions at multiple scales, such as basic primitives, including corners and smooth joints, as well as compound primitives such as ends, cranks, bumps, and dents, which are built from basic primitives. The representation is generated by convolving different size Gaussian kernels with the path-based contour's orientation function and tracking locations of orientation and curvature discontinuities. Boundary symbolic features are detected at each different scale, resulting in a multiple-scale interpretation of object contours.

In the case of the implemented dominant point extraction algorithm, only the basic primitives have been considered since they are sufficient to segment object contours into piecewise linear segments and circular arcs. These basic primitives, which are illustrated in Figure 1 (a), are the corner and the smooth joint, corresponding respectively to an orientation (θ) discontinuity and a curvature (κ) discontinuity.

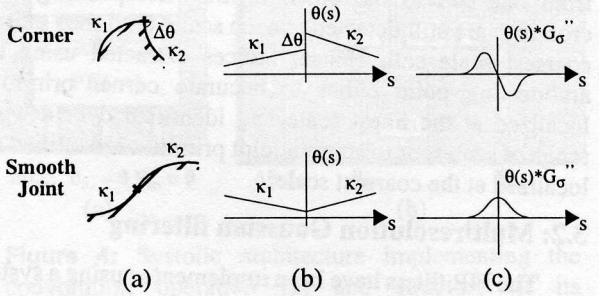

Figure 1: Dominant point primitives considered in the implemented algorithm (a) with their θ-s parametric representation (b) and the result of the convolution with the second derivative of a Gaussian function (c).

The main steps of the proposed algorithm are depicted in Figure 2. Starting from a raw list of connected edge elements which has been previously extracted using edge-extraction and edge-tracking processes, the initial operation consists of parametrizing the orientation of the tangent to the contour as a function of its arc length. From this point, corners and smooth joints, whose parametric representations are shown in Figure 1 (b), are extracted using different procedures. Relevant corner primitives are detected using a multiresolution analysis which is carried out by convolving the parametric representation with a set of Gaussian smoothing functions. As shown in Figure 1 (c), a corner primitive convolved with the second derivative of a Gaussian function is characterized by a zero-crossing flanked by two extrema of opposite polarity. Thus, corner locations are identified at each scale by detecting their corresponding zero-crossings. The resulting multiresolution streams are combined into a unified representation using a process of scale-space integration. As originally proposed by Witkins [9], this integration combines the accurate locations provided by the finest filters with the noise rejection properties of the coarsest ones in order to reliably extract relevant corners with accurate localization. It consists of identifying significant events at a coarse scale and localizing them accurately by tracking their occurrence in the scale-space domain, from the coarsest scale to the finest.

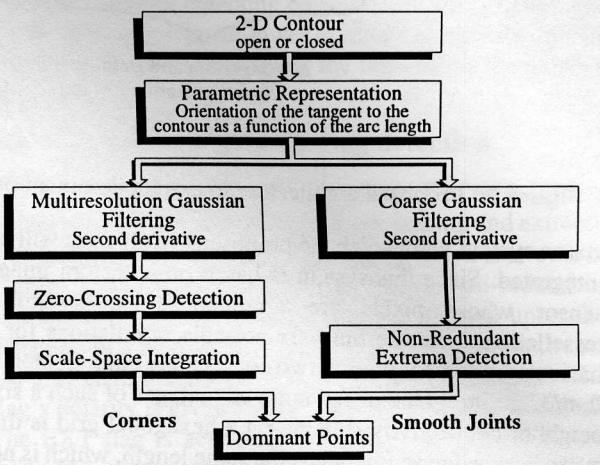

Figure 2: Main steps of the proposed dominant point extraction algorithm.

As shown in Figure 1 (c), smooth joints are characterized by so-called "non-redundant" extrema of the convolved representation (redundant extrema are those associated with zero-crossings). However, these non-redundant extrema are very difficult to detect in practice because they are usually very weak in magnitude and consequently, very sensitive to noise. As a result, smooth joints can only be reliably identified and localized using a coarse filter. Their reliable extraction requires a robust estimation of curvature, which can only be achieved in practice using a coarse filter.

3: VLSI implementation

3.1: Architecture

The VLSI architecture implementing the previously described algorithm is illustrated in Figure 3. It receives as input a list of edge orientations (i.e. a chain-code) describing the contour to be processed. Such a chain-code is extracted from the scene using the embedded edge-extraction and edge-tracking capabilities of the machine

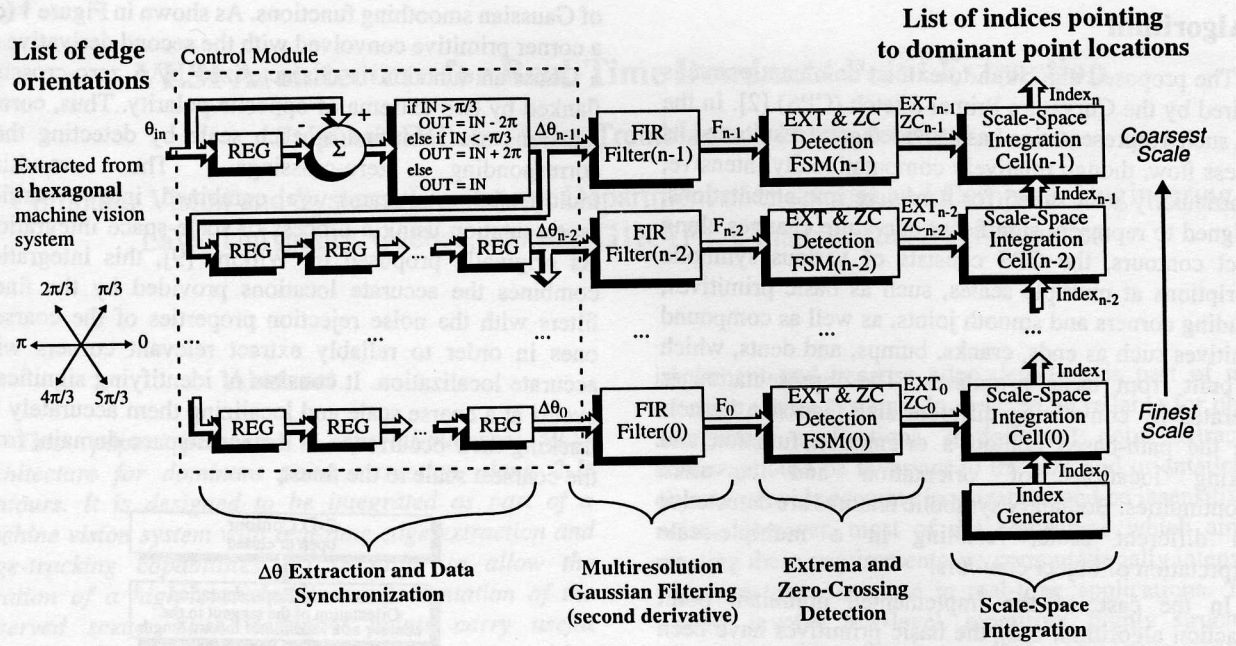

Figure 3: Proposed architecture to extract dominant points along 2-D contours.

vision system with which the proposed architecture will be integrated. Since this system is based on a custom image sensor whose pixels are arranged in a hexagonal tessellation, there are only six possible orientations for a basic edge segment joining two consecutive edge elements: 0, $\pi/3$, ..., $5\pi/3$. One of the main advantages of such a six-neighbor connectivity scheme on a hexagonal grid is that every basic edge segment has the same length, which is not the case with the conventional eight-neighbor connectivity scheme on a Cartesian grid.

Prior to being smoothed using a set of FIR Gaussian filters, the input list of edge orientations θ_{in} is first processed by a control module which computes its differential chain-code $\Delta\theta$ i.e the variation of orientation between consecutive edge elements, then adds delays in the distinct $\Delta\theta_i$ paths (where i is the scale number ranging from 0 to n-1) that are each connected to an individual FIR filter. The computation of the differential chain-code is aimed to prevent false zero-crossings of the filter outputs caused by 0-2π wrap-around effects whereas delay insertion is aimed to compensate for the scale-dependent pipeline latencies of the FIR filters in order for the data flow to be synchronized at the inputs of the scale-space integration cells.

The multiresolution analysis required for corner detection is computed in parallel using a set of n FIR filters. The output of each filter feeds a Finite-State-Machine (FSM) which detects extrema and zero-crossing locations. Extracted at each scale, this information is used in the scale-space integration process. However, only extrema detected at the coarsest scale are useful since, as mentioned in Section 2, smooth joints can only be reliably identified and localized at the coarsest scale. As shown in Figure 3, the scale-space integration process is implemented using a linear array of cellular automata. Indices pointing to accurate corner locations extracted using the finest filter enter the connected scale-space integration cell, then flow from one cell to the other, if the corresponding zero-crossings are still detected at each scale, until they exit the coarsest scale cell. Hence, indices extracted using this architecture point either to accurate corner primitives localized at the finest scale and identified over a whole range of scales, or to smooth joint primitives identified and localized at the coarsest scale.

3.2: Multiresolution Gaussian filtering

The FIR filters have been implemented using a systolic architecture. Recall that a systolic architecture consists of a set of interconnected processing elements (PEs) through which inputs and outputs flow in a rhythmic fashion, like blood circulating to and from the heart [10]. This concurrent use of many locally interconnected PEs allows the computation throughput to be increased up to the available I/O bandwidth, thus balancing computation with I/O.

A possible architecture implementing the convolution operation is shown in Figure 4 (a). In this architecture, the inputs $\Delta\theta$ flow systolically in one direction while the reconstructed orientations θ and the results F flow in the opposite direction. For the discrete convolution operation defined as:

$$F_{out}(s) = C_{-w}\theta(s-w) + C_{-w+1}\theta(s-w+1) + \ldots + C_{w}\theta(s+w) \quad (1)$$

where w is the width of the filter (there are 2w+1 PEs) and

C_j the coefficient of the j^{th} PE, each PE performs the following operations:

$$\Delta\theta_{out} = \Delta\theta_{in} \quad (2)$$

$$\theta_{out} = \theta_{in} + \Delta\theta_{in} \quad (3)$$

$$F_{out} = F_{in} + C\theta_{out} \quad (4)$$

Thus, at each clock cycle, each PE (whose structure is shown in Figure 4 (b)) propagates its input $\Delta\theta_{in}$ to the previous PE, updates the reconstructed orientation θ_{out} from its input $\Delta\theta_{in}$ and the reconstructed orientation θ_{in} of the previous PE, and updates the partial result F_{out} from the partial result F_{in} of the previous PE, the reconstructed orientation θ_{out}, and the filter coefficient C stored locally in a register. The computation of the reconstructed orientation θ_{out} consists of accumulating the variations of orientation $\Delta\theta_{in}$ using a range larger than the [0, 2π] interval in order to avoid false zero-crossings caused by 0-2π discontinuities.

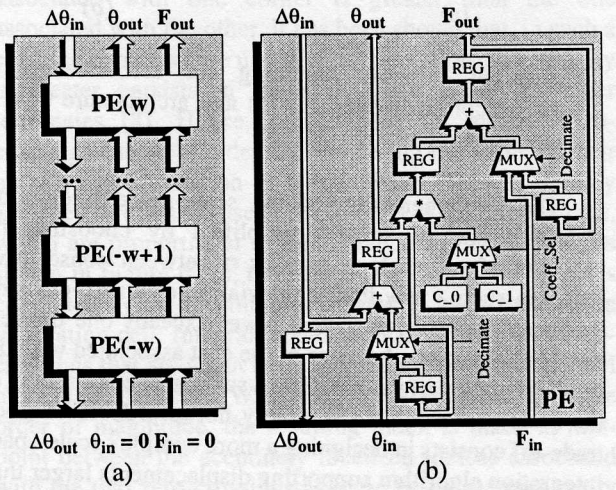

Figure 4: Systolic architecture implementing the convolution operation (a) and structure of its processing elements (PEs) (b).

The main disadvantage of this architecture is that, at any time, only 50% of the PEs are performing useful computation. It can be shown that the inputs $\Delta\theta$ are required to enter the cell PE(w) at a rate of one datum every two clock cycles [10]. Consequently, the outputs F_{out} are generated at the same rate. However, an efficiency of 100% can be achieved by interleaving two convolution computations using two coefficient registers (C_0 and C_1) and a multiplexer (Coeff_Sel) in each PE.

It is a fact that the implementation of Gaussian filters requires very wide convolution kernels, especially those having a large standard deviation σ. Using the widely used discrete implementation rule which consists of sampling continuous Gaussian functions and truncating them to the smallest integer larger than or equal to 3σ [11], it turns out that several PEs would be required to implement coarse filters. For example, 65 PEs would be required to implement a Gaussian filter having a standard deviation equal to 10.7. In order to reduce the cost associated with the implementation of coarse filters, we have implemented a special feature, similar to a decimation process, which allows wider filters (2(2w+1), 4(2w+1), ...) to be approximated using the same number of PEs (2w+1). This consists of feedback loops in the $F_{out} \rightarrow F_{in}$ and $\theta_{out} \rightarrow \theta_{in}$ paths of each PE, allowing d consecutive input samples to be multiplied by the same coefficient, where d is the decimation step. That is, instead of filtering the full resolution input stream using a standard deviation σ for the Gaussian kernel, this feature is equivalent to replace each d consecutive sample by the sum of its d neighbors and filter the resulting 1/d resolution list using a standard deviation σ/d for the Gaussian. Such a technique has been used elsewhere [12] for detecting edges in intensity images and has produced, at a much lower computational cost, results similar to those obtained using the full-resolution filtering approach.

3.3: Extrema and zero-crossing detection

As illustrated in Figure 3, the output F of each filter is routed to an FSM which extracts zero-crossing and extrema information. An example of such an extraction is shown in Figure 5 where a zero-crossing and a non-redundant extrema are detected. The *ZC* flag is activated upon the detection of a zero-crossing whose two associated extrema are greater in magnitude than an arbitrary threshold. The flag's polarity is given by the value of the ZC_POL signal. The EXT flag is activated upon the detection of a non-redundant extrema whose magnitude is greater than an arbitrary threshold. Its polarity is given by the value of the EXT_POL signal. Note that the EXT flag is not activated upon the detection of the extrema associated with the zero-crossing. The remaining VALID and VALID_POL signals are used in the scale-space integration process. Activated over a region of arbitrary width surrounding a zero-crossing detected at scale σ_i, the VALID signal feeds the previous (i-1) scale-space integration cell in order to define a region over which one or many zero-crossings detected at

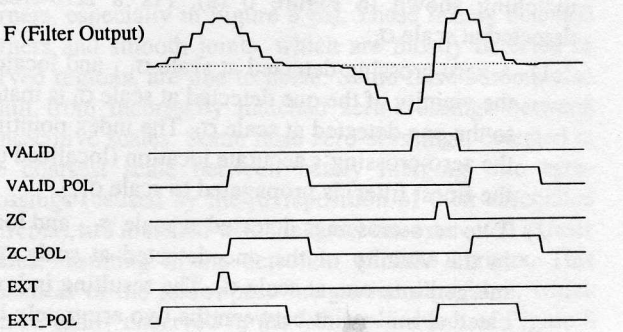

Figure 5: Zero crossing and extrema information extracted from the output F of each filter.

scale σ_{i-1} are combined into only one, then matched to the one detected at scale σ_i. The width of this region, which defines the maximum displacement of a zero-crossing between two consecutive scales, is determined according to the variation of the scale parameter σ between consecutive scales. The VALID_POL signal, whose value is equal to that of the ZC_POL signal, is also routed to the previous scale-space integration cell in order to ensure consistency in the polarity of the matched zero-crossings.

3.4: Scale-space integration

The scale-space integration process performed on multiresolution zero-crossing streams is meant to combine the accurate locations provided by the finest filters with the noise rejection properties of the coarsest ones in order to reliably extract relevant corners with accurate localization. This consists in tracking the occurrence of corresponding zero-crossings between consecutive scales and propagating their index from the finest scale, where they are accurately localized, to the coarsest, where the noise has been attenuated. This fine to coarse scale propagation of indices pointing to accurate corner locations is equivalent to the coarse to fine tracking proposed by Witkins [9]. Instead of first identifying significant events at the coarsest scale and accurately localizing them by tracking their occurrence in the scale-space domain, from the coarsest scale to the finest, we first accurately localize all events at the finest scale and discriminate the relevant ones from the noisy ones by tracking their occurrence and propagating their index, from the finest scale to the coarsest.

The use of the Gaussian kernel as a smoothing function to generate multiple-scale representations has been studied extensively over the past [13]. It has been shown that the Gaussian kernel is unique in having the property of not creating new features as the scale parameter σ increases. That is, while moving from fine to coarse scales, some existing zero-crossings persist; some disappear; some merge into a single one, but no new zero-crossing can be created. In order to correctly track the behavior of zero-crossings in the scale-space, the scale-space integration algorithm must support the four types of matching shown in Figure 6 (a). For a zero-crossing detected at scale σ_i:

1) A zero-crossing detected at scale σ_{i-1} and located in the vicinity of the one detected at scale σ_i is matched to the one detected at scale σ_i. The index pointing to the zero-crossing's accurate location (localized using the finest filter) is propagated to scale σ_{i+1}.
2) Two zero-crossings detected at scale σ_{i-1} and located in the vicinity of the one detected at scale σ_i are merged into one at scale σ_i. The resulting index (located at mid-point between the two accurately localized indices associated with the two zero-crossings) is propagated to scale σ_{i+1}.
3) A zero-crossing detected at scale σ_{i-1} and located in the vicinity of the one detected at scale σ_i is matched to the one detected at scale σ_i. However, the index pointing to the zero-crossing's accurate location is not propagated to scale σ_{i+1} since no corresponding zero-crossing is detected at scale σ_{i+1}.
4) No zero-crossing is detected at scale σ_{i-1} in the vicinity of the one detected at scale σ_i. Although not possible in theory, such a situation can occur in practice due to noise and arbitrary thresholds of the system.

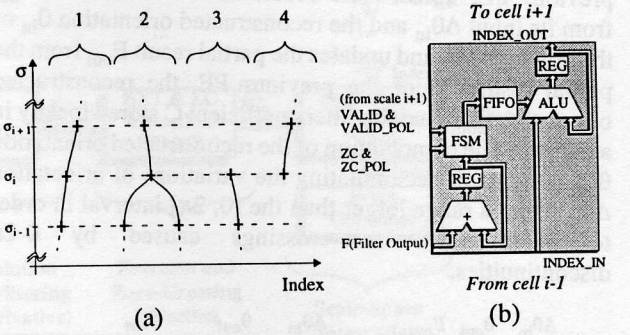

Figure 6: Possible matching for zero-crossings between consecutive scales (a) and architecture of a scale-space integration cell (b).

The implementation of the scale-space integration algorithm can be greatly simplified by choosing the variation of the scale parameter σ between consecutive scales in such a way that the deviation of a zero-crossing (between consecutive scales) never exceeds one discrete unit [14]. However, because of the cost associated with the implementation of the FIR filters, such an over-sampling of the scale-space domain is simply not affordable. A good trade-off consists in designing a more complex scale-space integration algorithm supporting displacements larger than one discrete unit between consecutive scales in order to use fewer filters to sample the same range of scales. This is achieved mainly through the use of the VALID signal which defines the maximum deviation of a zero-crossing between two consecutive scales.

The architecture of the scale-space integration cell is depicted in Figure 6 (b). From the VALID and VALID_POL signals generated at scale σ_{i+1} and the ZC and ZC_POL signals associated with the current scale σ_i, an FSM decodes, for each zero-crossing, the type of matching between the current and the next scale (case 1, 2, 3, or 4 in Figure 6 (a)), then stores this information into a First In First Out (FIFO) memory. Once the index pointing to the accurate location of a zero-crossing is available at the input INDEX_IN, the corresponding matching information is read from the FIFO in order to control a simple Arithmetic Logic Unit (ALU) computing the index of the resulting zero-crossing. Depending on the type of matching decoded by the FSM, the ALU may directly passe the input index to its output, store the index in the temporary register

without propagating it to the next cell in order to merge it with the next index, or compute the mean between the index stored in the temporary register and the one available at the input. The first operation corresponds to a simple propagation of the index from one scale to the other (case 1) while the following two correspond to a merge of two zero-crossings into one (case 2). Furthermore, some control signals not shown in the figure can impede the propagation of an index when no corresponding zero-crossing is detected at the next scale σ_{i+1} (case 3), or invalidate an index when no corresponding zero-crossing is detected at the previous scale σ_{i-1} (case 4).

Because the scale-space domain is discretely sampled, some zero-crossings can be incorrectly matched between consecutive scales. For example, two nearby zero-crossings can be merged together even if they don't in the continuous representation. This corresponds to two nearby corners of the same polarity, one of which is weaker than the other. That is to say, the orientation discontinuity associated with one corner is greater than the one associated with the other. It has been shown that, in such a case, the corner whose underlying orientation discontinuity is greater persists in the scale-space while the other terminates [8]. Hence, such incorrectly merged zero-crossings can be avoided by comparing the value of their underlying orientation discontinuities. This is mainly achieved through the use of the integral of the filter outputs, which are proportional to the variation of orientation. As shown in Figure 6 (b), the integral of each filter output is computed by an accumulator located in each scale-space integration cell. Integral values associated with two zero-crossings that are about to be merged are used by the FSM to validate the merge. When the two values are of the same order of magnitude, the resulting index is taken at mid-point between the accurately localized indices associated with the two zero-crossings as these zero-crossings really merge together in the continuous representation. However, when one integral value is greater than the other, the resulting index corresponds to the one pointing to the accurately localized corner having the greater integral value since that corner persists in the scale-space while the other terminates.

4: Simulation results

The proposed dominant point extraction architecture has been modeled using the VHDL language and simulated using real edge images as inputs. These images have been acquired from the hexagonal machine vision system with which the proposed architecture will be integrated. Thus, the results obtained are very representative of what will be obtained in practice, when the architecture will be implemented and integrated with that system. The simulated architecture consists of six 17-tap FIR filters, six extrema and zero-crossing extraction FSMs, and six scale-space integration cells. The filters implement a set of Gaussian smoothing functions whose standard deviation ranges from $\sigma = 2.0$ to $\sigma = 10.7$ in increments of $\sqrt{2}$. According to the 3σ truncation rule, this results in a set of filters having a width varying from 6 to 32. However, because the width of the filters has been physically limited to 8 (i.e. 17 taps), wider filters have been approximated using the decimation process described in Section 3.2.

A complete simulation example is presented in Figure 7 for a simple object shape containing both types of dominant point primitives. The orientation of the tangent to each basic edge segment as a function of the contour arc length is shown in Figure 7 (a) in conjunction with the results of the multiresolution filtering with the second derivative of Gaussian functions. It can be easily observed that fine scale filters provide many non-significant extrema and zero-crossings, especially in curved regions, while coarse scale filters only extract relevant features. However, although it cannot be easily observed in the figure, features detected using coarse filters are often not accurately localized, especially in regions where several nearby events are influencing each other (see the end of the parametric representation, for example). Figure 7 (b) shows the object contour overlaid with the resulting dominant points. As can be observed, all geometrically relevant dominant points have been accurately detected, especially the corners, which have been extracted and localized using the scale-space integration process. However, a false smooth joint has also been detected. This can be explained by the fact that, as discussed in Section 2, smooth joints are very sensitive to noise because their response to the second derivative of a Gaussian function is usually very weak in magnitude. This is particularly true in this case, as can be observed in the response to the coarsest filter (F_5), between s = 400 and s = 500.

Other simulation results obtained from more complex scenes are presented in Figure 8. In this case, the grey-level intensity images are shown along with the corresponding edge images overlaid with the detected dominant points. Again, it can be observed that all relevant dominant points have been accurately detected but a few false ones have also been extracted. Many of these false detections correspond to smooth joints but some also consist of corners, especially in Figure 8 (b). These falsely detected corners and smooth joints, which are mostly detected in curved regions, are due to noise. Some false corners also result from incorrectly matched zero-crossings between consecutive scales. Some false zero-crossings, detected at the coarsest scale between nearly merging true zero-crossings (caused by the juxtaposition of their associated extrema), are matched with the noisy ones detected at finer scales, resulting in the detection of false corners. This weakness of the scale-space integration algorithm, which can be easily observed in the left-hand object of Figure 8 (d), is attributed to the discrete sampling of the scale-space domain.

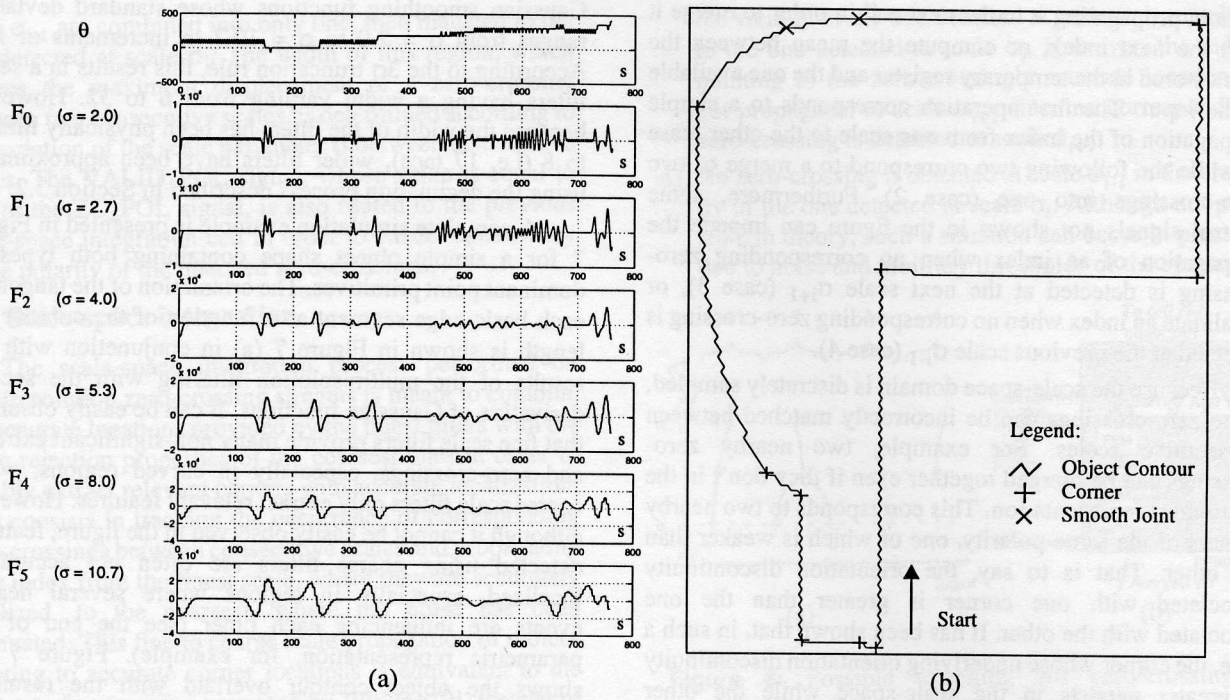

Figure 7: Simulation example showing the multiresolution Gaussian filtering of the contour's θ-s parametric representation (a) and the resulting dominant points (b).

5: Conclusion

In this paper, we presented a VLSI architecture aiming at extracting dominant points in real-time. The proposed architecture, whose process flow is inspired by that of the CPS, consists of a set of FIR Gaussian filters implemented using a systolic architecture, a set of extrema and zero-crossing detection FSMs, and a set of scale-space integration cells.

The simulation results obtained from the VHDL model of the proposed architecture are very promising but not perfect. In the examples that have been presented, all geometrically relevant dominant points have been accurately detected (no miss) with only few false detections. It has been observed that many of these false detections consisted of smooth joints since their filtered response is very sensitive to noise. Even though these false detections could be probably eliminated by properly adjusting the thresholds of the system, we believe it is preferable, from a practical point of view, to ensure there is no missed dominant point to the detriment of few false detections since the amount of processing required to discard false detections is usually less than that require to detect missed dominant points. The detection of a missed dominant point requires another parsing of the raw list of edge elements while a false detection can be discarded, once the scene database is created, using a simple post-processing performed on the parameters describing the contour primitives (linear segments and circular arcs) fitted between consecutive dominant points.

The integration of this architecture to the custom machine vision system developed in our laboratory will accelerate the image understanding process since it will be able to parse raw lists of edge elements at a rate of one datum every two clock cycles, with a latency of the order of 16w clock cycles, where w is the width of the filters (w = 8). Assuming a reasonable clock frequency, the proposed dominant point extraction architecture will be much faster than any software implementation of a similar algorithm.

Regarding the hardware implementation of the proposed architecture, a 20k-gate VLSI ASIC integrating a 17-tap FIR filter supporting two interleaved convolution computations, two extrema and zero-crossing detection FSMs, and two scale-space integration cells has been designed, fabricated, and successfully tested. It has been implemented using Nortel's 0.8μ BiCMOS technology. The control module, which is presently under development, will be implemented using a Xilinx FPGA. An operating frequency of 20 Mhz is anticipated.

6: Acknowledgments

This work was made possible in part by support from the Institute for Robotic and Intelligent Systems of Canada and through grants from FCAR of Québec and NSERC of Canada. The Canadian Microelectronics Corporation (CMC) provided software, hardware, and fabrication support through the Nortel, Mitel, and Gennum foundries.

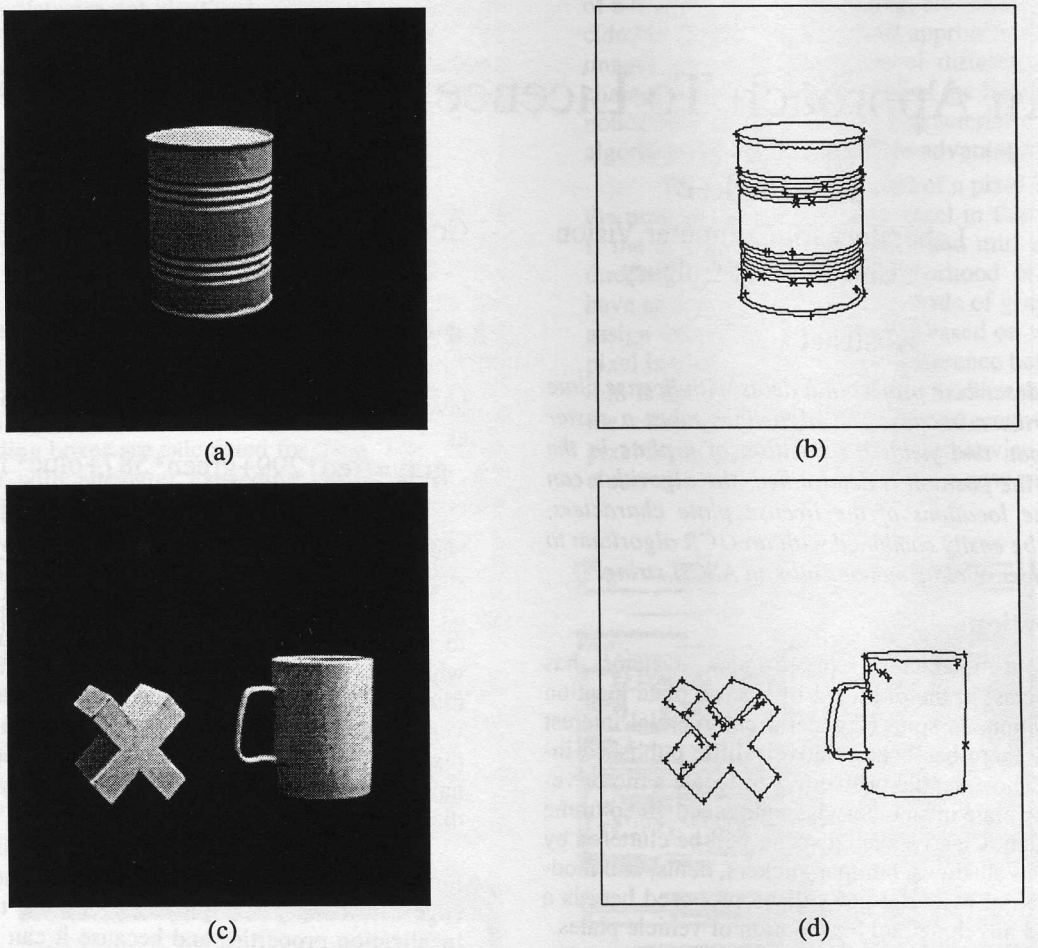

Figure 8: Other simulation results obtained from more complex scenes. Grey-level images are shown in (a) and (c) while the corresponding edge image overlaid with the detected dominant points are shown in (b) and (d).

7: References

[1] S. Dallaire, M. Tremblay and D. Poussart, "Mixed Signal VLSI Architecture for Real-Time Computer Vision", *to be published in Journal of Real-Time Imaging - special issue on Special-Purpose Architectures for Real-Time Imaging*, 1997.

[2] H. Asada and M. Brady, "The Curvature Primal Sketch", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 1, pp. 2-14, Jan. 1986.

[3] F. Attneave, "Some Informational Aspects of Visual Perception", *Psychol. Rev.*, Vol. 61, pp. 183-193, 1954.

[4] L. Kitchen and A. Rosenfeld, "Gray-Level Corner Detection", *Pattern Recognition Lett.*, Vol. 1, pp. 95-102, 1982.

[5] O. A. Zuniga and R. Haralick, "Corner Detection Using the Facet Model", *Proc. IEEE CVPR*, pp. 30-37, 1983.

[6] A. Rosenfeld and E. Johnston, "Angle Detection on Digital Curves", *IEEE Trans. on Computer*, Vol. C-22, pp. 875-878, Sept. 1973.

[7] C.-H. Teh and R. T. Chin, "On the Detection of Dominant Points on Digital Curves", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-11, No. 8, pp 859-872, Aug. 1989.

[8] A. Rattarangsi and R. T. Chin, "Scale-Based Detection of Corners of Planar Curves", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-14, No. 4, pp. 430-449, Apr. 1992.

[9] P. A. Witkins, "Scale-Space Filtering", *Proc. 8th Int. Joint Conf. on Artificial Intelligence*, Kalsruhe, West Germany, pp. 1019-1021, 1983.

[10] H. T. Kung, "Why Systolic Architectures?", *Computer Magazine*, pp. 37-46, Jan. 1982.

[11] D. G. Lowe, "Organization of Smooth Image Curves at Multiple-Scales", *Proc. 2th International Conference on Computer Vision*, Tampa, Florida, pp. 558-567, 1988.

[12] A. Huertas and G. Medioni, "Detection of Intensity Changes with Subpixel Accuracy Using Laplacian-Gaussian Masks", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 5, pp. 651-664, Sept. 1986.

[13] J. Badaud, A. P. Witkins, M. Baudin and R. O. Duda, "Uniqueness of the Gaussian Kernel for Scale-Space Filtering", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 1, pp. 26-33, Jan. 1984.

[14] F. Bergholm, "Edge Focusing", *IEEE trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, No.6, pp. 726-741, Nov. 1987.