Highly Concurrent VLSI Computing Structures for DCA

JÁN GLASA
Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9, 845 07 Bratislava
SLOVAK REPUBLIC

Abstract: In this paper highly concurrent pipelined computing structures based on a constrained digital contour smoothing are described. The smoothing minimizes the undersampling, digitizing and quantizing error and so it is able to improve the stability of invariants calculation. The word-level and bit-level systolic arrays for completely pipelined calculation of the constrained least-squares digital contour smoothing are described. They represent the heart of the concurrent pipelined VLSI computing structures which enable to calculate invariants, such as the length, the area, the moment invariants and to perform the smoothed contour encoding, in parallel. The computing structures achieve very high throughputs and can be realized on a single VLSI chip. They can be used for purposes of real-time digital curves analysis.

Key-Words: Digital curves analysis, digital contour smoothing, circulant Toeplitz matrices, bit-level systolic arrays, pipelined architectures, VLSI

1 Introduction
Digital curves analysis (DCA) plays an important role in modern image processing [10, 13, 21, 22]. It covers several sophisticated tools which can have a principal impact for successful analysis, representation and shape characterization of real objects investigated in digital images [1, 3]. Development of efficient (often real-time) techniques for different fundamental problems of DCA is then of a major interest [2, 6, 14, 19]. These reasons influence the development of VLSI systems utilizing parallelism, such as systolic arrays, pipelined architectures, string processors and wavefront arrays. Serious attention is also paid to introduction of new methods and approaches to achieve the improvement of accuracy of the shape invariants estimation.

For the purposes of DCA, several methods for a constrained digital contour smoothing have been suggested [18, 8]. They are based on the least-squares approximation to functions on equidistant subsets of points by orthogonal polynomials and on the approximation to functions by non-negative kernels. In [18, 9], bit-level systolic arrays for the constrained, so called feasible digital contour smoothing have been suggested. These arrays are completely pipelined on a bit-level and achieve very high throughputs.

In this paper, highly concurrent pipelined VLSI computing structures for purposes of real-time digital curves analysis are described.

2 Digital Contour Smoothing
Let a digital picture be a finite rectangular array represented by a finite square grid Ω, where the distance between neighbouring grid points of Ω is equal to 1. Let a digital contour be a planar simple closed digital curve Γ ≡ \( \bigcup_{j=0}^{N} S_j \), where \( S_j \) is a linear segment of the length 1 whose end points \((x_j, y_j)\) and \((x_{(j+1)\mod N}, y_{(j+1)\mod N})\) belong to grid points of Ω and for each \( j \) there are exactly two points \((x_{(j-1)\mod N}, y_{(j-1)\mod N})\) and \((x_{(j+1)\mod N}, y_{(j+1)\mod N})\) for which

\[
\begin{align*}
| x_j - x_{(j-1)\mod N} | + & | y_j - y_{(j-1)\mod N} | = 1 \\
| x_{(j+1)\mod N} - x_j | + & | y_{(j+1)\mod N} - y_j | = 1.
\end{align*}
\]

Let us denote

\[
X = \begin{pmatrix} x_0 & x_1 & \ldots & x_N \\ y_0 & y_1 & \ldots & y_N \end{pmatrix}^T.
\]

A digital contour smoothing in the least-squares sense is defined by a linear operator \( \frac{1}{c} C \) 
which is applied on \( X \) [18, 9],

\[
\frac{1}{e} C X = X',
\]

where \( C \) is an \((N+1) \times (N+1)\) circulant Toeplitz matrix and \( e \) is the sum of all elements of a row of \( C \). The coefficients of \( C \) are defined by the least-squares approximation (for more details see [8]).

These operators are position invariant [18]. It means that the smoothed contour \( \frac{1}{e} C X = X' \) has the same centroid as the original digital contour \( X \) itself.

A subset of these operators has the property that they are feasible [18]. A linear operator \( \frac{1}{e} C \) is feasible if

\[
| x_j - x_j' | < \frac{1}{2}, \quad | y_j - y_j' | < \frac{1}{2}
\]

for all \( j \), where \( x_j, y_j \) are elements of \( X \) and \( x_j', y_j' \) are elements of \( X' \). According to this definition the feasible operator is defined by the constrained least-squares smoothing with the constraint defined by (2), i.e., it generates points which lie in the interior of the corridor

\[
\bigcup_{j=0}^{N} \{(u, v) \in R^2 : (| x_j - u | \leq \frac{1}{2}) \land (| y_j - v | \leq \frac{1}{2})\}.
\]

It has been shown that the operator \( \frac{1}{e} C \) defined by polynomials of the third degree and by seven points in the least-squares sense has the form [18]

\[
\frac{1}{e} C = \begin{pmatrix}
7 & 6 & 3 & -2 & -2 & 3 & 6 \\
6 & 7 & 6 & 3 & -2 & -2 & 3 \\
3 & 6 & 7 & 6 & 3 & -2 & -2 \\
-2 & 3 & 6 & 7 & 6 & 3 & -2 \\
-2 & 3 & 6 & 7 & 6 & 3 & -2 \\
3 & -2 & 3 & 6 & 7 & 6 & 3 \\
6 & 3 & -2 & -2 & 3 & 6 & 7
\end{pmatrix},
\]

where only non-zero elements of \( C \) are registered. For this operator it holds that [18]

\[
| x_j - x_j' | < \frac{10}{27}, \quad | y_j - y_j' | < \frac{10}{27}.
\]

The value \( \frac{10}{27} \) is the greatest value less than \( \frac{1}{2} \) among all values defined by feasible least-squares smoothing operators and so it performs in some sense ”maximal” feasible smoothing.

The digital contour smoothing by (3) can be described by a set of direction vectors related to (3). They are defined by

\[
\begin{align*}
ca_{ix} &= cx_i' - cx_{(i+1)\mod N} \\
ca_{iy} &= cy_i' - cy_{(i+1)\mod N}
\end{align*}
\]

where the values of \( x_i', y_i \) and \( x_{(i+1)\mod N}, y_{(i+1)\mod N} \) correspond to the coordinates of mid points of two subsequent digital arcs of the length of 7 points.

\[
\begin{align*}
y_{i+1}' & \quad a_y \quad a_i \\
x_{i+1}' & \quad a_x \\
x_i' & \quad a_{ix} \quad a_{iy}
\end{align*}
\]

Fig. 1. Notation to (5).

\[
\begin{pmatrix}
(15,0),(19,0),(21,0),(25,0),(14,1),(20,1),(22,1),(17,2), \\
(19,2),(23,2),(12,3),(16,3),(18,3),(22,3),(11,4),(17,4), \\
(25,4),(14,5),(16,5),(20,5),(9,6),(13,6),(15,6),(19,6), \\
(14,7),(18,7),(11,8),(13,8),(17,8),(12,9),(16,9),(11,10), \\
(15,10),(14,11),(18,11),(13,12).
\end{pmatrix}
\]

In order to identify all possible direction vectors related to (3), it is necessary to analyze all possibilities of digital arcs of the length of 8 points which begin in \((0,0)\). The corresponding direction vectors (5) in the first octand are as follows [18]:

\[
(15,0),(19,0),(21,0),(25,0),(14,1),(20,1),(22,1),(17,2), \\
(19,2),(23,2),(12,3),(16,3),(18,3),(22,3),(11,4),(17,4), \\
(25,4),(14,5),(16,5),(20,5),(9,6),(13,6),(15,6),(19,6), \\
(14,7),(18,7),(11,8),(13,8),(17,8),(12,9),(16,9),(11,10), \\
(15,10),(14,11),(18,11),(13,12).
\]

Fig. 2. Direction vectors (5) related to (3).

The vectors in the second octand are mirror symmetric according to the diagonal. According to this, it holds that \( ca_{ix} \in \langle -25, 25 \rangle \), \( ca_{iy} \in \langle -25, 25 \rangle \). The total number of all direction
vectors (5) is 272 and they can be represented by the scheme shown on Fig. 2 [8]. The unit distance of this array is $\sqrt{\frac{217}{24}}$. The minimal vector length has the value $\sqrt{\frac{217}{14}}$ and the maximal vector length has the value $\sqrt{\frac{217}{24}}$.

3 Applications

The linear operator (3) allows to smooth digital contours in the least-squares sense preserving a contour shape and to minimize the undersampling, digitizing and quantizing error. It is able to improve the stability of estimation of local and global invariants which are related to the continuous pre-digitized contours of original objects investigated. Note that the digital contours are assumed to correspond to the continuous contours (i.e., a suitable noise supression has been performed [20, 11, 15, 16]).

The direction vectors of (3) can be ordered according to their increasing angle with the $x$-axis, it holds $\alpha_i = 0 < \beta_i < \gamma_i$, and according to their decreasing norms if there are more vectors which have the same angle. Since $\alpha_i \leq \beta_i \leq \gamma_i$, they can be numbered by the numbers $i \in [0, 271]$ according to their ordering, a simple coding scheme for encoding the smoothed curve can be obtained (see below).

The area of the closed polygon $P$ bounded by the smoothed contour is defined by

$$P_1^{\text{c}} \text{CX} = \frac{1}{c} \sum_{i=0}^{N} c_{ai} s_i + c_{aiy} c_{aix},$$

where $s_i := s_i + 2c_{aiy}$; $s_0 := 0$. (6)

The length of the smoothed contour is defined by

$$L_1^{\text{c}} \text{CX} = \frac{1}{c} \sum_{i=0}^{N} (c_{aix})^2 + (c_{aiy})^2 \frac{1}{4}.$$  

For the first-order moments according to the $x$- and $y$-axis, it holds

$$M_1^{\text{c}} \text{CY} = \frac{1}{c} \sum_{i=0}^{N} c_{aiy}((cy_i')^2 + c_{aiy}(cy_i' + \frac{1}{2}c_{aiy})),$$

$$M_1^{\text{c}} \text{CY} = \frac{1}{c} \sum_{i=0}^{N} -c_{aiy}((cx_i')^2 + c_{aiy}(cx_i' + \frac{1}{2}c_{aiy})).$$

The center of gravity $(x_T, y_T)$ is defined by

$$x_T = \frac{M_1^{\text{c}} \text{CY}}{P_1^{\text{c}} \text{CX}}, y_T = \frac{M_1^{\text{c}} \text{CY}}{P_1^{\text{c}} \text{CX}}.$$

The second-order moments according to the $x$- and $y$-axis are defined by

$$M_2^{\text{c}} \text{CX} = \frac{1}{c} \sum_{i=0}^{N} c_{ai} ((cy_i')^3 + \frac{1}{2} c_{aiy} (cy_i')^2 + (c_{aiy})^2 (cy_i' + \frac{1}{3} c_{aiy})),$$

$$M_2^{\text{c}} \text{CX} = \frac{1}{c} \sum_{i=0}^{N} -c_{aiy} ((cx_i')^3 + \frac{1}{2} c_{aiy} (cx_i')^2 + (c_{aiy})^2 (cx_i' + \frac{1}{3} c_{aiy})).$$

The second-order moments according to the center of gravity, we have

$$M_2^{\text{c}} \text{CX} = \frac{1}{c} \sum_{i=0}^{N} c_{ai} ((cy_i')^3 + \frac{1}{2} c_{aiy} (cy_i')^2 + (c_{aiy})^2 (cy_i' + \frac{1}{3} c_{aiy})),$$

$$M_2^{\text{c}} \text{CX} = \frac{1}{c} \sum_{i=0}^{N} -c_{aiy} ((cx_i')^3 + \frac{1}{2} c_{aiy} (cx_i')^2 + (c_{aiy})^2 (cx_i' + \frac{1}{3} c_{aiy})).$$

For the smoothed central second order moments according to the center of gravity, we have

$$M_2^{\text{c}} \text{CX} = \frac{1}{c} \sum_{i=0}^{N} c_{ai} ((cy_i')^3 + \frac{1}{2} c_{aiy} (cy_i')^2 + (c_{aiy})^2 (cy_i' + \frac{1}{3} c_{aiy})),$$

$$M_2^{\text{c}} \text{CX} = \frac{1}{c} \sum_{i=0}^{N} -c_{aiy} ((cx_i')^3 + \frac{1}{2} c_{aiy} (cx_i')^2 + (c_{aiy})^2 (cx_i' + \frac{1}{3} c_{aiy})).$$

where $X_T = \begin{pmatrix} x_T & x_T & ... & x_T \\ y_T & y_T & ... & y_T \end{pmatrix}^T$.

Note that it depends on the concrete application when the normalization by $c$ will be performed. It can be done after the invariant has been calculated from the array $\text{CX}$.

4 Systolic arrays

The matrix-matrix multiplication $\text{CX}$ can be represented by two circulant convolutions

$$cx_i' = \sum_{j=-3}^{3} c_j x((i-j) \mod N),$$

$$cy_i' = \sum_{j=-3}^{3} c_j y((i-j) \mod N), i = 0, 1, ..., N$$

where $c_{-3} = c_3 = -2$, $c_{-2} = c_2 = 3$, $c_{-1} = c_1 = 6$, and $c_0 = 7$ denote the non-zero elements of the matrix (3). The circulant convolution (7) can be efficiently realized in serial as well as in parallel pipelined manner.

In [18], a linear systolic array for the calculation of (7) has been suggested. It has simple
cells (see Fig. 3a) separated by delay elements controlled by a common clock. The numbers of delay elements at z- and x-connections and the necessary input operations are shown on Fig. 3b. Since the primitive operations performed are operations on the word level, the array is qualified as a word-level systolic array with the throughputs (N+13)T, where T is the execution time of a single cell function.

In [18, 7, 9], a bit-level pipelined implementation of (7) on simple regular systolic structures has been investigated. The effective bit-level decomposition of (7) and represents a concrete bit-level implementation strategy [8, 9].

In [18], the matrix C is decomposed onto the sum of 2 matrices C(1) and C(2), where neighbouring non-zero elements from c_j^{(1)} or c_j^{(2)}, j = −3, −2, ..., 3 represent the neighbouring powers of 2 (see Table 1). Moreover, the last non-zero element from \{c_j^{(1)}\} and the first non-zero element from \{c_j^{(2)}\} are also neighbouring powers of 2.

Therefore, the calculation of (7) can be realized by 2 subsequent word-level systolic arrays, which perform the corresponding circulant convolutions. Then, each word-level cell in such array can be realized as a linear vertical systolic array of full adders (see Fig. 4a), which are separated by delay elements, and the whole calculation can be pipelined on a bit level.

In [7, 9], other particular decompositions of C and corresponding bit-level systolic pipelined strategies of the calculation of (7) have been investigated and higher throughputs of the arrays have been achieved.

\[ z \xrightarrow{c_j} z' \quad x \xrightarrow{c} x' \quad z' =: z + c_j x; \quad \begin{array}{c} x_3 \ x_2 \ x_1 \ x_N \ \cdots \ x_1 \ x_N \ x_{N-1} \ x_{N-2} \end{array} \]

(a)

(b)

Fig. 3. Word-level systolic array for the least-squares 7-point digital contour smoothing.

\[ c' \]

\[ z \xrightarrow{c} x \]

\[ z' \]

\[ x' \]

\[ c' \]

\[ \cdots 0_{(13)} \ \cdots 0_{(13)} \ \cdots x_N \ x_{N-1} \ x_{N-2} \ \cdots \]  

\[ \cdots 0_{(2)} \ 0_{(2)} \ 0_{(2)} \ \cdots x_N \ x_{N-1} \ x_{N-2} \ \cdots \]

\[ \cdots 0_{(1)} \ 0_{(1)} \ 0_{(1)} \ x_N \ x_{N-1} \ x_{N-2} \]

\[ \begin{array}{cccccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1
1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1
1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array} \]

(a)

(c)

Fig. 4. A bit-level systolic array for the least-squares 7-point digital contour smoothing.

NDE_\ast - number of delay elements on \ast-connections.
For the purposes of this paper, let us consider the case of 8-bit input data. In this case, the intermediate results can be represented by \(8 + \lceil \log_2 21 \rceil + 1 = 14\)-bit numbers. Then, a systolic array with maximal throughputs and optimalized numbers of its structural elements is shown on Fig. 4 [9]. The whole array is a 2-dimensional regular matrix of full adders arranged into 12 columns (corresponding to the non-zero elements of decomposition matrices) and 14 rows (corresponding to the bits of different significance), which are connected by delay elements (see Fig. 4b-c). The directions of z- and x-connections are constant for the whole array; the directions of x-connections are constant within one column and change from column to column. The shifting of x-data by just one bit position upwards, downwards, or horizontally, corresponds to the multiplication by \(2^1, 2^{-1},\) or \(2^0,\) respectively. The multiplication by -1 is realized by converting the x-inputs into their 2’s complements (inverters are denoted as small black circles, see Fig. 4b). The numbers of delay elements on x- and z-connections, constant within one column, change from column to column. The number of delay elements on c-connections is constant in the whole array. The vertical and horizontal pipelining of the calculation achieved by involving delay elements on c-connections of the array and by corresponding transformation of numbers of the delay elements on z- and x-connections requires the input data seewing to provide the proper data alignment (see Fig. 4b-c). The clock period of such completely pipelined system is controlled by the delay \(t\) of a single full adder. The primitive operations performed are operations on the bit level and so the array is qualified as a **bit-level systolic array**. This array was obtained by the analysis of all possible decompositions of \(C\) mentioned above and has maximal throughputs and minimal numbers of cells and delay elements used (for more details see [8, 9]). The throughputs of the array is \((N+36)t\), where \(t\) is the delay of a single full adder and \(N\) is the number of points of the digital contour.

### Table 1

<table>
<thead>
<tr>
<th>(j)</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(c_j)</td>
<td>-2</td>
<td>3</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>3</td>
<td>-2</td>
</tr>
<tr>
<td>(c_j^{(1)})</td>
<td>0</td>
<td>(2^0)</td>
<td>(2^1)</td>
<td>(-2^0)</td>
<td>(2^1)</td>
<td>(2^0)</td>
<td>(-2^1)</td>
</tr>
<tr>
<td>(c_j^{(2)})</td>
<td>(-2^1)</td>
<td>(2^1)</td>
<td>(2^2)</td>
<td>(2^3)</td>
<td>(2^2)</td>
<td>(2^1)</td>
<td>0</td>
</tr>
</tbody>
</table>

#### 5 Concurrent pipelined computing structures

In this part, highly concurrent pipelined computing structures which are able to calculate invariants related to digitized contours investigated will be described. The hearth of the computing structures represents the systolic array for the feasible least-squares digital contour smoothing described above which can be denoted by the scheme shown on Fig. 5.

![Fig. 5. Scheme of the systolic array for the digital contour smoothing.](image)

The most commercially available vision systems are based on moment invariants [4, 12, 17]. The first and the second-order moments (non-normalized) of the smoothed curve can be realized by a parallel pipelined computing scheme shown on Fig. 6.

![Fig. 6. Scheme of a computing structure for moments estimation.](image)
can be suggested (see Fig. 7). This pipeline comprises the following functional elements:
systolic array for digital contour smoothing,
string generator \((a_i b_i)\) a string, each
string element \(a_i, b_i \in \langle 0, 50 \rangle\), i.e., each element
requires six bits) and 4K 32-bites LUT memory
\((a_i b_i)\) represents an address on which the values
of \(L_i\) and \(N_i\) are stored whereby for each value
16 bits are reserved). On the output of the pipeline the values of \(L_i\) and \(N_i\) are obtained in
parallel, so that the scheme achieves the effect
of multiprocessing and enables to avoid the
calculation of \(L_i\) and \(N_i\).

\[
\begin{align*}
x_{i-3} & \quad cx' \quad a_i := cx'_i - cx'_{i+1} + 25 \quad a_i b_i LUT \quad L_i \quad \Sigma L_i \\
y_{i-3} & \quad cy' \quad b_i := cy'_i - cy'_{i+1} + 25 \quad \Sigma N_i 
\end{align*}
\]

Fig. 7. Scheme of a computing structure for
the length estimation and the smoothed
contour coding.

The area of the smoothed curve can be realized
by a concurrent computing structure shown on
Fig. 8, where \(R\) performs (6).

\[
\begin{align*}
x_{i-3} & \quad cx' \quad cx_{i+2} := cx'_i - cx'_{i+1} + 25 \quad c_{aiy} \\
y_{i-3} & \quad cy' \quad cy_{i+2} := cy'_i - cy'_{i+1} + 25 \quad c_{ayx} \quad 2e_{y'z} \quad I \quad \Sigma 
\end{align*}
\]

Fig. 8. Scheme of a computing structure for
the smoothed contour area estimation.

6 Conclusion
In this paper, highly concurrent pipelined VLSI
computing structures for digital curves analysis
are described. The heart of these computing
structures is represented by the systolic array for
the constrained least-squares digital contour
smoothing. The computing structures enable to
estimate invariants related to digitized contours,
such as the area, the length, the first- and the
second-order moment invariants and encoding of
digitized curves, in parallel. They can be
realized on single VLSI chips and are dedicated for
real-time digital curves analysis.

Acknowledgement. This work was partially supported by National Scientific Grant Agency, grant No. 2-4149-24.

References:
[1] M. Bennamoun, G. Mamic, Object Recognition,
Aircraft Identification by Moments Invariants,
[6] K.S. Fu, VLSI for Pattern Recognition and
Contour Smoothing, LNCS, Vol. 399, Springer-
Verlag, Berlin, 1989, pp. 105-120.
Contour Smoothing and their Parallel Imple-
mentation on Systolic Arrays, PhD. theses, 1993.
[10] R.C. Gonzales, R.E. Woods, Dig. Image Process-
Classification, Proc. of the Int. Conf. on Mech.
Invariants. IRE Trans. Comput. Mach., Vol.9,
1962, pp. 240-258.
[14] P.A. Laplante, A.D. Stoyenko, A. Stoyenko,
[16] R. Lukac, Binary LUM Smoothing, IEEE Sig-
Vol.64, 1979, pp. 697-699.
[18] N. Petkov, F. Sloboda, A Bit-level Systolic
Array for Digital Contour Smoothing, Parallel
[19] N. Petkov, Systolic Parallel Processing, North
Digital Filters: Principles and Applications,