Analog Neural Networks for Real-Time Constrained Optimization

R. Domínguez-Castro †, A. Rodríguez-Vázquez, J. L. Huertas †, E. Sánchez-Sinencio ‡ and B. Linares-Barranco †‡
†Dto. Electrónica y Electromagnetismo, Universidad de Sevilla, 41012-Sevilla, SPAIN
‡Texas A&M University, Dept. of Electrical Engineering, College Station, TX 77843, USA

Abstract

We explore architectures and circuit techniques for the implementation of general *piecewise constrained optimization* problems using VLSI techniques. *Discrete-time* analog techniques are considered due to their inherent *accuracy, programmability and reconfigurability*. A general architecture is introduced for minimization of piecewise functions by using gradient schemes. Switched-capacitor (SC) building blocks featuring improved characteristics in terms of area occupation and operation speed are presented. The implementation of the architectures by using the newest Switched-Current (SI) techniques is also discussed. Finally, the layout of a 3 μm CMOS SC prototype for a quadratic optimization problem with linear constraints is given.

Introduction

The field of *analog computation* deserved big attention during the sixties [1], [2]. Although the basic concept was theoretically well proven at this time, this concept was of no practical use because of the many limitations of the implementations available in this pre-VLSI era. However, in spite of this and of the subsequent prevalent dominance of conventional digital computers, analog computation have continued to be recognized as the ideal solution for those applications where *real-time* processing is required.

The current status of analog VLSI technology makes it possible to overcome many of the problems in the implementation of analog computers. In particular, there exists a strong renovated interest in the design of new analog computational models based on some aspects of biological neural nets [3], [4]. It has been encouraged for recent proposals demonstrating the application of these *artificial neural networks* in solving difficult *optimization roblems* [5].

In [5] a special-purpose neural-like analog computer architecture was reported to solve linear constrained optimization problems in *real-time*. This preliminary work has been further extended by Chua and Kennedy [6] and others [7]. All of these approaches use conventional *active-C* design techniques which require operational amplifiers, resistors and capacitors. The resulting circuits are thus not the best suited for monolithic implementations.

Recently, the authors have obtained preliminary results in the direction of practical IC implementations of analog programming solvers by using switched-capacitor techniques [8], [9]. This paper explores several lines that were opened in these previous works. A more general architecture is presented covering the case of a piecewise objective function. Also, SC building blocks with enhanced performance are proposed to implement this architecture. Finally, building blocks are presented for the implementation of the architecture by using SI techniques.

Architectures for Constrained Optimization

Optimization problems are usually formulated as *minimization* (or maximization) problems. For *constrained optimization*, the formulation is made in terms of a cost function, $\Phi(\tilde{\mathbf{x}})$, subjected to a set of *constraints*. Solving such problem is hence the process of finding the point inside the region defined by the constraints (*feasibility region*) where the value of the cost function is the minimum one.

The common strategy to solve the constrained optimization problem by analog computation consists of two step. First, *penalty functions* are used to define an equivalent unsconstrained problem with a pseudo-cost function, $\Psi(\tilde{\mathbf{x}})$ [9], [10]. Then, an analog system is built to solve this equivalent problem by using *gradient* techniques,

$$\frac{d\mathbf{x}}{dt} = -\frac{1}{\tau}\nabla\Psi(\mathbf{x}) \qquad , \tau > 0 \tag{1}$$

The formulation in [8], [9] assumes the cost function is defined by a single expression valid in the whole feasibility region. Here, a more general problem is considered where the feasibility region is divided into several subregions, each one corresponding to a different expression of the cost function. This general constrained optimization problem can be formulated as follows,

$$\textit{Minimize} \qquad \Phi(\tilde{\mathbf{x}}) = \begin{cases} \Phi_1(\tilde{\mathbf{x}}) & , \tilde{\mathbf{x}} \in R_1 \\ \vdots & \\ \Phi_p(\tilde{\mathbf{x}}) & , \tilde{\mathbf{x}} \in R_p \\ \vdots & \\ \Phi_P(\tilde{\mathbf{x}}) & , \tilde{\mathbf{x}} \in R_P \end{cases} \tag{2}$$

Subject to the constraints

$$F_k(\tilde{\mathbf{x}}) \geq 0 \ , \quad 1 \leq k \leq Q$$

An example of such a kind of problems can be found in Fig.1, corresponding to a two-dimensional piecewise-linear scalar function.

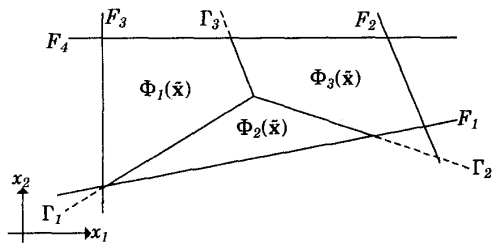


Figure 1:Two-dimensional piecewise-linear constrained optimization problem.

Observe the feasibility region is divided by the curves $\Gamma_1(\tilde{\mathbf{x}})$, $\Gamma_2(\tilde{\mathbf{x}})$, $\Gamma_3(\tilde{\mathbf{x}})$ into three regions. In the more general case, a set of L functions (Γ_l, $1 \leq l \leq L$) will be required to define the P subregions of the feasibility interval. In this general case, each region R_p, will be defined by the condition that all corresponding boundary function, (Γ_l and/or F_k) are positive. For instance, the region R_1 in Fig.1 is defined by F_3, F_4, Γ_1 and Γ_3 via the following inequalities

$$F_3(\mathbf{x}) \geq 0, \quad F_4(\mathbf{x}) \geq 0, \quad \Gamma_1(\mathbf{x}) \leq 0, \quad \Gamma_3(\mathbf{x}) \geq 0,$$

In a more compact notation we will say R_1 is defined by the condition that all the components of a vectorial function $\mathbf{R}_1 = \{F_3, F_4, -\Gamma_1, \Gamma_3\}$ are positive.

By extending the penalty strategy given in [9] to this general piecewise problem the following pseudo-cost function can be calculated,

CH2868-8/90/0000-1867$1.00 © 1990 IEEE

$$\Psi(\mathbf{x}) = \sum_{p=1}^{P} U(\mathbf{R}_p)\Phi_p(\mathbf{x}) + \mu \sum_{k=1}^{Q} U(-F_k) P_k\left|F_k(\mathbf{x})\right| \qquad (3a)$$

where we define

$$U(\mathbf{G}) = \begin{cases} 1 & ,\text{if each component of } \mathbf{G} \text{ is positive} \\ 0 & ,\text{otherwise} \end{cases} \qquad (3b)$$

and where $P_k[F_k]$ is assumed to monotonically increase as the constraint F_k decrease from zero. In the more common case,

$$P_k\left|F_k(\mathbf{x})\right| = \begin{cases} |F_k(\mathbf{x})| \\ F_k^2(\mathbf{x}) \end{cases} \qquad (3c)$$

Starting from this pseudo-cost function the following discrete-time companion gradient system can be formulated,

$$x_i(n+1) = x_i(n) - \frac{1}{\tau_o}\left[\sum_{p=1}^{P} U(\mathbf{R}_p)\frac{\partial \Phi_p}{\partial x_i} + \mu \sum_{k=1}^{Q} U(-F_k)\frac{\partial P_k}{\partial F_k}\frac{\partial F_k}{\partial x_i}\right]\Bigg|_{\mathbf{x}(n)} \qquad (4)$$

which corresponds to the conceptual block diagram of Fig.2. This hence represents a general architecture for the solution of constrained optimization problems using discrete-time analog techniques.

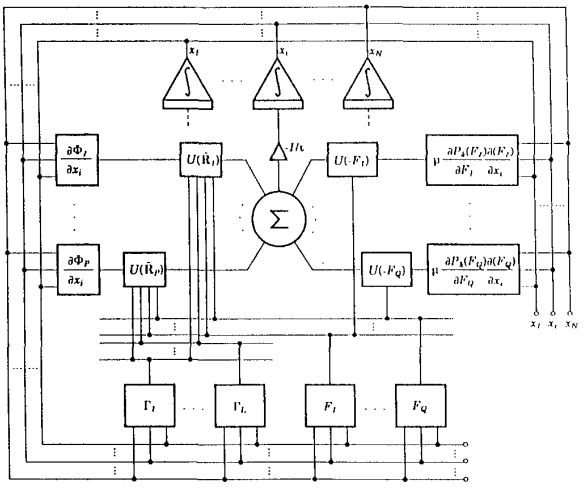


Figure 2:Architecture for a discrete-time constrained optimizer with a piecewise cost function.

The integrators in Fig.2 can be viewed as the elementary computational units ("neurons") of a neural-like circuit architecture. Observe the interconnections among "neurons" are strongly nonlinear in the more general case. Consider the particular case in which the pieces of the scalar function are quadratic,

$$\Phi_p(\mathbf{x}) = \sum_{i=1}^{N} a_i x_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} (g_{ij})_p x_i x_j \qquad , (g_{ij})_p \doteq (g_{ji})_p \qquad (5)$$

and the constraints and the boundary functions are linear. In this case, only *analog switches* and *comparators* are required to implement the nonlinear interconnections. It is illustrated in Fig.3a where the *i-th* "neuron" and corresponding inputs are shown for a piecewise quadratic problem with linear sections and linear constraints and assuming the absolute-value penalty is used. The signal controlling the analog switches in Fig.3a can be obtained

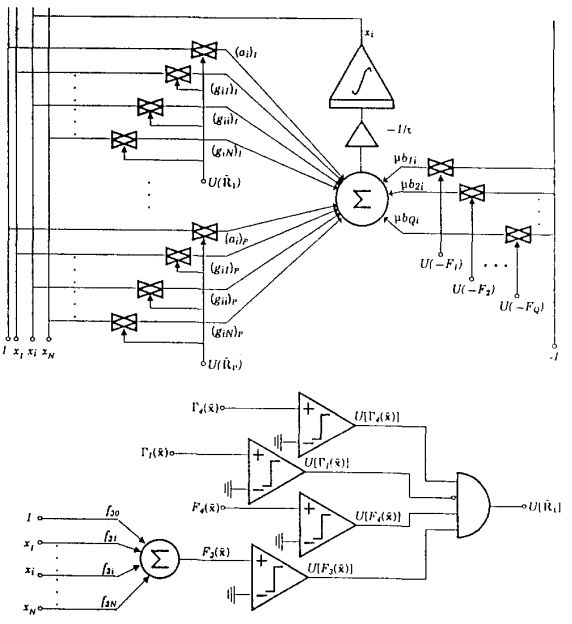


Figure 3:a) i-th "neuron" and corresponding inputs b) example of boundary encoder.

via logical operations on the outputs of weighted-summer/ comparators as it is illustrated in Fig.3b for the region R_1 of Fig.1.

Design of the SC building blocks

According to Fig.3, the only blocks required for the implementation of piecewise-quadratic constrained optimization problems with linear boundaries are just weighted-summer/integrators and /comparators. State of the art implementations for these blocks can be found elsewhere [11], [12], [9]. There are however several practical considerations concerning our specific application which should be taken into account for proper design.

Area-optimized integrators.

Let us consider the summing/integrator. In a typical optimization problem, some of the "neurons" may require the values of the corresponding weights to vary in a wide range. Also, stability considerations dictate the need to scale all the weights by a very small value in order the solution point to remain bounded inside a given interval. Since in conventional summing/integrators the weights are directly given by ratios of capacitors, very large areas and power consumptions may result if very small and very different weights have to be implemented. Obviously this is not convenient for "neural-like" circuits.

A family of new SC summing/integrators have been developed which overcome this drawback of the conventional implementations. Fig.4 serves to illustrate both the basic concept and the properties of the family.

A conventional one-input SC integrator is shown (Fig.4a) together with an area optimized design (Fig.4b). Assuming the area is proportional to the capacitor values and using a weight $1/\tau$ for the input we get,

$$A_{NOR} \propto C_1 + \frac{A_{MOD}^2}{4}$$
$$A_{NOR} \propto C_1\left\{1 + \tau\left[1 + \frac{C_5}{C_4}\right]\right\} \qquad (6)$$

where A_{NOR} holds for Fig.4a and A_{MOD} for Fig.4b. A

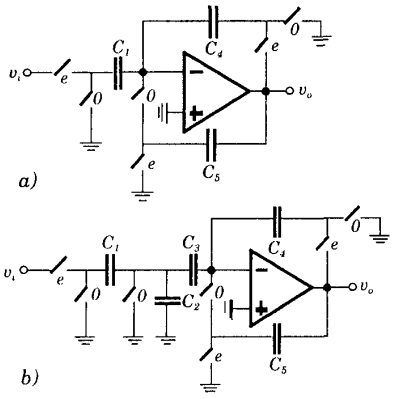
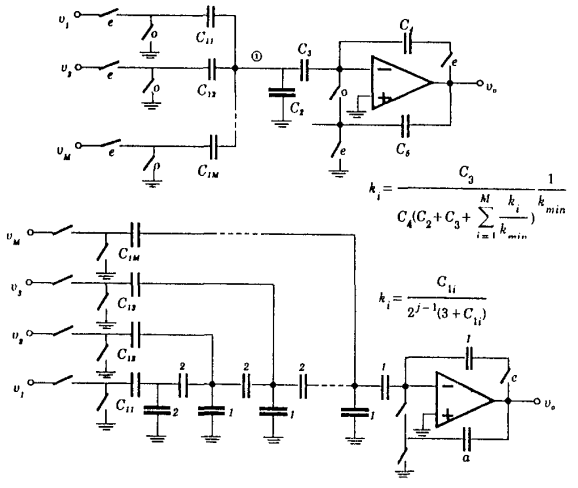


Figura 4:a) Insensitive Amplifier [Malo84]; b) Modification small weights.

significant reduction in the area can be observed when the modified structure is compared to the conventional one.

The design concept in Fig.4 can be extended to the general summing/integrator case. Fig.5 show several alternatives implementations. The corresponding expressions for the weights are shown along with the figures.

The circuits in Fig.5 are insensitive to the input offset voltage of the opamp. They are however sensitive to the parasitic capacitors associated to actual MOS capacitors. In the case of Fig.5a all the weights are modified by the same factor. In its application to analog optimizers it means a change in the speed of operation of the circuits but not in the accuracy of the solution point. For Fig.5b different weights are modified by different factors which may not only affect the speed of convergence but also the accuracy of the solution point.



$$k_i = \frac{C_3}{C_4(C_2 + C_3 + \sum_{i=1}^{M} \frac{k_i}{k_{min}})} \frac{1}{k_{min}}$$

$$k_i = \frac{C_{1i}}{2^{j-1}(3 + C_{1i})}$$

Figure 5: Area-optimized summing/integrators.

Comparator

For most practical cases, the solution point of a piecewise constrained optimization problem is either on the boundary between two adjacents region inside the feasibility region or on the boundary of this latter region. Fig.6 shows a typical trajectory of a discrete-time constrained optimization solver. It corresponds to the problem of Fig.1, whose theoretical solution point is at the crossing point of the three boundary lines Γ_1, Γ_2 and Γ_3. As can be seen, the trajectory is oscillating back and forth around the line Γ_3. The comparator whose output codifies the position of the point relative to this line must hence change state in each iteration. Since the amplitude of the oscillations must, on the other hand, stay bounded to a small value to ensure stability of the optimizer, the amplitude of the step at the input of the comparator will be very small and, as a consequence, the transient to change state very large. To overcome this problem, we propose to use the dynamic comparator of Fig. 6 where a positive feedback is temporarily applied during the one of the clock phases. By using this comparator a reduction of even two orders magnitude can be achieved in regard to the resolution time of a conventional high-gain comparator.

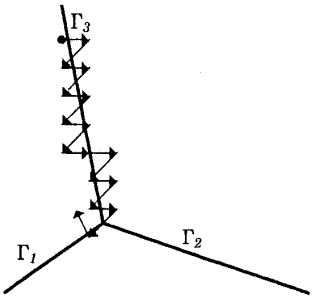


Figure 6: Typical trajectory for a piecewise optimizer.

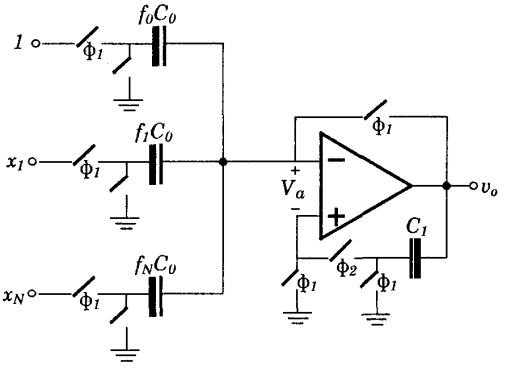


Figure 7:A dynamic positive feedback SC comparator

Switched-Currents Building Blocks

The primitives for the monolithic implementation of SC circuits are MOS transistors and MOS capacitors. Recently, a new design technique have been proposed for the design of analog sampled-data circuits which only requires MOS transistors [13]. This is very appealing for implementation in digital MOS technologies where the available capacitors are bulky and inaccurate. Also in SI circuits the processing is not made on voltages, as it is the case in SC circuits, but on currents. It means that the summing operation can be directly made by exploiting KCL. Also, the reduction in dynamic range which results as a consequence of the scaling down is less restrictive for current-based circuits than it is for voltage-based ones.

Fig.8 shows several elementary SI building blocks for the solution of piecewise-quadratic optimization problems with linear boundaries. The same architectural considerations as for SC circuits apply for this family of SI blocks [9].

Discussion of Results

Using sampled-data analogue techniques provides a natural way to implement "neural-like" constrained optimizers in monolithic form. Several SC prototypes have

been built. In particular, Fig.9 shows one microphotograph corresponding to a quadratic problem with linear constraints. It has been designed in a 3μm n-well double-poly technology and conventional SC building blocks [9]. Preliminary testing give results that are in accordance with the expected theoretical behavior. New prototypes are currently under progress which uses the enhanced building blocks proposed in this paper. A family of enhanced SI building blocks is also currently under development. Simulation results for this family shows very promising results in terms of convergence speed.

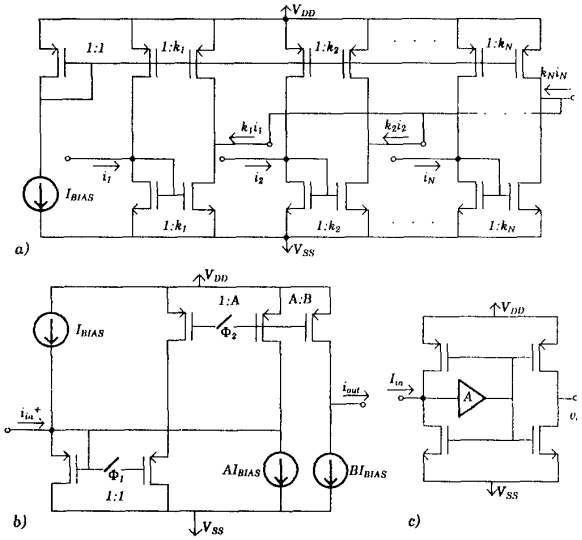


Figure 8:a) SI weighted-summer, b) SI integrator, c) Current comparator

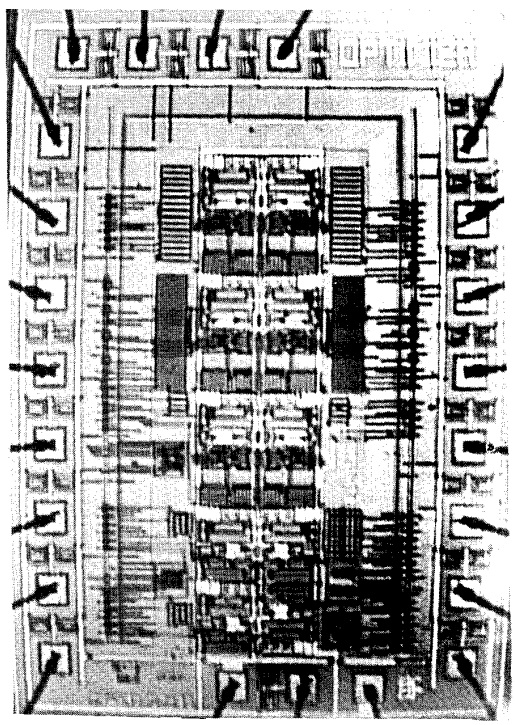


Figure 9:A 3μm CMOS prototype

References

[1] A. Hausner: *"Analog and Analog/Hybrid Computer Programming"*. Prentice-Hall 1971.

[2] G.A. Korn and T.M. Korn: *"Electronic Analog and Hybrid Computers- 2nd Edition"*. Mc Graw-Hill 1972.

[3] C.A. Mead: *"Analog VLSI and Neural Systems"*. Addison Wesley 1989.

[4] V. Vemuri: *"Artificial Neural Networks: Theoretical Concepts"*. IEEE Comp. Society Press 1988.

[5] D.A. Tank and J.J. Hopfield: "Simple Neural Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit". *IEEE Trans. Circuits Systems*, Vol. 33, pp 533-541, May 1986.

[6] M.P. Kennedy and L.O. Chua: "Neural Networks for Nonlinear Programming". *IEEE Trans. Circuits and Systems*, Vol. 35, pp 554-562, May 1988.

[7] M.J.S. Smith and C.L. Portmann: "Practical Design and Analysis of a Simple "Neural" Optimization Circuit". *IEEE Trans. Circuits Systems*, Vol. 36, pp 42-50, January 1989.

[8] A. Rodríguez-Vázquez et al. : "Switched-Capacitor "Neural" Networks for Linear Programming". *Electronic Letters*, Vol 24, pp 496-498, April 1988.

[9] A. Rodríguez-Vázquez et al.: "Nonlinear Switched-Capacitor "Neural" Networks for Optimization Problems". *IEEE Trans. Circuits and Systems*, Vol CAS-37, March 1990. (to appear)

[10] G. V. Vanderplaats: *"Numerical Optimization Techniques for Engineering Design: with Applications"*. McGraw-Hill 1984.

[11] R. Gregorian and G.C. Temes: *"Analog MOS Integrated Circuits for Signal Processing"*. Wiley-Interscience 1986.

[12] P.E. Allen and D.R. Holberg: *"CMOS Analos Circuit Design"*. Holt, Rinehart and Winston, 1987.

[13] J.B. Hughes et al.: "Switched Currents-A New Technique for Analog Sampled-Data Signal Processing". *Proc. IEEE-ISCAS'89, pp 1584-1587, 1989.*