International Journal of Automation and Computing 6(3), August 2009, 285-294
DOI: 10.1007/s11633-009-0285-2

A New Method of Embedded Fourth Order with Four Stages to Study Raster CNN Simulation

R. Ponalagusamy* S. Senthilkumar

Department of Mathematics, National Institute of Technology, Tiruchirappalli 620 015, India

Abstract: A new Runge-Kutta (PK) fourth order with four stages embedded method with error control is presented in this paper for raster simulation in cellular neural network (CNN) environment. Through versatile algorithm, single layer/raster CNN array is implemented by incorporating the proposed technique. Simulation results have been obtained, and comparison has also been carried out to show the efficiency of the proposed numerical integration algorithm. The analytic expressions for local truncation error and global truncation error are derived. It is seen that the RK-embedded root mean square outperforms the RK-embedded Heronian mean and RK-embedded harmonic mean.

Keywords: Raster scheme, cellular neural network (CNN), numerical integration techniques, edge detection, new embedded Runge Kutta root mean square (RKARMS (4, 4)) method, truncation errors.

1 Introduction

Cellular neural network (CNN) was first proposed by Chua and Yang[1, 2] as an implementable alternative to fully-connected neural networks. Furthermore, CNN[3] has many important applications and has been widely studied for theoretical foundations and practical applications in real-time image and video signal processing, robotic and biological visions, and higher brain vision. Roska et al.[4,5]have presented the first widely used simulation system that allows the simulation of a large class of CNN and is especially suited for image processing applications[6]. It also includes signal processing, pattern recognition, and solving ordinary and partial differential equations, etc. Lee and Gyvez[7] introduced Euler, improved Euler, predictor-corrector, and fourth-order (quartic) Runge-Kutta (RK) algorithms in raster CNN simulation. Using RK-Butcher′ fifth-order method, the edge detection problem has been analyzed using raster CNN simulation[8]. To distinguish faces of various angles during face recognition, an algorithm of the combination of approximate dynamic programming (ADP) called action dependent heuristic dynamic programming (ADHDP), and particle swarm optimization (PSO) is presented by Lu[9]. Xiao et al.[10] proposed a method for moving shadow detection based on edge information, which can effectively detect the cast shadow of a moving vehicle in a traffic scene. A novel method of shape retrieval based on shape impression of human′s Kansei have been developed by Koda et al.[11]

It is of interest to state that embedded methods are actually two methods built into one. The first method is of order p, and the second has order $p+1$. The difference between these methods provides an error estimate for the first method with order p. Error estimates by these methods have been derived by Merson[12] and Fehlberg[13]. Evans and Yaakub[14] introduced a new embedded Runge-Kutta RK (4,4) method that is actually two different RK methods but of the same order $p = 4$. Yaacob and Sanugi[15] adapted RK-embedded harmonic mean for solving the ordinary differential equations. Evans and Yaacob[16] proposed a fourth order RK method based on the Heronian mean formula.

It is known that the general p-stage RK method for solving $\dot{y}(x) = f(x, y(x))$ is defined by

$$y_{n+1} = y_n + h\sum_{i=1}^{p} b_i k_i$$

where

$$k_i = f(x_n + c_i h, y_n + h\sum_{i=1}^{p} a_{ij} k_i)$$

$$c_i = \sum_{j=1}^{p} a_{ij}, \quad i = 1, 2, 3, \cdots, p$$

$$b_i = \sum_{j=1}^{p} a_{ij}, \quad i = 1, 2, 3, \cdots, p$$

with p dimensional vectors c and b and the $(p \times p)$ matrix $A(a_{ij})$. The RK$(p, p+1)$ methods with a built-in error estimate have been proposed by Merson[12] and Fehlberg[13]. Further, the above can be represented in Butcher array form of order $p+1$.

$$\begin{array}{c|cccccc}
0 & & & & & & \\
c_2 & a_{21} & & & & & \\
c_3 & a_{31} & a_{32} & & & & \\
\vdots & \vdots & \vdots & & & & \\
c_p & a_{p,1} & a_{p,2} & \cdots & a_{p,p-1} & & \\
c_{p+1} & a_{p+1,1} & a_{p+1,2} & \cdots & a_{p+1,p-1} & a_{p+1,p} & \\
\hline
 & b_1 & b_2 & \cdots & b_{p-1} & b_p & b_{p+1}
\end{array}$$

The following Butcher array representation has order p.

Manuscript received August 5, 2008; revised February 10, 2009
This work was supported as a part of Technical Quality Improvement Programme (TEQIP).

*Corresponding author. E-mail address: rpalagu@nitt.edu

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & \vdots & & & \\
c_p & a_{p,1} & a_{p,2} & \cdots & a_{p,p-1} & \\
\hline
 & a_{p+1,1} & a_{p+1,2} & \cdots & a_{p+1,p-1} & a_{p+1,p}
\end{array}$$

The values of y_{n+1} from a given value of y_n are obtained from the above two methods of order $p+1$ and p, respectively and the difference of the results computed by those methods is used to determine the error estimate.

The RK root mean square (RKARMS (4,4)) method is found to have comparatively less local truncation error (LTE), global truncation error (GTE), and error estimate, and plays a key role in computing numerical solutions of industrially applicable problems as well as image processing problems. Keeping this in view, a modest effort has been made to develop a new numerical method with less computational error.

In this paper, the edge detection problem is solved with different approaches using presently developed new RK-embedded root mean square and raster scheme. The rest of the paper is organized as follows. In Section 2, the embedded RKARMS (4,4) method is discussed in detail with proof. In addition, the error control of RKARMS(4,4) is analyzed, and its subsection deals with LTE, GTE, and estimation of error. Section 3 gives an outline idea about the functions of CNN and its structure. Section 4 deals with the technical rationale of raster CNN simulation and its performance. Moreover, the subsection discusses the pseudo code for image-based raster behavioral CNN simulation. Section 5 discusses various numerical integration techniques. Section 6 discusses the simulation and experimental results of the given images. Finally, conclusion is given in Section 7.

2 Development of RK-embedded RKARMS method

Consider the ordinary differential equation of initial value problem

$$\dot{y}(x) = f(x, y(x)), \quad x \geqslant x_0 \tag{1}$$

where $y(x_0) = y_0$, $f : \mathbf{R} \times \mathbf{R}^m$ is sufficiently differentiable in a neighborhood of the exact solution $(x, y(x)), x \in [a, b]$.

An embedded explicit RK pair is adopted to find numerical solution of (1). A general s-stage RK pair is written in the form of Butcher array.

$$\begin{array}{c|c}
C & A \\
\hline
 & b^{\mathrm{T}} \\
 & \hat{b}^{\mathrm{T}} \\
\hline
 & E^{\mathrm{T}}
\end{array} \tag{2}$$

The symbols defined by C, A, and b^{T} have order p, and those defined by C, A, and $\hat{b}^{\mathrm{T}}$ have order $(p+1)$. Using these two methods, the values of y at $x = x_{n+1} = x_{n+h}$, can be expressed as

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i \tag{3}$$

$$\hat{y}_{n+1} = y_n + h\sum_{i=1}^{s} \hat{b}_i k_i$$

where h is the step size, and

$$k_i = f(x_n+c_i h, y_n+h\sum_{i=1}^{s} a_{ij}k_i), \quad c_i = \sum_{j=1}^{s} a_{ij}, \quad i = 1, 2, \cdots$$

with s dimensional vectors c and b and the $(s \times s)$ matrix $A(a_{ij})$. From the embedded form, LTE may be computed from the formula: $LTE = y_{n+1} - \hat{y}_{n+1}$. It is of interest to note that LTE controls the step size. The four-stage method with the Butcher array form is written as

$$\begin{array}{c|cccc}
0 & & & & \\
c_2 & a_{21} & & & \\
c_3 & a_{31} & a_{32} & & \\
c_4 & a_{41} & a_{42} & a_{43} & \\
\hline
 & b_1 & b_2 & b_3 & b_4
\end{array} \tag{4}$$

The fourth-order RK arithmetic mean (RKAM) can be written in the Butcher array form

$$\begin{array}{c|cccc}
0 & & & & \\
\frac{1}{2} & \frac{1}{2} & & & \\
\frac{1}{2} & 0 & \frac{1}{2} & & \\
1 & 0 & 0 & 1 & 0 \\
\hline
1 & \frac{1}{6} & \frac{2}{3} & \frac{2}{3} & \frac{1}{6}
\end{array}$$

$$y_{n+1} = y_n + \frac{h}{3}\left(\frac{k_1+k_2}{2} + \frac{k_2+k_3}{2} + \frac{k_3+k_4}{2}\right) \tag{5}$$

where

$$\begin{aligned}
k_1 &= f(x_n, y_n) \\
k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_1}{2}) \\
k_3 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_2}{2}) \\
k_4 &= f(x_n + h, y_n + hk_3).
\end{aligned} \tag{6}$$

From (4)–(6), the RK methods with Butcher array can also be written in the modified form

$$\begin{array}{c|ccc}
0 & & & \\
\frac{1}{2} & \frac{1}{2} & & \\
\frac{1}{2} & 0 & \frac{1}{2} & \\
1 & 0 & 0 & 1 \\
\hline
 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3}
\end{array} \tag{7}$$

The fourth-order RK root mean square (RKRMS) is given by

$$y_{n+1} = y_n + \frac{h}{3}\left(\sqrt{\frac{k_1^2+k_2^2}{2}} + \sqrt{\frac{k_2^2+k_3^2}{2}} + \sqrt{\frac{k_3^2+k_4^2}{2}}\right) \tag{8}$$

where

$$\begin{aligned} k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1) \qquad (9) \\ k_3 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{16}hk_1 + \frac{7}{16}hk_2) \\ k_4 &= f(x_n + h, y_n + \frac{1}{8}hk_1 - \frac{17}{56}hk_2 + \frac{33}{28}hk_3). \end{aligned}$$

Combination of RKAM and RKRMS ((5) and (8)) is referred to as RKARMS (4,4), and can be formulated as

$$\begin{aligned} k_1 &= f(x_n, y_n) = k_1^* \\ k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_1}{2}) = k_2^* \\ k_3 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_2}{2}) \\ k_4 &= f(x_n + h, y_n + hk_3) \\ k_3^* &= f(x_n + \frac{1}{2}h, y_n - \frac{1}{48}hk_1 + \frac{25}{48}hk_2) \\ k_4^* &= f(x_n + h, y_n + \frac{1}{8}hk_1 - \frac{17}{56}hk_2 + \frac{33}{28}hk_3) \\ y_{n+1} &= y_n + \frac{h}{3}\left(\frac{k_1+k_2}{2} + \frac{k_2+k_3}{2} + \frac{k_3+k_4}{2}\right) \end{aligned}$$

$$\begin{aligned} y_{n+1}^* = y_n + \frac{h}{9}\Big(k_1^* + 2(k_2^* + k_3^*) + k_4^* + \\ \sqrt{|k_1^* + k_2^*|} + \sqrt{|k_2^* + k_3^*|} + \sqrt{|k_3^* + k_4^*|}\Big). \end{aligned} \tag{10}$$

The Butcher array form for embedded RKARMS(4,4) is expressed as

$$\begin{array}{c|ccc} 0 & & & \\ \frac{1}{2} & \frac{1}{2} & & \\ \frac{1}{2} & 0 & \frac{1}{2} & \\ 1 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{2} & \frac{1}{16} & \frac{7}{16} & \\ 1 & \frac{1}{8} & -\frac{17}{56} & \frac{33}{28} \\ \hline & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \hline & E^{\mathrm{T}} & & \end{array}$$

Hence, by (2)

$$b^{\mathrm{T}} = y_{n+1}^{AM}$$

$$y_{n+1}^{AM} = y_n + \frac{h}{3}\left(\frac{k_1+k_2}{2} + \frac{k_2+k_3}{2} + \frac{k_3+k_4}{2}\right)$$

$$\begin{aligned} \hat{b}^{\mathrm{T}} &= y_{n+1}^{RMS} = \\ & y_n + \frac{h}{9}\Bigg(k_1^* + 2(k_2^* + k_3^*) + k_4^* + \\ & \sqrt{|k_1^* + k_2^*|} + \sqrt{|k_2^* + k_3^*|} + \sqrt{|k_3^* + k_4^*|}\Bigg) \end{aligned}$$

and the estimation of the LTE, $E^{\mathrm{T}} = \left|b^{\mathrm{T}} - \hat{b}^{\mathrm{T}}\right|$, where y_{n+1}^{AM} and y_{n+1}^{RMS} are numerical approximations at x_{n+1} obtained by arithmetic mean (AM) and root mean square (RMS), respectively. In the RKARMS $(4,4)$ method, four stages are required to obtain the solution, which share the same set of vectors k_1 and k_2 using b^{T} and $\hat{b}^{\mathrm{T}}$ approximately, but k_3 and k_4 use b^{T} while k_3^* and k_4^* use $\hat{b}^{\mathrm{T}}$.

2.1 Error control analysis for RKARMS (4,4)

It is well known that the primary types of error in the numerical solution of ordinary and partial differential equations are truncation and rounding errors. Let us now study and compare LTE and GTE. Henrichi[17] and Lambert[18, 19] discussed in detail the following theorem.

Theorem 1. Let the function $f(x,y)$ be defined and continuous for all points (x,y) such that $a \leqslant x \leqslant b$, $-\infty < y < \infty$, where a and b are finite. If f satisfies a Lipschitz condition, i.e., there exists a constant L such that $|f(x,y) - f(x,y^*)| \leqslant L(y - y^*)$ for $a \leqslant x \leqslant b$ and all y, y^*, then for any initial value a, there exists a unique solution of the initial value problem $\dot{y} = f(x,y)$, $y(a) = \alpha$ where L is called the Lipschitz constant $(17-19)/(18-20)$.

Definition 1. The LTE at the point x_{n+1} is the difference between the computed value y_{n+1} and the value at the point x_{n+1} on the solution curve that goes through the point (x_n, y_n).

Definition 2. The GTE in the point x_{n+1} is defined as $y_{n+1} - y(x_{n+1})$, where $y(x)$ denotes the solution of the given initial value problem.

2.2 Derivation of LTE for RKARMS (4,4)

According to Lotkin[20] and Ralston[21], the error estimate for fourth order RK schemes is given by $|\psi(x_n, y_n : h)| \leqslant (73/720)ML^4$, where M and L are positive constants. To control the step size h, use (10) to obtain an estimate LTE for the RKARMS$(4,4)$ as

$$LTE = y_{n+1} - y_{n+1}^*.$$

The LTE for the classical fourth-order RKAM is

$$y_{n+1}^{AM} = y_n + LTE_{AM}$$

and for RK based on RMS is

$$y_{n+1}^{RMS} = y_n + LTE_{RMS}$$

where LTE_{AM} and LTE_{RMS} are the LTEs in RKAM and RKRMS. The difference between the RKAM and RKRMS at x_{n+1} gives an error estimate as

$$y_{n+1}^{AM} - y_{n+1}^{RMS} = LTE_{AM} - LTE_{RMS}.$$

The LTE for RKAM is given by

$$LTE_{AM} = \frac{h^5}{2880}[-24ff_y^4 + f^4 f_{yyyy} + 2f^3 f_y f_{yyy} - 6f^3 f_{yy}^2 + 36f^2 f_y^2 f_{yy}] \tag{11}$$

where the LTE of the RKRMS is given by

$$LTE_{RMS} = \frac{h^5}{184320}[-429ff_y^4 - 64f^4 f_{yyyy} - 48f^3 f_y f_{yyy} - 96f^3 f_{yy}^2 - 2454f^2 f_y^2 f_{yy}]. \tag{12}$$

The absolute difference between $LTE_{AM} - LTE_{RMS}$ is given by

$$|LTE_{AM} - LTE_{RMS}| = \frac{h^5}{184320}[1107ff_y^4 + 128f^4 f_{yyyy} + 176f^3 f_y f_{yyy} + 288f^3 f_{yy}^2 + 4758f^2 f_y^2 f_{yy}]. \tag{13}$$

By following an argument suggested by Lotkin[20], if we assume that the following bounds for f and its partial derivatives hold for $x \in [a,b]$ and $y \in (-\infty, \infty)$, we have

$$|f(x,y)| < Q$$
$$\left|\frac{\partial^{i+j} f(x,y)}{\partial x^i \partial y^j}\right| < \frac{P^{i+j}}{Q^{j-1}}, \quad i+j \leqslant p \tag{14}$$

where P and Q are positive constants, and p is the order of the method.

In the present analysis, $p = 4$. Using (14), one may arrive at

$$\left.\begin{array}{l} |f_y| < P \\ |ff_y^4| < \dfrac{Q^4 P^4}{Q^3} \\ |f^4 f_{yyyy}| < \dfrac{Q^4 P^4}{Q^3} \\ |f^3 f_y f_{yyy}| < Q^3 P \dfrac{P^3}{Q^2} \\ |f^3 f_{yyy}^2| < Q^3 (\dfrac{P^2}{Q})^2 \\ |f^2 f_y^2 f_{yy}| < Q^2 P^2 (\dfrac{P^2}{Q}) \end{array}\right\} < P^4 Q. \tag{15}$$

From (13) and (14), we obtain

$$LTE_{AM} - LTE_{RMS} \leqslant (\frac{6457}{184320}) P^4 Q h^5. \tag{16}$$

Hence,

$$\left|y_{n+1}^{AM} - y_{n+1}^{RMS}\right| \leqslant \frac{6457}{184320} P^4 Q h^5. \tag{17}$$

We assume the tolerance $TOL = 0.00005$. Taking $\left|y_{n+1}^{AM} - y_{n+1}^{RMS}\right| \leqslant TOL$ and using (17), step-size (ΔT) selection can be determined in order to control the error

$$\frac{6457}{184320} P^4 Q h^5 < TOL$$

or

$$h < \left(\frac{28.54764 \cdot TOL}{P^4 Q}\right)^{\frac{1}{5}}. \tag{18}$$

2.3 GTE for RKAM

Yaakub and Evans[22] proved that the GTE for RKAM is of order h^4.

2.4 GTE for fourth-order four-stage RKRMS method

With respect to the fifth decimal place, Taylor′s series at $x = x_n$ may be written as

$$y(x_n + h) = y(x_n) + hf + \frac{h^2}{2} ff_y + \frac{h^3}{6}(ff_y^2 + f^2 f_{yy}) + \frac{h^4}{27}(f^3 f_{yyy} + \frac{1}{6} f^2 f_y f_{yyy}) + \frac{h^5}{120}(ff_y^4 + f^4 f_{yyyy} + 7f^3 f_y f_{yyyy} + 11f^2 f_y^2 f_{yy} + f^3 f_{yy}^2). \tag{19}$$

The LTE for RKRMS is the difference between (8) and (19):

$$LTE_{RMS} = \frac{h^5}{184320}[-429ff_y^4 - 64f^4 f_{yyyy} - 48f^3 f_y f_{yyy} - 96f^3 f_{yy}^2 - 2454f^2 f_y^2 f_{yy}]. \tag{20}$$

By using the procedure adapted by Evans and Yaakub[14] to evaluate the GTE for the RKAM, we have for the RMS:

$$\varepsilon_{n+1} \leqslant \varepsilon_n + hL\varepsilon_n + \frac{h^2}{2} L\varepsilon_n + \frac{h^3}{6} L\varepsilon_n + \frac{h^4}{24} L\varepsilon_n + \frac{h^5}{184320} y^v(\xi)$$

where $0 < \xi < 1$ and

$$|\varepsilon_{n+1}| \leqslant \left(1 + hL + \frac{h^2 L}{2} + \frac{h^3 L}{6} + \frac{h^4 L}{24}\right) |\varepsilon_n| + \frac{h^5}{184320} M \leqslant (1 + c_1) |\varepsilon_n| + B_1$$

$$|y^v(x)| \leqslant M$$

where

$$c_1 = L \left(\sum_{p=1}^{n} \frac{h^{p-1}}{p!}\right) h$$
$$A_1 = 1 + c_1$$
$$B_1 = \frac{h^5}{184320} M.$$

A simple induction proof leads to

$$|\varepsilon_n| \leqslant A_1^n |\varepsilon_0| + (\sum_{k=0}^{n-1} A_1^k) B_1.$$

For $A_1 \neq 1$, and $\varepsilon_0 = 0$, the term $|\varepsilon_n|$ can be written as

$$|\varepsilon_n| \leqslant (\frac{A_1^n - 1}{A_1 - 1}) B_1. \tag{21}$$

If we use the inequality $1+x \leqslant \mathrm{e}^x$, we get

$$A_1^n = (1+c_1)^n = \left(1 + hL\sum_{p=1}^{4}\frac{h^{p-1}}{p!}\right)^n \leqslant \mathrm{e}^{c_1 n} = \mathrm{e}^{Lhn}(\sum_{p=1}^{4}\frac{h^{p-1}}{p!}) = \mathrm{e}^{DL(x_n-x_0)} \tag{22}$$

where

$$D = \sum_{p=1}^{4}\frac{h^{p-1}}{p!}.$$

By substituting (22) into (21), for ε we obtain

$$|\varepsilon_n| \leqslant \frac{h^4}{184320\,LD}M(\mathrm{e}^{DL(x_n-x_0)}-1)$$

and therefore, the GTE for the fourth-order RMS method is of order h^4. The LTE represents the magnitude of the error involved in the numerical solution obtained by the corresponding numerical technique. From the above discussion, we can conclude that if the LTE of a numerical method is $O(h^{p+l})$, then the GTE is $O(h^p)$. The estimate of the GTE cannot be used for practical error estimation or error control because the value from the GTE is less accurate than the LTE.

2.5 Estimation of error for RKARMS (4,4)

It is significant to point out that in the RKARMS(4,4) method with error control program, we choose the error estimate as the difference between the fourth order RKAM method and the RK method. From (17), the error estimate is expressed as

$$ERREST = |Y_{AM} - Y_{RMS}|\,\frac{6457}{184320}. \tag{23}$$

3 Outline of CNN

The first-order nonlinear differential equation defining the dynamics of a cellular nonlinear network cell[1,2] is given by

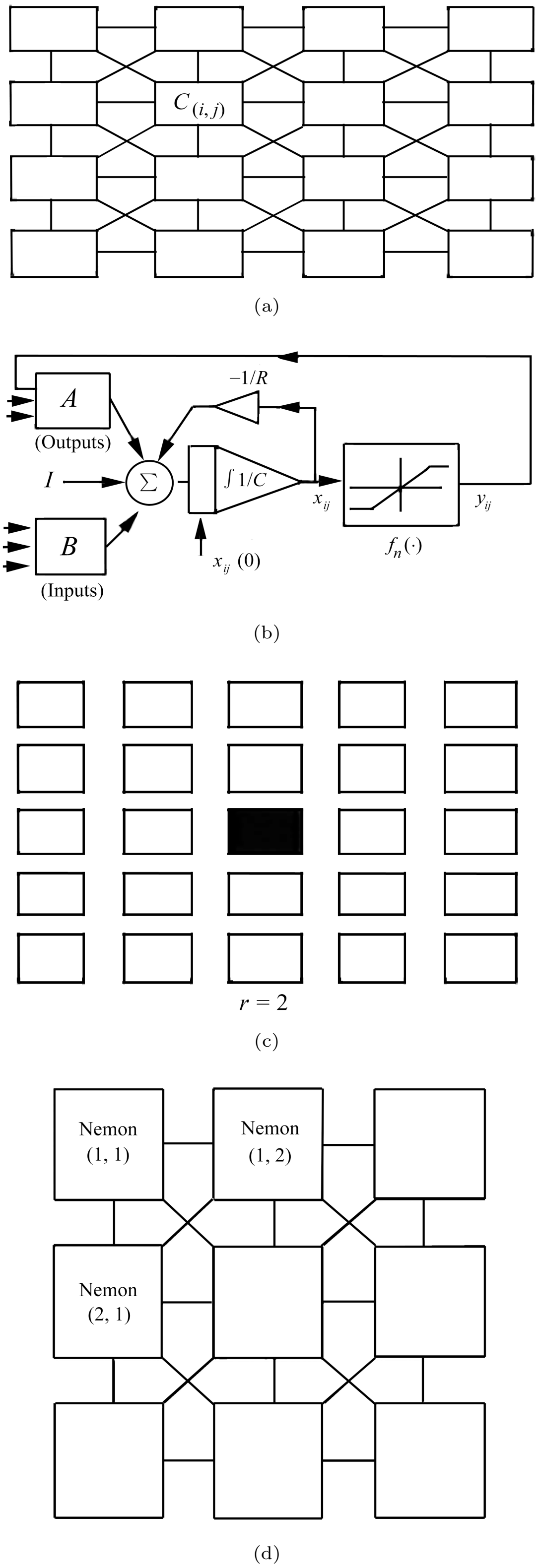
$$c\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t} = -\frac{1}{R}x_{ij}(t) + \sum_{c(k,l)\in N(i,j)} A(i,j;k,l)y_{kl}(t) + \sum_{c(k,l)\in N(i,j)} B(i,j;k,l)u_{kl}(t) + I,$$
$$1 \leqslant i \leqslant M; 1 \leqslant j \leqslant N \tag{24}$$

and the output equation is given by

$$y_{ij}(t) = \frac{1}{2}\left(|x_{ij}(t)+1| - |x_{ij}(t)-1|\right), \quad 1 \leqslant i \leqslant M; 1 \leqslant j \leqslant N. \tag{25}$$

where x_{ij} is the state of cell $C(i,j)$, $x_{ij}(0)$ is the initial condition of the cell, c is a linear capacitor, R is a linear resistor, I is an independent current source, $A(i,j;k,l)y_{kl}$ and $B(i,j;k,l)u_{kl}$ are voltage controlled current sources for all cells $C(k,l)$ in the neighborhood $N(i,j)$ of cell $C(i,j)$, and y_{ij} indicates the output equation. CNN is an analog nonlinear dynamic processor array, which is shown in Fig. 1 (a).

From (1), it is observed that the summation operators of each cell is affected by its neighboring cells. $A(\cdot)$ represents the output of neighboring cells and is called feedback operator, $B(\cdot)$ in turn affects the input control and is known as the control operator.

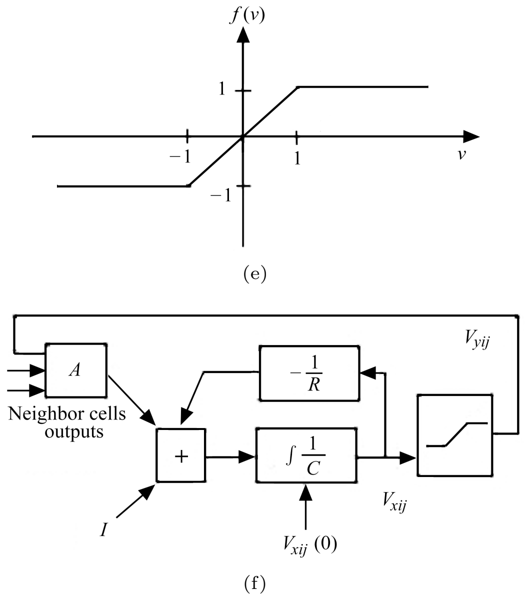


(a)

(b)

(c)

(d)



(e)

(f)

Fig. 1 CNN[1,2]. (a) Array structure; (b) Block diagram; (c) Neighborhood of cell $C(i,j)$ for $r = 2$; (d) Representation of 3×3 CNN; (e) The characteristic of the nonlinear controlled source; (f) Equivalent block diagram of a CNN cell

It is of interest to note that the entry values of matrices $A(\cdot)$ and $B(\cdot)$ are dependent on the application chosen by the user which are space invariant and are referred to as cloning templates. A current bias I and cloning templates establish the transient behavior of the cellular nonlinear network. A continuous-time cell implementation is shown in Fig. 1 (b) as an equivalent block diagram. CNNs have a set of analog values as input and its programmability is done via cloning templates. Fig. 1 (c) shows the neighborhood of cell $C(i,j)$ for $r = 2$, and Fig. 1 (d) is a representation of 3×3 CNN. Thus, programmability is one of the most attractive properties of CNNs.

4 Technical rationale of raster CNN simulations and performance

Equation (1) is space-invariant, which means that $A(i,j;k,l) = A(i-k, j-1)$ and $B(i,j;k,l) = B(i-k, j-1)$ for all i, j, k, l. Therefore, the solution of the system of difference equations can be seen as a convolution process between the image and the CNN processors. The fundamental approach is to imagine a square subimage area centered at (x,y), with the subimage being in the same size as the templates involved in the simulation. The center of this subimage is then moved from pixel to pixel starting, say, at the top left comer and applying A and B templates at each location (x,y) to solve the differential equation. This procedure is repeated for each time step, for all the pixels in the image. An instance of this image scanning-processing is referred to as “iteration”. The processing stops when it is found that the states of all CNN processors have converged to steady-state values, and the outputs of its neighbor cells are saturated, e.g., they have $a \pm 1$ value[1,2]. This whole simulating approach is referred to as raster simulation. Raster CNN simulation is an image scanning-processing technique for solving the system of difference equations of CNN. A simplified pseudo code presented below gives the exact notion of the raster/single layer approach[7].

The Pseudo code for image based raster behavioral CNN simulation is as follows

Step 1. During initial stage, get the input image, initial conditions, and templates from the end user.

```
/* M,N = Number of rows and columns of the 2D image */
while (converged-cells < total number of cells)
{
for ($i = 1$; $i \leqslant M$; $i++$)
for ($j = 1$; $j \leqslant N$; $j++$)
{
if (convergence-flag[$i$][$j$])
continue;
/* current cell already converged*/
```

Step 2. Calculate the next state

$$x_{ij}(t_{n+1}) = x_{ij}(t_n) + \int_{t_n}^{t_{n+1}} f'(x(t_n))\mathrm{d}t.$$

Step 2.1. Calculating by RK-embedded Heronian mean method.

$$\begin{aligned}
k_1 &= f(x_n, y_n) = k_1^* \\
k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_1}{2}) = k_2^* \\
k_3 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_2}{2}) \\
k_4 &= f(x_n + h, y_n + hk_3) \\
k_3^* &= f(x_n + \frac{1}{2}h, y_n - \frac{1}{48}hk_1 + \frac{25}{48}hk_2) \\
k_4^* &= f(x_n + h, y_n - \frac{1}{24}hk_1 + \frac{47}{600}hk_2 + \frac{289}{300}hk_3) \\
y_{n+1} &= y_n + h[\frac{k_2}{6} + \frac{k_3}{6} + \frac{2}{3}(\frac{k_1k_2}{k_1+k_2}) + \frac{2}{3}(\frac{k_3k_4}{k_3+k_4})].
\end{aligned}$$

Step 2.2. Calculating by RK-embedded harmonic mean method.

$$\begin{aligned}
k_1 &= f(y_n) \\
k_2 &= f(y_n + \frac{hk_1}{2}) \\
k_3 &= f(y_n - \frac{hk_1}{8} + \frac{5hk_2}{8}) \\
k_4 &= f(y_n - \frac{hk_1}{4} + \frac{7hk_2}{20} + \frac{9hk_3}{10}) \\
y_{n+1} &= y_n + h[\frac{k_2}{6} + \frac{k_3}{6} + \frac{2}{3}(\frac{k_1k_2}{k_1+k_2}) + \frac{2}{3}(\frac{k_3k_4}{k_3+k_4})].
\end{aligned}$$

Step 2.3. Calculating by RK-embedded root mean square method.

$$\begin{aligned}
k_1 &= f(x_n, y_n) = k_1^* \\
k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_1}{2}) = k_2^* \\
k_3 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_2}{2}) \\
k_4 &= f(x_n + h, y_n + hk_3) \\
k_3^* &= f(x_n + \frac{1}{2}h, y_n - \frac{1}{48}hk_1 + \frac{25}{48}hk_2) \\
k_4^* &= f(x_n + h, y_n + \frac{1}{8}hk_1 - \frac{17}{56}hk_2 + \frac{33}{28}hk_3).
\end{aligned}$$

Step 3. Check the convergence criteria.

$$\text{If } \left(\frac{\mathrm{d}x_{ij}(t_n)}{\mathrm{d}t}\right) = 0 \text{ and } y_{kl} = \pm 1, \forall c(k,l) \in N_r(i,j)$$

```
{
convergence-flag[$i$][$j$] = 1;
converged-cells++ ;
}
}
```

Step 4. Update the state values of the entire Image.

```
for ($i = 1$; $i \leqslant M$; $i + +$)
for ($j = 1$; $j \leqslant N$; $j + +$)
{
if (convergence-flag[$i$][$j$]) continue;
$x_{ij}(t_n) = x_{ij}(t_{n+1})$;
}
Number of iteration++;
}
End
```

For simulation purposes, a discretized form of (24) is solved within each cell to simulate its state dynamics. One common way of processing a large complex image is using a raster approach[1,2]. This approach implies that each pixel of the image is mapped onto a CNN processor[23]. That is, it has an image processing function in the spatial domain, expressed as

$$\phi(x, y) = \chi(\eta(x, y)) \tag{26}$$

where $\phi(\cdot)$ the processed image, $\eta(\cdot)$ is the input image, and χ is an operator on $\eta(\cdot)$ defined over the neighborhood of (x, y). It is an exhaustive process from the view of hardware implementation. For practical applications, in the order of 256×256 pixels, the hardware would require a large number of processors, which would make its implementation unfeasible. At this juncture, an alternative choice to this scenario is to make the image processing operator multiplex.

5 Integration techniques: numerical approaches

It is understood that a system of nonlinear differential equation is to describe the CNN. It implies that we need to discretize the differential equation for performing behavioral simulation. To perform computation, a normalized time differential equation for describing CNN is used by Nossek et al.[24]

$$\begin{aligned} f'(x(mT)) &= \frac{\mathrm{d}x_{ij}(mT)}{\mathrm{d}t} = -x_{ij}(mT) + \\ &\sum_{c(k,l)\in N_r(i,j)} A(i,j;k,l)y_{kl}(mT) + \\ &\sum_{c(k,l)\in N_r(i,j)} B(i,j;k,l)u_{kl} + I \\ &1 \leqslant i \leqslant M; 1 \leqslant j \leqslant N \end{aligned} \tag{27}$$

$$\begin{aligned} y_{ij}(mT) &= \frac{1}{2}\left(|x_{ij}(mT)+1| - |x_{ij}(mT)-1|\right), \\ 1 &\leqslant i \leqslant M; 1 \leqslant j \leqslant N \end{aligned} \tag{28}$$

where T is the normalized time. For the purpose of the initial-value problem, well established single step methods of numerical integration techniques are used[23,25].

These methods can be derived using the definition of the definite integral

$$x_{ij}((m+1)T) - x_{ij}(mT)) = \int_{T_m}^{T_{m+1}} f'(x(mT))\mathrm{d}(mT). \tag{29}$$

The RK methods are single step algorithms for the behavioral simulation of CNN, and they vary in the way they evaluate the integral presented in (4).

5.1 Fourth-order RK method based on embedded means

5.1.1 RK-embedded Heronian mean

The fourth-order RK-embedded Heronian mean[26] is given by

$$y_{n+1} = y_n + \frac{h}{3}[\frac{k_1+k_2}{2} + \frac{k_2+k_3}{2} + \frac{k_3+k_4}{2}] \tag{30}$$

$$\begin{aligned} y^*_{n+1} = y_n + \frac{h}{9}[k_1 + 2(k_2+k_3) + k_4 + \\ \sqrt{|k_1+k_2|} + \sqrt{|k_2+k_3|} + \sqrt{|k_3+k_4|}] \end{aligned} \tag{31}$$

where

$$\begin{aligned} k_1 &= f(x_n, y_n) = k^*_1 \\ k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_1}{2}) = k^*_2 \\ k_3 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_2}{2}) \\ k_4 &= f(x_n + h, y_n + hk_3) \\ k^*_3 &= f(x_n + \frac{1}{2}h, y_n - \frac{1}{48}hk_1 + \frac{25}{48}hk_2) \\ k^*_4 &= f(x_n + h, y_n - \frac{1}{24}hk_1 + \\ &\frac{47}{600}hk_2 + \frac{289}{300}hk_3). \end{aligned} \tag{32}$$

5.1.2 RK-embedded Harmonic mean

The fourth-order RK-embedded Harmonic mean[15] is given by

$$y_{n+1} = y_n + h[\frac{k_2}{6} + \frac{k_3}{6} + \frac{2}{3}(\frac{k_1k_2}{k_1+k_2}) + \frac{2}{3}(\frac{k_3k_4}{k_3+k_4})] \tag{33}$$

$$\begin{aligned} k_1 &= f(y_n) \\ k_2 &= f(y_n + \frac{hk_1}{2}) \\ k_3 &= f(y_n - \frac{hk_1}{8} + \frac{5hk_2}{8}) \\ k_4 &= f(y_n - \frac{hk_1}{4} + \frac{7hk_2}{20} + \frac{9hk_3}{10}). \end{aligned} \tag{34}$$

5.1.3 Present method: RK-embedded root mean square

The newly proposed fourth-order RK-embedded root mean square (see Section 2) is given by

$$y_{n+1} = y_n + \frac{h}{3}\left(\sqrt{\frac{k_1^2+k_2^2}{2}} + \sqrt{\frac{k_2^2+k_3^2}{2}} + \sqrt{\frac{k_3^2+k_4^2}{2}}\right) \tag{35}$$

$$\begin{aligned} y^*_{n+1} = y_n + \frac{h}{9}\Big(k_1 + 2(k_2+k_3) + \\ k_4 + \sqrt{|k_1+k_2|} + \sqrt{|k_2+k_3|} + \sqrt{|k_3+k_4|}\Big) \end{aligned}$$

where

$$\begin{aligned} k_1 &= f(x_n, y_n) = k^*_1 \\ k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_1}{2}) = k^*_2 \end{aligned}$$

$$\begin{aligned}
k_3 &= f(x_n + \frac{h}{2}, y_n + \frac{hk_2}{2}) \\
k_4 &= f(x_n + h, y_n + hk_3) \\
k_3^* &= f(x_n + \frac{1}{2}h, y_n - \frac{1}{48}hk_1 + \frac{25}{48}hk_2) \\
k_4^* &= f(x_n + h, y_n + \frac{1}{8}hk_1 - \\
&\quad \frac{17}{56}hk_2 + \frac{33}{28}hk_3). \qquad (36)
\end{aligned}$$

6 Discussion on simulation and experimental results

Using high power workstation, all the simulated outputs are presented below. It is observed that the actual CPU time used is equal to the simulation time. Figs. 2 (b) and (c) illustrate the results of the raster simulator obtained from a complex image of 256 × 256 pixels. The results of the raster simulator obtained from a complex image of 256 × 256 pixels are depicted, respectively in Figs. 2 (b) and (c) using embedded RK-Heronian mean and embedded RK-harmonic mean. On the original image, in order to get the images shown in Figs. 2 (b), (c), and (d), respectively, we have applied an averaging template followed by an edge detection template. It is observed in Figs. 2 (b), (c), and (d) that the edges obtained by RK-embedded root mean square are better than those obtained by embedded RK-heronian mean and embedded RK-harmonic mean. Furthermore, by raster CNN simulation the simulation took 185.43 s.

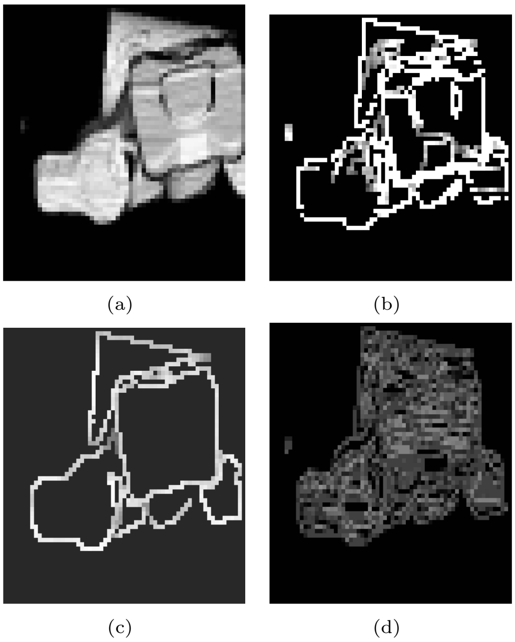


Fig. 2 Simulation and experimental results. (a) Original input image; (b) After averaging and edge detection templates by employing RK-embedded root mean square algorithm; (c) After averaging and edge detection templates by employing RK-embedded Heronian mean square algorithm; (d) After averaging and edge detection templates by employing RK-embedded harmonic mean algorithm.

It is seen from Fig. 3 that RK-embedded root mean square allows us to select a maximum step-size (ΔT) as compared to other two methods irrespective of the selection of templates. Fig. 4 shows that the importance of selecting an appropriate time step-size (ΔT). The results in Fig. 4 were obtained by simulating a small image of size 256 × 256 pixels using edge detection template on an image.

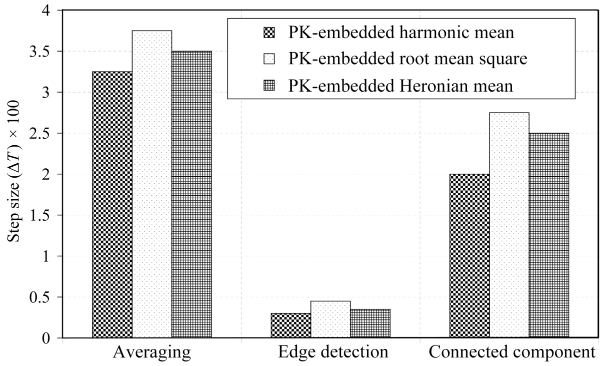


Fig. 3 Maximum step size (ΔT) yields the convergence for three different templates

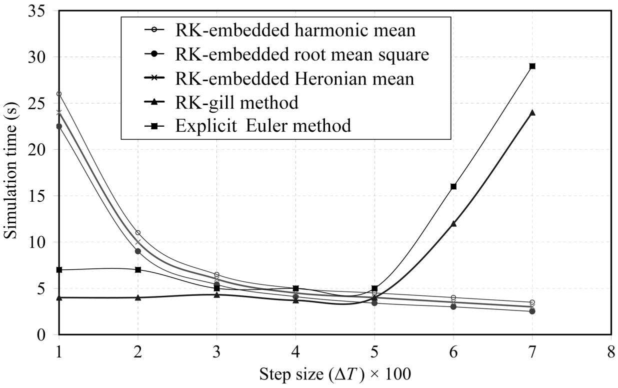


Fig. 4 Comparisons of five numerical integration techniques using the edge detection template

It is of significance to see in Fig. 4 that for a larger step-size (ΔT), RK-embedded root mean square takes shorter simulation time than embedded RK-Heronian mean and embedded RK-harmonic mean. Furthermore, if the step size chosen is too small, it might take many iterations, hence, it will take a longer time to achieve convergence. However, on the other hand, if the step size taken is too large, it might not converge at all, or it would converge to erroneous steady state values for the cases of adapting explicit Euler and RK-gill methods. One of the remarkable results of the present investigation is that even for a larger time step-size (ΔT), the embedded RK-root mean square gives better simulation results.

It is seen from Fig. 5 that the maximum step-size (ΔT) yields the convergence for three different templates using six different numerical integration techniques.

In Table 1, it can be observed that LTE, GTE, and error estimation of the proposed RK-embedded root mean square method are less than the RK-embedded Heronian mean and RK-embedded harmonic mean.

Table 1 Comparison of LTE, GTE, error estimation for different fourth-order four stage RK-embedded techniques

RK-embedded methods	LTE	GTE	Error estimation
RK-embedded root mean square	$LTE_{AM} - LTE_{RMS} \leqslant \frac{6457}{184320})$ $P^4Qh^5 = \left\|y^{AM}_{n+1} - y^{RMS}_{n+1}\right\| \leqslant \frac{6457}{184320}P^4Qh^5$	$\|\varepsilon_n\| \leqslant (\frac{h^4}{184320\,LD})\times$ $M(\mathrm{e}^{DL(x_n-x_0)} - 1)$	$ERREST = \|Y_{AM} - Y_{RMS}\|\,\frac{6457}{184320}$
RK-embedded Heronian mean	$LTE_{AM} - LTE_{HeM} \leqslant (\frac{121809}{1658880})$ $P^4Qh^5 = \left\|y^{AM}_{n+1} - y^{HeM}_{n+1}\right\| \leqslant \frac{121809}{1658880}P^4Qh^5$	$\|\varepsilon_n\| \leqslant (\frac{h^4}{1658880\,LD})\times$ $M(\mathrm{e}^{DL(x_n-x_0)} - 1)$	$ERREST = \|Y_{AM} - Y_{HeM}\|\,\frac{121809}{1658880}$
RK-embedded Harmonic mean	$LTE_{AM} - LTE_{HeM} \leqslant (\frac{5469}{69120})$ $P^4Qh^5 = \left\|y^{AM}_{n+1} - y^{HeM}_{n+1}\right\| \leqslant \frac{5469}{69120}P^4Qh^5$	$\|\varepsilon_n\| \leqslant (\frac{h^4}{69120LD})\times$ $M(\mathrm{e}^{DL(x_n-x_0)} - 1)$	$ERREST = \|Y_{AM} - Y_{HeM}\|\,\frac{5469}{69120}$

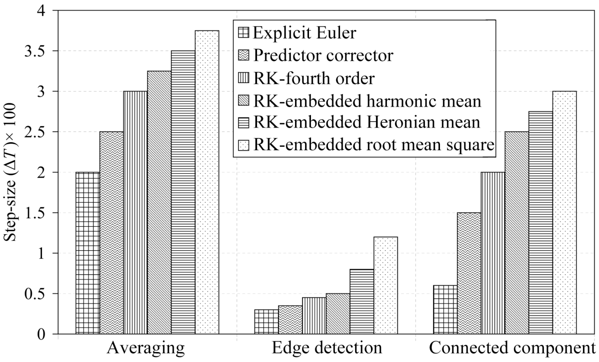


Fig. 5 Maximum step-size (ΔT) yields the convergence for three different templates using six numerical integration techniques

7 Conclusions

A novel integration algorithm by formulating an embedded method involving RK methods based on AM and RMS with error control for general CNNs is proposed. By using the newly proposed embedded method, a versatile algorithm for simulating raster CNN arrays are implemented for any kind as well as any size of input image. It is pertinent to pin-point out here that through simulation results and comparison of RK-embedded root mean square guarantees the accuracy of the detected edges and greatly reduces the impact of random noise on the detection results in comparison with the RK-embedded Heronian mean and RK-embedded harmonic mean. It is of interest to mention that using RK-embedded root mean square, the edges of the output images are proved to be feasible and effective by theoretic analysis and simulation.

References

[1] L. O. Chua, L. Yang. Cellular Neural Networks: Theory. *IEEE Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1257–1272, 1988.

[2] L. O. Chua, L. Yang. Cellular Neural Networks: Application. *IEEE Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1273–1290, 1988.

[3] L. O. Chua. *CNN: A Paradigm for Complexity*, World Scientific, 1998.

[4] T. Roska. CNNM Users Guide, Version 5.3x, Budapest, Hungary, 1994.

[5] T. Roska, L. O. Chua. The CNN Universal Machine: An Analogic Array Computer. *IEEE Transactions on Circuits Systems II: Analog and Digital Signal Processing*, vol. 40, no. 3, pp. 163–173, 1993.

[6] R. C. Gonzalez, R. E. Woods, S. L. Eddins. *Digital Image Processing Using MATLAB*, Pearson Education Asia, 2005.

[7] C. C. Lee, J. P. De Gyvez. Single-layer CNN Simulator. In *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 6, pp. 217–220, 1994.

[8] V. Murugesh, K. Murugesan. Comparison of Numerical Integration Algorithms in Raster CNN Simulation. *Lecture Notes in Computer Science*, vol. 3285, pp. 115–122, 2004.

[9] Y. Z. Lu. A Novel Face Recognition Algorithm for Distinguishing Faces with Various Angles. *International Journal of Automation and Computing*, vol. 5, no. 2, pp. 193–197, 2008.

[10] M. Xiao, C. Z. Han, L. Zhang. Moving Shadow Detection and Removal for Traffic Sequences. *International Journal of Automation and Computing*, vol. 4, no. 1, pp. 38–46, 2007.

[11] Y. Koda, I. Kanaya, K. Sato. Modeling Real Objects for Kansei-based Shape Retrieval. *International Journal of Automation and Computing*, vol. 4, no. 1, pp. 14–17, 2007.

[12] R. H. Merson. An Operational Method for the Study of Integrating Processes. In *Proceedings of the Symposium on Data Processing, Weapon Research Establishment*, Salisbury, Australia, 1957.

[13] E. Fehlberg. Low Order Classical Runge Kutta Formulas with Step-size Control and Their Application to Some Heat Transfer Problems, NASA Technical Report 315, George C. Marshall Space Flight Center, Marshall, Australia, 1969.

[14] D. J. Evans, A. R. Yaakub. A New Runge Kutta RK (4,4) Method. *International Journal of Computer Mathematics*, vol. 58, no. 3, pp. 169–187, 1995.

[15] N. Yaacob, B. Sanugi. A New Fourth-order Embedded Method Based on the Harmonic Mean. *Mathematika*, vol. 14, pp. 1–6, 1998.

[16] D. J. Evans, N. Yaacob. A Fourth order Runge-Kutta Method Based on the Heronian Mean Formula. *International Journal of Computer Mathematics*, vol. 58, no. 1, pp. 103–115, 1995.

[17] P. Henrici. *Discrete Variable Methods in Ordinary Differential Equations*, John Wiley & Sons, New York, USA, 1962.

[18] J. D. Lambert. *Computational Methods in Ordinary Differential Equations*, John Wiley & Sons, New York, USA, 1973.

[19] J. D. Lambert. *Stiffness in Computational Techniques for Ordinary Differential Equations*, I. Gladwell, D. K. Sayers (ed.), Academic Press, London, UK, pp. 19–46.

[20] M. Lotkin. On the Accuracy of RK Methods. *MYAC*, vol. 5, pp. 128–132, 1951.

[21] R. H. Ralston. Runge-Kutta Methods with Minimum Error Bonds. *Mathematics and Computer*, vol. 16, pp. 431–437, 1957.

[22] A. R. Yaakub, D. J. Evans. A New Fourth Order Runge-Kutta Method Based on the Root Mean Square Formula, Computer Studies Report No. 862, Lappeenranta University of Technology, Finland, 1993.

[23] L. O. Chua, T. Roska. The CNN Universal Machine Part 1: The Architecture. In *Proceedings of International Workshop on Cellular Neural Networks and Their Applications*, pp. 1–10, 1992.

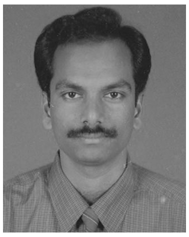
[24] K. K. Lai, P. H. W. Leong. Implementation of Time-multiplexed CNN Building Block Cell. In *Proceedings of 5th International Conference on Microelectronics for Neural Networks*, IEEE Press, Lausanne, Switzerland, pp. 80–85, 1996.

[25] K. K. Lai, P. H. W. Leong. An Area Efficient Implementation of a Cellular Neural Network. In *Proceedings of the 2nd New Zealand International Two-stream Conference on Artificial Neural Networks and Expert Systems*, pp. 51–54, 1995.

[26] R. Ponalagusamy, S. Senthilkumar. Multilayer Raster CNN Simulation by Arithmetic and Heronian Mean RKAHeM(4,4). In *Proceedings of International Conference of Information Engineering, World Congress on Engineering*, London, UK, IAENG Journal and Book, ISBN: 978-988-98671-5-7, and *Lecture Notes in Engineering and Computer Science*, vol. 1, pp. 713–718, 2007.

R. Ponalagusamy received the M. Sc. degree in applied mathematics from Madurai Kamaraj University, India in 1981. He received the Ph. D. in computational biomechanics from the Indian Institute of Technology, Bombay, India in 1986. The Environmental Research Corporation, Tokyo, Japan, awarded him a post-doctoral fellowship for two years (1987–1988) at Chuo University, Tokyo, Japan. During June, 1989 to June, 1996, he was working as a lecturer and doing research in the field of computational fluid mechanics. During July, 1996 to June, 2005, he was working as an assistant professor and doing research in the Development of Mathematical and Computer Models in the field of processing of advanced materials, namely 1) computer aided metal flow analysis in streamlined extrusion die, 2) formability, localized, and siffusion necking, 3) wrinkling tendency of sheets metals and materials, and 4) development of yield criterion for sheet metals. From July, 2005 to till date, he has been working as a professor and guiding research students in the field of development of parallel algorithm. He is also guiding research students in the fields of parallel numerical computing and digital image processing. He has been one of the regional editors in several international Journals. He is editor-in-chief for *International Journal of Mathematics and Engineering with Computers* (IJMAEC). He received Best Teacher Award for 2007–2008 from National Institute of Technology, Tiruchirappalli, Tamilnadu, India. He has written two books on theory of engineering plasticity and engineering mathematics. He is a life member in Indian Society of Theoretical and Applied Mechanics and Indian Society of Technical Education. He is one of the members of the World Congress in Computer Science, Computer Engineering, and Applied Computing, Las Vegas, Nevada, USA, during 2005–2008.

His research interests include computational experimentation on biofluid mechanics, plastic flow, finite element method, computer models on metal forming and powder metallurgical materials, wavelets, digital image processing, and development of parallel algorithms.



Sukumar Senthilkumr received his B. Sc. degree in mathematics from Madras University, India, M. Sc., degree in mathematics from Bharathidasan University, India in 1996, M. Phil degree in mathematics from Bharathidasan University in 1999, and M. Phil degree in computer science and engineering from Bharathiar University in 2000. Also, he received PGDCA and PGDCH in computer applications and computer hardware from Bharathidasan University in 1996 and 1997, respectively. He is currently a Ph. D. candidate in the Department of Mathematics at National Institute of Technology, Tiruchirappalli, Tamilnadu, India. He served as a lecturer, in the Department of Computer Science at Asan Memorial College of Arts and Science, Chennai, Tamilnadu, India.

His research interests include digital image processing and numerical methods.