Integer Lattice Gas Automata for Computational Electromagnetics

Joanne R. Treurniet, Neil R. S. Simons, and Greg E. Bridges, *Member, IEEE*

***Abstract*—Integer lattice gas automata (ILGA) are combined with the transmission-line matrix (TLM) method to yield a new electromagnetic-field computation algorithm using very low-precision integer variables. Lattice gas automata can be evaluated using look-up tables on special-purpose hardware and do not require floating-point arithmetic. In this paper, we present a TLM motivated ILGA with emphasis placed on algorithms that demonstrate minimal dissipation.**

***Index Terms*—Computational electromagnetics, lattice gas automata, TLM.**

I. Introduction

TRANSMISSION-LINE matrix (TLM) methods are widely used in the numerical solution of Maxwell's equations [1]. The evaluation of the TLM method and other electromagnetic (EM) analysis techniques is based on the use of floating-point processors. Lattice gas automata (LGA) have been investigated as an alternative approach for the solution of partial differential equations [2], [3]. LGA can be evaluated using look-up tables on special-purpose hardware and do not require floating-point arithmetic. A generalization of LGA using low-precision integer variables as opposed to single-bit variables have also been proposed [4] and are referred to as integer lattice gas automata (ILGA). We will show that we can combine the optimal features of both of these algorithms into a "TLM-motivated" algorithm for the solution of EM field problems. Special-purpose computer architectures such as the CAM-8 cellular automata (CA) machine [5] can be used for the efficient evaluation of this algorithm.

The straightforward conversion of the TLM algorithm to an integer arithmetic environment is not possible. Roundoff error introduces statistical noise that grows with the square root of time. This noise can eventually become larger than the desired signals within the computation. In an integer-based algorithm, a dissipative behavior is also present. To reduce the dissipative errors, we have developed an algorithm based on low-precision integer arithmetic (ILGA) rather than single-bit variables (LGA) in order to implement TLM-like scattering events. By enforcing conservation laws for fields and energy in our algorithm, results show that the dissipation can be significantly reduced using only 4-bit integer variables.

This paper will outline the algorithm and present the results of a benchmark test of a two-dimensional waveguide structure.

Manuscript received January 5, 1999.

J. R. Treurniet was with the Communications Research Center, Ottawa, Ont., Canada K2H 8S2. She is now with Ionizing Radiation Standards, National Research Council, Ottawa, Ont., Canada KIA 0R6 (e-mail: treurnie@irs.phy.nrc.ca).

N. R. S. Simons is with the Communications Research Center, Ottawa, Ont. Canada K2H 8S2 (e-mail: neil.simons@crc.ca).

G. E. Bridges is with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, Man., Canada R3T 2N2 (e-mail: bridges@ee.umanitoba.ca).

Publisher Item Identifier S 0018-9480(00)04661-5.

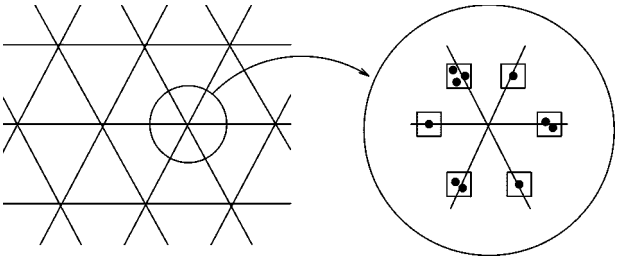


Fig. 1. ILGA using $M = 2$ bits per direction on a hexagonal lattice. Each direction can hold as many as $2^M - 1$ particles, where M is the number of bits. The distance between nodes along each of the three mesh directions is Δl [8].

II. Hexagonal TLM Method

Although the TLM algorithm may, in principle, be applied to different lattices [6], [7], we base this discussion on a hexagonal lattice. The propagation characteristics of the hexagonal mesh are more isotropic than those of a rectangular mesh for both TLM and finite-difference algorithms [8]. The TLM operation for voltage propagating on a hexagonal lattice (shown in Fig. 1) is given by [8]

$$\begin{pmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \end{pmatrix}^{t+1} = \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 & -2 & 1 & 1 \\ 1 & 1 & 1 & 1 & -2 & 1 \\ 1 & 1 & 1 & 1 & 1 & -2 \\ -2 & 1 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 1 & 1 & 1 \\ 1 & 1 & -2 & 1 & 1 & 1 \end{bmatrix} \begin{pmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \end{pmatrix}^{t}$$

or

$$[V]^{t+1} = C_{\mathrm{TLM}}[V]^t \tag{1}$$

where C_{TLM} is the scattering matrix. All variables are represented by a floating-point value and the update operation requires floating-point operations. Note that we have presented the scattering matrix here in LGA notation, so that incident and reflected voltages exchange indexes symmetrically about the origin of the node. Equation (1) applies to a single node on the lattice. In one time step, this equation is simultaneously applied to all nodes on the lattice.

III. ILGA

CA have the property of being completely discretized, in space, time, and dependent variable [9]. This differs from

traditional TLM, finite-difference and finite-element schemes, which are discretized in time and space, but continuous in dependent variable. The dynamics of a cellular automaton are specified by a "rule" that determines what the state of the system will be at time step $t + \Delta t$ based on its state and the state of its neighbors at time t.

LGA are special types of CA for which there is some association between the state at a site and that site's neighbors on the lattice and for which the rule that determines collisions at a node obeys some conservation law, e.g., mass or momentum [2]. A time step is composed of two phases. The first is known as the "collision" phase. It is in this phase that the rule is applied. The second phase is known as "streaming" (or "advection") phase, in which all particles on the lattice move to adjacent sites (i.e., a westbound particle moves one step west). These particles may be thought of as variables moving about the lattice. In the usual model, the state at a node may be described in terms of single-bit variables. If a node has one of its directions occupied, the bit for that direction is turned "on" (set to one); if it is unoccupied, that bit is turned "off" (set to zero). These models may also allow rest particles that can be used to change the propagation velocity [10].

For an ILGA, the state at a node may be described in terms of multiple-bit variables [4], i.e., a site may have more than one particle traveling in any direction. An integer then represents the number of particles in a given direction, as opposed to only 1 bit in the standard LGA. Fig. 1 shows the ILGA on a hexagonal lattice using 2 bits per direction, thus allowing 0–3 particles to exist per direction at a site.

IV. TLM-Motivated ILGA Collision Rule

For an integer variable hexagonal lattice, we specify the state of a cell by a vector of six integers, $[N]$, one for each direction. If M-bit integer variables are used, up to $2^M - 1$ particles per direction are allowed, $0 < N_i < 2^M - 1$. The TLM operation may be applied as an ILGA collision rule if truncation due to the use of integer arithmetic is accounted for [11]. For the hexagonal lattice of Fig. 1, a collision rule is constructed by splitting the number of particles in each direction into a quotient, i.e., N_Q, which is divisible by three and a remainder N_R. Specifying $[N] = [N_Q] + [N_R]$, the TLM collision operation (1) can then be exactly applied to N_Q without roundoff. The ILGA collision operation is defined by applying the TLM operator to N_Q and a streaming operator to N_R as

$$[N_R]^t = \mathrm{mod}\{[N]^t, 3\} \tag{2}$$

$$[N_Q]^t = [N]^t - [N_R]^t \tag{3}$$

$$[N]^{t+1} = C_{\mathrm{TLM}}[N_Q]^t + I_6[N_R]^t \tag{4}$$

where C_{TLM} was defined in (1), and I_6 is the 6×6 identity matrix. Note that the collision operation as applied by (4) does not simply imply rounding down to the nearest integer divisible by three. The operation exactly conserves the total number of particles and the particle momentum (and, thus, local E- and H-fields). Local energy density is not exactly conserved, but is approximately conserved since the particles undergoing the TLM operation conserve energy density and the streamed particles act as a perturbation to this. The nonexact conservation of energy in the model leads to a dissipation of fields as they propagate in the lattice and is discussed later in the paper. Alternate operators, in place of the trivial streaming operator I_6, that still obey the same conservations can also be found, such as the single-bit FHP rule [10] used in fluid dynamics models.

TABLE I
Percentage of States Involved in the TLM Operation as a Function of Number of Bits

Bits	% TLM
2	7.74
3	55.44
4	83.55
5	98.00

The operation (4) handles roundoff due to integer arithmetic, but does not handle possible overflow or underflow of variables. All variables before and after collision must be in the range $0 < N_i < 2^M - 1$. If the operation defined by (4) does not satisfy this condition, we alter the original state by a fractional amount α_k as

$$[N']^t = \mathrm{int}\{\alpha_k[N]^t\} \tag{5}$$

$$[N'']^t = [N]^t - [N']^t \tag{6}$$

$$\alpha_k = k/(2^M - 1) \tag{7}$$

where $k \epsilon \{0, 1, 2, \cdots, 2^M - 1\}$ and $\mathrm{int}\{x\}$ is the integer portion of x. A streaming operation is then applied to N' and the TLM operation is applied to the remaining N'' following the rule defined in (4) as

$$[N]^{t+1} = C_{\mathrm{TLM}}\left[N''_Q\right]^t + I_6\left[N''_R\right]^t + I_6\left[N'\right]^t. \tag{8}$$

The procedure (5)–(7) is repeated for $k = 0, 1, 2$ to $2^M - 1$ until a valid collision operator is obtained where all $0 < N_i < 2^M - 1$.

When the number of bits M used in the model is small, collisions will be dominated by the streaming operation. As the number of bits increases, however, the proportion of particles undergoing a TLM operation increases and the proportion of particles streamed decreases. Table I gives the percentage of particles that undergo a TLM collision event for $M = 2, 3, 4, 5$ bit variables. Table I was determined by summing the proportion of particles that undergo a TLM collision for all possible input vector combinations and weighting these by the probability of their occurrence. We observe that, for even 5-bit variables, almost all collision events obey the TLM operation. For 5-bit variables, energy density is almost exactly conserved, and we expect that the dissipation will be very small. This is examined in Section VI.

V. Look-Up-Table-Based Computation

Since our model is entirely integer based, a look-up table can be used to describe the state of a node after a collision occurs. If

one allows a long integer (32 bits) to represent the state of one lattice site, then each of the six directions at a node would have up to 5 bits available to specify the number of particles in that direction. For 4 bits per direction, for example, a 24-bit integer describes the state. A simple look-up table can then be configured to directly correspond to the integer representing the new state of the node. For M bits per direction, there are 2^{6M} possible states available, hence, for 4 bits per direction, a $4 \times 2^{24} = 67.1 \times 10^6$ byte look-up table is required. Using 5 bits per direction becomes impractical since the look-up table requires 4.29×10^9 bytes. In this case, the collisions can be determined without the use of the look-up table, i.e., by determining collisions as the simulation proceeds, but at a great sacrifice to speed.

One of the motivations for using the LGA approach is that it is amenable to implementation on a fine-grain parallel architecture, such as the cellular automata machine 8 (CAM-8) [5]. The CAM-8 contains one or more modules containing 10^6 16-bit cells along with the hardware for updating them. Each cell represents a node on the lattice. Therefore, a square lattice can hold as many as 15 particles in each of its four directions [11]. The CAM-8 is capable of performing 200×10^6 site updates per second on spaces consisting of 32×10^6 sites. For a hexagonal lattice, the CAM-8 is not as desirable due to the extra number of particles representing the state and, therefore, the large look-up table size. The 16 bits would only yield a maximum of 2 bits per direction for this approach, which may not sufficiently reduce the dissipation (see Table I).

CAM-8 operates on 16 bits per site at any instant of time. Implementations involving cell sizes of more than 16 bits can be accommodated by parsing the particle interactions into 16-bit operations. It is desirable to minimize the number of 16-bit operations required. In general, an N-bit collision process requires a 2^N-sized look-up table. If an N-bit collision operator ($N = K + 16$) look-up table is to be parsed in a brute-force manner into 16-bit operations, 2^K look-up tables are required. It is, therefore, desirable to exploit any symmetries or factorizations to parse a collision operator involving more than 16 bits. For example, in [12], the implementation of a face-centered hyper-cubic (FCHC) LGA on CAM-8 is described. The FCHC LGA requires 24 particles per cell, however, Alder, *et al.* were able to split the 24-bit collision process into two 16-bit collision events.

Evaluation of the algorithm as implemented on the hexagonal lattice was performed using a C program, while evaluation of the algorithm as implemented on the square lattice was performed using CAM-8 [11].

VI. Results

It is common to characterize an LGA by determining its equilibrium properties. One way of characterizing the state of a lattice at a particular time step is to accumulate the occurrence of every possible state for N_i over the entire lattice. Once these accumulated states remain (relatively) constant versus time, equilibrium has been reached. Usually this occurs after tens of time steps for the ILGA investigated here.

To show that the TLM method is approximated by the ILGA algorithm, the equilibrium distributions were determined for both models. Fig. 2 compares the equilibrium distributions of particles as a fractional occupation for the ILGA with 4 bits per direction and for the floating-point TLM method. The TLM equilibrium distribution was obtained through simulation of a randomly initialized TLM mesh. Voltage pulses within a TLM mesh were initialized using a uniform distribution from 0 to 1. The TLM space using these random initial values was simulated for several thousand time steps in order to obtain an equilibrium distribution of voltages. The floating-point TLM equilibrium voltage values were binned using 0.0625 intervals. If a particle is interpreted as a packet of voltage, their distributions are similar. The tails of the ILGA distribution must necessarily be cut off at zero and $2^M - 1$, which is the main source of the deviation in the distribution from the TLM model.

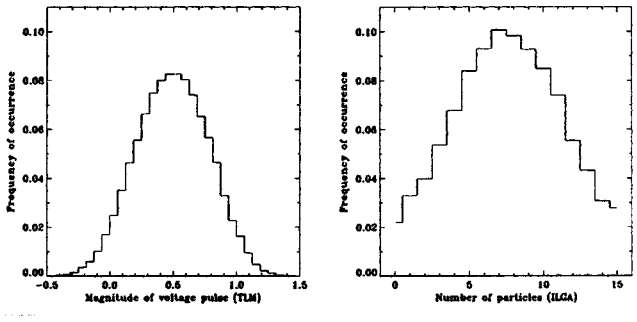


Fig. 2. Comparison of the equilibrium distributions for the two methods. The TLM distribution is similar to that of the ILGA.

An inherent dissipation is present within simulations using the TLM-motivated ILGA. The streamed particles (discussed in Section IV) are similar to those seen in ILGA simulations of fluid dynamics, in which a viscosity is inherent. These do not obey TLM-like energy conserving collisions. The dissipation is analogous to a viscosity, the effect of which can be determined by solving the Navier–Stokes equation to give the variation in density of particles with time $\delta\rho$ as [13]

$$\delta\rho \propto \exp\left(-k^2\nu t\right) \tag{9}$$

where $k = 2\pi/\lambda$ defines the wavenumber and ν is the viscosity. If the viscosity is zero, we obtain lossless behavior. Hence, the validity of the ILGA model as applied to electromagnetics is dependent on showing that the viscosity can be eliminated in some manner. Quantification of the viscosity has been performed through derivation of the dispersion relation of the method [14].

As the number of bits allocated per direction increases, the proportion of states that obey TLM-like collision operations increases, as seen in Table I, and the model approximates the floating-point TLM method. We, therefore, expect the effect of damping to decrease as the number of bits increases. In addition, from (9), we expect the effect of the damping to disappear as $\Delta l/\lambda$ approaches zero (the grid spacing becomes infinitesimal) since $k = 2\pi\Delta l/\lambda$.

To study the effect of damping in our ILGA algorithm and its dependence on the number of bits used to represent the variables, we have simulated a two-dimensional cross section of a rectangular waveguide of dimension $a \times b$.

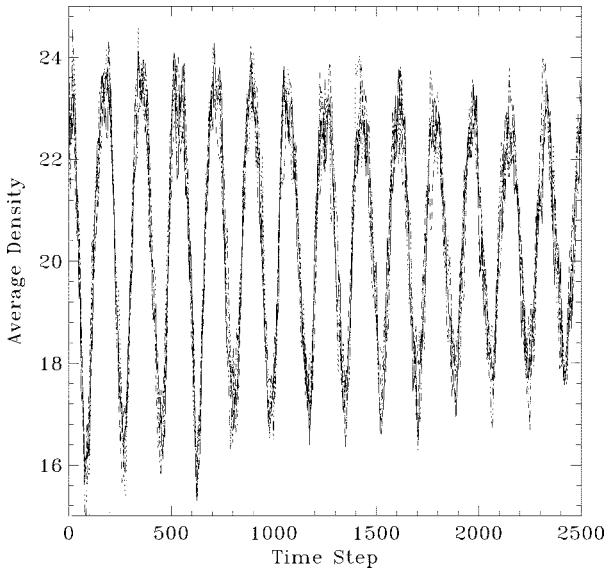


Fig. 3. Five trials for an FHP collision rule [10] for 3 bits per direction, initialized to a $(1,0)$ mode. The y-axis is the average density of particles in the sample space.

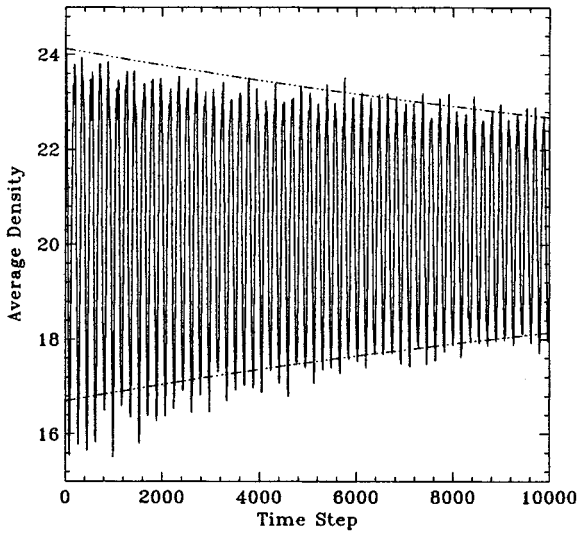


Fig. 4. Averaged data and envelope curve using 3 bits per direction, initialized to a $(1,0)$ mode.

The TE propagation equations for a waveguide operating in an $(m,\ n)$ mode are described by [15]

$$H_z = A\ \cos\left(\frac{m\pi x}{a}\right)\ \cos\left(\frac{n\pi y}{b}\right) e^{-\gamma z}$$
$$E_z = 0 \tag{10}$$

where A is a constant, and E_z and H_z are the z components of the electric- and magnetic-field strengths, respectively. If the density of particles is taken to be proportional to the field strength, an $a \times b$ lattice may be initialized to reflect the field strength. The cutoff frequency of a particular mode is

$$f_{m,n} = \frac{c_s}{2}\sqrt{\left(\frac{m}{a}\right)^2 + \frac{4}{3}\left(\frac{n}{b}\right)^2} \tag{11}$$

where $c_s = \Delta l/(\sqrt{2}\,\Delta t)$. The 4/3 term is due to the asymmetry of a hexagonal grid in the x and y dimensions (see Fig. 1). The distance between grid points in the x-direction is Δl, however, the distance between grid lines in the y-direction is $\sqrt{(3)}\Delta l/2$. In (10), x and y are expressed in terms of Cartesian coordinates, while a and b in (11) refer to the number of grid points in the x- and y-directions, respectively.

The simulation size was $a = b = 64$, corresponding to a physical space of $64 \times (\sqrt{3}/2)64$. The sample space was a 10×10 section in a region containing a maximum at initialization. Several different modes were tested, in particular $(m,\ n) = (0,1)$, (1,0), (1,1) and (2,1), for up to 5 bits per direction. To excite the lattice, the probability p of filling a bit at a node was defined as

$$p = 0.5 + 0.1\ \cos\left(\frac{m\pi x}{a}\right)\ \cos\left(\frac{n\pi y}{b}\right) \tag{12}$$

to achieve a 50% density perturbed according to (10). Five different initial distributions in H_z were used.

Fig. 3 shows the density versus time step for five different initial $(1,0)$-mode distributions, using a traditional LGA rule that conserves mass and momentum, but does not attempt to optimally apply the C_{TLM} collision operator. This can be considered as analogous to the FHP rule [10] using 3 bits per direction, as applied in [4]. The results are noisy due to the small grid used. An increase in grid size to 128×128 considerably reduces the noise, but requires four times the computation time. We use ensemble averaging over five trials in our results. In the remainder of this paper, we present data obtained from application of the ILGA described in Section IV.

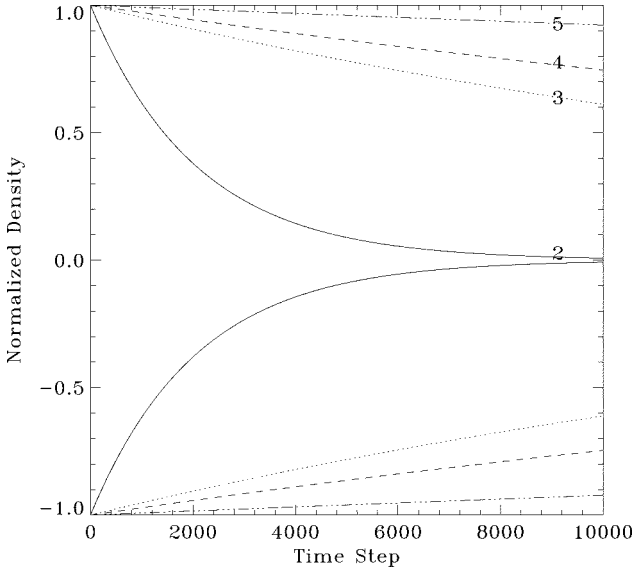


Fig. 5. Envelope curves for 2–5 bits per direction, initialized to a $(1,0)$ mode. Note that since the amplitudes and average values increase with the number of bits used, these curves have been normalized such that they all lie between -1 and $+1$.

To obtain a viscosity from these curves, an average of the five data sets was taken and the result fit to the analytical expression

$$H_z(t) = A\ \sin(\omega t + \phi)\ \exp(-Bt) + C \tag{13}$$

TABLE II
VISCOSITY FOR ALL RULES AND ALL MODES

Bits	ν_{01}	ν_{10}	ν_{11}	ν_{21}
2	0.20±0.01	0.20±0.02	0.18±0.02	0.14±0.03
3	0.0197±0.0007	0.0204±0.0009	0.0175±0.0005	0.0154±0.0005
4	0.0093±0.0004	0.0122±0.0005	0.0048±0.0002	0.0054±0.0002
5	0.00580±0.00009	0.0034±0.0001	0.00667±0.00007	0.00421±0.00004

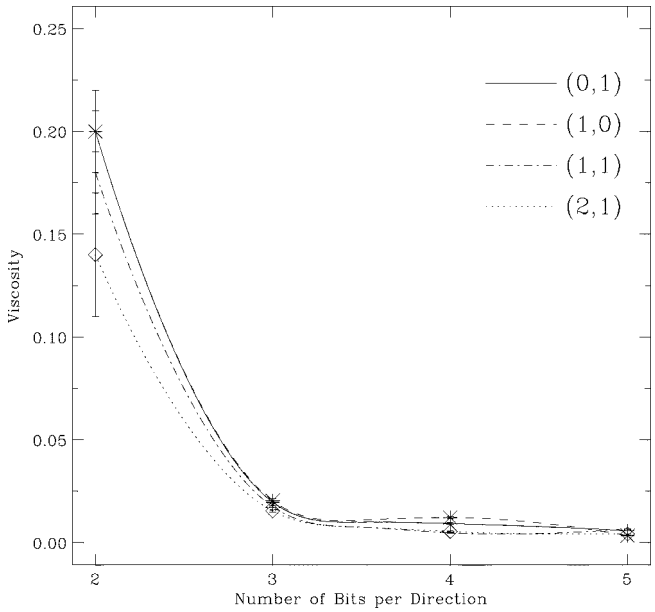


Fig. 6. Viscosity as a function of number of bits for the $(1,0)$, $(0,1)$, $(1,1)$, and $(2,1)$ modes.

using the Levenberg–Marquardt method [16]. Here, $B = k^2\nu$ and ν is the viscosity. The uncertainty on each data point was taken as the standard deviation.

For the $(1,0)$ mode, the raw data is shown with the exponential envelope curve for 3 bits per direction in Fig. 4. The transient response is provided for over 50 periods. The raw data shown is the average over the five data sets obtained for different random number generator seeds. Fig. 5 shows the exponential envelope curves for a $(1,0)$ mode for 2–5 bits per direction. As expected, the damping decreases as the number of bits increases. In addition, the damping observed in simulations of the ILGA presented in this paper is significantly less than that observed in simulations of the ILGA of [4].

The viscosity values calculated from the fit parameters are given in Table II for $(1,0)$, $(0,1)$, $(1,1)$, and $(2,1)$ modes and shown graphically in Fig. 6. These modes correspond to plane-wave propagation through the mesh at angles of 0°, 90°, 45°, and 26.6° to the x-axis, respectively. These results show that the viscosity is slightly anisotropic, which is unexpected for a hexagonal lattice.

Results were also obtained using 3 bits per direction for the $(2,0)$ and $(4,2)$ modes for comparison to the $(1,0)$ and $(2,1)$ modes, respectively. A 128 × 128 lattice was used for these higher modes since the results decay rapidly on a smaller lattice. For these cases, $\nu_{20} = 0.024 \pm 0.001$ and $\nu_{42} = 0.017 \pm 0.002$, showing good agreement with the $(1,0)$ and $(2,1)$ results of $\nu_{10} = 0.0204 \pm 0.0009$ and $\nu_{21} = 0.0174 \pm 0.0007$. A $(3,0)$ mode gives $\nu_{30} = 0.026 \pm 0.007$. We have observed that the uncertainty in the viscosity increases considerably for higher modes.

VII. DISCUSSION

We have developed an algorithm that successfully combines TLM and ILGA methods for the numerical solution of EM field problems. The dissipative error encountered is significantly reduced for larger numbers of bits. In [14], the dispersion relation of the algorithm is derived in order to quantify the dissipative errors.

The algorithm's total discretization of time, space, and dependent variable has the advantage of implementation on a CAM-8 style processor, thus eliminating the need for a floating-point processor. Since the model is entirely integer-based, a look-up table can be used to describe the state of a node after a collision occurs, resulting in a relatively small computational requirement to update the individual cells.

As a preliminary investigation into the relative computational effort (CE) required for the TLM-motivated ILGA, we use the method presented in [17] to compare to the standard TLM, using a square lattice. In two dimensions, for physical dimensions $D_x \times D_y$, the number of cells is $D_x/\Delta l_x$ and $D_y/\Delta l_y$, respectively. The relative CE in two dimensions is then

$$\frac{\mathrm{CE_{TLM}}}{\mathrm{CE_{ILGA}}} = \frac{N^t_{\mathrm{TLM}} N^x_{\mathrm{TLM}} N^y_{\mathrm{TLM}} N^{\mathrm{ops}}_{\mathrm{TLM}}}{N^t_{\mathrm{ILGA}} N^x_{\mathrm{ILGA}} N^y_{\mathrm{ILGA}} N^{\mathrm{ops}}_{\mathrm{ILGA}}} \tag{14}$$

where N^t is the number of time steps, $N^{x,y}$ is the number of cells in the x or y dimension, and N^{ops} is the number of operations required per cell per time step. The time steps, i.e., $\Delta t = \Delta l/\sqrt{(2)}$, are the same for both algorithms in two dimensions. From our numerical experiments, we have found that 2–3 TLM-motivated ILGA cells per dimension are required (at 4 bits per direction) for every cell in the TLM method per dimension to achieve the same degree of accuracy. Taking the worst case, $\Delta l_{\mathrm{TLM}} = 3\Delta l_{\mathrm{ILGA}}$. We obtain

$$\frac{\mathrm{CE_{TLM}}}{\mathrm{CE_{ILGA}}} = \left(\frac{\Delta l_{\mathrm{ILGA}}}{\Delta l_{\mathrm{TLM}}}\right)^3 \frac{N^{\mathrm{ops}}_{\mathrm{TLM}}}{N^{\mathrm{ops}}_{\mathrm{ILGA}}} \tag{15}$$

$$\frac{\mathrm{CE_{TLM}}}{\mathrm{CE_{ILGA}}} = \left(\frac{1}{27}\right) \frac{N^{\mathrm{ops}}_{\mathrm{TLM}}}{N^{\mathrm{ops}}_{\mathrm{ILGA}}}. \tag{16}$$

For a square lattice, the TLM method requires 16 floating-point multiplications and 12 additions per cell at each time step. However, the ILGA requires only a single look-up-table operation per time step. The ILGA can be evaluated using special-purpose

hardware such as CAM-8 where parallelism is easily and inexpensively realized. A comparison of actual computation speeds would require comparison of a floating point architecture with a look-up-table-based computing architecture.

To solve the most general EM-field problems, an algorithm such as the three-dimensional symmetrical condensed node (SCN) TLM is required [18], as well as graded meshes and permittivity stubs. The general form of these scattering matrices contains real number entries, not easily divisible by integers. We are currently investigating the applicability of the approach described here to these more complex algorithms. An LGA (using single-bit variables) for solving Maxwell's equations in three dimensions is described in [19]. The algorithm described here is compatible with the three-dimensional LGA.

REFERENCES

[1] C. Christopoulos, *The Transmission Line Modeling Method (TLM)*. Piscataway, NJ: IEEE Press, 1993.

[2] B. Boghosian, "Lattice gases," in *1989 Lectures in Complex Systems, SFI Studies in the Sciences of Complexity*, E. Jen, Ed. Reading, MA: Addison-Wesley, 1990, vol. Lecture Vol. II.

[3] G. D. Doolen, U. Frisch, B. Hasslacher, S. Orszag, and S. Wolfram, Eds., *Lattice Gas Methods for Partial Differential Equations*. Santa Fe, NM: Santa Fe Inst., 1990.

[4] B. M. Boghosian, J. Yepez, F. J. Alexander, and N. H. Margolus, "Integer lattice gases," *Phys. Rev. E*, vol. 55, pp. 4137–4147, 1997.

[5] N. Margolus, "CAM-8: a Computer architecture based on cellular automata," in *Pattern Formation and Lattice Gas Automata*, ser. Fields Inst. Series. Providence, RI: Amer. Math. Soc., 1995.

[6] P. Russer, "On the field theoretical foundation of the transmission line matrix method," presented at the 1st Int. TLM Modeling Workshop, Victora, B.C., Canada, 1995.

[7] N. R. S. Simons and J. LoVetri, "Derivation of two-dimensional TLM algorithms on arbitrary grids using finite element concepts," presented at the 1st Int. TLM Modeling Workshop, Victoria, B.C., Canada, 1995.

[8] N. R. S. Simons and A.-R. Sebak, "New transmission line matrix node for two-dimensional electromagnetic field problems," *Canadian J. Phys.*, vol. 69, pp. 1388–1398, 1991.

[9] S. Wolfram, *Cellular Automata and Complexity*. Reading, MA: Addison-Wesley, 1994.

[10] U. Frisch, D. d'Humières, B. Hasslacher, P. Lallemand, Y. Pomeau, and J.-P. Rivet, "Lattice gas hydrodynamics in two and three dimensions," *Complex Syst.*, vol. 1, p. 599, 1987.

[11] G. E. Bridges and N. R. S. Simons, "Extremely low precision integer cellular array algorithm for computational electromagnetics," *IEEE Microwave Guided Wave Lett.*, vol. 9, pp. 1–3, Jan. 1999.

[12] C. Alder, B. M. Boghosian, E. G. Flekkoy, N. Margolus, and D. H. Rothman, "Simulating three-dimensional hydrodynamics on a cellular-automata machine," *J. Statist. Phys.*, vol. 81, p. 105, 1995.

[13] D. d'Humières and P. Lallemand, "Numerical simulations of hydrodynamics with lattice gas automata in two dimensions," *Complex Syst.*, vol. 1, pp. 599–632, 1987.

[14] J. R. Treurniet, N. R. S. Simons, G. E. Bridges, and M. Cuhaci, "Evaluation of dissipation within an ILGA for computational electromagnetics," *Int. J. Numer. Modeling*, to be published.

[15] E. C. Jordan and K. G. Balmain, *Electromagnetic Waves and Radiating Systems*. Engelwood Cliffs, NJ: Prentice-Hall, 1968.

[16] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, ser. 2. Cambridge, U.K.: Cambridge Univ. Press, 1992.

[17] N. R. S. Simons, R. Siushansian, J. LoVetri, and M. Cuhaci, "Comparison of the transmission line matrix and finite-difference time-domain methods for a problem containing a sharp metallic edge," IEEE Trans. Microwave Theory Tech., to be published.

[18] P. B. Johns, "A symmetrical condensed node for the TLM method," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-35, pp. 370–377, 1987.

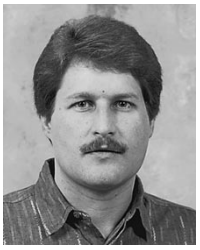
[19] N. R. S. Simons, G. E. Bridges, and M. Cuhaci, "A lattice gas automaton capable of modeling three-dimensional electromagnetic fields," *J. Comput. Phys.*, vol. 151, pp. 816–835, 1999.

Joanne R. Treurniet received the B.Sc. degree in chemistry and physics and the M.Sc. degree in computational physics from Charleton University, Ottawa, Ont., Canada, in 1994 and 1997, respectively.

She is currently a Technical Officer with Ionizing Radiations Standards, National Research Council of Canada, Ottawa, Ont., Canada.

Neil R. S. Simons received the Ph.D. degree from the University of Manitoba, Winnipeg, Man., Canada, in 1994.

From 1987 to 1989, he was with Quantic Laboratories, Winnipeg, Man., Canada, where he was involved with problems related to transmission-line effects on printed circuit boards. Since 1994, he has been a Research Scientist with the Advanced Antenna Technology Group, Communications Research Centre, Ottawa, Ont., Canada. His research interests include computational electromagnetics, application of numerical methods to the analysis and design of microwave antennas and circuits, use of statistical methods for characterization of electromagnetic interference (EMI)/electromagnetic compatibility (EMC) problems, and the characterization of EM environments.



Greg E. Bridges (S'80–M'82) was born in Winnipeg, Man., Canada. He received the Ph.D. degree from the University of Manitoba, Winnipeg, Man., Canada, in 1989.

In 1989, he joined the Department of Electrical and Computer Engineering, University of Manitoba, where he is currently a Professor. In 1996, he was a Visiting Scientist at the Communications Research Centre, Ottawa, Ont., Canada. His research interests involve computational electromagnetics, study of high-frequency transmission lines, and development of new nanoprobing tools for performing internal microelectronics diagnostics and failure analysis.