Ultrahigh performance three-dimensional electromagnetic relativistic kinetic plasma simulation[a)]

K. J. Bowers,[b)] B. J. Albright, L. Yin, B. Bergen, and T. J. T. Kwan
Plasma Theory and Applications (X-1-PTA), Los Alamos National Laboratory, MS F699, P.O. Box 1663, Los Alamos, New Mexico 87545, USA

(Received 8 November 2007; accepted 15 January 2008; published online 5 March 2008)

The algorithms, implementation details, and applications of VPIC, a state-of-the-art first principles 3D electromagnetic relativistic kinetic particle-in-cell code, are discussed. Unlike most codes, VPIC is designed to minimize data motion, as, due to physical limitations (including the speed of light!), moving data between and even within modern microprocessors is more time consuming than performing computations. As a result, VPIC has achieved unprecedented levels of performance. For example, VPIC can perform ~0.17 billion cold particles pushed and charge conserving accumulated per second per processor on IBM's Cell microprocessor—equivalent to sustaining Los Alamos's planned Roadrunner supercomputer at ~0.56 petaflop (quadrillion floating point operations per second). VPIC has enabled previously intractable simulations in numerous areas of plasma physics, including magnetic reconnection and laser plasma interactions; next generation supercomputers like Roadrunner will enable further advances. © *2008 American Institute of Physics*.
[DOI: 10.1063/1.2840133]

I. INTRODUCTION

In many settings in plasma physics, first principles particle-in-cell (PIC) simulation[1,2] is the only viable tool for understanding the physics. By their nature, large scale explicit PIC simulations require large computing resources. Traditional programming methodologies, evolved from experience on large vector single instruction multiple data (SIMD) machines, no longer utilize these resources efficiently; minimizing data motion is now paramount. This work gives an overview of the 3D relativistic electromagnetic PIC code VPIC. VPIC uses state of the art algorithms and optimizes data flows used by these algorithms for modern supercomputers. Several example simulations enabled by VPIC's unprecented performance are shown. The techniques used in VPIC, outlined below, can be applied to other PIC simulations in widespread use in plasma physics and have even been used with success in radically different fields like computational biochemisty (see, for example, Ref. 3).

II. VPIC ALGORITHMS

A. PIC overview

VPIC integrates the relativistic Maxwell–Boltzmann equations in a linear background medium,

$$\partial_t f_s + c\gamma^{-1}\vec{u}\cdot\nabla f_s + \frac{q_s}{m_s c}(\vec{E} + c\gamma^{-1}\vec{u}\times\vec{B})\cdot\nabla_{\vec{u}} f_s = (\partial_t f_s)_{\text{coll}}, \quad (1)$$

$$\partial_t \vec{B} = -\nabla\times\vec{E}, \quad (2)$$

$$\partial_t \vec{E} = \varepsilon^{-1}\nabla\times\mu^{-1}\vec{B} - \varepsilon^{-1}\vec{J} - \varepsilon^{-1}\sigma\vec{E}. \quad (3)$$

Above, $f_s(\vec{r},\vec{u},t)$ is the smooth part of the instantaneous phase-space distribution of a species s with particle charge q_s and mass m_s. c is the speed of light in vacuum, $\vec{u}$ is the normalized momentum, and $\gamma(\vec{u})=\sqrt{1+u^2}$ is the relativistic factor. $\vec{E}(\vec{r},t)$ and $\vec{B}(\vec{r},t)$ are the electric and magnetic field and $\vec{J}(\vec{r},t)=\Sigma_s\int d\vec{u} q_s c\gamma^{-1}\vec{u} f_s$ is the current density. $\varepsilon(\vec{r})$, $\mu(\vec{r})$, and $\sigma(\vec{r})$ are the background medium diagonal permittivity, permeability, and conductivity tensors. $(\partial_t f_s)_{\text{coll}}$ represents discrete particle effects (e.g., short range Coulomb collisions, excitation, and ionization).

Since a direct discretization of f_s is prohibitive, PIC simulations sample it with a collection of computational particles—each computational particle typically represents many physical particles. Equation (1) is replaced by the computational particle equations of motion

$$d_t r_{s,n} = c\gamma_{s,n}^{-1}\vec{u}_{s,n}, \quad (4)$$

$$d_t u_{s,n} = \frac{q_s}{m_s c}[\vec{E}(\vec{r}_{s,n},t) + c\gamma_{s,n}^{-1}\vec{u}_{s,n}\times\vec{B}(\vec{r}_{s,n},t)]. \quad (5)$$

f_s for the computational particles obeys Eq. (1) outside of discrete particle collisional effects. A smooth $\vec{J}$ is computed from the particle motion for use in Eqs. (2) and (3). Because $\vec{J}$ is smooth, $\vec{E}$, $\vec{B}$, and $\vec{J}$ can be sampled on a mesh and interpolated between the mesh and particles.

PIC simulations of these equations can be distinguished by the time stepping, field discretization, interpolation, and collision methods. VPIC uses state-of-the-art techniques, many of which are in use in other codes.[4–9] The methods are summarized below for completeness; more in-depth analysis of PIC simulation can be found in Refs. 1 and 2. For the sake of brevity, collisional methods will not be discussed in this paper.

[a)]Paper YI2 4, Bull. Am. Phys. Soc. **52**, 352 (2007).
[b)]Invited speaker. Electronic mail: kevin.j.bowers@gmail.com. Guest scientist. Presently at: D. E. Shaw Research LLC, 120 W. 45th St., 39th Fl, New York, New York 10036.

B. Time and space discretization

The formal solution of $d_t X = \mathcal{L}X$ for an operator $\mathcal{L}$ is $X(\delta_t) = e^{\delta_t \mathcal{L}} X(0)$. A second order operator splitting[10] for Eqs. (2)–(5) is

$$e^{\delta_t \mathcal{L}_{ub}/2} e^{\delta_t \mathcal{L}_{ue}/2} e^{\delta_t \mathcal{L}_r/2} e^{\delta_t \mathcal{L}_B/2} e^{\delta_t \mathcal{L}_E} e^{\delta_t \mathcal{L}_B/2} e^{\delta_t \mathcal{L}_r/2} e^{\delta_t \mathcal{L}_{ue}/2} e^{\delta_t \mathcal{L}_{ub}/2},$$

where $\mathcal{L}_B$ is associated with Eq. (2), $\mathcal{L}_E$ is associated with Eq. (3), $\mathcal{L}_r$ is associated with Eq. (4), $\mathcal{L}_{ue}$ is associated with the electric force contribution to Eq. (5), and $\mathcal{L}_{ub}$ is associated with the magnetic force contribution to Eq. (5). Grouping the updates for $\vec{E}$ and $\vec{B}$ separately from the others in the repeated application of this splitting and noting the effect of $e^{\delta_t \mathcal{L}_r/2}$ is independent of $\vec{E}$ and $\vec{B}$ yields the well known velocity Verlet field advance, $e^{\delta_t \mathcal{L}_B/2} e^{\delta_t \mathcal{L}_E} e^{\delta_t \mathcal{L}_B/2}$, and leapfrog particle advance, $e^{\delta_t \mathcal{L}_r/2} I_J e^{\delta_t \mathcal{L}_r/2} e^{\delta_t \mathcal{L}_{ue}/2} e^{\delta_t \mathcal{L}_{ub}} e^{\delta_t \mathcal{L}_{ue}/2}$ (I_J indicates that $\vec{J}$ is computed for the field advance but the state is unchanged) that advance $\vec{u}_{s,n}$ from $t - \delta_t/2$ to $t + \delta_t/2$ and $\vec{r}_{s,n}$, $\vec{E}$ and $\vec{B}$ from t to $t + \delta_t$.

The simulation domain is divided into a regular mesh of identical rectangular cells with potentially irregular but cell aligned boundaries. $\vec{E}$ and $\vec{B}$ are staggered; E_x is sampled at the middle of x-directed cell edges, B_x is sampled at the middle of yz-oriented cell faces and similarly for the y and z components. $\vec{J}$ is sampled the same as $\vec{E}$. A variety of particle and field boundary conditions are supported on the domain boundaries including particle absorbing, particle reflecting, particle refluxing, perfect electric conductor, perfect magnetic conductor, field emitting, and field absorbing (first order Higdon[11]).

C. Particle advance

In exact arithmetic, exactly applying $e^{\delta_t \mathcal{L}_r}$ is trivial, $\vec{r}_{s,n} \leftarrow \vec{r}_{s,n} + \delta_t c \gamma_{s,n}^{-1} \vec{u}_{s,n}$. Applying $e^{\delta_t \mathcal{L}_{ue}}$ is similarly trivial, $\vec{u}_{s,n} \leftarrow \vec{u}_{s,n} + (q_s \delta_t / m_s c) \vec{E}(\vec{r}_{s,n}, t)$. Applying $e^{\delta_t \mathcal{L}_{ub}}$ is more involved, $\vec{u}_{s,n} \leftarrow$ rotate $\vec{u}_{s,n}$ around $\vec{B}(\vec{r}_{s,n}, t)$ by $q_s \delta_t |\vec{B}(\vec{r}_{s,n}, t)| / (m_s \gamma_{s,n})$ radians; this can be done efficiently with a modified Boris push.[12] To avoid aliasing of cyclotron motion above the simulation Nyquist frequency if this done exactly, VPIC uses a sixth order rotation angle approximation that is also more efficient to compute. The resulting update still conserves energy like the exact update but will not unphysically couple high frequency cyclotron motion to low frequency phenomena. A similar method is used in Ref. 8.

VPIC uses an energy conserving interpolation scheme for the particle fields. For example, E_x is bilinearly interpolated from the four E_x edge samples and B_x is linearly interpolated from two B_x face samples of the cell containing a particle. While this scheme does not yield as smooth a field interpolation as a momentum conserving trilinear interpolation, this interpolation is consistent with a finite element time domain treatment of Eqs. (2)–(5) for this discretization[6] and is easier to implement in simulations with irregular boundary conditions as no resampling of field components is required.

VPIC uses the charge conserving method of Villasenor and Buneman[13] to compute $\vec{J}$ for the field advance. A particle makes a $\vec{J}$ contribution in each cell it passes through during a time step. In exact arithmetic, the resulting $\vec{J}$ satisfies the discretized charge conservation equation implicit in the field advance described below for a trilinear ρ accumulation. Though this is often considered prohibitively expensive, this eliminates issues that can arise from the build up of Gauss' law violations. Further, determining all the cells a particle passed through provides efficient and robust particle boundary interaction detection.

D. Field advance

Exactly applying $e^{\delta_t \mathcal{L}_B}$ is trivial, $\vec{B} \leftarrow \vec{B} - \delta_t \nabla \times \vec{E}$. Exactly applying $e^{\delta_t \mathcal{L}_E}$ is more involved but can be done via exponential differencing, $\vec{E} \leftarrow e^{-\delta_t \varepsilon^{-1} \sigma} \vec{E} + \sigma^{-1}(1 - e^{-\delta_t \varepsilon^{-1} \sigma})(\nabla \times \mu^{-1} \vec{B} - \vec{J})$ (note that the second term coefficient asymptotes to $\delta_t \varepsilon^{-1}$ in the limit $\sigma \to 0$). The needed curls are computed with second order accurate finite differencing.

A charge conserving $\vec{J}$ and the above imply a discretized Gauss' law. However, in finite precision, arithmetic error can cause small violations of this law to accumulate. To accommodate, VPIC periodically applies Marder passes[14] tuned specifically to clean arithmetic error induced violations. While this method is local and inexpensive, it suffices to use it infrequently to keep this law satisfied to near machine precision.

For wavelengths comparable to the cell dimensions, the discretized speed of light can deviate significantly from c and relativistic particle motion can generate nonphysical Cherenkov radiation at these wavelengths. To combat this, VPIC uses the transverse current adjustment method discussed in Ref. 6. Effectively, a computationally inexpensive current that obeys $\vec{J}_T = \tau \partial_t (\vec{J}_T - \nabla \times \mu^{-1} \vec{B})$ is included that damps short wavelength radiation on a time scale τ while leaving the discretized charge conservation properties unchanged.

E. Stability

In vacuum, the field advance reduces to a basic FDTD algorithm[15] and the time step and mesh spacing must satisfy the Courant condition, $(c\delta_t/\delta_x)^2 + (c\delta_t/\delta_y)^2 + (c\delta_t/\delta_z)^2 < 1$. Additionally, the particle advance usually requires the time step and cell dimensions to satisfy $\omega_p \delta_t < 2$ and $\delta_{x,y,z} \approx \lambda_d$, where ω_p is the peak plasma frequency and λ_d is the plasma Debye length.[1,2] Given particles cannot exceed c, satisfying the Courant condition and the Debye criterion typically implicitly satisfy the plasma frequency criterion. Though the time step is stable for any cyclotron frequency, it is usually desirable to resolve it to keep dynamics accurate. Also, sampling f_s typically requires between tens and thousands of particles per cell depending on the above parameters and phenomena being studied to avoid nonphysical computational particle collisional effects.

III. VPIC IMPLEMENTATION

A. Performance guidelines and optimization regime

Most modern supercomputers consist of a large number of parallel processing nodes connected by network. The nodes themselves contain one or more processors and

TABLE I. VPIC implementation rules of thumb. Data access estimates the time to initiate a data transfer between the processor and a level in its memory hierarchy. Data movement estimates the time to move the next 32-bits in a transfer. Internode figures were obtained from benchmarks of typical high performance cluster interconnects. The single precision figure corresponds to a 2.5 GHz processor completing a 4-vector SIMD instruction every clock. Other rules of thumb were similarly extrapolated from various data sheets and benchmarks. Similar rules of thumb were applied in Ref. 3.

Operation		Time	Rel. cost
Data access	Internode	10 μs	100 000
(Latency)	Memory	50 ns	500
	L2 Cache	5.0 ns	50
	L1 Cache	1.0 ns	10
Data movement	Internode	5.0 ns	50
(32-bit)	Memory	0.5 ns	5
	L2 Cache	0.2 ns	2
	L1 Cache	0.1 ns	1
Single precision	FLOP	0.1 ns	1

memory. Processor performance has improved more rapidly than memory performance over recent decades. To compensate, modern processors use a deep memory hierarchy with faster (though smaller and more limited) "cache" memories located close to the processor. Approximate rules of thumb for the cost of various operations on processing nodes are given in Table I. Achieving high performance requires minimizing latency, bandwidth and computation, in roughly that order. As physical constraints (such as c) are already a noticeable component in many costs, the ratio between processor and memory performance (particularly latency) should not be expected to improve significantly in the future.

Fortunately, in most parallel relativistic codes, data needed for computations on a node are spatially localized to that node or very nearby; internode latency and bandwidth are naturally optimized. For example, in particle dominated simulations, VPIC has smoothly scaled to some of the largest machines available (several thousand nodes) without any special tuning effort when the simulation domain could be statically decomposed to yield an approximately uniform number of particles per node (even some highly nonuniform plasmas can be so decomposed, especially if using variable particle charges).

Accordingly, in VPIC, local data motion is the dominant concern. Unfortunately, on most processors, programmers have no control over cache. Rather, a processor heuristically caches data it anticipates will be used. Given this and Table I, minimizing local data motion requires grouping data needed to perform critical operations contiguously and accessing it sequentially when possible. As computation and storage are virtually free compared to data motion, replicating computations and/or data is often worthwhile.

In 3D PIC simulations, there tend to be many particles per cell on average and more field and particle data per node than can fit in any cache on that node. In such a simulation, computation time is heavily dominated by the particle advance. VPIC is designed for this regime. For the sake of brevity, only particle advance implementation considerations are given and, unless otherwise stated, simulations will be assumed to be in this regime.

B. Single precision

Single precision implementations can be made to run significantly faster than their double precision counterparts as most modern processors provide 4-vector single precision SIMD capabilities and single precision requires half the data movement as double precision. For typical time steps, cell dimensions and particles per cell, discretization error exceeds single precision arithmetic error. Thus, single precision should be acceptable for such simulations provided it does not introduce nonphysical artifacts (e.g., large violations of otherwise conserved quantities during the simulation). VPIC uses single precision to help achieve high performance but takes care to structure operations to minimize its impact; some cases are discussed below.

Unfortunately, languages like C and FORTRAN are not expressive enough (e.g., data alignment restrictions) to allow compilers to use 4-vector SIMD in operations as complex as those in VPIC. To compensate, VPIC has a C language extension that allows portable 4-vector SIMD code to be written and converted automatically to high performance 4-vector SIMD instructions on a wide variety of platforms. A similar approach was used in Ref. 3.

A full discussion of single precision issues is beyond the scope of this paper. For examples of single precision use in applications with far more demanding precision requirements, see Refs. 3 and 16.

C. Single pass processing and particle data layout

Since there is more particle data than can fit in any cache, it is desirable to limit the number of times a particle is touched during a time step lest performance be limited by moving particle data to and from memory. This can be achieved by processing the majority of the particles in a single pass. This, combined with the particle advance, yields VPIC's inner loop structure,

```
for each particle
  interpolate fields
  update momentum
  update position
  accumulate current
end for
```

To further minimize the cost of moving particle data, particle data is stored contiguously, memory aligned, and organized for 4-vector SIMD. A particle looks like

```
struct {float dx, dy, dz; int i;
        float ux, uy, uz; float q; }
```

As a result, the inner loop streams through particle data once using large aligned memory transfers under the hood—the ideal memory access pattern.

D. Field interpolation and particle sorting

Field interpolation can severely impact performance if the data choreography is not considered. For example, if particles are stored in a random order relative to the cells, mesh fields will be accessed in a random order by the inner loop. Because there is more mesh field data than can be cached, these accesses will often require memory transfers. The situation is made worse if mesh fields are directly accessed as several memory transfers may be necessary to load noncontiguous field data. Worse still if the field components are stored in separate arrays. In this worst case, VPIC's field interpolation could involve up to 13 memory transfers per particle under the hood (assuming adjacent x data is contiguous in memory, 4 for E_x, 2 for E_y, E_z, B_y and B_z, and 1 for B_x). Further exacerbating this, far more data than requested would be transferred; modern cores roundup the amount of memory transferred in small requests like this to, at least, the (larger) memory-core path width.

To make field interpolation efficient, an array of interpolation coefficients is precomputed and saved in a contiguous, aligned, 4-vector SIMD compatible layout,

```
struct { float ex, dex_dy, dex_dz, d2ex_dydz;
         float ey, dey_dz, dey_dx, d2ey_dzdx;
         float ez, dez_dx, dez_dy, d2ez_dxdy;
         float bx, dbx_dx, by,     dby_dy;
         float bz, dbz_dz, pad0,       pad1; }
```

Additionally, the particle array is periodically sorted by the containing cell index. Because the particles do not move far per time step, sorting is infrequent—every tens to hundreds of time steps. Nevertheless, the sorting can be done efficiently in $O(N)$ operations.[17]

As a result, all the particles in a given cell are processed approximately sequentially and the interpolation coefficients necessary for these particles can be loaded once from memory and then cached. The interpolation coefficients themselves are accessed approximately sequentially in large aligned transfers a near minimal number of times. Even though storing the interpolation coefficients requires over three times as much memory as the mesh field data, the net impact is to reduce memory transfers to minimal levels by making more efficient use of cache.

Note that this approach generalizes to other interpolation strategies (e.g., trilinear momentum conserving interpolation). Likewise, already having the particles sorted allows various particle-particle collision models to be implemented more efficiently.

This field interpolation approach is less effective when there are very few particles per cell on average or when a large fraction of the cells are devoid of particles. In these cases, the field interpolation coefficient computation time is not compensated by a reduction in the particle field interpolation time. However, given that the cost of forming unused coefficients is small and that cells containing multiple particles still see some benefit, this effect is mild. Further, this regime is atypical; sampling of f_s typically requires many particles per cell on average.

High temperature particle distributions generally require more frequent sorting to maximize overall performance than cold particle distributions. Fortunately, the finite speed of light in relativistic simulation (or, more generally, time step limitations due to numerical aliasing instabilities induced by particles moving through too many cells in a step) limit the rate at which the particle array becomes disorganized at even the highest temperatures. Interestingly, nonthermal particle distributions (for example, fast moving cold drifting particle beams) tend to need less performance sorting; as these stay more organized physically, they stay more organized in memory as well. Also, the frequent sorting that occurs when using various particle-particle collision models usually eliminates the need for additional performance sorting.

E. Position representation

Positions are given by the containing cell index and the offset from the cell center, normalized to the cell dimensions. Having the index makes determining which interpolation coefficients to load trivial. The interpolation computation and current accumulation are likewise very efficient as the particle offsets can be used directly.

This representation is also critical in single precision. When an absolute position representation is used, some coordinate information encodes the containing cell index. Thus, there is less information available to encode the offset. Consider a 1D simulation with $2^{10}=1024$ cells over the region (0,1). In single precision absolute coordinates, particles in cell 0 see a worst case absolute position resolution of $2^{-24}/2^{10}\sim 5.8\times 10^{-11}$ while particles in cells 512–1023 see a resolution of $2^{-24}\sim 6.0\times 10^{-8}$. In index plus offset representation, all particles see a resolution of $2^{-25}/2^{10}\sim 2.9\times 10^{-11}$ (the sign bit provides one bit of position resolution) regardless of which cell contains them. This is better everywhere than the absolute representation and does not exhibit numerical anisotropy. Interestingly, VPIC has identical numerical properties for each cell regardless how the cell mesh is translated, oriented or reflected in absolute coordinates.

This representation requires that the coordinate system for the particle offsets need to be transformed when a particle crosses into a different cell. This cost is trivial and can be elegantly incorporated into the charge conserving current accumulation. Additionally, this representation generalizes naturally to more advanced PIC algorithms (e.g., boundary fitted curvilinear meshes).

F. SIMD, current accumulation, and exceptions

Given the single precision usage and SIMD compatible data layouts, all particle processing described above is ideal for 4-vector SIMD. Unfortunately, current accumulation is not. Determining the cells through which a particle passed varies from particle to particle; one particle might remain in the cell in which it started while the next might cross through several. To utilize 4-vector SIMD, VPIC exploits that particles do not cross cell boundaries often. Consider a fairly extreme simulation with a uniform isotropic light species (e.g., electrons) with $\omega_p\delta_t\sim 0.2$ and $\delta_x\sim\lambda_d$ (by the Courant condition, the thermal velocity would be $\sim 0.1c$) and a heavy species (e.g., hydrogen) at the same temperature. Only $\sim 41\%$ of

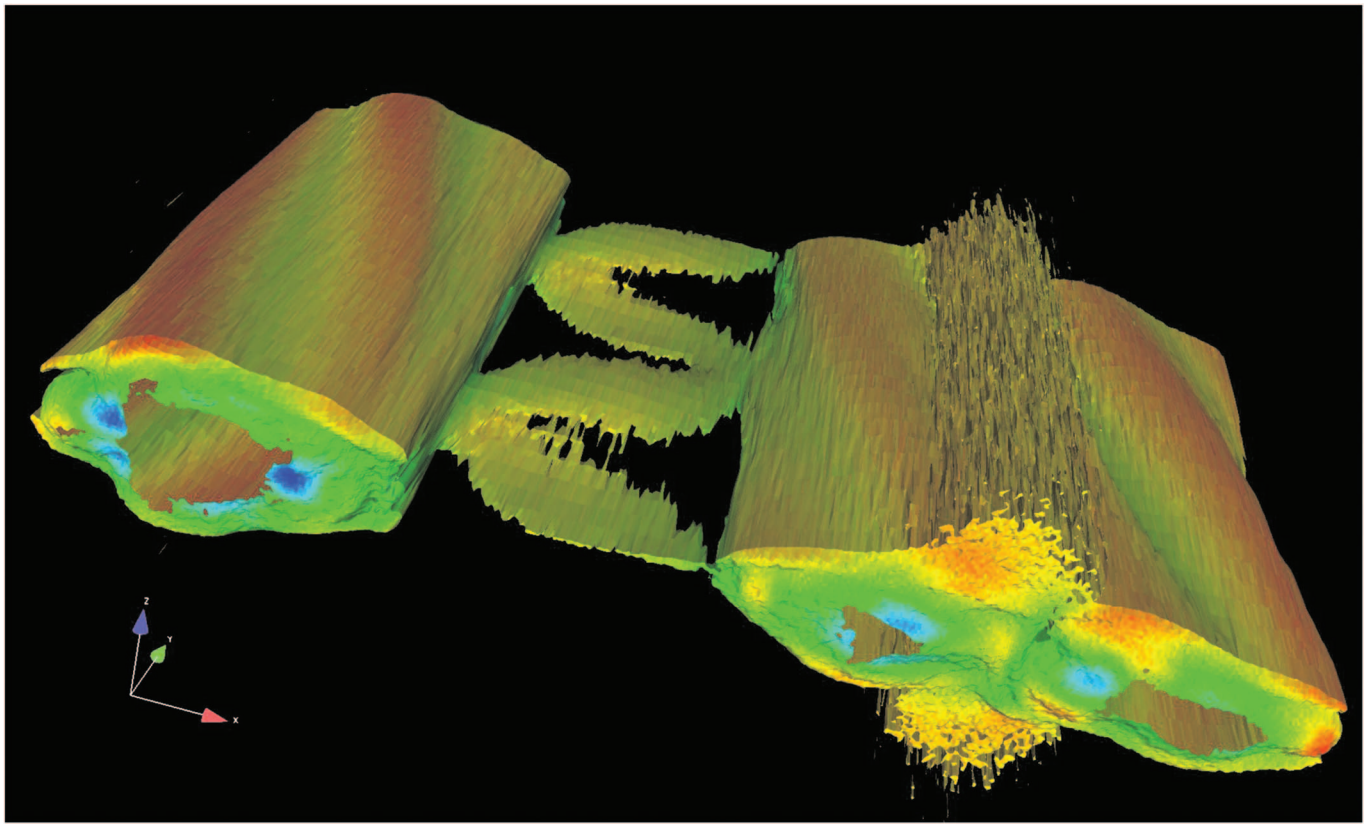

FIG. 1. (Color) 3D VPIC simulation of magnetic reconnection in a pair plasma with an intermediate guide field. Displayed are isosurfaces of electron density colored by the magnetic field B_y (y is the initial current and guide field direction). The isosurfaces undulate from the interaction of the tearing modes (wave vector along x) with the drift-kink modes (along y) (Ref. 20).

light species particles and ~1.1% of heavy species particles will cross a cell boundary in a time step. VPIC advances 4 particles at a time with 4-vector SIMD by assuming none of the 4 particles cross cell boundaries. Particles that do cross are detected and make zero current contribution during this process. These particles are processed in scalar code subsequently.

Other particle distributions may have different performance characteristics, but, as noted above, simulation stability criteria (especially in 3D) generally limit the number of cell boundaries a particle will cross in a time step to a few with most not crossing at all. Further, in many simulations with a drifting particle species, the simulation frame can be chosen to reduce such potential performance impacts—this can even improve numerical quality and yield significantly less stringent simulation stability criteria.[18]

Like the interpolation coefficients, current contributions from particle motion in a cell are made to a contiguous aligned set of partial currents which is then postprocessed into $\vec{J}$ prior to the field advance. The same benefits as described for interpolation coefficients apply here.

During cell crossing current accumulation, if a particle hits an "exceptional" boundary (e.g., needs communication to neighbor node, needs absorbed, needs refluxed), the index and remaining particle displacement are saved to an exception list for later processing. This has several benefits: No additional passes through the particles are necessary to find exceptions, exception handling (often slow and application specific) is cleanly separated from the high performance general particle advance and exception handling does not pollute the instruction or data caches while the particle advance is running. Exception processing itself does not use any dramatic techniques. In 3D, exception processing time typically scales as the simulation domain surface area (and any boundaries within that domain); this is negligible compared to the particle advance time which scales as the simulation domain volume.

G. IBM Cell considerations for Roadrunner

VPIC was recently ported to IBM's Cell processor. Each cell processor contains one general purpose core and eight high performance 4-vector SIMD processor cores ("SPE"). Each SPE can only directly access 256 KB of high performance memory specific to it. This "local store" is analogous to cache, but, unlike conventional cache, memory transfers to and from it must be explicitly managed. Though nontrivial, this makes it possible to design high performance application specific cache protocols.[19]

On the SPEs, particle data is triple buffered in blocks of $N=512$; while a block is processed, the next is loaded and the previous is stored. An N line software cache of read-only interpolation coefficients and read-write partially accumulated currents handles inner loop random memory access. It has a fully associative least recently used policy (i.e., a line can hold any cell's data and the least recently used line is

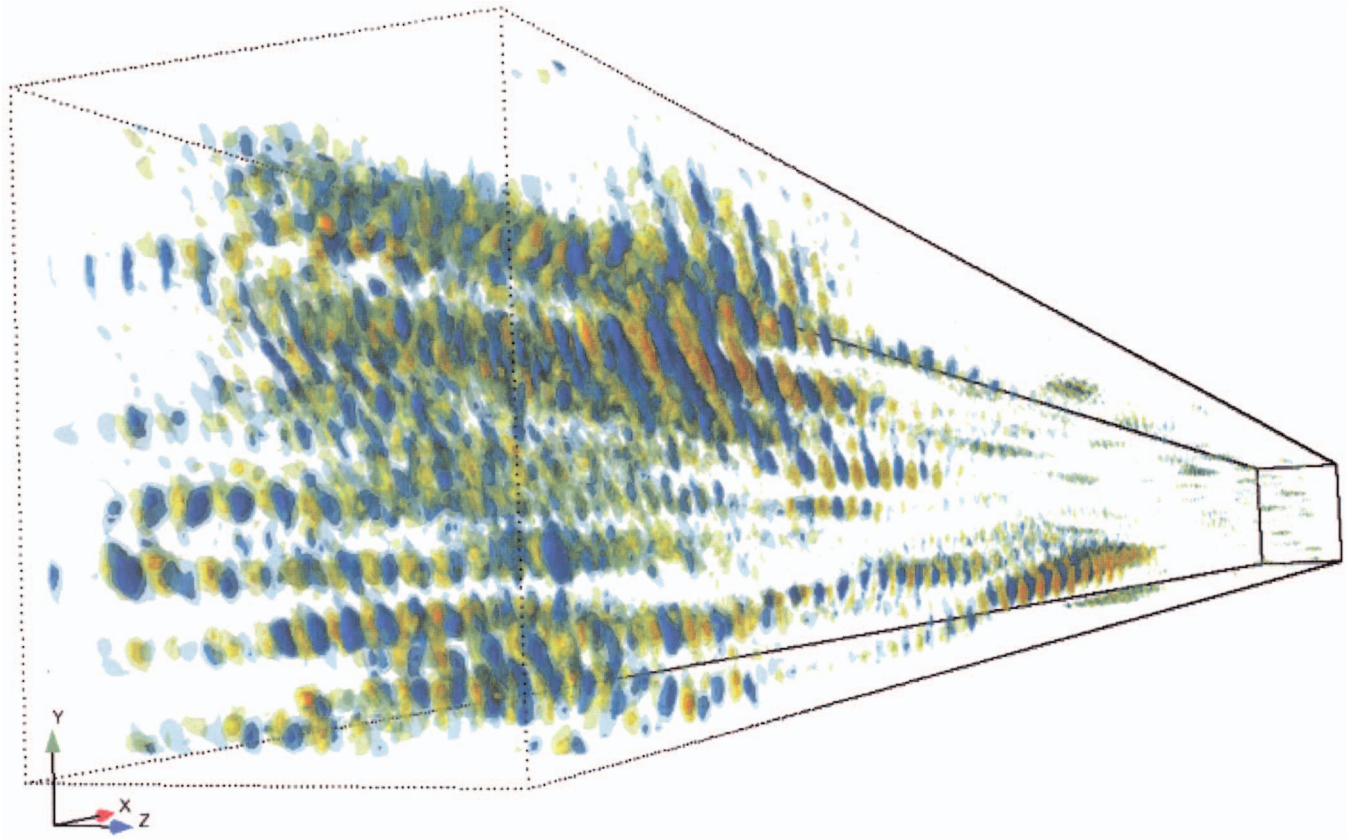

FIG. 2. (Color) 3D VPIC simulation of stimulated Raman scattering (SRS) in the kinetic regime. Isosurfaces of longitudinal electrostatic field show filament structures resulting from trapped electron self-focusing of Langmuir waves during SRS saturation. The laser is launched from the simulation geometry near face (Ref. 22).

replaced when caching new data); the last N distinct requests are guaranteed to be cached. Before processing a block, requests are made for all cells referenced in the block. Most of these will already be in the cache. Nonredundant memory transfers are initiated for the remainder and block processing begins. The net impact is minimal memory transfers are used, memory transfers are overlapped with computation and the inner loop operates exclusively out of cache (cell crossing current accumulation makes additional requests following the inner loop).

While such a cache may seem expensive, each SPE can be faster (even clock-for-clock) than a conventional processor core at cold particle advances. Presently, VPIC performs at ~173 million cold particles advanced per second on a 3.2 GHz cell (~49% of measured peak memory bandwidth and ~21% of theoretical aggregate SPE single precision floating point performance). By comparison, VPIC runs at ~12.6 million PA/s on a single core 2.2 GHz AMD Opteron (used in the systems that ran the below examples). As Los Alamos's Roadrunner supercomputer plans to use ~13 000 cell eDP processors, a sustained rate up to ~2.25 trillion PA/s could be achieved in particle dominated simulations. As a cold particle advance takes ~246 FLOP per particle (warm particle advances take more, depending on simulation details such as temperature, for cell crossing current accumulation), this is equivalent to ~0.556 quadrillion FLOP per second.

IV. VPIC APPLICATIONS

VPIC has been applied with success in both basic and applied plasma science. Here, we give three examples of computationally demanding simulations enabled by the high efficiency of VPIC. The first is a large-scale simulation of 3D magnetic reconnection in an electron-positron plasma, as shown in Fig. 1. The initial configuration is a Harris equilibrium with an intermediate guide field. The simulation follows the dynamics of 16 billion particles on a $1000\times100\times1000$ mesh (on a physical domain of $200d_i\times20d_i\times200d_i$ where d_i is the ion inertial length). Shown are electron density isosurfaces with color indicating the magnetic field B_y (y is the direction of initial current) at a time when a dominant diffusion region has formed and influenced by the drift-kink mode. The calculation ran for 36 h of wall-clock time on 500 AMD Opteron cores. More details on the 3D reconnection dynamics and the interplay between tearing and kink instability can be found in Ref. 20 (another example of a VPIC magnetic reconnection simulation is in Ref. 21).

The second example in Fig. 2 is a 3D simulation of stimulated Raman scattering (SRS) in the kinetic regime ($k\lambda_d=0.34$, where k is the initial Langmuir wavenumber and λ_d is the plasma Debye length). The simulation used 18 billion particles on a mesh of size $2436\times486\times486$ (on a physical domain of $90\times18\times18$ microns) run on 1008 AMD

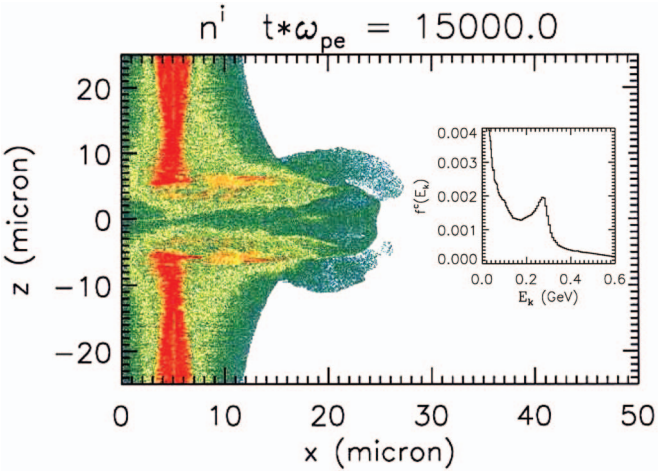

FIG. 3. (Color) Large-scale 2D VPIC simulation of ultraintense laser interaction with a thin target in the "break-out afterburner" regime. Shown are contours of density for the carbon target as the laser, launched from the simulation left edge, has propagated to the simulation right edge; the inset displays the carbon energy spectrum, obtained along a 1 μm slice centered at $z=0$, with a quasimonoenergetic feature at ~275 MeV (Refs. 23 and 24).

Opteron cores. The electron plasma waves self-localize to regions of narrow transverse extent, as shown by the filament structures in the longitudinal electric field. SRS saturates via the trapped electron self-focusing of Langmuir waves.[22]

The final example shown in Fig. 3 is a grid-dominated simulation of short-pulse laser interaction with a solid density carbon target ($n_e/n_{cr}=660$, where n_{cr} is the critical density). This large-scale 2D simulation used 1.23 billion cells (on a physical domain of 50×50 microns) and ran on 1020 AMD Opteron cores; within the ultrathin 30-nm-thick target, each species is represented by 1.7×10^5 particles/cell. Shown is the ion density after the 1 μm laser (a Gaussian beam of FWHM 156 fs at an intensity of 10^{21} W/cm^2) launched from the simulation left edge, has propagated to simulation right edge at time 312 fs. The inset displays the carbon energy spectrum with a quasimonoenergetic feature at ~275 MeV. In the simulation, quasimonoenergetic and GeV carbon ions are generated via the "break-out afterburner" acceleration mechanism.[23,24]

V. CONCLUSION

If the LANL Roadrunner supercomputer is acquired as planned, VPIC calculations will be possible in late 2008 at scales well beyond what is practicable today: Simulations with 10^{12} particles, 10^9 cells, and 10^6 time steps may be prosecuted in a matter of days. This opens up the possibility for kinetic modeling of, e.g., 3D magnetic reconnection at high ion-to-electron mass ratio for a hydrogen plasma, 3D dynamics of stimulated laser scattering in multispeckles, 3D high resolution ultraintense laser-matter interaction, and *ab initio* kinetic thermonuclear burn, to name a few. The techniques described herein that enable VPIC's ultrahigh explicit PIC performance can be applied fruitfully to other plasma simulation codes and platforms. In this sense, VPIC is the first of a new generation of high performance plasma simulation codes.

ACKNOWLEDGMENTS

This work was performed under the auspices of the United States Department of Energy by the Los Alamos National Security LLC Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396.

The authors acknowledge the assistance of Dr. Hui Li and Dr. Bill Daughton. 3D visualization was performed using the EnSight Gold software by CEI Inc. The authors thank Dr. Jeremy Margulies and Dr. Eric Nelson for their assistance. Simulations shown were run on the LANL Flash and Lightning supercomputers.

[1] C. K. Birdsall and A. B. Langdon, *Plasma Physics via Computer Simulation* (McGraw-Hill, New York, 1985).
[2] R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles* (Institute of Physics, Philadelphia, 1988).
[3] K. J. Bowers, E. Chow, H. Xu, R. O. Dror, M. P. Eastwood, B. A. Gregersen, J. L. Klepeis, I. Kolossvary, M. A. Moraes, F. D. Sacerdoti, J. K. Salmon, Y. Shan, and D. E. Shaw, in *SC '06: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing* (ACM, New York, 2006).
[4] T. J. T. Kwan and C. M. Snell, *Lecture Notes in Physics* (Springer-Verlag, Berlin, 1985), Vol. 240.
[5] J. P. Verboncoeur, A. B. Langdon, and N. T. Gladd, Comput. Phys. Commun. **87**, 199 (1995).
[6] J. W. Eastwood, W. Arter, N. J. Brealey, and R. W. Hockney, Comput. Phys. Commun. **87**, 155 (1995).
[7] M. E. Jones, D. Winske, S. R. Goldman, R. A. Kopp, V. G. Rogatchev, S. A. Bel'kov, P. D. Gasparyan, G. V. Dolgoleva, N. V. Zhidkov, N. V. Ivanov, Y. K. Kochubej, G. F. Nasyrov, V. A. Pavlovskii, V. V. Smirnov, and Y. A. Romanov, Phys. Plasmas **3**, 1096 (1996).
[8] J. D. Blahovec, L. A. Bowers, J. W. Luginsland, G. E. Sasser, and J. J. Watrous, IEEE Trans. Plasma Sci. **28**, 821 (2000).
[9] C. Nieter and J. R. Cary, J. Comput. Phys. **196**, 448 (2004).
[10] R. I. McLachlan and G. R. W. Quispel, in *Acta Numerica* (Cambridge University Press, Cambridge, 2002), Vol. 11, pp. 341–434.
[11] R. L. Higdon, Math. Comput. **47**, 437 (1986).
[12] J. P. Boris, in *Proceedings of the 4th Conference on Numerical Simulation of Plasmas*, edited by J. P. Boris and R. A. Shanny (Naval Research Laboratory, Washington, D.C., 1970), pp. 3–67.
[13] J. Villasenor and O. Buneman, Comput. Phys. Commun. **69**, 306 (1992).
[14] B. Marder, J. Comput. Phys. **68**, 48 (1987).
[15] K. S. Yee, IEEE Trans. Antennas Propag. **14**, 302 (1966).
[16] R. A. Lippert, K. J. Bowers, B. A. Gregersen, R. O. Dror, M. P. Eastwood, J. L. Klepeis, I. Kolossvary, and D. E. Shaw, J. Chem. Phys. **126**, 046101 (2007).
[17] K. J. Bowers, J. Comput. Phys. **173**, 393 (2001).
[18] J. L. Vay, Phys. Rev. Lett. **98**, 130405 (2007).
[19] J. A. Kahle, M. N. Day, H. P. Hofstee, C. R. Johns, T. R. Maeurer, and D. Shippy, IBM J. Res. Dev. **49**, 589 (2005).
[20] L. Yin, W. Daughton, H. Karimabadi, B. J. Albright, K. J. Bowers, and J. Margulies, "Fully kinetic 3D simulations of collisionless reconnection in large-scale pair plasmas," Phys. Rev. Lett. (to be published).
[21] K. J. Bowers and H. Li, Phys. Rev. Lett. **98**, 035002 (2007).
[22] L. Yin, B. J. Albright, K. J. Bowers, W. Daughton, and H. A. Rose, Phys. Rev. Lett. **99**, 265004 (2007).
[23] L. Yin, B. J. Albright, B. M. Hegelich, K. J. Bowers, K. A. Flippo, T. J. T. Kwan, and J. C. Fernández, Phys. Plasmas **14**, 056706 (2007).
[24] B. J. Albright, L. Yin, K. J. Bowers, B. M. Hegelich, K. A. Flippo, T. J. T. Kwan, and J. C. Fernández, Phys. Plasmas **14**, 094502 (2007).