Cell Accelerated Cryoablation Simulation

Daniel J. Blezek[1], David G. Carlson[2], Lionel T. Cheng[1], Matthew R. Callstrom[1], and Bradley J. Erickson[1]

[1] blezek.daniel@mayo.edu, Mayo Clinic, Rochester MN, 55905, USA
[2] IBM, Rochester MN 55901, USA

Abstract. Tracking growth of lethal ice is critical to success of percutaneous ablations using supercooled probes introduced into cancerous lesions. Physicians lack planning tools which provide accurate estimation of the ice formation. Simulation is computationally demanding, but must rapid for clinical utility. We develop the computational framework for the simulation, acceleration strategies for multi-core Intel x86 and IBM Cell architectures, and preliminary validation of the simulation. Our data shows streaming SIMD implementation has better performance and scalability, with good agreement between simulation and manually identified ice ball boundaries.

1 Introduction

Minimally-invasive methods of lesion ablation, utilizing image guidance, have become increasing popular in recent years. One key procedure is destruction of neoplastic lesions through freezing, or cryoablation. It is often used for tumors in patients where surgical extirpation is not a viable option [1].

Skeletal metastases are often painful. The current standard of care is radiation therapy. However, relief from pain lags treatment and 20 to 30% of patients do not experience relief at all. Palliative cryoablation has emerged as an immediate source of relief [2]. To the best of our knowledge, no clinically oriented simulation systems exist for musculo-skeletal (MSK) cryoablation procedures. The procedure is illustrated in Fig. 1; bone and cryoprobe isosurfaces are rendered with the 0°C isotherm.

Suitable lesions are accurately localized through radiological imaging techniques. One or more thin probes (cryoprobes) are introduced into the body via a small skin puncture, and positioned within the lesions under real-time imaging guidance [3]. Recent advances in cryoprobe technology have provided unprecedented flexibility to the physician; insulated probes allow percutaneous approaches and elimination of cryogens allow fine control over the heat flux. Once probe positions are confirmed, expansion of argon gas near the probe tip (the Joule-Thompson effect) progressively freezes the adjacent tissue, forming an "ice ball". The temperature for ensuring cell death is "lethal ice" at -20 to -40°C. Ice formation, changes in cellular volume and loss of blood supply induce cell death in lethal ice. Current clinical practice is a 10 minute freeze, 10 minute thaw and 10 minute re-freeze.

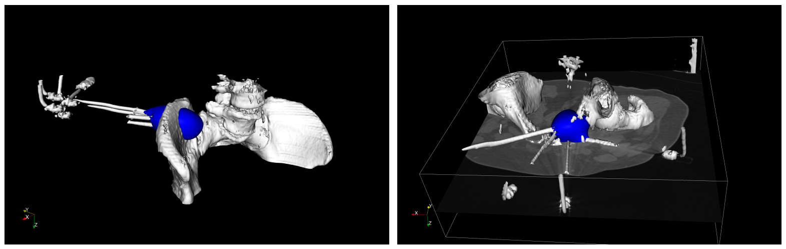

(a) Left illiac bone (b) Right superic pubic ramus

Fig. 1. MSK cryoablation procedures. Cryoprobes may be used in parallel (a) or in a "spoke" pattern (b). Four cryoprobes are used in both examples.

Imaging plays a key role in the planning, intra-procedural monitoring and subsequent follow-up of cryoablation cases. Planning consists of lesion visualization and careful selection of probe trajectory to maximize tumor destruction and minimize injury to normal tissue. Parallel (Fig. 1(a)) and crossing (Fig. 1(b)) probe configurations are common. The interventional radiologist integrates many factors when formulating his plan, i.e., location of the lesion relative to critical structures, heat conducting properties of surrounding structures, amount of blood perfusion that might carry heat away from the region, and lesion shape. During the procedure, ice ball growth is monitored through serial imaging studies, allowing the operator to modify growth rate by titration of gas flow.

Physicians performing MSK cryoablation estimate the number and position of cryoprobes from general manufacturer specifications and prior experience. This suggests the need for tools to assist in planning cryoablation procedures, which in turn is expected to improve clinical outcomes. To be useful, a simulation system must rapidly provide accurate temperature maps based on physical properties of cryoprobes and tissue-specific thermal characteristics. This point can not be overstated: if the system does not provide adequate response, no matter how accurate, it can not be clinical useful. Once a plan is submitted for simulation, physicians expect immediate visualization of the temperature maps. Seconds of run time for each simulated minute are expected. In this work, we have endeavored to develop and validate a highly accelerated solver first on the Intel x86 architecture and then on IBM's Cell Broadband Engine architecture.

2 Methods

2.1 Solution to the Bioheat Equation

Based on the plan configured by the planning interface, the simulation system solves the heat transfer equations. The planning system provides the location, size, and heat flux of each cryoprobe. The simulation determines tissue properties for each voxel in the CT image and boundary conditions for a partial differential equation (PDE) system. The simulation incorporates models of cryoprobe

Table 1. Bioheat equation symbols and units

Symbol	Description	Units
C	specific heat	$MJm^{-3}K^{-1}$
T	temperature	K
k	thermal conductivity	$Wm^{-1}K^{-1}$
$\dot{w}_b$	blood perfusion	unitless
C_b	specific heat of blood	$MJm^{-3}K^{-1}$
T_b	blood temperature	K
$\dot{q}_{met}$	metabolic heat generation	K
β	relaxation parameter	unitless
W	$\dot{w}_b C_b / C$	unitless
$N(T_t)$	neighborhood of T_t	
Δx_i^2	square distance from T_t to T_t^i	mm
	Tissue specific temperature-dependent	
$\tilde{C}(T)$	effective specific heat	$MJm^{-3}K^{-1}$
$\tilde{k}(T)$	effective thermal conductivity	$Wm^{-1}K^{-1}$
$\tilde{q}_{met}(T)$	effective metabolic heat	K
$\tilde{w}_b(T)$	effective blood perfusion	unitless

physical properties and intrinsic tissue properties include thermal conductivity, specific heat, perfusion, etc. Our derivation follows the work of Rabin [4] and Deng [5].

Our simulation model incorporates the important liquid-solid phase transition [5]. Tissue moves from "fresh" to "mush" to "frozen", each state having different thermal properties[6]. Symbols and unites used in this work are defined in Table 1. The bioheat equation [7] is given in (1). The time rate of change in temperature is governed by the divergence of the temperature gradient (Laplacian for scalar fields), blood perfusion and metabolic heat generation. Using finite differences (2) with a relaxation parameter β, (1) can be iteratively solved as shown in (3). The solution domain is divided into isotropic $1mm^3$ elements, resulting in a time step (Δt) of 0.3 sec. or 180 iterations per minute of simulated freezing (see [5] for details). Associated with each tissue class are several temperature dependent parameters($\tilde{C}(T)$,$\tilde{k}(T)$,$\tilde{q}_{met}(T)$, and $\tilde{w}_b(T)$ in Table 1) modeling phase change.

$$C\frac{\partial T}{\partial t} = \nabla \cdot (k\nabla T) + \dot{w}_b C_b (T_b - T) + \dot{q}_{met} \tag{1}$$

$$T_t = \beta T_{t+\Delta t} + (1-\beta)T_t \tag{2}$$

$$\begin{aligned} T_{t+\Delta t} =& \frac{1}{1 - W\beta\Delta t}\Bigg(1 - W(1-\beta)\Delta t T_t + \frac{(\dot{q}_{met} + \dot{w}_b C_b T_a)\Delta t}{C} \\ &+ \sum_{T_t^i \in N(T_t)} \frac{k\Delta t (T_t^i - T_t)}{C\Delta x_i^2}\Bigg) \end{aligned} \tag{3}$$

2.2 Implementation

CT images are acquired approximately every 2 minutes during the procedure. For each subject, the CT image corresponding to the end of the first freeze cycle were identified. All CT volumes were resampled to $1mm^3$ isotropic resolution, forming the T_t volume for $t = 0$. A volume of tissue classes (C_{tissue}) was initialized. Initial conditions were constructed from manually identified cryoprobe voxels fitted to a line and assigned the "tissue class" of probe; each voxel's temperature was fixed at $-170°$C for the duration of the simulation. The remaining tissue was initialized to 37°C and assigned to the generic tissue class.

The basic algorithm for solving the heat transfer equation is shown in Algorithm 1. For each slice, row pointers are initialized, including the neighborhood rows $N(P_{row})$ around P_{row}. $T_{t+\Delta t}$ is calculated by the Update($\cdot$) function. The algorithm iterates until the simulation time T_{sim} is reached.

Algorithm 1 Basic simulation algorithm.

```
Initialize volumes $T_t$, $T_{t+\Delta t}$ and $C_{tissue}$
while $t < T_{sim}$ do
  for $slice = 1$ to $N_{slices}$ do
    for $row = 1$ to $N_{rows}$ do
      $P_{row} = T_t.getSlice(slice).getRow(row)$
      Load pointers $N(P_{row}) = \{P_{above}, P_{below}, P_{row+}, P_{row-}\}$
      for $idx = 1$ to $N_{columns}$ do
        {the Update($\cdot$) function implements (3)}
        $T_{t+\Delta t}[idx] = \text{Update}(constants[P_{tissue}[idx]], N(P_{row}[idx]))$
      end for
    end for
  end for
  $t = t + \Delta t$
end while
```

2.3 Acceleration strategies

Symmetric multi-processors are ubiquitous. OpenMP has emerged as a simplified alternative to explicit management of threads [8,9]. The main strength of OpenMP lies in the ability to convert serial loops to parallel through `#pragma` compiler directives. The resulting code is readily understandable, and may be executed in serial for debugging/validation purposes. OpenMP naturally fits into medical image processing by allowing loops over all the slices in a volume to be executed on all the available SMP cores. Both commercial high-performance compilers, e.g. the Intel compiler, and open source compilers, e.g. GCC implement the OpenMP standard. Our initial acceleration strategy augments Algorithm 1 with OpenMP directives, processing slices in parallel on all available CPUs (Algorithm 2). The code was compiled by the Intel compiler and GCC on an server (IBM HS21 Blade, CentOS 5). Table 2 shows compiler parameters and versions.

Algorithm 2 OpenMP algorithm.

```
while $t < T_{sim}$ do
  for all $Core$ in available CPUs do {Execute in parallel}
    for all $slice \in Core_{slices}[Core]$ do
      for $row = 1$ to $N_{rows}$ do
        $P_{row} = T_t.getSlice(slice).getRow(row)$
        Load pointers $N(P_{row}) = \{P_{above}, P_{below}, P_{row+}, P_{row-}\}$
        for $idx = 1$ to $N_{columns}$ do
          {the Update($\cdot$) function implements (3)}
          $T_{t+\Delta t}[idx] = \text{Update}(constants[P_{tissue}[idx]], N(P_{row}[idx]))$
        end for
      end for
    end for
  end for
  $t = t + \Delta t$
end while
```

Table 2. Compiler details

Name / Version	Parameters
GNU GCC 4.1.2 20070626	-O3 -fopenmp
Intel Compiler for Linux version 10.1 20080112	-O3 -ipo -no-prec-div -xT -openmp
XL C/C++ for Multicore Acceleration V9.0	-O5 (SPU compilation only)

IBM's Cell Broadband Engine [10] is a useful platform for acceleration of medical image registration [11], and shows potential for broad applicability. Though the Cell is akin to a general purpose CPU, i.e., Intel x86, understanding of its strengths and weaknesses are critical to successful acceleration. Current generations of the Cell have 16 synergistic processing units (SPU). Each SPU has 256KiB of local storage and a single-instruction, multiple-data (SIMD) vector processor. Data must be explicitly moved from the main memory into the local store. The Cell has excellent direct memory access (DMA) abilities; up to 16 asynchronous DMA requests may be simultaneously pending per SPU. While the small local storage may seem like a significant limitation, relative to graphical processing units, it is spacious. The SPU local storage is, in essence, an explicitly managed L1 cache.

Development to a streaming SIMD algorithm for Cell was our second acceleration phase (Algorithm 3). Data-starvation and CPU-starvation pitfalls have been avoided by overlapping computation and asynchronous DMA requests. In parallel, each SPU processes an independent regions of interest (ROI). SPUs are dual issue, executing fixed- and floating-point SIMD operations in parallel with load/store/permute instructions. Double buffering is achieved through two input buffers and two output buffers. Each SPU initiates filling the input buffers on lines 5 and 6, possibly waiting for input (line 8) and free output buffers (line 9) before processing. During processing (lines 10-16), input and output buffers are asynchronously DMA'ed. After processing the current output buffer is sent (line 18) and a future chunk requested (line 17). The streaming algorithm is augmented by the SIMDUpdate(·) function, implementing (3) with SIMD vector operations (line 15).

Algorithm 3 Streaming SIMD algorithm for Cell.

```
1:  initialize $barrier$
2:  **while** $t < T_{sim}$ **do**
3:    **for all** $SPU$ in available $SPUs$ **do** {Execute in parallel}
4:      $ROI = SPU_{ROIs}[SPU]$
5:      start($DMA_{retrieve}, 0$)
6:      start($DMA_{retrieve}, 1$)
7:      **for** $chunkId = 0$ to $N_{chunk}$ **do**
8:        $T_t$ = wait($DMA_{retrieve}, chunkId$)
9:        $T_{t+\Delta t}$ = waitIfNotSent($DMA_{send}$) {two buffers are available}
10:       **for** $idx = 1$ to $N_{voxels}$ by 4 **do**
11:         { The "vec" operator loads 4 floats into SIMD vector}
12:         $V_{tissue}$ = vec($Tissue_{row}[idx \ldots idx + 4]$)
13:         $V_t$ = vec($T_t[idx \ldots idx + 4]$)
14:         $N(V_t) = \{V_{prev}, V_{next}, V_{slice+}, V_{slice-}, V_{row+}, V_{row-}\}$
15:         $T_{t+\Delta t}[idx \ldots idx + 4]$ = SIMDUpdate($constant, V_{tissue}, V_t, N(V_t)$);
16:       **end for**
17:       start($DMA_{retrieve}, chunkId + 2$) {$chunkId + 1$ is "in-flight"}
18:       start($DMA_{send}, T_{t+\Delta t}$)
19:     **end for**
20:   **end for**
21:   wait($barrier$) {Synchronize all SPUs}
22:   $t = t + \Delta t$
23: **end while**
```

2.4 Data and Validation

Ice ball borders are clearly visualized on CT acquired at the end of the first freeze stage [12,13]. Contours were traced by an experienced radiologist, providing the "ground truth" ice ball extent (see Fig. 4). The algorithms were validated by comparing the 0°C isotherm from simulation to the ice ball border. Four measures were used [14]. The Dice Similarity Coefficient (DSC) is a measure volume overlap (ranging from 0 to 1). Mean absolute distance (MAD) is the mean of the absolute distances from each point in one isotherm to the nearest boundary point in the "ground truth". Mean signed distance (MSD) calculates the mean of the signed distances. The Hausdorff distance finds the largest distance between the two contours. Thick slices necessitate calculating each measure in 2D on each slice.

3 Results

Table 3. Procedure details

ID	Probes	T_{sim}	Lesion Location	Resolution(mm^3)	Isotropic Size(MiB)
1	4	8	left lower anterior chest wall	$0.54x0.54x5$	14.2
2	1	14	right pubic ramus	$0.54x0.54x2$	8.6
3	4	10	left iliac bone	$0.74x0.74x3$	32.2
4	4	14	right superior pubic ramus	$0.66x.066x5$	24.2
5	3	14	right acetabular roof	$0.58x0.58x5$	23.2
6	3	12	left sacrum	$0.86x0.86x5$	27.7

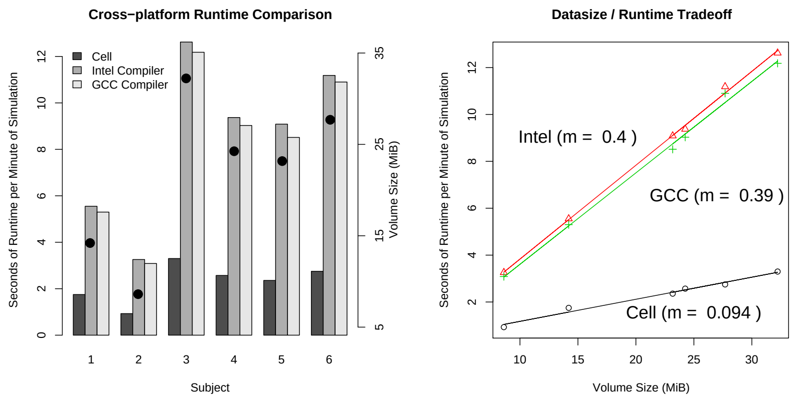
This retrospective study was compliant with the Health Insurance Portability and Accountability Act and had institutional review board approval: informed consent was waived. From the population of patients undergoing percutaneous cryoablation treatment for palliation of painful metastases [2,15] at our institution, six patients were selected at random (Table 3).

Figure 2(a) summarizes the relative performance of the algorithm. The three programs were executed once to preload data, then executed five times and the execution times averaged. The x86 server is a 2.66GHz eight core x86 machine (HS21 Blade, IBM, Armonk, New York) running CentOS 5. The Cell server is a 3.2GHz 16 SPU machine running Fedora 7 (QS21 Blade, IBM, Armonk, New York). Normalized execution time is expressed in seconds per minute of simulated time, a measure intuitively useful for the physician. The Cell consistently outperforms the x86 server.

Figure 2(b) shows architecture scalability relative to data size. The x86 server scales as 0.4 seconds per MiB, while the Cell server scales as 0.1 second per MiB, indicating the Cell processor more gracefully scales to large data volumes.

Figure 3 compares the SMP-related speedup of the Intel and Cell programs. The x86 program was run on 1, 2, 4, and 8 CPUs with execution time of five runs averaged. Speedup under GCC was comparable but omitted for brevity. The Cell program was executed with 1, 2, 4, 8 and 16 SPUs and five runs were averaged. SMP speedup curves relative to the single CPU/SPU were calculated. While the x86 exhibits nearly linear speedup with increasing numbers of cores, the Cell is approximately linear to 8 SPUs, but sub-linear at 16 SPUs.

The similarity measures outlined in §2.4 were computed for each run of the algorithm. The measures were identical for all runs of the two x86 programs. The Cell runs were likewise identical. The results are given in Table 4. The results for Subject 2 were not included in the means as the ice ball extents could



(a) Normalized execution time (b) Scalability

Fig. 2. Comparison of execution time, data size and scalability. Run times over five executions of each algorithm were averaged and compared on subject by subject basis (a). Scalability was measured as the slope of normalized run time vs. volume size (b). All timings are expressed in seconds of run time per minute of simulation time, a measure intuitive to physicians.

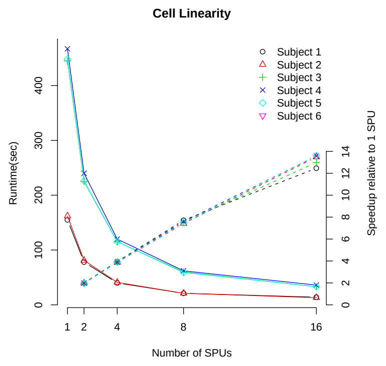
not be fully visualized (note the low DSC measurements). A DSC score of 0.8 or more is considered good agreement. This was achieved in 3 of 6 cases for the x86, and nearly 4 of 6 for Cell. Differences in similarity scores arise from implementation (OpenMP vs. streaming SIMD) and hardware floating point implementations.

Qualitative validation of the simulation results are shown in Fig. 4. After 8 minutes of freezing, ice ball extents are clearly visualized in CT (Fig. 4(a)). The 0°C isotherm was manually traced (Fig. 4(b)). Simulation results (Fig. 4(c)) demonstrate good visual agreement with manual delineation (Fig. 4(d)).

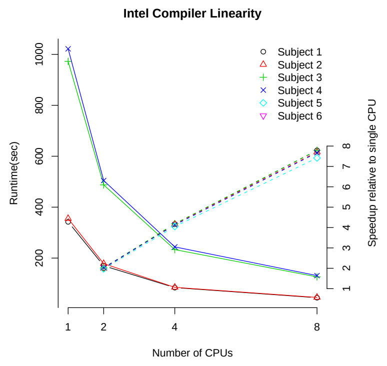
4 Discussion

Our first acceleration strategy, OpenMP, exhibited excellent performance. OpenMP is a simple programming model, scaling well up to 8 CPUs. Somewhat surprisingly, the commercial Intel compiler was outperformed by the open source GCC compiler. The compiler flags were set based on documented recommendations from the vendor, but extensive exploration of flag combinations is beyond the scope of this work.

The second acceleration strategy, a streaming SIMD algorithm implemented for Cell, surpassed the x86 in execution speed. The Cell implementation benefited from a SIMD instruction set and the explicit management of local store (cache) memory. Implementation was less straightforward than OpenMP. Our simulator



(a) Cell SMP speedup



(b) x86 SMP speedup

Fig. 3. SMP speedup for Cell (a) and x86 (b) relative to the single SPU case. x86 results are presented for the Intel compiler, GCC results are comparable.

system is a balance of computation and data movement directly benefiting from Cell's unique capabilities. Key to the Cell's performance is double buffering performing simultaneous DMA transfers and SIMD computations.

Our experiments showed better scalability in the Cell. With a slope of 0.1 seconds of execution per minute simulation per MiB, the Cell scales four times better than the x86. Increased scalability allows solutions computed at finer resolutions, if demanded by the physician. Performance under different conditions will be more uniform in the Cell.

Low values of MAD and MSD indicate good agreement of simulation results with manual identification of ice ball borders. Large Hausdorff distances indicate large local deviations in contours.

Study Limitations. Our study is far from comprehensive, and lacks several important experiments. Ice Ball shapes produced by the simulation must conform to reality. The measures used in this study fail to account for differences in shape, focusing on contour distances. If simulated ice balls have proper shape, physicians will have increased confidence in the simulator results.

Our performance evaluation of x86 and Cell was biased. For developer expediency, the x86 implementation did not use SIMD instructions. Cell architecture demands SIMD implementation for reasonable performance. If SIMD instructions on the x86 were implemented, run times may improve. We suspect the x86 has saturated the main memory bus, thus SIMD instructions will not improve the scalability, i.e. slope of Fig. 2(b). Development burden of managing the Cell's local store is offset by deterministic performance. While we have not endeavored to do so, tweaking this algorithm for x86 cache performance is a daunting task. Clearly, additional investigation is required.

Table 4. Simulation contours compared to ground truth[a].

	Cell				x86			
Subject	DSC	MAD	MSD	Haus.	DSC	MAD	MSD	Haus.
1	0.84	2.80	1.68	14.8	0.81	3.20	2.40	15.5
2[b]	0.41	7.36	7.12	19.7	0.40	7.54	7.35	20.4
3	0.79	5.11	−3.30	18.6	0.83	4.38	−2.49	18.6
4	0.79	3.24	1.30	17.9	0.75	3.64	2.05	17.9
5	0.73	4.38	2.34	20.2	0.70	4.76	3.01	20.2
6	0.78	4.65	−0.04	25.0	0.81	4.21	0.61	25.0
Mean[b]	0.79	4.03	0.40	19.3	0.78	4.04	1.12	19.4

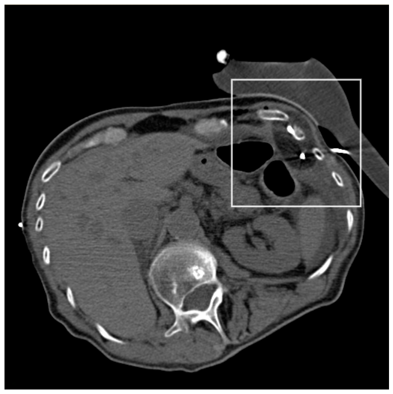
[a] All units are in *mm*
[b] Ice contours were not visualized for Subject 2; data were omitted from the mean.

Comparative Works. Prostate cryoablation simulation has been previously studied [6,5,16,17,18,19] Fast simulation is required for optimization of cryoprobe geometry [20]. Validation of simulation results has been limited to temperature measurements at locations near the cryoprobe tip [21]. *In vivo* measurements show the CT-visualized ice ball border corresponds with the 0°C isotherm [13]. Our future plans include experiments to couple CT visualization with point-wise temperature measurements for validation.

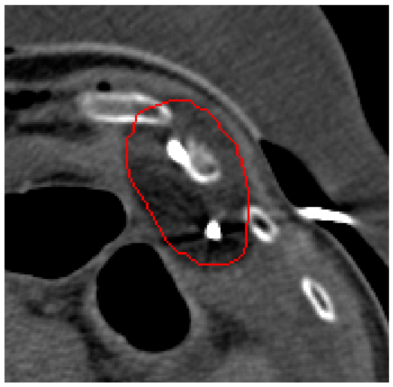
Conclusion. Simulation and validation of cryoablation procedures is imperative for physician acceptance. Though common practice at our institution [3,22,1,15,2,23], percutaneous MSK cryoablation is a nascent technology in the broader clinical community. Availability of a validated planning system for MSK cryoablation will speed procedure acceptance at institutions will small procedure volumes. In such instances, physicians will not be able to build the experience necessary to mentally plan cryoablation procedures. Instead, they will turn to simulation systems as outlined in this work. Our results demonstrate the utility of the Cell architecture for cryoablation simulation.

References

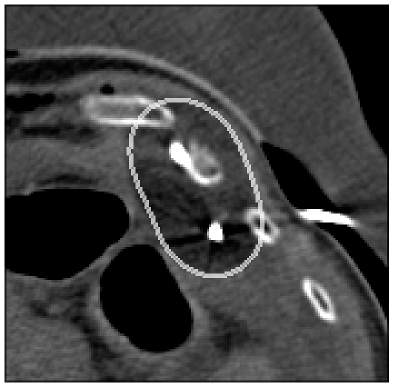
1. Atwell, T.D., Farrell, M.A., Leibovich, B.C., Callstrom, M.R., Chow, G.K., Blute, M.L., Charboneau, J.W.: Percutaneous renal cryoablation: Experience treating 115 tumors. J Urol (Apr 2008)
2. Callstrom, M.R., Charboneau, J.W.: Image-guided palliation of painful metastases using percutaneous ablation. Techniques in Vascular and Interventional Radiology **10** (2007) 120–131
3. Atwell, T.D., Farrell, M.A., Callstrom, M.R., Charboneau, J.W., Leibovich, B.C., Frank, I., Patterson, D.E.: Percutaneous cryoablation of large renal masses: techni-



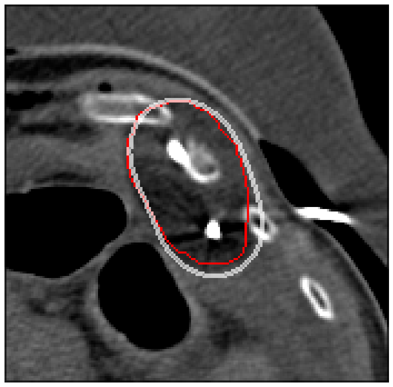
(a) CT after 8 minutes of freeze



(b) Manual ice ball delineation



(c) Simulation 0° isotherm



(d) Manual and simulation

Fig. 4. Qualitative evaluation of simulation results. Ice ball borders are clearly visualized after 8 minutes of freezing in (a). Borders were manually delineated in (b), 0°C isotherm was simulated (c) and compared to manual in (d).

cal feasibility and short-term outcome. AJR Am J Roentgenol **188**(5) (May 2007) 1195–1200
4. Rabin, Y., Korin, E.: An efficient numerical solution for the multidimensional solidification (or melting) problem using a microcomputer. International Journal of Heat Mass Transfer **36**(3) (1993) 673–683
5. Deng, Z.S., Liu, J.: Numerical simulation of 3-D freezing and heating problems for combined cryosurgery and hyperthermia therapy. Numerical Heat Transfer, Part A **46** (2004) 587–611
6. Zhang, Y.T., Liu, J.: Numerical study on three-region thawing problem during cryosurgical re-warming. Medical Engineering and Physics **24** (2002) 265–277
7. Pennes, H.H.: Analysis of tissue and arterial blood temperature in resting human forearm. Journal of Applied Physics **1** (1948) 93–122
8. Dagum, L., Menon, R.: OpenMP: an industry standard API for shared-memory programming. IEEE Computational Science & Engineering **5**(1) (Jan.–March

1998) 46–55
9. Sato, M.: OpenMP: parallel programming API for shared memory multiprocessors and on-chip multiprocessors. In: Proc. 15th International Symposium on System Synthesis. (2002) 109–111
10. Kahle, J.A., Day, M.N., Hofstee, H.P., Johns, C.R., Maeurer, T.R., Shippy, D.: Introduction to the Cell multiprocessor. IBM Journal of Research and Development **49** (2005) 589–604
11. Ohara, M., Yeo, H., Savino, F., Iyengar, G., Gong, L., Inoue, H., Komatsu, H., Sheinin, V., Daijavad, S., Erickson, B.: Real-time mutual-information-based linear registration on the Cell Broadband Engine processor. In: IEEE International Symposium on Biomedical Imaging: From Nano to Macro. (2007)
12. Permpongkosol, S., Link, R.E., Kavoussi, L.R., Solomon, S.B.: Temperature measurements of the low-attenuation radiographic ice ball during ct-guided renal cryoablation. Cardiovasc Intervent Radiol **31**(1) (2008) 116–121
13. Sandison, G.A., Loye, M.P., Rewcastle, J.C., Hahn, L.J., Saliken, J.C., McKinnon, J.G., Donnelly, B.J.: X-ray CT monitoring of iceball growth and thermal distribution during cryosurgery. Phys Med Biol **43**(11) (Nov 1998) 3309–3324
14. Crum, W., Camara, O., Hill, D.: Generalized overlap measures for evaluation and validation in medical image analysis. IEEE Transactions on Medical Imaging **25**(11) (Nov. 2006) 1451–1461
15. Callstrom, M.R., Atwell, T.D., Charboneau, J.W., Farrell, M.A., Goetz, M.P., Rubin, J., Sloan, J.A., Novotny, P.J., Welch, T.J., Maus, T.P., Wong, G.Y., Brown, K.J.: Painful metastases involving bone: Percutaneous image-guided cryoablation–prospective trial interim analysis. Vascular and Interventional Radiology **241**(2) (2006) 572–580
16. Rabin, Y., Lung, D.C., Stahovich, T.F.: Computerized planning of cryosurgery using cryoprobes and cryoheaters. Technology in Cancer Research & Treatment **3**(3) (June 2004)
17. Jankun, M., Kelly, T.J., Zaim, A., Young, K., Keck, R.W., Selman, S.H., Jankun, J.: Computer model for cryosurgery of the prostate. Computer Aided Surgery **4** (1999) 193–199
18. Kim, C., O'Rourke, A.P., Mahvi, D.M., Webster, J.G.: Finite-element analysis of *ex vivo* and *in vivo* hepatic cryoablation. IEEE Transactions on Biomedical Engineering **54**(7) (2007) 1177–1185
19. Rewcastle, J.C., Sandison, G.A., Hahn, L.J., Saliken, J.C., McKinnon, J.G., Donnelly, B.J.: A model for the time-dependent thermal distribution within an iceball surrounding a cryoprobe. Physics in Medicine and Biology **43** (1998) 3519–3534
20. Baissalov, R., Sandison, G.A., Reynolds, D., Muldrew, K.: Simultaneous optimization of cryoprobe placement and thermal protocol for cryosurgery. Physics in Medicine and Biology **46** (2001) 1799–1814
21. Fortin, A., Belhamadia, Y.: Numerical prediction of freezing fronts in cryosurgery: Comparison with experimental results. Computer Methods in Biomechanics and Biomedical Engineering **8**(4) (August 2005) 241–249
22. Atwell, T.D., Farrell, M.A., Callstrom, M.R., Charboneau, J.W., Leibovich, B.C., Patterson, D.E., Chow, G.K., Blute, M.L.: Percutaneous cryoablation of 40 solid renal tumors with us guidance and ct monitoring: initial experience. Radiology **243**(1) (Apr 2007) 276–283
23. Callstrom, M.R., Charboneau, J.W., Goetz, M.P., Rubin, J., Atwell, T.D., Farrell, M.A., Welch, T.J., Maus, T.P.: Image-guided ablation of painful metastatic bone tumors: a new and effective approach to a difficult problem. Skeletal Radiology **35**(1) (Jan 2006) 1–15