Massively Parallel Energy Space Exploration for Uncluttered Visualization of Vascular Structures

Yongkweon Jeon, Joong-Ho Won, *Member, IEEE*, and Sungroh Yoon*, *Senior Member, IEEE*

***Abstract*—Images captured using computed tomography and magnetic resonance angiography are used in the examination of the abdominal aorta and its branches. The examination of all clinically relevant branches simultaneously in a single 2-D image without any misleading overlaps facilitates the diagnosis of vascular abnormalities. This problem is called *uncluttered single-image visualization* (USIV). We can solve the USIV problem by assigning energy-based scores to visualization candidates and then finding the candidate that optimizes the score; this approach is similar to the manner in which the protein side-chain placement problem has been solved. To obtain near-optimum images, we need to explore the energy space extensively, which is often time consuming. This paper describes a method for exploring the energy space in a massively parallel fashion using graphics processing units. According to our experiments, in which we used 30 images obtained from five patients, the proposed method can reduce the total visualization time substantially. We believe that the proposed method can make a significant contribution to the effective visualization of abdominal vascular structures and precise diagnosis of related abnormalities.**

***Index Terms*—Abdominal aorta, energy-space exploration, GPGPU, parallelization, single-image visualization.**

I. Introduction

MEDICAL imaging techniques such as computed tomography (CT) and magnetic resonance angiography (MRA) facilitate the examination of the abdominal aorta and its branches. Medical doctors can diagnose vascular abnormalities using images from the CT and/or MRA. It aids diagnosis to visualize all clinically relevant branches of the abdominal aorta simultaneously in a single 2-D stylistic image without any misleading overlap between branches [1]. This visualization problem is termed the *uncluttered single-image visualization* (USIV) [2], [3]. Fig. 1 shows an example.

The existing solution to the USIV problem can be summarized as follows: different spatial configurations of the bounding boxes of the vessels are calculated in a 2-D plane on the basis of the tubular 3-D geometry of the abdominal aorta and its branches. An optimization algorithm is then used to minimize a score function that considers overlaps among bounding boxes and deviation from the input. The resulting spatial configuration is almost similar to the projection of the input in a prespecified direction (mostly anteroposterior).

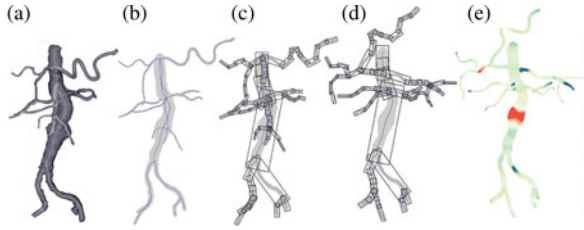

Fig. 1. Graphical illustration of the USIV algorithm [3]. (a) Surface rendering of the abdominal aortic vessel tree. (b) 2-D projection. (c) Bounding box representations. (d) Uncluttered bounding box configuration after minimizing the energy function. (e) Stylistic visualization of (d).

Because of the large number of possible bounding box configurations, the optimization technique critically affects the efficiency of the existing methods. Simulated annealing (SA) has been used in [2], but, despite using multiple speedup techniques, this method is computationally intensive and requires a fair amount of domain knowledge to limit the search space. The relation between the USIV problem and the protein side-chain placement (SCP) problem has been identified in [3], and the USIV problem has been reformulated as an integer linear programming (ILP) problem [4] over samples from the search space, inspired by the ILP-based solution to SCP [5]. The ILP-based solution shows better performance than the SA-based solution because in the latter, the search space samples are precomputed using graphics processing units (GPUs); however, the improvement by the ILP-based solution is rather limited because of unoptimized utilization of GPU resources.

In this study, we further enhance the ILP-based solution to USIV by incorporating four optimization techniques—three GPU-related techniques and one general technique. These techniques enable us to efficiently explore the optimization search space, where the exploration corresponds to enumerating scores represented by energy functions in the SCP formulation. According to our experiments, in which we used 30 images obtained from five patients, the proposed method can run 801 times faster than the SA-based solution and 13 times faster than the unoptimized ILP-based technique.

Manuscript received March 17, 2012; revised July 20, 2012; accepted July 23, 2012. Date of publication August 21, 2012; date of current version December 14, 2012. This work was supported by the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology under Grant 2011-0009963. *Asterisk indicates corresponding author*.

Y. Jeon is with the Department of Electrical and Computer Engineering, Seoul National University, Seoul 151-744, Korea (e-mail: yonkham@gmail.com).

J.-H. Won is with the School of Industrial Management Engineering, Korea University, Seoul 136-713, Korea (e-mail: wonj@korea.ac.kr).

*S. Yoon is with the Department of Electrical and Computer Engineering, Seoul National University, Seoul 151-744, Korea (e-mail: sryoon@snu.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBME.2012.2214386

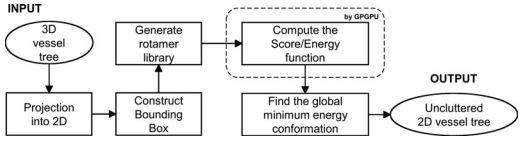

Fig. 2. Overview. The energy computation step is accelerated by GPUs.

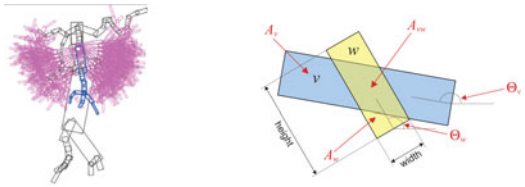

Fig. 3. (a) Rotamer library generation for the superior mesenteric artery. The thick blue boxes denote the reference configuration and the thin purple boxes denote the Monte Carlo sampled configurations. The black boxes are other vessels fixed in sampling. (b) Configuration of a pair of bounding boxes.

II. USIV

A. Overview of the USIV Algorithm

Fig. 2 shows the overall flow for solving the USIV problem [2], [3]. First, we project the tubular 3-D geometry of the abdominal aorta and its branches into a 2-D vessel tree. For each vessel segment in this tree, we consider *vessel parts* designed to resemble the actual shape of the segment; a branch of the abdominal aorta is then represented by a set of bounding boxes called a *rotamer*. All feasible rotamer conformation of a branch is called the *rotamer library* of that branch. The rotamer library of a branch is generated using the Monte Carlo sampling method [6]. Fig. 3(a) shows a rotamer library of the superior mesenteric artery. We anchor the vessel parts that correspond to the aorta and the common iliac arteries. This reduces the search space without significantly affecting the accuracy of the model anatomy. This set of anchored vessel parts is called the *backbone* of the abdominal aortic vessel tree. The configuration of the vessel parts is evaluated by a scoring function expressed in terms of energy, as defined in the SCP problem [5]. We associate two types of energy with rotamers. For each rotamer, we compute its self-energy, and for two rotamers, we calculate the pairwise energy between them. The main contribution of this study is to accelerate the calculation of the self and pairwise energy for all the rotamers. Finally, we minimize the energy function reformulated as an ILP problem [4] over the precomputed samples from the optimization search space.

B. Scoring Function

A configuration of vessel parts is represented by a vector $(\Theta_1, \ldots, \Theta_{|V|})$ of their orientations, where V denotes the index set of the vessel parts. Our goal is to find a configuration that minimizes the overlap between bounding boxes and the deviation from the reference configuration. To encode these objectives, we define a function $\mathcal{E}$ that scores a configuration of vessel parts as follows [2]:

$$\mathcal{E}(\Theta_1, \ldots, \Theta_{|V|}) = \Omega(\Theta_1, \ldots, \Theta_{|V|}) + \lambda \cdot \Delta(\Theta_1, \ldots, \Theta_{|V|}) \tag{1}$$

where Ω and Δ are the total overlap and the total deviation of the given configuration, respectively, and λ is a Lagrangian multiplier for achieving a tradeoff between the two objectives that are typically in conflict.

For a pair of bounding boxes (v, w), the areas of v and w, and their intersection are denoted by A_v, A_w, A_{vw}, respectively, as shown in Fig. 3(b). Computing Ω and Δ depends on two metrics called the *overlap* and the *deviation* metrics, respectively. The overlap metric is given by

$$\rho(v, w) = \alpha \cdot \max(A_{vw}/A_v, A_{vw}/A_w) \tag{2}$$

where constant $\alpha = 1$ if (v, w) is not a joint-sharing pair[1]; otherwise, $0 < \alpha < 1$. The deviation metric is

$$\delta(v, w) = \left\| \begin{pmatrix} \cos(\Theta_v - \Theta_w) \\ \sin(\Theta_v - \Theta_w) \end{pmatrix} - \begin{pmatrix} \cos(\Theta_v^{\text{ref}} - \Theta_w^{\text{ref}}) \\ \sin(\Theta_v^{\text{ref}} - \Theta_w^{\text{ref}}) \end{pmatrix} \right\|^2 \tag{3}$$

where Θ_i^{ref} is the orientation of box i in the reference.

C. Similarity to Protein SCP

It has been observed in [3] that USIV is similar to SCP, which is a key problem in protein structure prediction [5]. The immovable aorta and its branches correspond to the backbone and the side chains of a protein. The deviation and overlap scores in (1) are analogous to the covalent energy between bonded atoms and the van der Waals energy that avoids atomic overlaps, respectively.

Given n vessel parts and the fixed geometry of the reference aorta, we can redefine the scoring function [see (1)] using the SCP terminology as follows [3]:

$$\mathcal{E}(\mathbf{r}) = \sum_{i=1}^{n} \mathcal{E}_{\text{self}}(r_i) + \sum_{i<j}^{n} \mathcal{E}_{\text{pair}}(r_i, r_j) \tag{4}$$

where r_i is a rotamer at the ith residue (branch), $\mathbf{r} = (r_1, \ldots, r_n)$, $\mathcal{E}_{\text{self}}(r_i)$ is the self-energy of r_i including its interaction with the anchored backbone (aorta), and $\mathcal{E}_{\text{pair}}(r_i, r_j)$ is the pairwise interaction energy between r_i and r_j. The $\mathcal{E}_{\text{pair}}$ and $\mathcal{E}_{\text{self}}$ terms are given by

$$\mathcal{E}_{\text{pair}}(r_i, r_j) = \sum_{v \in r_i, w \in r_j} \rho(v, w) + \sum_{v \in r_i, w \in r_j, (v,w) \in J} \delta(v, w) \tag{5}$$

$$\mathcal{E}_{\text{self}}(r_i) = \sum_{v, w \in r_i} \rho(v, w) + \sum_{v, w \in J} \delta(v, w) + \mathcal{E}_{\text{pair}}(r_i, b) \tag{6}$$

where ρ and δ are the overlap and deviation metrics, respectively, b is the aorta backbone, and J is a set of joint-sharing vessel part pairs, as explained in Section II-B.

[1]Either a parent–child pair or a sibling for which overlap is unavoidable.

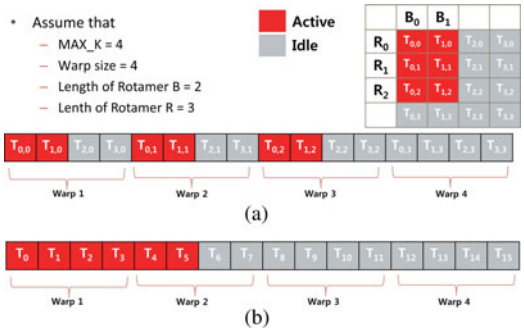

Fig. 4. (a) Example explaining why 2-D indexing and a single-grid launch is inefficient. (b) One-dimensional indexing is better for minimizing mixed warps.

III. Parallelization and Optimization

A. GPU Parallelization of Energy Computation

There is little dependence among the computation of self and pairwise energies of different configurations, thus providing abundant data-level parallelism suitable for operations on single-instruction multiple-data (SIMD) machines such as GPUs. A simple approach would be to index each GPU thread by a 2-D index (x, y), where x and y are the indices of each bounding box in two rotamers. Each thread reads the appropriate rotamer information (e.g., coordinates, angles, and parents) from the global memory of the GPU device and then calculates the overlap and deviation for the corresponding pair of rectangles. To obtain the final self or pairwise energy of the rotamer, we employ a parallel sum reduction technique, in which each thread stores the computation result in its dedicated area of the GPU's shared memory. Although 2-D thread indexing is easy to understand, it does not have the best performance. The next sections present four optimization techniques for performance improvement.

B. Optimization 1: Maximizing Thread Utilization

Assume that rotamers B and R consist of n and m number of boxes, respectively. As stated earlier, to calculate the pairwise energy between B and R, we might use $n \times m$ threads, assigning one to each of the $n \times m$ combinations. A basic approach would use this type of 2-D thread indexing along with a single launch of the GPU grid, but this will significantly degrade the performance.

The reason for this degradation is explained using an example in Fig. 4(a). To complete the entire computation in a single-grid launch, assume that the maximum size of a rotamer is set to `MAX_K`, and `MAX_K` $\times$ `MAX_K` threads are statically assigned for computing the pairwise energy of every rotamer pair. Furthermore, assume that threads are 2-D indexed by $T_{x,y}$. In the example, $n = 2$, $m = 3$, and `MAX_K`$= 4$. In the table in Fig. 4(a), only six threads are active out of sixteen available threads in this suboptimal scheme, leaving much scope for improvement. Another problem is that a warp (one SIMD group [7]) of threads can consist of active and inactive threads at the same time. Inactive threads wait for the active threads to complete their processes, thus wasting scalar processors. In Fig. 4(a), we assume that the warp size is four. Then, three out of four warps comprise active and inactive threads.

Using 1-D thread indexing can help reduce the number of mixed warps, as shown in Fig. 4(b). If we use 1-D thread indexing, warp 1 is fully active, warps 3 and 4 are idle, and only warp 2 is a mixed warp. If 1-D indexing is used, the number of mixed warps is limited to one. Moreover, if we launch multiple grids, then the number of threads to be assigned can be decided according to the size of rotamers involved in the computation instead of always assigning `MAX_K` $\times$ `MAX_K` threads. For example, in Fig. 4(b), warps 3 and 4 would not have been created if we assigned only $2 \times 3 = 6$ threads in the beginning instead of $4 \times 4 = 16$.

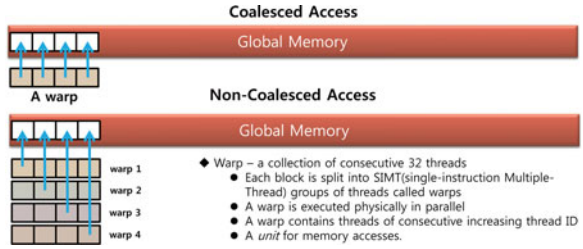

Fig. 5. Comparison of coalesced and noncoalesced memory access [7].

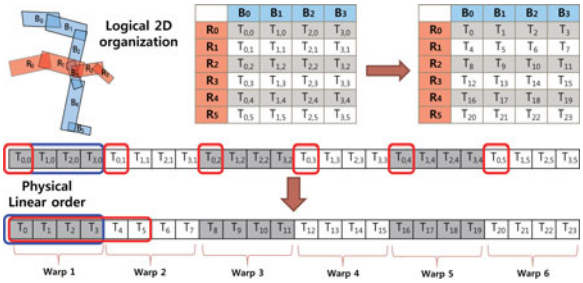

Fig. 6. Facilitating coalesced memory access by 1-D indexing.

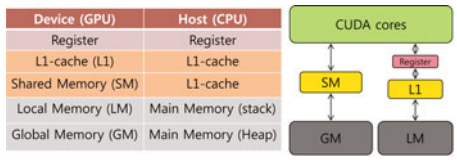

Fig. 7. Comparison of memory hierarchy in CPUs and GPUs [7].

C. Optimization 2: Coalesced Memory Access

The latency of global and local memories in the GPU is high, and we need to coalesce the access to such memory for higher performance, as explained in Fig. 5. This concept of coalesced memory access is an important optimization technique in GPU programming [7].

We state that 1-D thread indexing is better than 2-D thread indexing for coalesced memory access. Fig. 6 shows an example supporting this fact. Assume that a warp is composed of four threads and that two rotamers R and B consists of six boxes (R_0–R_5) and four boxes (B_0–B_3), respectively. To compute the pairwise energy between R and B, we could use 2-D thread indices $T_{x,y}$ (x: index for B; y: index for R) as shown in the left

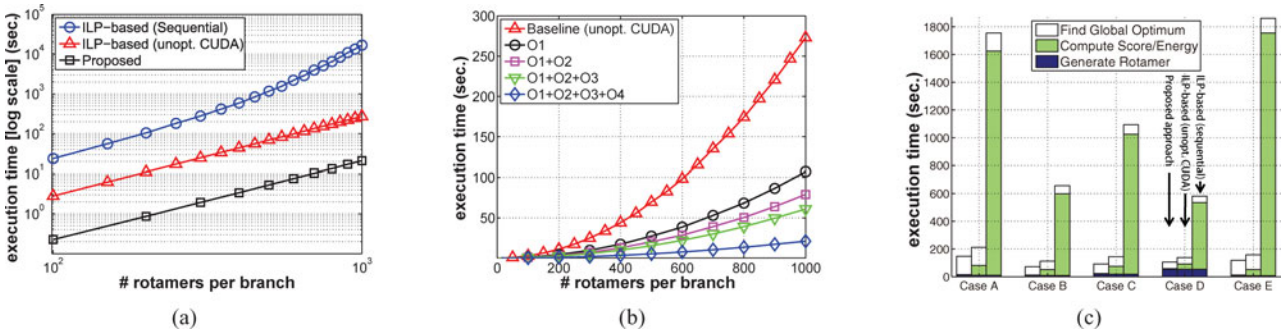

Fig. 8. (a) Running time for the energy-computation step: sequential method versus the ILP-based (unoptimized CUDA parallelization) method [3] versus the proposed method (data: Case B). (b) Effect of the proposed optimization techniques on the time taken for energy computation. Baseline: the ILP-based approach [3], Data: Case B [2]. (c) Breakdown of the total running time (number of rotamers per branch = 400).

table in Fig. 6. Threads $T_{x,0}$ can read the information for boxes B_0–B_3 from memory through coalesced access. However, if threads $T_{0,y}$ read the data for R_0–R_5 in memory, then we need six cycles to fetch the data. This will significantly degrade the overall performance. To address this issue, we use 1-D thread indices T_z, as shown in the right table in Fig. 6, using which we can obtain the data for R_0–R_5 in only two cycles. In this specific example, the indices of B and R can be calculated by $\lfloor z/4 \rfloor$ and z mod 4, respectively.

D. Optimization 3: Increasing L1 Cache Size

The memory hierarchy of a GPU system is different from that of CPU systems. In Fig. 7, the "register-L1-LM (local memory)" line corresponds to the conventional hierarchy for the stack (the device manages its usage), whereas the "shared memory (SM)–global memory" line resembles the heap (the user allocates/releases it). Since the data stored in the SM are shared by all threads in a thread block, the SM works as a fast, convenient communication medium among the threads. L1 works as the cache for automatic variables, and it is desirable to have a sufficiently large L1 to reduce cache misses. In the GPU device we use, the size of L1 and SM is user configurable but the combined size is fixed as 64 kB. We use the SM to store the rotamer information and intermediate results. Through memory usage profiling, we have discovered that the information stored for each block is less than 16 kB. In contrast, each thread uses many local variables for computation; hence, a large L1 cache is appropriate. Thus, we have decided to increase the size of L1 as much as possible (i.e., 48 kB) and decrease the size of SM (i.e., 16 kB).

E. Optimization 4: Eliminating Redundant Computation

To compute the overlap metric by (2), we need to calculate three types of areas: A_v, A_w, and A_{vw}. However, A_v and A_w are constant for a rotamer library and can be computed only once for the reference configuration, and reused later without recomputation. This is a rather straightforward technique, but was not spotted in [3].

IV. Results and Discussion

For evaluation, we used the CT data used in [2] and [3], which was five datasets labeled, Cases A–E (collected from five patients, six images per case). The setup used was as follows: an Intel 2.66-GHz Core i5 processor with 8-GB RAM, an NVIDIA GeForce GTX465 (1 GB DDR5; 352 scalar processors), 64-bit Ubuntu Linux 10.10, and Compute Unified Device Architecture (CUDA) Toolkit 4.0.17.

Fig. 8(a) shows the comparison of the time taken to compute the energy scores of all rotamers in Case B. We varied the number of rotamers from 100 to 1000. With respect to sequential and partially parallelized ILP-based methods, our approach is 801× and 13× faster in this experiment. Fig. 8(b) shows how the proposed optimization techniques affect the time taken to compute all the energy values in Case B for different rotamer library sizes. The baseline is the unoptimized ILP-based approach [3] partially parallelized with CUDA. The amount of improvement by each of the four optimization techniques was 2.58× (O1), 1.35× (O2), 1.30× (O3), and 3.00× (O4) by using the baseline for comparison. Thus, the proposed method improves the performance of the baseline by 13×. The improvements by O1 and O4 were the maximum. This result implies that a large amount of energy computation is redundant and removing the redundancy can substantially reduce the running time. Furthermore, we can confirm the importance of maximizing the number of active threads by carefully allocating thread runs. Fig. 8(c) shows the breakdown of the total running time of the proposed and the ILP-based approaches [3]. The running time includes unparallelized steps such as the rotamer library generation and the optimization. The number of rotamers per branch was 400.

Comparison of the SA-based and ILP-based methods in terms of the deviation metric suggests that their performance is highly similar ($p = 0.05$, test of noninferiority with 15% tolerance) [3]. Since this study accelerates the ILP-based method without modifying the final visualization, the same conclusion applies. It was also found that the monotone increasing trend between the subjective distortion rating and the (algorithmic) deviation metric is significant ($p < 0.0001$, Jonckheere–Terpstra test [8]). In other words, branches with a lower distortion rating had lower branch rotation and lower branch distortion metrics. Since the outputs of the SCP-based method behave similarly to those of

the SA-based one, we expect that clinicians would perceive them likewise.

Overall, our approach was very effective in accelerating the energy-space exploration involved in the USIV. It took less than 3 min to process an image in all the tested cases. For near real-time visualization, we are planning to parallelize the rotamer library generation and the optimization steps. The former is based on Monte Carlo simulation, and we expect it to be suitable for parallelization.

REFERENCES

[1] J. Meaney, J. C. Weg, T. L. Chenevert, D. Stafford-Johnson, B. H. Hamilton, and M. R. Prince, "Diagnosis of pulmonary embolism with magnetic resonance angiography," *New Engl. J. Med.*, vol. 336, no. 20, pp. 1422–1427, May 1997.

[2] J.-H. Won, J. Rosenberg, G. D. Rubin, and S. Napel, "Uncluttered single-image visualization of the abdominal aortic vessel tree: Method and evaluation," *Med. Phys.*, vol. 36, no. 11, pp. 5245–5260, Nov. 2009.

[3] J.-H. Won, Y. Jeon, J. Rosenberg, S. Yoon, G. D. Rubin, and S. Napel, "Uncluttered single-image visualization of vascular structures using GPU and integer programming," *IEEE Trans. Vis. Comput. Graphics*, in press. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/22291148

[4] A. Schrijver, *Theory of Linear and Integer Programming*. Hoboken, NJ: Wiley, 1998.

[5] C. L. Kingsford, B. Chazelle, and M. Singh, "Solving and analyzing side-chain positioning problems using linear and integer programming," *Bioinformatics*, vol. 21, no. 7, pp. 1028–1036, Apr. 2005.

[6] G. S. Fishman, *Monte Carlo Concepts, Algorithms, and Applications*. New York: Springer, 1996.

[7] R. Farber, *CUD Application Design and Development*. San Mateo, CA: Morgan Kaufmann, 2011.

[8] A. R. Jonckheere, "A distribution-free k-sample test against ordered alternatives," *Biometrika*, vol. 41, no. 1–2, pp. 133–145, Jun. 1954.