计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2015.20140356
52(6):1431-1442,2015

一种基于并行 SNESIM 的空间数据重建方法

张 挺[1] 杜 奕[2,3] 黄 涛[3] 李 雪[3]
[1](上海电力学院计算机科学与技术学院 上海 200090)
[2](上海第二工业大学计算机与信息学院 上海 201209)
[3](中国科学技术大学工程科学学院近代力学系 合肥 230027)
(tingzh@shiep.edu.cn)

A Reconstruction Method for Spatial Data Using Parallel SNESIM

Zhang Ting[1], Du Yi[2,3], Huang Tao[3], and Li Xue[3]
[1](*College of Computer Science and Technology, Shanghai University of Electric Power, Shanghai* 200090)
[2](*School of Computer and Information, Shanghai Second Polytechnic University, Shanghai* 201209)
[3](*Department of Modern Mechanics, School of Engineering Science, University of Science and Technology of China, Hefei* 230027)

Abstract The application of spatial data is becoming increasingly large. Interpolation can effectively reconstruct the unknown data in space, which is actually a process of data reproduction, and also a process of reproducing data with higher resolution from original data. Interpolation methods are divided into two branches: definite interpolation and indefinite interpolation. On one hand, the uncertainty of indefinite interpolation shows in selecting certain stochastic interpolation ways; on the other hand, the uncertainty is reflected by selecting the interpolation parameters using probability principles. Multiple-point simulation (MPS) is an important indefinite interpolation method in reconstructing spatial data, and single normal equation simulation (SNESIM), as a frequently used MPS method, has been used in three-dimensional reconstruction of categorical spatial data in many fields currently. However, due to the large burdens on CPU and memory brought in by SNESIM, its practical application has been limited greatly. To overcome this limitation, SNESIM is parallelized using compute unified device architecture(CUDA). A proper size of data template is chosen using the entropy theory of training image (TI) and the reconstruction quality is improved by the integration of soft data and hard data. Compared with the CPU-based SNESIM method, the CUDA-based one shows the better reconstruction efficiency of spatial data.

Key words spatial data; multiple-point simulation (MPS); compute unified device architecture (CUDA); entropy; soft data

摘 要 空间数据的应用领域正在不断扩大.数据插值可以有效重建空间未知数据.数据插值就是一个数据再生的过程,即由原始数据再生出具有更高分辨率的数据.插值方法分为"确定"性插值和"不确定"

收稿日期:2014-04-22;**修回日期**:2014-09-23
基金项目:中国科学院战略性先导科技专项项目(XDB10030402);国家"九七三"重点基础研究发展计划基金项目(2011CB707305);国家科技重大专项基金项目(2011ZX05009-006);上海市自然科学基金项目(11ZR1413700,12ZR1412000);上海市优秀青年教师培养资助计划项目(ZZsdl12002,ZZsdl13015);上海电力学院人才引进基金项目(K2012-004,K2013-019,K2014-020);上海第二工业大学校级重点学科建设软件服务工程基金项目(XXKZD1301)
通信作者:杜 奕(duyi0701@126.com)

性插值方法.不确定性插值方法的不确定性一方面表现在选用的插值方式具有随机性,另一方面表现在插值参数的选取和确定需要依赖于概率统计原则.多点随机模拟法(multiple-point simulation, MPS)是实现空间数据不确定插值重建的重要手段.单一标准方程模拟(single normal equation simulation, SNESIM)作为一种常用的 MPS 方法,目前已经用于多个领域的离散型空间数据三维重建.但是由于 SNESIM 给 CPU 和内存带来的负荷较大,大大限制了其实际应用.为了克服这种局限性,基于统一计算设备架构(compute unified device architecture, CUDA)实现 SNESIM 的并行化,并在计算训练图像(training image, TI)熵的基础上选择合适的数据模板尺寸;同时,通过整合软硬数据提高重建质量.与以往基于 CPU 的重建方法相比,基于 CUDA 的 SNESIM 并行算法显示出更好的空间数据重建效率.

关键词 空间数据;多点随机模拟;统一计算设备架构;熵;软数据

中图法分类号 TP391.41

空间数据是指包含空间特征的数据集合[1-5].目前重建大范围真实有效的空间数据存在一定的难度,出现这种情况的主要原因是科学实验和勘探开发的费用都非常高,而数据插值成为重建空间数据的一个有效手段[6-7].插值方法大体可分为"确定"性插值方法和"不确定"性插值方法[8-11].

目前多点随机模拟方法(multiple-point simulation, MPS)是不确定性插值的研究热点[12-13].由 Strebelle 提出的单一标准方程模拟(single normal equation simulation, SNESIM)是离散型空间数据重建的代表性 MPS 方法[14].最初的 MPS 在每次模拟一个点时都要重新扫描一遍训练图像(training image, TI),以获得对应点的条件概率分布函数(conditional probability distribution function, cpdf),这严重影响了模拟速度.SNESIM 使用"搜索树"数据结构存储 cpdf,可以加快重建数据的速度.

空间数据重建过程中的待处理数据量一般很大,包括 SNESIM 在内的很多 MPS 方法往往基于 CPU 的串行处理.为了提高模拟效率,产生了一些并行方法用于 MPS 方法加速.Nunes 和 Almeida[15] 给出了 3 个经典序贯模拟方法的并行实现.Peredo 和 Ortiz[16]实现了并行的模拟退火方法.Huang 等人[17-18]给出了基于 GPU 的并行 MPS 方法实现.通过对 MPS 实施并行处理,可望解决其 CPU 和内存的瓶颈问题.

在文献[18]的基础上,本文提出一种基于统一计算设备架构(compute unified device architecture, CUDA)的 SNESIM 并行算法以实现空间数据重建.数据模拟过程中结合软硬条件数据的使用,大大提高了重建准确性;通过图像熵理论来确定模板尺寸;实验结果对并行后的模拟效率进行了详细讨论和分析.

1 SNESIM 概述

设数据模板为 $\boldsymbol{\tau}_n$,它是由 n 个向量组成的几何形态,$\boldsymbol{\tau}_n=\{\boldsymbol{h}_\alpha;\alpha=1,2,\cdots,n\}$.设模板中心位置为 $\boldsymbol{u}$,模板其他位置 $\boldsymbol{u}_\alpha=\boldsymbol{u}+\boldsymbol{h}_\alpha(\alpha=1,2,\cdots,n)$[19].图 1(a)(b)分别给出了二维和三维数据模板示意图,二维数据模板中心为 $\boldsymbol{u}$,三维数据模板中心用深灰色表示.假定一种属性 S 可取 K 个状态值$\{s_k;k=1,2,\cdots,K\}$,d_n 表示 n 个向量在位置 $\boldsymbol{u}_\alpha$ 的 $S(\boldsymbol{u}_1),\cdots,S(\boldsymbol{u}_n)$分别为状态值 $s_{k_1},\cdots,s_{k_n}$:

$$d_n=\{S(\boldsymbol{u}_\alpha)=s_{k_\alpha};\alpha=1,2,\cdots,n\}. \quad (1)$$

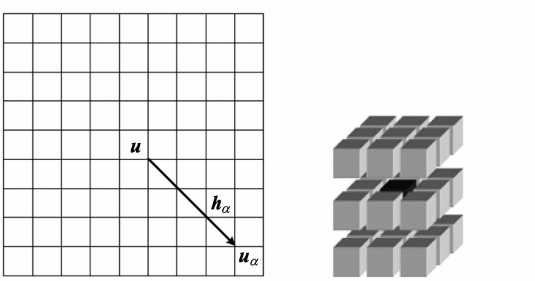


Fig. 1 Data templates.

图 1 数据模板

一般采用数据模板扫描训练数据(通常表现为训练图像形式)来捕获模式,这些模式可以视为训练数据的结构特征.例如,图 2(a)所示的一个 3×3 的数据模板扫描一个二维训练图像,图 2(b)是捕获的一个模式.如果按照每次移动一个节点,从左到右、从上到下进行扫描,那么就可以获得训练数据全部的模式库,如图 3 所示[20].对于三维训练图像同理可以建立类似的模式库.

在应用任一给定的数据模板对训练图像扫描的过程中,当数据模板捕获一个模式(由待模拟点 $\boldsymbol{u}$ 和

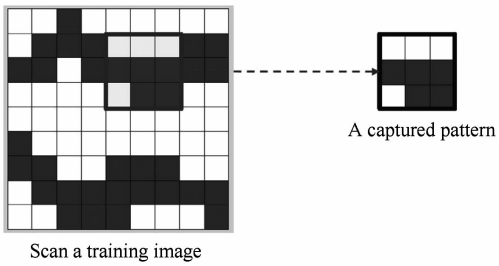


Fig. 2　Capture a pattern in a training image.

图 2　捕获训练图像中的一个模式

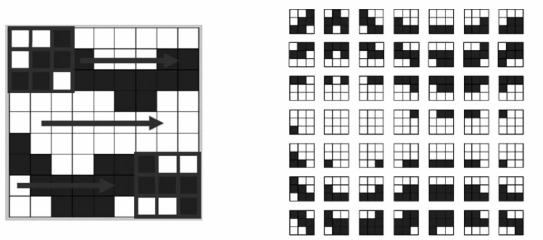


Fig. 3　Capture a pattern database from a training image.

图 3　获取训练图像的模式库

其所在数据模板内的条件数据所组成),称为一个重复. 如果模式库中某种模式重复出现 N 次,则称为 N 个重复. 对于任一待模拟点 $\boldsymbol{u}$,需要确定在给定 n 个条件数据值 $S(\boldsymbol{u}_\alpha)$ 的情况下,属性 $S(\boldsymbol{u})$ 取 K 个状态值中任一个状态值的条件概率分布函数 cpdf. 根据贝叶斯条件概率公式,该 cpdf 可表达为

$$Prob\{S(\boldsymbol{u})=s_k \mid d_n\}=\frac{Prob\{S(\boldsymbol{u})=s_k \text{ and } S(\boldsymbol{u}_\alpha)=s_{k_\alpha};\alpha=1,\cdots,n\}}{Prob\{S(\boldsymbol{u}_\alpha)=s_{k_\alpha};\alpha=1,\cdots,n\}},\tag{2}$$

其中,分母为某个模式出现的概率;分子为模式和待模拟点 $\boldsymbol{u}$ 取某个状态值的情况同时出现的概率. 因此,式(2)也可以表示为

$$Prob\{S(\boldsymbol{u})=s_k \mid S(\boldsymbol{u}_\alpha)=s_{k_\alpha};\alpha=1,\cdots,n\}=\frac{c_k(d_n)}{c(d_n)},\tag{3}$$

其中,$c(d_n)$ 为某个模式重复的总数量;$c_k(d_n)$ 为重复模式中待模拟点 $S(\boldsymbol{u})$ 等于 s_k 的数量.

为了避免反复扫描训练图像,在 SNESIM 算法中采用了"搜索树"的数据结构来存储 cpdf 以加速重建过程. 对于任意一个待模拟节点 $\boldsymbol{u}$,其对应的 cpdf 可以直接从预先定义的搜索树中检索. 尽管该方法提高了整个模拟过程的速度,但当数据模板和训练图像较大时,仍然会消耗大量的 CPU 时间和内存空间[14].

2　结合软硬数据的 MPS 模拟

设 A 是一个二进制的变量,用于表示位于位置 $\boldsymbol{u}$ 的状态值 s_k 是否出现;B 表示以 $\boldsymbol{u}$ 为中心的数据模板 τ_n 内的硬数据,它包括原始的硬数据和已模拟节点的状态值;C 为软数据. $P(A)$为事件 A 产生的概率;$P(A|B)$表示在已知硬数据 B 的情况下事件 A 发生的条件概率;$P(A|C)$表示在已知软数据 C 的条件下事件 A 发生的概率;$P(A|B,C)$表示 B 和 C 同时已知的情况下事件 A 发生的概率.

定义一个事件出现的"先验距离"为[19]

$$a=\frac{1-P(A)}{P(A)}\in[0,+\infty).\tag{4}$$

类似地,可以定义在硬数据 B 已知情况下事件 A 出现的"先验距离"为

$$b=\frac{1-P(A|B)}{P(A|B)}\in[0,+\infty).\tag{5}$$

在软数据 C 已知情况下事件 A 出现的"先验距离"为

$$c=\frac{1-P(A|C)}{P(A|C)}\in[0,+\infty).\tag{6}$$

在 B 和 C 均已知情况下事件 A 出现的"先验距离"为

$$x=\frac{1-P(A|B,C)}{P(A|B,C)}\in[0,+\infty).\tag{7}$$

由式(7)可得:

$$P(A|B,C)=\frac{1}{1+x}\in[0,1].\tag{8}$$

假定在已知和未知 B 的情况下,C 对于 A 是否出现的相对贡献都是相同的,那么可以得到:

$$\frac{x}{b}=\frac{c}{a}.\tag{9}$$

由式(8)(9)可得:

$$P(A|B,C)=\frac{a}{a+bc}\in[0,1].\tag{10}$$

3　数据模板尺寸选择

一般数据模板的大小由用户主观选择. 一方面模板尺寸需要尽可能小,以降低 CPU 负荷和内存需求;另一方面模板尺寸需要足够大,以捕获训练图像中的特征信息. 因此,本文根据熵理论来选择合适的模板尺寸. 熵可以用来表示训练图像中模式的统计特性,可以定义为[20]

$$Entropy_j=-\sum_{i=1}^{K}p_i\lg p_i,\tag{11}$$

其中,K 表示第 j 种数据模板捕获的对应某随机变量的模式种类;p_i 表示出现某种模式的概率.一般来说,熵越高表示随机性越大.因此,随着模板尺寸的增大,训练图像中模式的熵也随之增加.但是,当模板尺寸达到一定程度时,随机性增长幅度变慢甚至停止,因为模板增大到已包含足够多的统计信息.熵随模板尺寸增加而增加的过程分成 2 个阶段:1)由于模板尺寸从小变大,导致随机性快速增大,因而熵也快速变大;2)当模板尺寸最优时,此时几乎包含了训练图像中的所有统计信息,则熵的增长速度变慢甚至停止增长.为了消除模板尺寸带来的影响,采用平均熵来表示统计信息变化,平均熵定义为[20]

$$Entropy_m = \frac{1}{m_T}\sum_{j=1}^{m_T} Entropy_j, \tag{12}$$

其中,m_T 是从数据模板中抽取的不同模式的数量.可以看出,平均熵 $Entropy_m$ 在开始阶段将随着模板尺寸的变化而迅速变化,当模板尺寸达到最优时则变化速度趋于平缓.

为了方便绘制平均熵变化曲线,设曲线横轴变化量为 num.选择最优三维数据模板尺寸的方法如下:

1) 所有数据模板的初始尺寸为 $\lfloor\sqrt{num^3}\rfloor$(例如 $num=3$,模板尺寸为 5);使用数据模板扫描训练图像,获得出现各种模式的概率 p_i;利用式(12)计算 $num=3$ 条件下对应的平均熵.

2) 分别在 $num=4,5,\cdots,0.4\times\max\{N_x,N_y,N_z\}$(此处 N_x,N_y,N_z 为给定训练图像分别在 x,y,z 三个方向上的长度)时重复 1),计算每个 num 对应的平均熵并最终生成一条平均熵曲线.

3) 根据 2)的平均熵曲线,获得关于 num 的熵变化斜率.一般来说,随着模板尺寸的增大,斜率逐渐变小并趋近于零.因此,可以计算每个模板尺寸对应的斜率,并认为斜率小于给定阈值所对应的第 1 个模板尺寸为最优模板尺寸.

4 SNESIM 算法的并行实现

GPU(graphic processing unit)与 CPU(central processing unit)是完全不同的计算架构.最初 GPU 是针对视频数据处理的需求而设计的,但现在其已成为实现并行计算的重要硬件.CUDA 可以更方便地生成基于 GPU 的并行计算程序.在文献[18]给出的基于 GPU 加速 SNESIM 的基础上,我们结合软硬数据进行空间数据重建,并采用熵理论实现了搜索模板尺寸的最优选择.整个并行重建过程采用了典型的主从架构:主过程将整个问题划分成多个子问题并将其分配到多个子过程中,整个模拟过程如图 4 所示.内核 1 和内核 2 的每步循环中,在不同

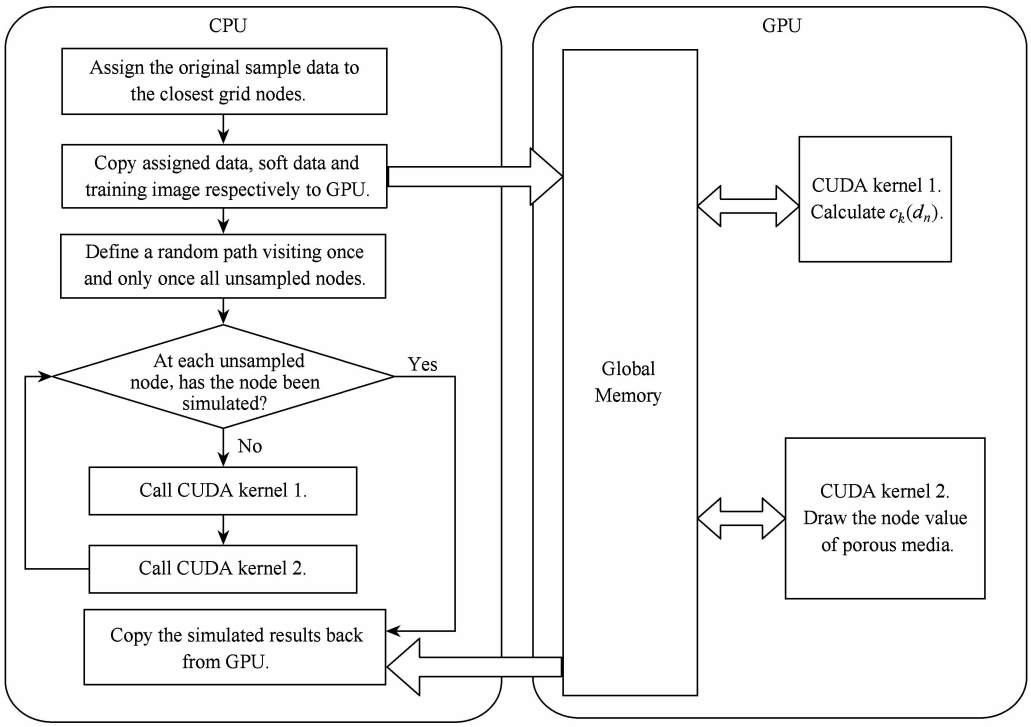


Fig. 4 Flowchart of parallel CUDA-based SNESIM.

图 4 基于 CUDA 的 SNESIM 的并行实现流程

的单道指令多道数据流中选择相同路径的条件分支放到同一步循环中,提高 GPU 内存的访问效率,优化分支分歧.

在重建过程中,1)将原始条件数据分配到最近的网格节点,初始的训练图像和模拟节点从 CPU 内存转入到 GPU 全局存储器. 2)定义随机路径对待模拟节点进行访问,如果访问到的节点还未模拟,则程序转入 GPU 的 CUDA 内核函数 1,利用 CUDA 内核函数 1 计算第 k 个变量的重复数 $c_k(d_n)$. 利用式(12)求得平均熵,通过比较平均熵曲线斜率获得最优模板尺寸;在最优模板情况下,利用式(10)进行软硬数据的整合. CUDA 内核函数 2 利用蒙特卡洛方法提取模拟值,详细方法参见文献[14]. 3)将模拟结果返回 CPU. 可见,整个重建过程中由 CPU 负责初始数据集的导入和重建结果的导出,重建的其他步骤在 GPU 中完成.

4.1　计算 $c_k(d_n)$

由于 SNESIM 基于逐点扫描进行空间数据重建,故令每个 CUDA 线程对应训练图像中的一个节点. 通过给定的搜索模板扫描当前重建区域,每个 GPU 线程判断当前模式是否在训练图像的模式库中. 如果在,则 $c_k(d_n)$ 总数量加 1. 因为此时搜索模板的最优尺寸尚未确定,需要对 num 从 3 到 $0.4\times\max\{N_x,N_y,N_z\}$ 循环,以计算每种模板尺寸对应的 $c_k(d_n)$.

首先将训练图像划分成若干块,块中的每个节点对应一个 CUDA 的线程. 在三维重建时,假设训练图像的尺寸为(TI_x,TI_y,TI_z),每个 CUDA 块尺寸为$(cuda_x,cuda_y,cuda_z)$,则 CUDA 块尺寸定义为

$$B_i = floor\left(\frac{TI_i}{cuda_i}\right)+1,\quad i=x,y,z, \tag{13}$$

其中,$floor$ 表示向下取整. CUDA 的内核 1 实现了计算训练图像$c_k(d_n)$的功能,具体的算法实现如下[18]:

算法 1. CUDA 内核 1:计算 $c_k(d_n)$.

① void *calculate_ckdn*().

/ * 主要参数:

training_image[*ti_x*][*ti_y*][*ti_z*]为训练图像在(*ti_x*,*ti_y*,*ti_z*)状态值;

sim_pos_x,*sim_pos_y*,*sim_pos_z* 为待模拟节点位置;

sim_grid[*simgrid_x*][*simgrid_y*][*simgrid_z*]为模拟网格在(*simgrid_x*,*simgrid_y*,*sim_pos_z*)处的状态值;

data_template_length 为数据模板长度;

state_code 为状态值的代号;

c_kd_n_*result*[*data_template_length*][*state_code*]模板中心状态值为 *state_code* 且模板长度为 *data_template_length* 时的重复数. * /

② *ti_pos_x* = *blockIdx.x* × *blockDim.x* + *threadIdx.x*;

③ *ti_pos_y* = *blockIdx.y* × *blockDim.y* + *threadIdx.y*;

④ *ti_pos_z* = *blockIdx.z* × *blockDim.z* + *threadIdx.z*;

⑤ *state_code*=*training_image*[*ti_pos_x*][*ti_pos_y*][*ti_pos_z*];

⑥ *data_template_length*=1;

⑦ while *data_template_length*≤ $floor(\sqrt{(0.4\times\max\{N_x,N_y,N_z\})^3})$ do

⑧　*ti_data_event_nodeval* = *get_ti_data_event_val*(*training_image*, *ti_pos_x*, *ti_pos_y*, *ti_pos_z*, *data_template_length*);

⑨　*sg_data_event_nodeval* = *get_sg_data_event_val*(*sim_grid*, *sim_pos_x*, *sim_pos_y*, *sim_pos_z*, *data_template_length*);

⑩　if (*ti_data_event_nodeval* = *sg_data_event_nodeval*) or (*sg_data_event_nodeval* 未知)

⑪　　*AtomicAdd* (*ckdn_result*[*data_template_length*][*state_code*],1);

⑫　　*data_template_length*++;

⑬　else

⑭　　*data_template_length*−−;

⑮　　break;

⑯　end if

⑰ end while

算法 1 中,根据 CUDA 线程的 ID,行②～④给出了训练图像中当前节点位置坐标;行⑤给出了当前节点对应的状态值;行⑥初始化数据模板的长度. 当数据模板的长度小于等于 $floor(\sqrt{(0.4\times\max\{N_x,N_y,N_z\})^3})$ 时,行⑧～⑨利用 *get_ti_data_event_val*()和 *get_sg_data_event_val*()分别捕获训练图像和模拟网格中的模式. 如果 2 个模式相同或模拟网格中的模式

节点值未知,则用于存储 $c_k(d_n)$数量的 $c_kd_n\_result$ 加 1,此过程通过 CUDA 函数 *AtomicAdd*()实现,然后数据模板长度也加 1;否则,减少数据模板长度并结束循环.

4.2 提取待模拟节点的状态值

由于 CUDA 的内核 1 中记录了 $c_k(d_n)$,一方面可以利用式(3)得到 cpdf,存储在搜索树中作为 $P(A|B)$;另一方面可以计算对应的模式出现概率 p_i,从而计算出不同搜索模板所对应的平均熵来选择最优模板尺寸.一旦模板尺寸确定,可结合软硬数据提取待模拟节点状态值.软数据 $P(A|C)$直接来源于输入参数,最后通过式(10)得到整合软硬数据的条件概率.这些过程由 CUDA 内核 2 实现.具体算法思路如下所示[18]:

算法 2. CUDA 内核 2:提取节点状态值.

① void *draw_node_value*().

/* 主要参数:

data_template_length 为数据模板长度;

optimal_template_size 为最优数据模板的尺寸;

state_num 为状态值的种类;

sim_pos_x,*sim_pos_y*,*sim_pos_z* 为待模拟节点位置;

sim_grid[*simgrid_x*][*simgrid_y*][*sim_pos_z*]为模拟网格在(*simgrid_x*, *simgrid_y*, *sim_pos_z*)处的状态值;

state_code 为状态值的代号;

$c_kd_n\_result$[*data_template_length*][*state_code*]是模板中心状态值为 *state_code* 且模板长度为 *data_template_length* 时的重复数;

cd_n_result[*data_template_length*]为模板长度 *data_template_length* 时的重复数;

soft_data[*sim_pos_x*][*sim_pos_y*][*sim_pos_z*]为软数据在(*simgrid_x*, *simgrid_y*, *sim_pos_z*)处的状态值. */

② cd_n[*data_template_length*]=0;

③ *data_template_length*=1;

④ *sim_val*=−1;

⑤ while *data_template_length* ≤ $floor(\sqrt{(0.4\times\max\{N_x,N_y,N_z\})^3})$ do

⑥ cd_n[*data_template_length*]=0;

⑦ for *i*=1 to *state_num* do

⑧ cd_n[*data_template_length*]=cd_n[*data_template_length*]+$c_kd_n\_result$[*data_template_length*][*i*];

⑨ end for

⑩ *data_template_length*++;

⑪ end while

⑫ *optimal_template_size*=*get_optimal_size*(cd_n,$c_kd_n\_result$,…);

⑬ *simval*=*draw_node_simval_with_harddata_softdata*(cd_n,$c_kd_n\_result$, *optimal_template_size*,*soft_data*,…);

⑭ *sim_grid*[*sim_pos_x*][*sim_pos_y*][*sim_pos_z*]=*simval*.

算法 2 中,在数据模板长度一定的情况下,根据对 $c_k(d_n)$求和可得 $c(d_n)$(行⑧).由于可以根据 $c(d_n)$和 $c_k(d_n)$计算得到 $P(A|B)$,同时可以得到出现某种模式的概率 p_i,因此可以利用平均熵理论编写函数 *get_optimal_size*(),获得最优搜索模板大小(行⑫).最后利用结合软硬数据的模型编写函数 *draw_node_simval_with_harddata_softdata*(),提取待模拟节点状态值(行⑬),并赋值到模拟网格的相应位置(行⑭).

5 实验和分析

5.1 软硬数据准备

本文利用重建多孔介质结构进行空间数据重建的验证.实验采用的多孔介质是砂岩,砂岩样本平均孔隙度为 0.17 左右.图 5(a)(b)(c)给出了砂岩样本数据的 3 张横截面图,其中白色部分表示孔隙,黑色部分表示固体骨架.

(a) Section 1 (b) Section 2 (c) Section 3

Fig. 5 Three sections of the sample.

图 5 样品横截面图

本次实验所用的 2 份真实体数据均从上述样本采集,体数据大小均为 80×80×80 体素,分别作为训练图像和目标图像(target image),孔隙度分别为 0.178 和 0.175.训练图像和目标图像在统计学上的结构差异可以用变差函数来衡量[21].如果 2 幅图像在同一个方向上具有相似的变差函数曲线,那么说明这 2 幅图像在此方向上具有相似的结构特征.训练图像和目标图像在 x,y,z 方向上的变差函数曲线分别如图 6(a)(b)(c)所示,说明两者在孔隙结构

上非常接近.

图 7 和图 8 分别是训练图像和目标图像,可以看出,训练图像与目标图像中的孔隙都具有长连通性且孔隙形状非常不规则. 从目标图像中随机取 0.5%的体素点作为初始条件数据,如图 9 所示,其中孔隙点所占比例为 0.176.

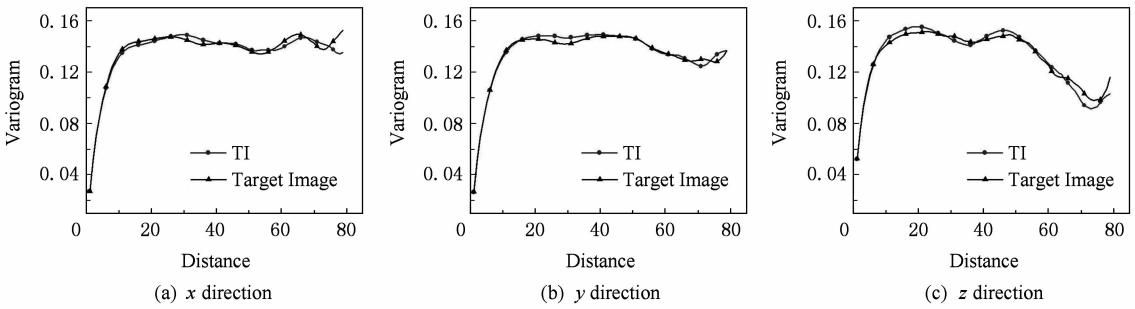


Fig. 6　Variograms of TI and the target image in x, y and z directions.

图 6　训练图像与目标图像在 x,y,z 方向上的变差函数

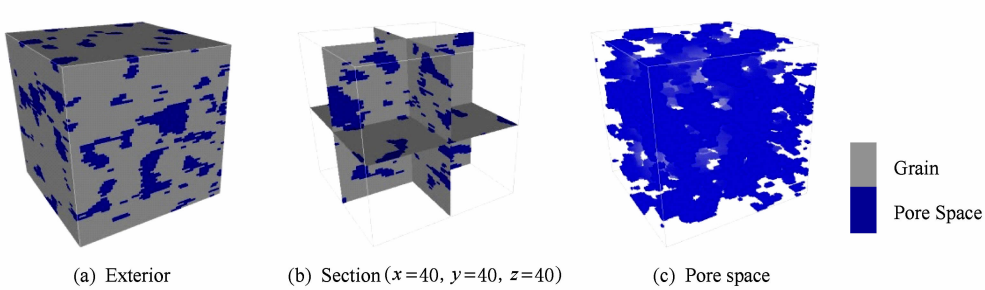


Fig. 7　Training image.

图 7　训练图像

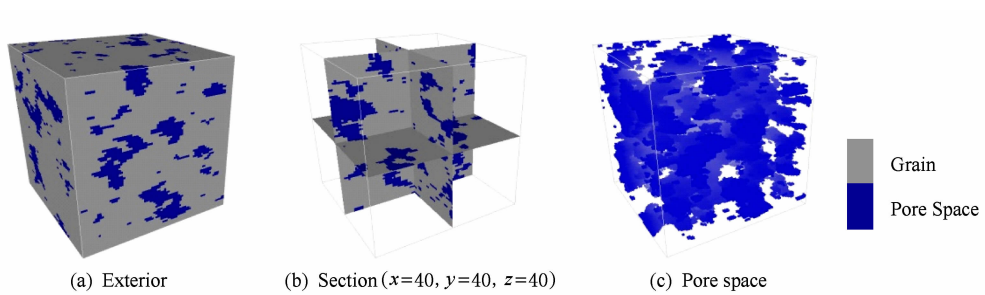


Fig. 8　Target image.

图 8　目标图像

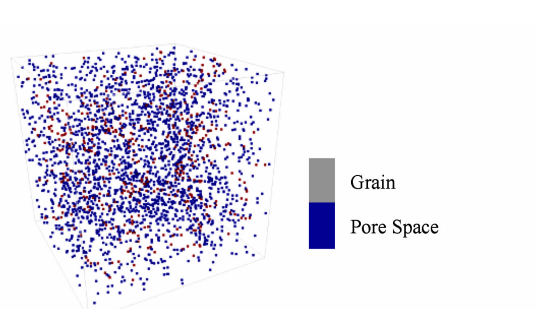


Fig. 9　Sample points.

图 9　采样点数据

结合软硬数据的算法在 CUDA 内核 2 中实现,其中硬数据来源于初始体数据的随机抽样(如图 9 所示),软数据则通过连续插值获得. 定义孔隙值为 1,骨架值为 0. 80×80×80 体素的“孔隙概率立方体”如图 10 所示,反映了砂岩中孔隙的概率分布,其各体素的状态值作为孔隙概率软数据. 同样可以得

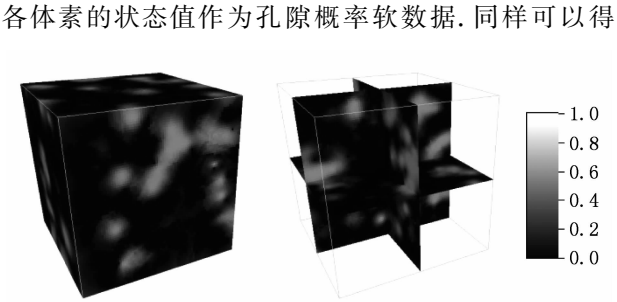


Fig. 10　Pore-space probability cube.

图 10　孔隙概率立方体

到一个 80×80×80 体素的“骨架概率立方体”,其各体素的状态值作为骨架概率软数据,如图 11 所示:

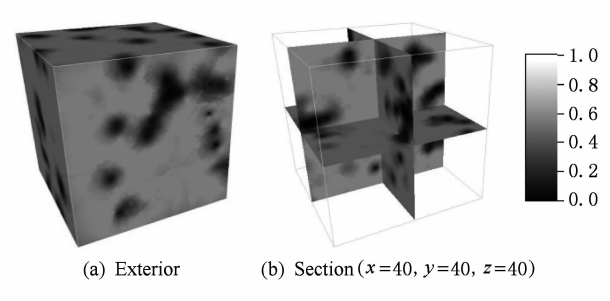


Fig. 11 Grain probability cube.

图 11 骨架概率立方体

5.2 基于 GPU 的多孔介质重建

本次实验所用的机器配置为: Intel i5 540 CPU、4 GB 主机内存;显卡采用 NVIDIA GeForce GTX 680,它的 CUDA 单元有 1 536 个,显存 2 048 MB, CUDA 版本为 4.2;运行环境为 Windows 7 和 VC 2010. 实验过程中,采用 CUDA 内核 2 计算每个 *num* 对应的平均熵,获得相应的平均熵曲线(如图 12 所示),可得到数据模板最优尺寸为 $\lfloor\sqrt{22^3}\rfloor=103$.

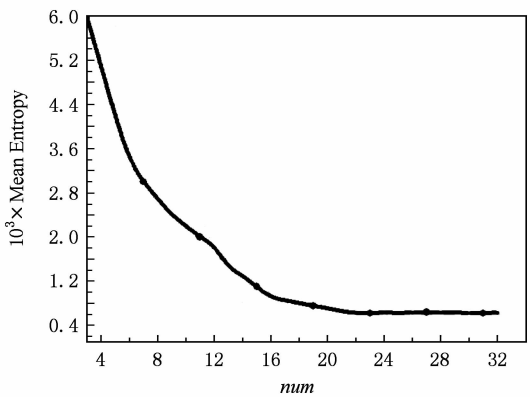


Fig. 12 Curve of mean entropy.

图 12 平均熵曲线

基于软硬数据和 GPU 实现砂岩重建的结果如图 13 所示. 可见重建图像具有与目标图像相似的孔隙和骨架结构,且孔隙具有较好的长连通性.

为了与使用软硬数据情况下的重建结果进行比较,又基于 GPU 分别在只使用硬数据和无条件数据 2 种情况下重建了多孔介质. 图 14 和图 15 分别给出了 2 种情况下的重建结果图(80×80×80 体素). 从图 14~15 可以看出,孔隙空间的重建效果良好.

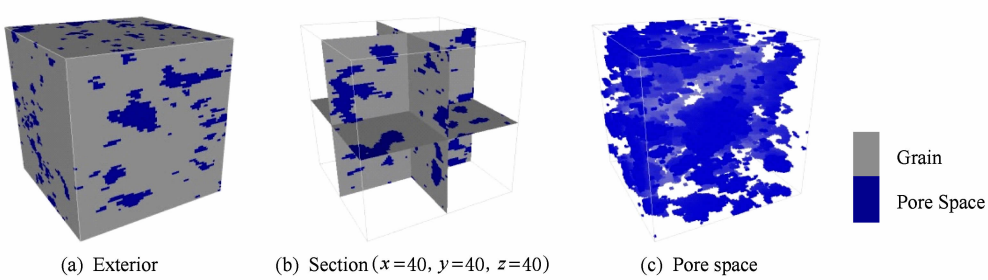


Fig. 13 GPU-based reconstructed sandstone images using soft data and hard data.

图 13 基于软硬数据和 GPU 的砂岩重建

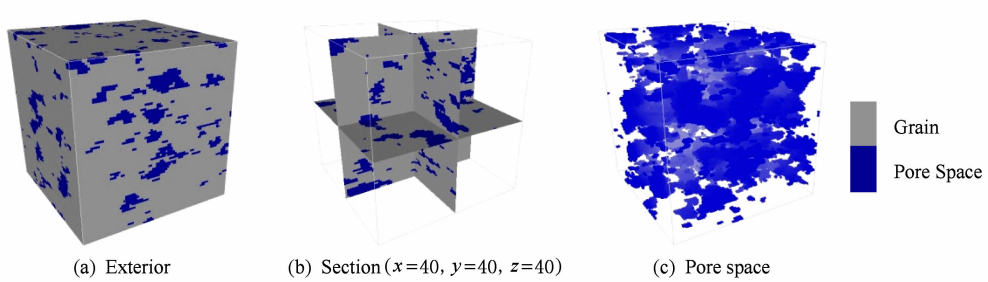


Fig. 14 GPU-based reconstructed sandstone images using hard data only.

图 14 基于硬数据和 GPU 的砂岩重建

采用基于 GPU 的 SNESIM 方法分别在无条件数据、只有硬数据和结合软硬数据这 3 种情况下各实现 10 次砂岩的三维图像重建,所有重建图像的大小均为 80×80×80 体素. 由每种情况下的 10 次重建结果可以得到在每个体素位置的方差值,各种情况下的方差值组成了一个 80×80×80 体素的“方差立方体”. 这些方差值只有 6 种取值:0, 0.09, 0.16, 0.21, 0.24, 0.25. 上述各种方差值对应的体素数目

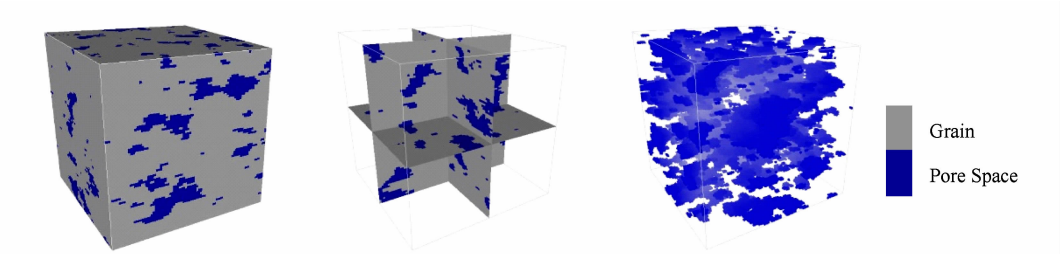


Fig. 15　GPU-based unconditional reconstructed sandstone images.

图 15　无条件数据的 GPU 砂岩重建

占“方差立方体”体素总数的百分比如图 16 所示:

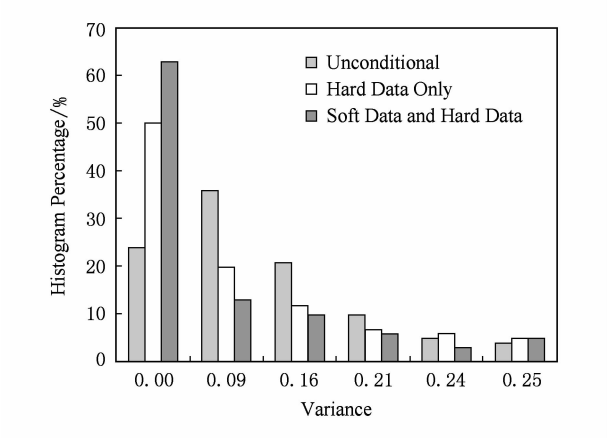


Fig. 16　Histogram percentage of different variances in three conditions.

图 16　3 种情况下方差值百分比的直方图

从图 16 可以看出,结合软硬数据时的方差值明显小于其他 2 种情况. 表 1 给出了这 3 种情况下的平均孔隙度,可以看出,结合软硬数据情况下的孔隙度与目标图像最为接近.

Table 1　Average Porosity in Three Conditions

表 1　3 种情况下的平均孔隙度

Item	Average Porosity
With Soft Data and Hard Data	0.174
With Hard Data Only	0.165
Unconditional	0.134

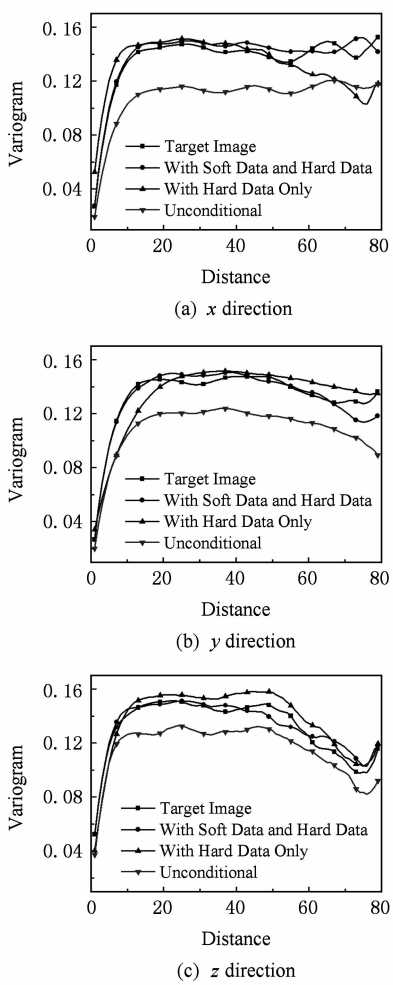


Fig. 17　Variograms of the target image and the reconstructed images in three conditions.

图 17　目标图像与 3 种情况下重建图像的变差函数

图 17 给出了目标图像与 3 种情况下重建图像的变差函数. 从图 17 可以看出,目标图像与软硬数据图像的变差函数最为接近,证明结合软硬数据有助于提高空间数据重建质量.

作为对比,在重建实验中分别选取了不同尺寸的数据模板与最优尺寸模板(103 个节点)进行重建实验. 图 18 给出了砂岩重建的结果,可以看出,与尺寸为 103 个节点的数据模板相比,52 个节点的数据模板重建结果包含更多的孤立孔隙空间,很难形成长连通区域,31 个节点的数据模板生成的孔隙结构最差;尺寸较大的数据模板(如 132 个节点和 181 个节点)可以生成与目标图像相似的结构并保留较好的孔隙长连通性.

对基于 CPU 的 SNESIM 算法也进行了重建实验. 实验中模板采用一系列固定尺寸(节点数量为

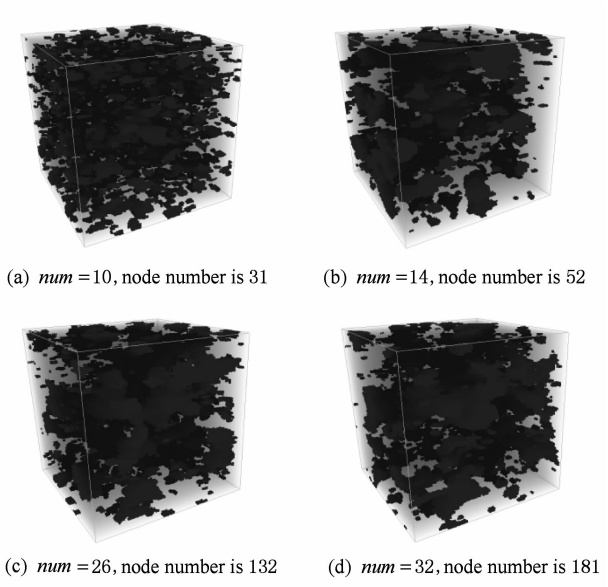


Fig. 18 GPU-based reconstructed pore spaces using different templates.

图 18 采用不同数据模板得到的 GPU 重建孔隙空间

20,40,60,…,200)代替最优尺寸,并在重建过程中结合软硬数据. 由于 CPU 和内存的限制,一重网格模板无法采用比较大的尺寸,但是可以利用网格逐渐密集化的多个数据模板来替代一个大而密集的模板对训练图像进行扫描. 因此,根据文献[22],在 CPU 实验中采用多重数据模板. 设网格重数 G,本文中最大采用三重网格,即 $G=3$. 图 19 给出了 CPU 条件下采用不同数据模板获得的重建孔隙空间的结果. 从图 19 可以看出,网格级数和搜索模板

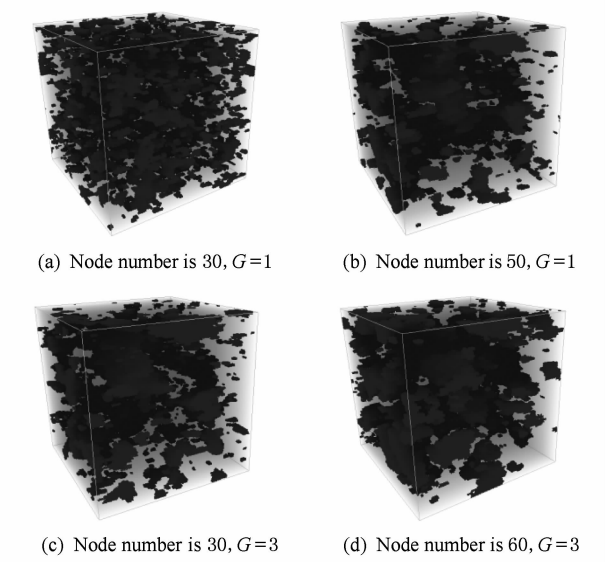


Fig. 19 3D reconstructed results on CPU using different data templates.

图19 CPU 条件下采用不同数据模板获得的三维重建结果

节点数目越大,孔隙空间的重建效果越好.

在基于 CPU 的 SNESIM 算法中,由于建立搜索树需要大量的内存需求,当使用一重模板($G=1$,节点数量大于 140)或者三重模板($G=3$,节点数量大于 100)时机器崩溃无法运行,因为此时的内存消耗超出了硬件负荷上限. 图 20 给出了基于 GPU 和 CPU 的 SNESIM 算法中不同尺寸数据模板对应的模拟时间和内存需求. 由于训练图像和模拟网格的大小固定,因此基于 GPU 的 SNESIM 的主机内存负荷是基本固定的. 在 CPU 情况下,当使用三重数据模板和一重数据模板时,主机内存负荷迅速增加. CPU(使用一重数据模板)和 GPU 情况下的模拟时间比 $Ratio_t$ 与内存比 $Ratio_m$ 定义如下[18]:

$$Ratio_t = T_c/T_g, \tag{14}$$

$$Ratio_m = M_c/M_g. \tag{15}$$

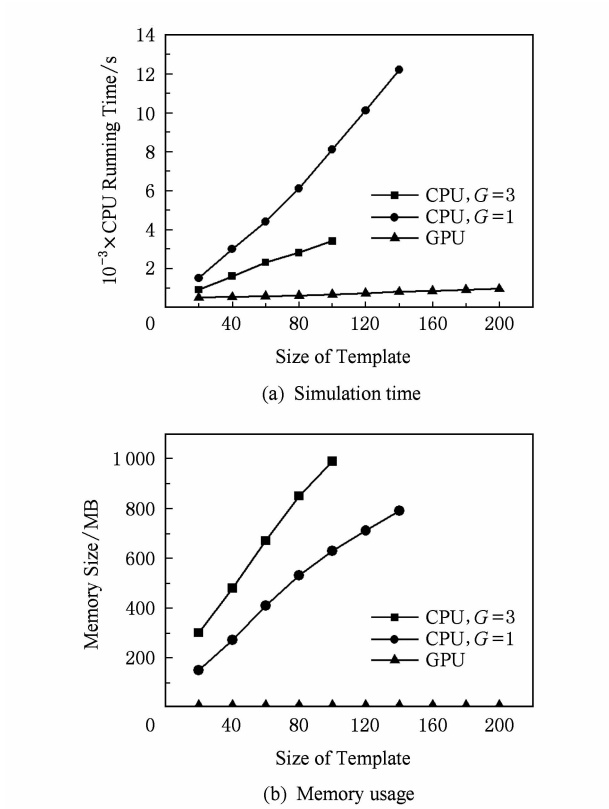


Fig. 20 Simulation time and memory usage on CPU and GPU.

图 20 基于 GPU 与 CPU 的模拟所需时间和内存

其中,T_c 和 T_g 分别是 CPU 版本和 GPU 版本的模拟时间;M_c 和 M_g 分别是 CPU 版本和 GPU 版本的最大内存负荷. 图 21 给出了 $Ratio_t$ 和 $Ratio_m$ 的变化曲线,均由数值模拟实验获得. 从图 21 可见,当数据节点达到 140 时,基于 CPU 的程序是基于 GPU 的程序花费时间的 15.25 倍,内存使用情况为 197.5

倍.由此可以看出,在数据模板尺寸较大时,GPU 版本的 SNESIM 算法可以大幅度地提高模拟效率.

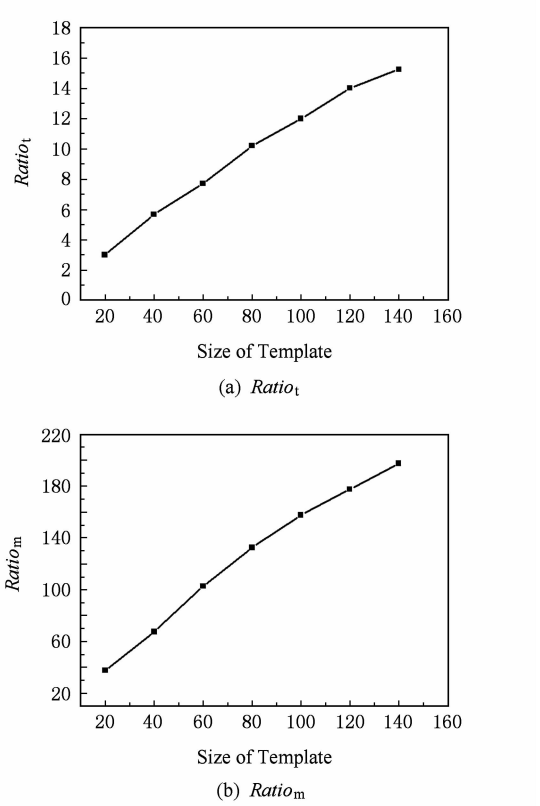


Fig. 21　Ratios of simulation time and memory on CPU and GPU.

图 21　基于 GPU 和 CPU 的模拟时间比和内存比

5.3　利用 LBM 计算渗透率

实验中利用格子玻尔兹曼方法(lattice Boltzmann method, LBM)分别计算不同条件下重建图像的渗透率[23].如表 2 所示,结合软硬数据的重建图像与目标图像的渗透率最为相似.

Table 2　Permeability of the Target Image and the Images in Three Conditions

表 2　3 种情况下图像与目标图像的渗透率

Item	Permeability /$10^{-3}\mu m^2$
Target Image	1.754
With Soft Data and Hard Data	1.691
With Hard Data Only	1.349
Unconditional	0.775

6　结　　论

基于 SNESIM 算法的空间数据重建已经应用于许多领域,但庞大的 CPU 负荷和内存需求是其应用瓶颈.为了解决这一问题,我们对 SNESIM 算法并行化以提高重建速度,并在算法实现中结合了软硬数据和选择最优模板尺寸的方法.通过结合软硬条件数据可以获得更好的模拟效果.在计算训练图像熵的基础上,形成平均熵曲线,可以获取最优模板尺寸.

利用砂岩进行空间数据重建实验.通过比较重建图像和目标图像的变差函数曲线、渗透率等结果表明本文提出的并行 SNESIM 算法能够保证重建质量且极大地减少 CPU 和内存的负荷.与当前基于 CPU 的 SNESIM 重建算法相比,基于 GPU 的重建算法的内存需求与数据模板尺寸关联度较小且模拟速度更快,从而更加适用于大范围的空间数据重建.

参　考　文　献

[1] Ren Jicheng. Research and implementation of parallel volume rendering techniques and visualization system for 3D data fields [D]. Beijing: Institute of Computing Technology, Chinese Academy of Sciences, 1999 (in Chinese)
(任继成. 三维数据场并行体绘制技术及可视化系统的研究与实现[D]. 北京:中国科学院计算技术研究所, 1999)

[2] Sheng Min. Research on the methods of nonlinear interpolation for digital image processing [D]. Hefei: Hefei University of Technology, 2009 (in Chinese)
(盛敏. 数字图像处理中非线性插值方法的应用研究[D]. 合肥:合肥工业大学, 2009)

[3] Xu Yanlei, Zhao Jiyin, Li Min, et al. The interpolation arithmetic study of medical image based on the order morphology [J]. Acta Electronica Sinica, 2010, 38(5): 1002-1007 (in Chinese)
(徐艳蕾, 赵继印, 李敏, 等. 基于顺序形态学的医学图像插值算法的研究[J]. 电子学报, 2010, 38(5): 1002-1007)

[4] Lu Zhibo. Research on enhancement and interpolation algorithms in medical image processing [D]. Zhengzhou: PLA Information Engineering University, 2007 (in Chinese)
(鲁志波. 医学图像增强与插值的算法研究[D]. 郑州:解放军信息工程大学, 2007)

[5] Chen K, Lorenz D A. Image sequence interpolation based on optical flow, segmentation, and optimal control [J]. IEEE Trans on Image Processing, 2012, 21(3): 1020-1030

[6] Pan Rijing. Quadratic B-spline interpolation curves with tangent constraints on data points [J]. Chinese Journal of Computers, 2007, 30(12): 2132-2141 (in Chinese)
(潘日晶. 满足数据点切向约束的二次 B 样条插值曲线[J]. 计算机学报, 2007, 30(12): 2132-2141)

[7] Paiement A, Mirmehdi M, Xie X, et al. Integrated segmentation and interpolation of sparse data [J]. IEEE Trans on Image Processing, 2014, 23(1): 110-125

[8] Du Yi, Zhang Ting, Lu Detang, et al. An interpolation method using an improved Markov model [J]. Journal of Computer Research and Development, 2012, 49(3): 565-571 (in Chinese)
(杜奕, 张挺, 卢德唐, 等. 一种基于改进的 Markov 模型的插值方法[J]. 计算机研究与发展, 2012, 49(3): 565-571)

[9] Xu Lin. Infinite interpolation on triangles [J]. Journal of Software, 2007, 18(2): 430-441 (in Chinese)
(徐琳. 三角形域上的超限插值方法[J]. 软件学报, 2007, 18(2): 430-441)

[10] Wang G Z, Fang L C. On control polygons of quartic pythagorean hodograph curves [J]. Computer Aided Geometric Design, 2009, 26(9): 1006-1015

[11] Wang Yanlong, Liu Jinhua, Zhang Ting. An interpolation method using the Markov model and COSGSIM [J]. Journal of Computer-Aided Design & Computer Graphics, 2010, 22(10), 1715-1720 (in Chinese)
(汪彦龙, 刘金华, 张挺. 结合 Markov 模型和 COSGSIM 的插值方法[J]. 计算机辅助设计与图形学学报, 2010, 22(10), 1715-1720)

[12] Guardiano F, Srivastava M R. Multivariate geostatistics: Beyond bivariate moments [EB/OL]. (1993-03-21) [2014-03-02]. https://pangea.stanford.edu/researchgroups/scrf/sites/default/files

[13] Zhang T F. Filter-based training pattern classification for spatial pattern simulation [D]. Palo Alto, CA: Stanford University, 2006

[14] Strebelle S. Sequential simulation drawing structures from training images [D]. Palo Alto, CA: Stanford University, 2001

[15] Nunes R, Almeida J A. Parallelization of sequential Gaussian, indicator and direct simulation algorithms [J]. Computers and Geosciences, 2010, 36(8): 1042-1052

[16] Peredo O, Ortiz J M. Parallel implementation of simulated annealing to reproduce multiple-point statistics [J]. Computers & Geosciences, 2011, 37(8): 1110-1121

[17] Huang T, Li X, Zhang T, et al. GPU-accelerated direct sampling method for multiple-point statistical simulation [J]. Computers & Geosciences, 2013, 57: 13-23

[18] Huang T, Lu D, Li X, et al. GPU-based SNESIM implementation for multiple-point statistical simulation [J]. Computers & Geosciences, 2013, 54: 75-87

[19] Journel A G. Combining knowledge from diverse data sources: An alternative to traditional data independence hypothesis [J]. Mathematical Geology, 2002, 34(5): 573-596

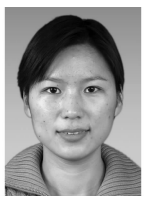
[20] Honarkhah M. Stochastic simulation of patterns using distance-based pattern modeling [D]. Palo Alto, CA: Stanford University, 2011

[21] Zhang Ting, Lu Detang, Li Daolun, et al. Research on statistical information reconstruction of images based on multiple-point geostatistics integrating soft data with hard data [J]. Journal of Computer Research and Development, 2010, 47(1): 43-52 (in Chinese)
(张挺, 卢德唐, 李道伦, 等. 基于软硬数据的多点信息统计法在图像统计信息重构中的应用研究[J]. 计算机研究与发展, 2010, 47(1): 43-52)

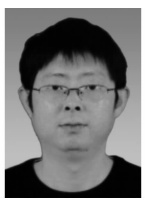
[22] Tran T T. Improving variogram reproduction on dense simulation grids [J]. Computers and Geosciences, 1994, 20(7/8): 1161-1168

[23] Zhang T, Li D, Lu D, et al. Research on the reconstruction method of porous media using multiple-point geostatistics [J]. Science in China: Physics, Mechanics & Astronomy, 2010, 53(1): 122-134

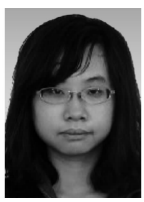
Zhang Ting, born in 1979. PhD and associate professor in Shanghai University of Electric Power. His main research interests include image reconstruction.



Du Yi, born in 1977. PhD and associate professor in Shanghai Second Polytechnic University. Member of China Computer Federation. Her current research interests include data mining.



Huang Tao, born in 1979. PhD and post-doctoral researcher in the University of Science and Technology of China. His current research interests include high performance computing.



Li Xue, born in 1988. PhD candidate in the University of Science and Technology of China. Her current research interests include high performance computing.