



Parallel programming of gradient-based iterative image reconstruction schemes for optical tomography

Andreas H. Hielscher^{a,b,*}, Sebastian Bartel^{b,1}

^a *Departments of Biomedical Engineering and Radiology, Columbia University, 500 West 120th Street, MC 8904, New York, NY 10027, USA*

^b *Department of Pathology, Downstate Medical Center, State University of New York, 450 Clarkson Avenue, Brooklyn, NY 11203, USA*

Received 1 March 2002; accepted 1 February 2003

KEYWORDS

Optical tomography;
Near-infrared (NIR);
Heterogeneous cluster;
Parallel computing

Summary Optical tomography (OT) is a fast developing novel imaging modality that uses near-infrared (NIR) light to obtain cross-sectional views of optical properties inside the human body. A major challenge remains the time-consuming, computational-intensive image reconstruction problem that converts NIR transmission measurements into cross-sectional images. To increase the speed of iterative image reconstruction schemes that are commonly applied for OT, we have developed and implemented several parallel algorithms on a cluster of workstations. Static process distribution as well as dynamic load balancing schemes suitable for heterogeneous clusters and varying machine performances are introduced and tested. The resulting algorithms are shown to accelerate the reconstruction process to various degrees, substantially reducing the computation times for clinically relevant problems.

© 2003 Elsevier Ireland Ltd. All rights reserved.

1. Introduction

Optical tomography (OT) is a fast developing new medical imaging modality that uses near-infrared (NIR) light ($650 < \lambda < 900$ nm) to probe various parts of the body. In general, laser diodes deliver light through optical fibers to several locations around the body part under investigation and transmitted light intensities are recorded. The technology for making sensitive light-transmission measurement

through the human body is now-a-days readily available [1–10]. Using this instrumentation, several pilot studies have proven the applicability of OT in medicine and its potential in functional tissue diagnostics is widely recognized. The most promising applications are monitoring of blood oxygenation [11–13], hemorrhage detection [14,15], functional imaging of brain activities [16–23], Alzheimer diagnosis [24,25], early diagnosis of rheumatic disease in joints [26–28], and breast cancer detection [29–33]. All these applications employ the fact that various disease processes and other physiological changes affect the optical properties (mainly absorption coefficient μ_a and scattering coefficient μ_s) in biological tissue. Therefore, different optical properties of different

* Corresponding author. Tel.: +1-212-854-5080;
fax: +1-212-854-8725.

E-mail address: ahh2004@columbia.edu (A.H. Hielscher).

¹ Current address: Laser Zentrum Hannover e.V.,
Hollerithallee 8, D-30419, Hannover, Germany.

Model: this model is a theory or algorithm that predicts a set of measured signals, \mathbf{M}_p , based on the position of the light source and the spatial distribution of optical properties $\zeta = [\mu_a(\mathbf{r}), \mu_s(\mathbf{r})]$. In this study we use a finite-difference scheme (FD) that is based on the time-dependent diffusion equation [37]. (2) **Analysis Scheme:** here an objective function, Φ , is defined, which describes the difference between the measured, \mathbf{M} , and predicted data, \mathbf{P} . In this work we define the objective function as least-square error norm, which is given by

$$\Phi(\zeta) = \chi^2(\zeta) \equiv \sum_s \sum_d \frac{(M_{s,d} - P_{s,d}(\zeta))^2}{2\eta_{s,d}^2} \quad (1)$$

The indices s and d identify different sources and detector positions. The parameter $\eta_{s,d}$ is a normalization constant, which for example can be set to $M_{s,d}(\zeta)$. In this case the error norm minimizes the sum of the squared percentage difference between measured and theoretical data. (3) **Updating Scheme:** once the objective function is defined, the task becomes to minimize Φ , starting from an initial guess $\zeta_0(\mathbf{r})$. This is accomplished in two substeps: (3a) First, the gradient of the objective function $d\Phi(\zeta_0)/d\zeta$ is calculated by means of the so-called adjoint-differentiation method. (3b) Second, given the gradient an iterative line minimization in the direction of the gradient is performed. This step is labeled inner iteration in Fig. 1 and consists of several forward calculations in which the optical parameters ζ are varied. Once the minimum along the line is found, a new gradient is calculated at this minimum (outer iteration) and another line-minimization is performed, now along a different direction in the ζ -space. These steps are repeated until a distribution ζ is found for which $\Phi(\zeta)$ is smallest. A more detailed description of the gradient MOBIIR scheme used in this work can be found in [37].

2.2. Parallelization

The appropriate decomposition of the original problem into smaller subtasks is most crucial to every parallel implementation of a given algorithm. We identified the finite differences (FD) algorithm that is used to calculate the predicted detector readings \mathbf{P} as the most time-consuming portion of our GIIR scheme. Here, the objective function $\Phi(\zeta)$ is determined for the current guess of optical properties ζ of the medium. The gradient calculation $\partial\Phi(\zeta)/\partial\zeta_i$ is also performed in this part of the code by adjoint differentiation, which consumes about the same time as one forward calculation. Approximately 95% of the computa-

tional time are spent in the finite-differences and adjoint-differentiation schemes. The exact fraction depends on the number of sources employed in the experimental setup, since both, the objective function $\Phi(\zeta)$ and the gradient $\partial\Phi(\zeta)/\partial\zeta_i$ are given as sums over all source-detector combinations. The experimental data consists of several sets of simultaneous detector readings corresponding to different source positions. One could activate all sources simultaneously, however, it proves to be advantageous to switch through all source positions sequentially to increase the amount of information gained. To match these experimental readings with the simulation and evaluate the objective function, each source present in the experiment requires a separate forward calculation. Hence, the problem of calculating Eq. (1) for N sources quite naturally breaks up into N mutually independent single source forward-calculations (SSFCs). We can rewrite the objective function (1) as

$$\Phi(\zeta) = \sum_s \sum_d \frac{(M_{s,d}(\zeta) - P_{s,d}(\zeta))^2}{2\eta_{s,d}^2} = \sum_s \Phi_s(\zeta) \quad (2)$$

Thus, instead of handling N sources sequentially as in the experiment, we may, therefore, share the total load amongst up to N processors, by assigning each a disjunct set of sources to evaluate. The objective function is then obtained by summing over the partial results (Fig. 1).

One SSFC typically requires on the order of 0.01–1 s depending on the size of the FD grid. This makes them appropriate candidates for sub-tasks to be distributed over the network. Shorter tasks would increasingly sense the latency times of the network, because TCP/IP protocol yields latency times of several milliseconds.

Another consideration is the network bandwidth, which in our case is ideally 100 Mb/s but may decrease to several 10 Kb/s in times of high traffic. If we assume a minimum transfer rate of 50 Kb/s, the amount of data transferred per SSFC should, therefore, not exceed 500 Bits to avoid unnecessary waiting time. This does, however, not pose a problem: All processors possess their private copy of all data structures involved in the reconstruction and are principally able to perform any SSFC autonomously. When the parallel code forks into different branches no data, other than the distribution scheme has to be communicated across the network.

Upon completion of the sub-tasks, when a global sum over all processors is formed, two data structures are to be transferred over the network to be available to all participating processors. First, the addends Φ_s of the objective function (Eq. (2)), i.e. single floating point values, are transferred.

Secondly, the gradient matrix, represents the largest chunk of data to be moved across the network—it consists of $n \times m$ values, where n and m fix the size of the FD grid. Typically, the grid size ranges between 20×20 and 50×50 pixels, so that a maximum of 80 Kb have to be moved across the cluster. However, the gradient calculation only occurs once per outer iteration, in about 1/15 of the invocations of the forward model.

Although the SSFC is an appropriate partitioning unit for parallelization, this choice leads to certain limitations. First of all, no more processors than sources may effectively be used. Moreover, certain configurations will lead to unfavorable load balancing. For instance, consider four SSFCs to be shared by three identical processors. Obviously, the 4th forward calculation will leave two processors idle. These limitations are unavoidable in the current splitting scheme and have to be considered when testing or using the algorithm.

3. Implementation

Parallel execution of a program on multiple processors requires a physical network and a communication interface between the nodes. All processors have to be aware of one another and be able to exchange data securely. The Message Passing Interface (MPI) is a platform independent standard, which was introduced by the Message Passing Interface Forum, in 1994 [58,59]. This interface defines a set of functions for inter-processor communication and data manipulation. MPI hides details of the underlying hardware and network protocol from the developer and provides a transparent interface between all participating nodes (it operates on top of protocols such as TCP/IP, e.g. above layer 4 in the OSI 7 Layer Network Model). Several implementations of the MPI for different architectures are freely available. We used the MPICH-package suitable for LINUX, IRIX, and SOLARIS, which includes C-libraries and several scripts for compiling and executing binaries in a parallel environment.

MPI is designed to support the distributed memory model, so all processors execute identical copies of the binary in their own memory space and have no direct access to data on other nodes. It is actually more appropriate to speak of different processes rather than of processors, since MPICH does not distinguish between physically different processors and multiple instances of a program on the same processor. If several processes are launched on one node, communication still uses the machine's TCP/IP interface but is routed directly through the internal loop-back. In the remainder of this paper

we will use the term processor and process synonymously for instances of a program, independent of their location.

When executing the parallel version of a binary, MPI assigns a unique identification number to each process through which it can be addressed by others. Communication can take place between two processors or groups of processors, within so-called communicators.

To take advantage of a parallel execution the code must of course be adapted to behave differently on each processor and handle disjunct portions of the total computational expense. The MPI provides functions to retrieve a processor's identification at run-time, so that the code can take different branches depending on which node it is executing. In the remainder of this section we will refer to respective processors IDs by n_{my_pe} and denote the total number of processors with n_{pe} .

Following the arguments given above, the total number of sources n_s is to be divided into n_{pe} subsets. Both, the objective function $\Phi(\xi)$ and the gradient $\partial\Phi(\xi)/\partial\xi_i$ are sums over all source contributions as indicated by Eq. (2). Within the actual implementation, these contributions are added up in a loop:

```
for (i = 0; i < ns; i++)
{
  Φ += Φi;
  grad Φ += grad Φi;
}
```

(3)

Therefore, our task is to break down this loop and to assign only part of the total interval $i \in [0 \dots n_s[= [0 \dots n_1[, [n_1 \dots n_2[, \dots, [n_{s-1} \dots n_s[$ to each processor. Let $n_{my_pe} \in [0 \dots n_{pe}[$ be the index of the executing processor, then the following code fragment causes all processors to evaluate the commands $\{\dots\}$ for disjunct sub-intervals only:

```
if (nmype == 0)
for (i = 0; i < n1; i++)
{...}
else if (nmype == 1)
for (i = n1; i < n2; i++)
{...}
:
else if (nmype == npe)
for (i = ns&0x00AD; i < ns; i++)
{...}
```

(4)

This code example merely illustrates the concept of assigning disjunct intervals to different processors. The actual implementation may, however, look somewhat different from the example above. In the following subsections we will introduce three

approaches for choosing the n_{pe} subsets appropriately and depict their implementation.

3.1. Static load distribution

In our first approach, the total number of sources $[0 \dots n_s]$ is broken into n_{pe} equal parts (if possible). Hence, if n_{pe} is not an integer denominator of n_s , the size of respective intervals will differ by 1. The corresponding implementation can be cast in a very compact form if we rewrite the loop (3) such that

```
for (i = n_mype; i < ns; i += npe)
{
  Φ + = ΦSource(JobId);
  grad Φ + = grad ΦSource(JobId);
}
```

(5)

where $\Phi_{\text{Source}}(\text{JobId})$ and $\text{grad } \Phi_{\text{Source}}(\text{JobId})$ represent function calls to a single source forward/gradient calculation (SSFC). Hence, processor 0 evaluates the sources 0, n_{pe} , $2n_{pe}$, ..., processor 1 evaluates sources 1, $n_{pe}+1$, $2n_{pe}+1$, ... and so forth. Fig. 2 depicts the splitting mechanism in a graphical form. Obviously, the respective subsets are always disjunct and completely fill the interval $[0 \dots n_s]$.

The major advantage of this approach, is that it produces virtually no overhead and requires no additional variables to define the boundaries of the respective intervals. In a homogeneous cluster of processors with n_{pe} being a denominator of n_s the loop (Eq. (5)) will execute exactly n_{pe} times as fast as on a single processor.

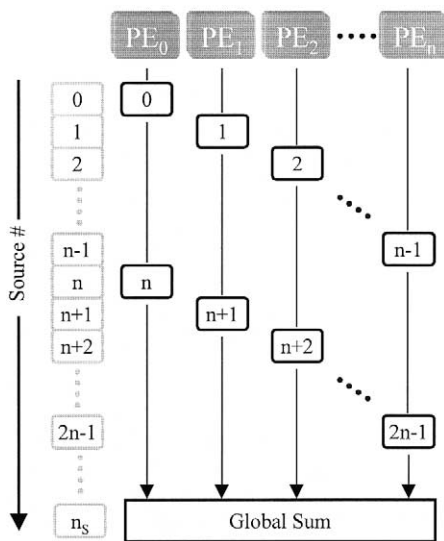


Fig. 2 Static distribution scheme. The total number of sources n_s is shared evenly across n PE's.

If the sources cannot be split up evenly across the cluster, some processors will be idle for the time necessary for a single forward calculation, leading to a decay in performance. The worst-case scenario occurs if the reconstruction is based on a single source only. In this case, all but processor 0 will be idle and no increase in performance will be observed. However, this is not a disadvantage of any particular distribution scheme but inherent to our approach to parallelization. Since one source calculation constitutes the smallest possible sub-task that can be assigned to a processor it is obviously useless to employ more than n_s processors in the reconstruction. While finer levels of subdivision can be envisioned, we will defer the discussion of corresponding schemes to Section 4.

The major disadvantage of the static load distribution becomes apparent in inhomogeneous clusters. Since the total number of sources is spread evenly across all participating processors, independent of their performance, slower processors will require more time to complete their share, forcing the rest of the cluster to wait. Consequently, the overall speed in a cluster of processors of performance P_i is determined by the slowest processor in the ensemble. The maximum performance P_Σ of the cluster is given by

$$P_\Sigma = n_{pe} \cdot \min\{P_i\} \quad (6)$$

3.2. Dynamic job assignment

To use the available processor performance in heterogeneous clusters more effectively we developed a flexible scheme to assign subtasks to individual processors. The scheme employs one dedicated processor, referred to as PE_0 , to manage the distribution of SSFC to the cluster. This PE_0 keeps track of the total number of sources and what fraction of them has been processed by any of the other processors. It does, however, not take part in the actual forward calculation.

All other processors, upon entering the forward model, post a job request to the dedicated processor and wait for an available source to evaluate. PE_0 responds to a request by sending out the next source index and marking this index as processed. Whenever a processor has finished a forward calculation, it sends another request to PE_0 to obtain the next, unprocessed source.

If no more sources are pending, the dedicated processor broadcasts a termination signal to the cluster and stops listening to requests. The notification to terminate the reconstruction loop passes through the same messaging channel as regular job assignments. But instead of a valid source index, a

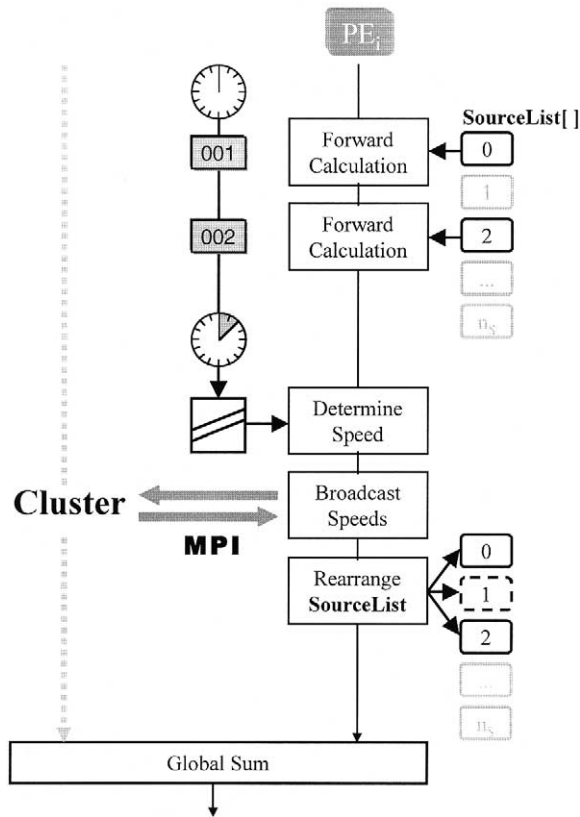


Fig. 3 Job-request/assignment. Processor PE_0 manages the distribution of SSFCs and assigns source IDs upon request by other PE 's.

predefined negative constant is broadcast to all processors and interpreted there accordingly. Fig. 3 illustrates the algorithm in detail. The essential code fragment has the following form:

```

if (nmype != 0) // on all but  $PE_0$ , request source ID
{
  // and perform SSFC
  RequestJob (& JobId);
  while (JobId != NO_JOB) // unless termination signal is sent...
  {
     $\Phi$  +=  $\Phi_{source}(\text{JobId})$ ; // do SSFC &
    grad  $\Phi$  += grad  $\Phi_{source}(\text{JobId})$ ; // gradient calculation
    RequestJob (& JobId); // request next source ID
  }
}
else // on  $PE_0$  assign source ID upon request
  // on  $PE_0$  listen to requests and
  { // assign jobs to processors...
    SrcCount = 0;
    while (SrcCount <= ns) // while we have unprocessed SSFCs {
      WaitRequest(& PeNo); // wait for requests from any PE
      AssignJob(PeNo, SrcCount) // assign next SSFC to calling PE
    }
    BeastTermination() // ... broadcast termination of all  $PE$ 's
  }

```

(7)

The concept of this job-request scheme is readily understood: Initially, each processor picks up a source index and performs the corresponding forward- or gradient calculation. Faster processors will complete their task in a shorter period of time than their slower counterparts and consequently be able to post another job request earlier. On average, if we consider many sources ($n_s \gg n_{pe}$), each processor will request a fraction $n(i)/n_s$ of all sources, that is proportional to its speed. Here, $n(i)$ is the number of sources handled by processor i .

Obviously this second scheme is much less susceptible to dawdling candidates than the static load distribution. Extremely slow processors will receive only a single job and have no more opportunity to request a second one. Furthermore, the request/assignment scheme adapts to changing processor performances during the image reconstruction automatically. Each participant will obtain as many sources to evaluate as it can handle at the time.

This advantage only becomes apparent, however, if more sources than processors are available, since even the slowest processor will handle at least one SSFC. The following considerations yield an estimate of the minimal number of SSFC, n_s , necessary to observe an improvement during parallel execution.

Consider a single processor PE_j with only a fraction $1/q$ of the performance p of all other members of the cluster. Let n_j be the number of sources handled by the slower candidate, compared with n sources handled by each of the other processors. Then the following relations hold:

$$n_j \simeq \frac{1}{q}n; \quad n_j \geq 1 \quad (8)$$

$$n_S = n \cdot (n_{pe} - 1) + n_j \quad (9)$$

From this we get:

$$n_j = \frac{n_S}{q(n_{pe} - 1) + 1} \stackrel{!}{\geq} 1 \quad (10)$$

Solving for n_S , the minimum number of SSFCs is given by

$$n_S \geq q(n_{pe} - 1) + 1 \quad (11)$$

Thus, approximately $n_{pe}q$ sources must be present to prevent processor j from slowing down the reconstruction unduly.

We note further, that some other pathological cases may be constructed, e.g. if all but one source position have been evaluated and the slowest processor manages to snatch the final source shortly before a faster colleague posts a request. In this unfortunate case, the cluster is forced to wait disproportionately long, while a faster processor could have completed the task earlier.

Again, as for the first scheme, the difficulties mentioned do not pose much of a problem, if we have $n_S \gg n_{pe}$. They originate from the relatively coarse partitioning of the reconstruction process.

However, two drawbacks are specific to this particular job distribution scheme. First, one processor, PE_0 is lost for the actual reconstruction process, since it is employed to manage the job assignment. The second and most important drawback, as will be shown later, is the communicational overhead involved with the request/assignment mechanism. A total of $2n_S + n_{pe}$ messages have to be passed across the network during each forward calculation. Although a single message is only some 10 bytes in size, the high latency time of TCP/IP networks causes significant idle times at each job request.

3.3. Dynamic load distribution

The third algorithm developed for the parallel image reconstruction combines the flexibility of the dynamic scheme introduced above with the modest communicational overhead of the static, first scheme. The key point of this concept is to determine individual processor speeds in terms of reconstructions per unit time at run-time. This information is then used to redistribute the total load across the cluster.

To keep track of the current partitioning of the complete set of sources $[1 \dots n_S]$ a static source list is maintained on all processors. It is defined as

$$\text{static int SourceList[NumberOfSources];} \quad (12)$$

where **SourceList**[i] contains the ID of the processor handling source position i . Initially, before any forward calculations are done, all processors are considered to be equal and the SourceList is populated evenly, similar to Eq. (3).

In the following, each processor scans the SourceList for elements with its own ID and performs the corresponding single-source forward calculations. The time t_i , required for these calculations is recorded, along with the number of sources n_i assigned to this processor, so that we can determine the effective speed, defined as:

$$v_i = \frac{n_i}{t_i} \quad (13)$$

Once all processors have finished their share of sources, they broadcast their current speed to all members of the cluster. The final step is to update the SourceList in view of the different v_i of individual processors, so that faster candidates will handle a bigger fraction of all sources in future. If we denote the number of sources to be processed on PE_i with n_i , and the time necessary with t_i , we are faced with the constrained minimization problem in T :

$$\begin{aligned} T(n_0, n_1, \dots, n_{n_{pe}-1}) &= \max\{t_0, t_1, \dots, t_{n_{pe}-1}\} \\ &= \max \left\{ \frac{n_0}{v_0}, \frac{n_1}{v_1}, \dots, \frac{n_{n_{pe}-1}}{v_{n_{pe}-1}} \right\}; \quad \text{with } n_S = \sum_{i=0}^{n_{pe}-1} n_i \end{aligned} \quad (14)$$

We employ the following algorithm to populate **SourceList**.

First, the processor speeds are normalized so that $\sum v_i = n_S$. If we now set

$$n_i = v_i \quad \forall i \in [0 \dots n_{pe}]; \quad n_j \in \mathbb{R} \quad (15)$$

both, the minimization and the constraint are satisfied, since

$$\begin{aligned} \text{(i)} \quad \frac{n_0}{v_0} &= \frac{n_1}{v_1} = \dots = \frac{n_{n_{pe}-1}}{v_{n_{pe}-1}} \\ \text{(ii)} \quad \sum_{i=0}^{n_{pe}-1} n_i &= \sum_{i=0}^{n_{pe}-1} v_i = n_S \end{aligned} \quad (16)$$

Note, that the n_i thus obtained are real, whereas we can only assign an integer number of sources to individual processors. To overcome this difficulty, we use a fast and simple method to generate integer values close to the optimal solution: The n_i are iteratively increased by a common factor q , such that $n_i \rightarrow (1+q)n_i$, until their round-down values add up to the total number of sources. Typically, the increment q is set to 0.05, which usually yields correct n_i within only five iterations.

```

PunchTime(); //mark current time
MyCount=0; //reset source counter
for (i=0; i < ns; i++) //scan SourceList for Jobs to do
{ //on this processor
if (SourceList[i] == nmype)
{
 $\Phi$  +=  $\Phi_{\text{source}}(i)$ ; //do SSFC &
grad  $\Phi$  += grad  $\Phi_{\text{source}}(i)$ ; //gradient calculation
MyCount++;
}
}
MyTime=PunchTime(); //record time required and
MySpeed=(double)MyCount/MyTime; //determine processor speed
UpdateSourceList(MySpeed, SourceList); //redistribute the sources
//according to processors speed

```

(17)

Finally, the n_i are used to populate and update **SourceList**. Note, that the calculations above are performed on all processors simultaneously, so there is no need for further communication. Once the information about processor speeds is shared across the cluster, the minimum solution of Eq. (14) is fully determined and requires no more interaction. Fig. 4 and the code below (Eq. (17)) summarize the essentials of the dynamic load distribution scheme.

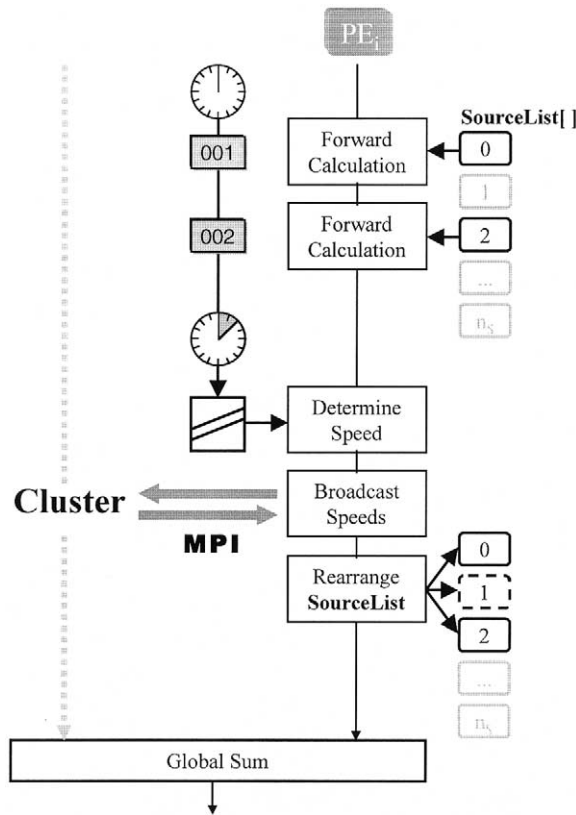


Fig. 4 Dynamic scheme: all processors determine their individual speed after completion of their share of forward calculations. After each iteration, the speeds are communicated to the cluster to re-evaluate the partitioning of all sources.

Two minor modifications, not shown in the coding example, were included to improve the stability of this algorithm: the current speed is biased by the average speed of all past forward calculations to dampen oscillations that tend to occur on multi-processor machines (we found that on these machines, single tasks are not tied to a processor but shifted around by the operating system. This resulted in a competition between different processes and oscillations in their performance. So-called processors (PE's) in the MPI-schemes are not necessarily identical to physical processors). Secondly, if a processor was not assigned at least on source, the last recorded speed is adopted rather than zero speed. This prevents individual processors from dropping out of the competition completely. Hence, the speed is determined as:

```

if (MyCount)
MySpeed = (double)MyCount /
MyTime + 0.5 * MyAverageSpeed;
else
MySpeed = MyLastSpeed;

```

(18)

This last scheme proves to be the most efficient distribution algorithm. It adapts to large discrepancies between different processors and may even exclude slow performers completely. However, it requires no more than one joint data exchange between all processors, after each forward calculation. The additional calculations do not consume significant computational time.

4. Results and performance

To test the performance of our distribution schemes quantitatively, we used an example from our recent research concerning optical tomographic joint imaging (Fig. 5). Based on cross-sectional images obtained from magnetic resonance imaging (MRI) of a human proximal-interphalangeal (PIP) finger

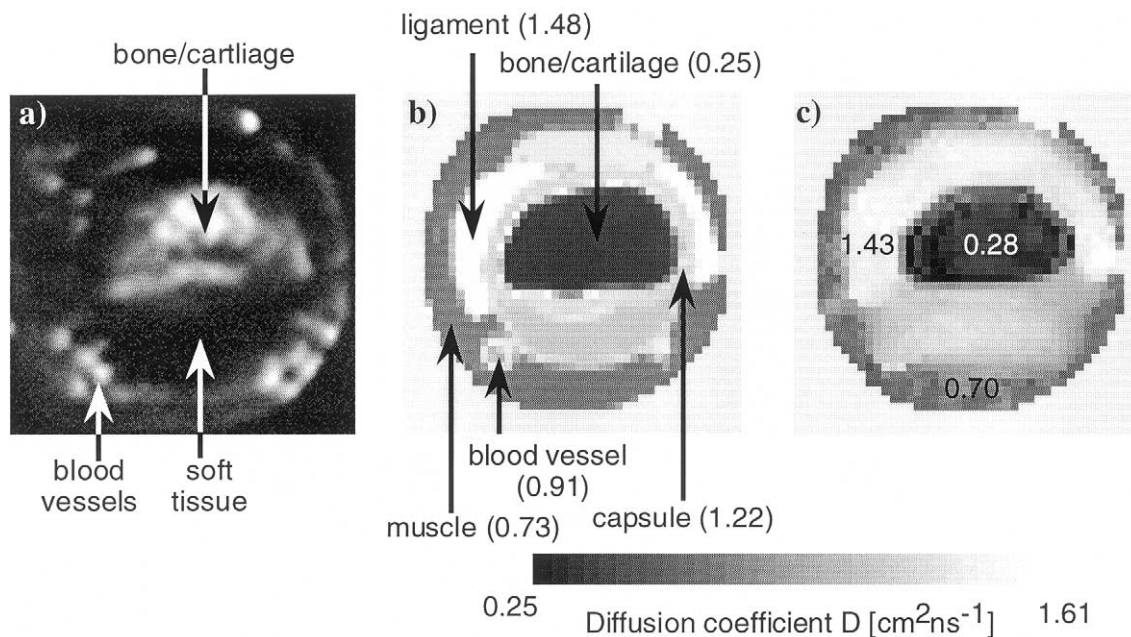


Fig. 5 Example used for testing the different parallel computing schemes. (a) Magnetic resonance image of a human PIP finger joint; (b) segmented image of the joint that shows the different tissue types (bone, muscle, tendon and ligament) and their different optical properties. This map was used to generate synthetic input data for the model-based iterative image reconstruction MOBIIR code; (c) result of the reconstruction of diffusion coefficients after 200 iterations.

joint we designed a numerical model that was used to generate synthetic data. We assigned optical properties to different tissues visible in the MRI cross-section images of the joint (see Fig. 5a and b). Since not all tissue types are clearly distinguishable by MRI data alone, standard anatomical information [60] was additionally used to uniquely identify the various positions of different tissues. Segmenting the MRI images in this way we obtained a two-dimensional slice of the optical properties in a finger joint. These images had a size of 44×44 pixels, with each pixel having dimensions of 0.4×0.4 mm. In order to simulate measurements of the finger joint, we placed 60 detectors and 60 sources around the joint and performed forward calculations that yielded detector readings for different source positions. This synthetic data was input to our gradient MOBIIR algorithm, which was started with a spatially homogeneous initial guess of the optical properties. The result of the reconstruction can be seen in Fig. 5c. For more details concerning this example see [27].

The results of a parallel implementation of the image reconstruction code do, in no way, differ from the serial execution. This is an obvious necessity of any parallel scheme and was tested thoroughly during the developing process. Therefore, the quality of our gradient-based image reconstruction is not subject to discussion at this point, and the reader

with interest in quality of optical tomographic imaging is referred to the previously cited publication [37–56] that address this problem.

In this study we are interested in the performance of the various parallel algorithm in terms of computational speed. In a first test, we determined the gain in reconstruction speed with increasing number of identical processors. Ideally, one would expect the total speed $v_t(n_{pe})$ to grow linearly with the number of processors, according to

$$v_t(n) = \gamma \cdot n_{pe} \cdot v_{pe} \quad (19)$$

where, v_{pe} is the single processor speed and $\gamma = 0 \dots 1$ the degree of parallelization. The latter quantifies the fraction of computational expense that is actually shared across the cluster. It becomes 1, if 100% of the code have been parallelized and no more sequential execution is done. In our case, given the large number of sources involved, we found that more than 99% of the time is spent within the forward model. Under the assumption that an optimal splitting of the total work is theoretically possible, i.e.

$$\frac{n_s}{n_{pe}} = n \in N \quad (20)$$

we can estimate $\gamma \approx 1$. Any deviation from Eq. (19) is, therefore, caused by overhead or idle time introduced by the parallelization scheme.

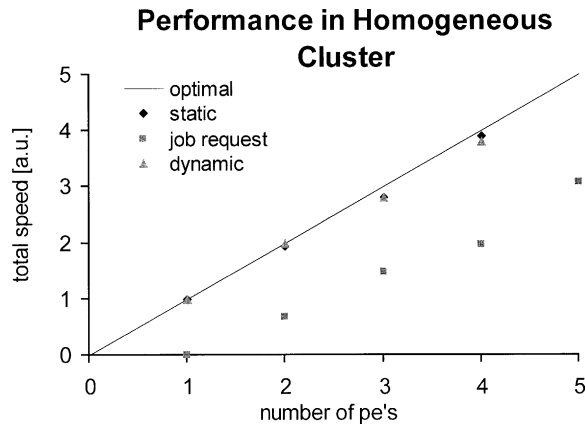


Fig. 6 Performance increase with increasing number of identical processors. The solid line indicates the performance of an idealized parallel scheme. Note that the job-request scheme requires at least two pe's since the first processor manages the distribution of forward calculations to the cluster.

Fig. 6 shows the dependency of $v_t(n_{pe})$ on the number of processors for all three schemes as well as the idealized case of $\gamma=1$. As expected, the static assignment of sources performs best, since it involves virtually no overhead and requires only minimal inter-processor communication. Almost equally good results are observed for the dynamic distribution scheme. The job-request scheme, however, performs poorly and yields only 50% of the maximum possible speed. Moreover, a gain due to parallelization only sets in for more than two processors, since one processor is necessary to handle the administrative tasks.

The second test was designed to study the algorithms' ability to perform in heterogeneous clusters, i.e. to adapt to different processor speeds. Ideally, in generalization of Eq. (16), the maximum speed of a parallel scheme is given by

$$v_t(v_0, v_1, \dots, v_{n_{pe}-1}) = \gamma \cdot \sum_{i=0}^{n_{pe}-1} v_i \quad (21)$$

In other words, the number of forward calculations per unit time performed on individual processors simply adds up to the total number of forward calculations of the cluster. Hence, slow processors should only affect the overall reconstruction speed insofar as they contribute to a lesser degree to the total work.

To quantify the performance of all three approaches, we employed 5 processors of identical speed but slowed down one processor (pe 1) artificially, by inserting waiting intervals of length

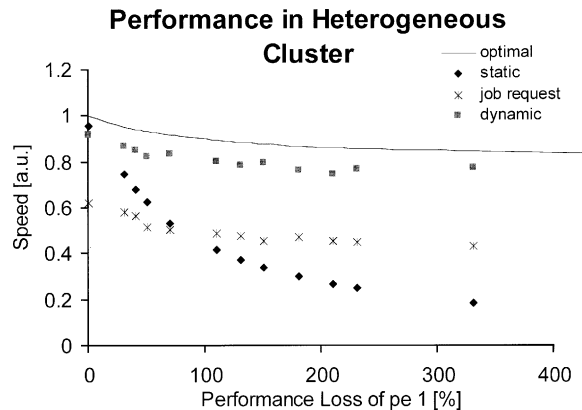


Fig. 7 Effect of a single slower processor (PE₀) in a cluster of five PE's. The ideally achievable speed of an optimal parallel scheme is given by the solid graph.

T_{wait} into the SSFC. In Fig. 7, the abscissa indicates the length of these intervals, relative to the actual computing time T_{SSFC} . Hence, a decay of 100% corresponds to $T_{wait} = T_{SSFC}$, rendering the processor half as fast as the normal. In the limit of $T_{wait} \rightarrow \infty$, only four processors participate in the reconstruction and the optimal speed approaches 0.8.

The results clearly state the superiority of the dynamic scheme over the other approaches. Over the whole range it yields more than 90% of the maximum possible optimal performance. Obviously, the algorithm is able to find an optimal distribution scheme according to each processor's capabilities. The speed of the static scheme, on the other hand quickly drops as pe 1 slows down. In fact, the overall speed is governed by the slowest processor and depends on its decay d like

$$v_i = v_{pe\ 1} \sim \frac{1}{1 + d/100} \quad (22)$$

It is interesting to note that the job-request/assignment algorithm behaves quite stable, as it does not decay to more than 40% relative to the optimal case (graph with solid line). We conclude that in spite of the overhead, the scheme adapts to the heterogeneous environment properly. In lower latency networks it might, therefore, provide a reasonable alternative to dynamic schemes.

5. Discussion

Given the results, the dynamic load distribution appears most suitable for parallel computation of problems involving optical tomographic image

reconstruction in a heterogeneous network. The dynamic scheme adapts to varying processors speeds and requires only little communicational overhead during the reconstruction process. This outcome is not surprising since the dynamic algorithm combines the positive aspects of the static-assignment and job-request approaches. Under realistic conditions, in a busy network the dynamic scheme is able to utilize more than 90% of each processor's time for the reconstruction process. Under the premise that an optimal distribution of all sources across the cluster is theoretically possible, the overall parallelization γ is, therefore, greater than 0.9. Single slower participants will contribute to a lesser degree but never affect the overall speed negatively.

There are no exceptional requirements on the execution environment to achieve this performance. The algorithm is, therefore, suitable for most of nowadays network architectures. In this study we executed the parallel reconstruction in a cluster of three Pentium-based dual-processor machines (450 and 750 MHz), running LINUX and three SGI-workstations across at least two passive hubs. Active switches would most likely raise the overall speed by several percent but are no necessity as our results show.

The algorithm yielded reconstructed images on a 44×44 grid with up to 12 sources—positions in approximately 30 s. This benchmark makes in situ image reconstruction in a medical environment feasible and will expand the range of clinical applications for NIR diagnostics.

As already noted in previous sections, the number of source-positions currently presents a limit for the parallelization. In typical experiments as pursued by our group, the setup includes 12–32 sources, which may be increased up to the total number of available processors, without loss in performance. In spite of this inherent limitation, the possibility to employ more sources in the reconstruction process poses a significant advantage. By raising the number of sources and/or detectors in a OT setup, the amount of information gained from the medium can be significantly increased. Due to the high costs of sensitive and highly dynamic detectors it is principally easier to use more sources rather than detectors [8]. However, on the reconstruction side, this choice amounts to additional computations of the light distribution originating from these sources. Previously, the time necessary to generate images from the experimental data rose accordingly. With parallel schemes as presented in this work the overall reconstruction time may be kept constant by utilizing additional processors in the reconstruction process.

6. Summary

OT is a novel emerging medical imaging modality that has shown great promise in a variety of clinical pilot studies. This new modality is based on measurements of transmitted intensities of non-ionizing, NIR light. Unlike X-rays, which traverses human tissue on a straight line, NIR light is strongly scattered in tissue and the image reconstruction problem involves computationally intensive, iterative, model-based algorithms. Long computation times have so far limited the practical clinical application of powerful model-based iterative image reconstruction (MOBIIR) scheme. In this paper we addressed this problem by analyzing, implementing, and testing various parallel execution schemes, which can be executed on an arbitrary number of UNIX-type machines in a local area network.

We found that in commonly employed MOBIIR schemes over 90% of the computational time is spent in forward solvers, which make prediction of the light intensities on the surface of the medium, given a guess of the optical properties inside the medium. Since typically many light sources are used in OT, and for each light source a forward problem has to be solved, the algorithm can be divided into n sub-tasks, where n is the number of light sources used in the experimental setup. These sub-tasks are then executed in parallel on distributed processors to accelerate the image reconstruction process, in the optimal case, by a factor close to n .

Three different parallel schemes (static load distribution, dynamic job assignment, dynamic load distribution) were tested in a heterogeneous cluster of networked PCs and workstations. The static load sharing makes almost optimal use of the available resources if used in a homogeneous cluster and the number of sources being a multiple of the number of processors. In heterogeneous networks, the dynamic scheme performs best, since it takes different processor speeds into account and avoids unnecessary waiting for slower participants to complete their share. This parallel algorithm is also insensitive to varying network loads. Moreover, it is suitable for arbitrary network architectures, as it based on the protocol-independent MPI standard. However, the associated communication overhead can be justified only in reconstruction problems that are sufficiently large so that the time necessary to complete a task is much larger than the latency for communicating the result. Current and future demands on image reconstruction certainly meet this condition and make the dynamic allocation scheme the preferable alternative.

Acknowledgements

This work was supported in part by the National Institute of Arthritis and Musculoskeletal and Skin Diseases, a division of the National Institute of Health (grant # R01 AR46255-01), and the Whitaker Foundation (grant # 98-0244).

References

- [1] C.H. Schmitz, M. Löcker, J.M. Lasker, A.H. Hielscher, R.L. Barbour, Instrumentation for fast functional optical tomography, *Review of Scientific Instruments* 73 (2) (2002) 429–439.
- [2] F.E.W. Schmidt, M.E. Fry, E.M.C. Hillman, J.C. Hebden, D.T. Delpy, A 32-channel time-resolved instrument for medical optical tomography, *Review of Scientific Instruments* 71 (2000) 256–265.
- [3] C.H. Schmitz, H.L. Graber, H. Luo, I. Arif, J. Hira, Y. Pei, A. Bluestone, S. Zhong, R. Andronica, I. Soller, N. Ramirez, S.–L.S. Barbour, R.L. Barbour, Instrumentation and calibration protocol for imaging dynamic features in dense-scattering media by optical tomography, *Applied Optics* 39 (2000) 6466–6485.
- [4] S.B. Colak, M.B. van der Mark, G.W. 't Hooft, J.H. Hoogenraad, E.S. van der Linden, F.A. Kuijpers, Clinical optical tomography and NIR spectroscopy for breast cancer detection, *IEEE Journal of Selected Topics in Quantum Electronics* 5 (1999) 1143–1158.
- [5] V. Ntziachristos, X.H. Ma, A.G. Yodh, B. Chance, Multichannel photon counting instrument for spatially resolved near infrared spectroscopy, *Review of Scientific Instruments* 70 (1) (1999) 193–201.
- [6] A.M. Siegel, J.J.A. Marota, D.A. Boas, Design and evaluation of a continuous-wave diffuse optical tomography system, *Optics Express* 4 (1999) 287–298.
- [7] B.W. Pogue, M. Testorf, T. McBride, U. Österberg, K. Paulsen, Instrumentation and design of a frequency-domain diffuse optical tomography imager for breast cancer detection, *Optics Express* 1 (1997) 391–403.
- [8] J.C. Hebden, S.R. Arridge, D.T. Delpy, Optical imaging in medicine I: experimental techniques, *Physics in Medicine and Biology* 42 (1997) 825–840.
- [9] H. Jess, H. Erdl, K.T. Moesta, S. Fantini, M.A. Franceschini, E. Gratton, M. Kaschke, Intensity modulated breast imaging: technology and clinical pilot study results, in: R.R. Alfano, J.G. Fujimoto (Eds.), *Advances in Optical Imaging and Photon Migration*, OSA Trends in Optics and Photonics, Vol. II, Optical Society of America, Washington, DC, 1996, pp. 126–129.
- [10] J.P. Vanhouten, D.A. Benaron, S. Spilman, D.K. Stevenson, Imaging brain injury using time-resolved near-infrared light scanning, *Pediatric Research* 39 (1996) 470–476.
- [11] P.J. Kirkpatrick, Use of near-infrared spectroscopy in the adult, *Philosophical Transactions of the Royal Society of London-Series B: Biological Sciences* 352 (1997) 701–705.
- [12] C. Hock, K. Villringer, F. Müller-Spahn, R. Wenzel, H. Heekeren, S. Schuh-Hofer, M. Hofmann, S. Minoshima, M. Schwaiger, U. Dirnagl, A. Villringer, Decrease in parietal cerebral hemoglobin oxygenation during performance of a verbal fluency task inpatients with Alzheimer's disease monitored by means of near-infrared spectroscopy (NIRS)—correlation with simultaneous CBF-PET measurements, *Brain Research* 755 (1997) 293–303.
- [13] T. Noriyuki, H. Ohdan, S. Yoshioka, Y. Miyata, T. Asahara, K. Dohi, Near-infrared spectroscopic method for assessing the tissue oxygenation state of living lung, *American Journal of Respiratory and Critical Care Medicine* 156 (1997) 1656–1661.
- [14] S.R. Hintz, W.F. Cheong, J.P. van Houten, D.K. Stevenson, D.A. Benaron, Bedside imaging of intracranial hemorrhage in the neonate using light: comparison with ultrasound, computed tomography, and magnetic resonance imaging, *Pediatric Research* 45 (1999) 60–65.
- [15] S.P. Gopinath, C.S. Robertson, C.F. Contant, R.K. Narayan, R.G. Grossman, B. Chance, Early detection of delayed traumatic intracranial hematomas using near-infrared spectroscopy, *Journal of Neurosurgery* 83 (1995) 438–444.
- [16] D.A. Benaron, S.R. Hintz, A. Villringer, D. Boas, A. Kleinschmidt, J. Frahm, C. Hirth, H. Obrig, J.C. Van Houten, E.L. Kermit, W. Cheong, D.K. Stevenson, Noninvasive functional imaging of human brain using light, *Journal of Cerebral Blood Flow and Metabolism* 20 (2000) 469–477.
- [17] M.A. Franceschini, E. Gratton, S. Fantini, V. Toronov, M. Filiaci, On-line optical imaging of the human brain with 160-ms temporal resolution, *Optics Express* 6 (2000) 49–57.
- [18] L.C. Henson, C. Calalang, J.A. Temp, D.S. Ward, Accuracy of a cerebral oximeter in healthy volunteers under conditions of isocapnic hypoxia, *Anesthesiology* 88 (1998) 58–65.
- [19] M. Tamura, Y. Hoshi, F. Okada, Localized near-infrared spectroscopy and functional optical imaging of brain activity, *Philosophical Transactions of the Royal Society of London-Series B: Biological Sciences* 352 (1354) (1997) 737–742.
- [20] G. Gratton, M. Fabiani, P.M. Corballis, D.C. Hood, M.R. Goodman-Wood, J. Hirsch, K. Kim, D. Friedman, E. Gratton, Fast and localized event-related optical signals (EROS) in the human occipital cortex: comparisons with the visual evoked potential and fMRI, *Neuroimage* 6 (1997) 168–180.
- [21] A. Maki, Y. Yamashita, E. Watanabe, H. Koizumi, Visualizing human motor activity by using non-invasive optical topography, *Frontiers of Medical and Biological Engineering* 7 (1996) 285–297.
- [22] C. Hirth, H. Obrig, J. Valdueza, U. Dirnagl, A. Villringer, Simultaneous assessment of cerebral oxygenation and hemodynamics during a motor task. A combined near infrared and transcranial Doppler sonography study, *Advances in Experimental Medicine and Biology* 411 (1997) 461–469.
- [23] R. Wenzel, H. Obrig, J. Ruben, K. Villringer, A. Thiel, J. Bernardinger, U. Dirnagl, A. Villringer, Cerebral blood oxygenation changes induced by visual stimulation in humans, *Journal of Biomedical Optics* 4 (1996) 399–404.
- [24] C. Hock, K. Villringer, F. Müller-Spahn, M. Hofmann, S. Schuh-Hofer, H. Heekeren, R. Wenzel, U. Dirnagl, A. Villringer, Near infrared spectroscopy in the diagnosis of Alzheimer's disease, *Annals of the New York Academy of Sciences* 777 (1996) 22–29.
- [25] A.J. Fallgatter, M. Roesler, L. Sitzmann, A. Heidrich, T.J. Mueller, W.K. Strik, Loss of functional hemispheric asymmetry in Alzheimer's dementia assessed with near-infrared spectroscopy, *Brain Research, Cognitive Brain Research* 6 (1997) 67–72.
- [26] U. Netz, J. Beuthan, H.J. Capius, H.C. Koch, A.D. Klose, A.H. Hielscher, Imaging of rheumatoid arthritis in finger joints by sagittal optical tomography, *Medical Laser Application* 16 (2001) 306–310.
- [27] A. Klose, A.H. Hielscher, K.M. Hanson, J. Beuthan, Three-dimensional optical tomography of a finger joint model for

- diagnostic of rheumatoid arthritis, in: D.A. Benaron, B. Chance, M. Ferrari, M. Kohl (eds.), *Photon Propagation in Tissue IV*, vol. 3566, Proceedings of the SPIE—The International Society for Optical Engineering, 1998, pp. 151–160.
- [28] A. Klose, V. Prapavat, O. Minet, J. Beuthan, G. Mueller, RA diagnostics applying optical tomography in frequency domain, Proceedings of the SPIE—The International Society for Optical Engineering, vol. 3196, 1997, pp. 194–204.
- [29] A.E. Cerussi, A.J. Berger, F. Bevilacqua, N. Shah, D. Jakubowski, J. Butler, R.F. Holcombe, B.J. Tromberg, Sources of absorption and scattering contrast for near-infrared optical mammography, *Academic Radiology* 8 (3) (2001) 211–218.
- [30] S. Nioka, Y. Yung, M. Shnall, S. Zhao, S. Orel, C. Xie, B. Chance, L. Solin, Optical imaging of breast tumor by means of continuous waves, *Advances in Experimental Medicine and Biology* 411 (1997) 227–232.
- [31] B.J. Tromberg, O. Coquoz, J.B. Fishkin, T. Pham, E.R. Anderson, J. Butler, M. Cahn, J.D. Gross, V. Venugopalan, D. Pham, Non-invasive measurements of breast tissue optical properties using frequency-domain photon migration, *Philosophical Transactions of the Royal Society of London-Series B: Biological Sciences* 352 (1997) 661–668.
- [32] R.R. Alfano, S.G. Demos, S.K. Gayen, Advances in optical imaging of biomedical media, *Annals of the New York Academy of Sciences* 820 (1997) 248–270 (Discussion 271).
- [33] M.A. Franceschini, K.T. Moesta, S. Fantini, G. Gaida, E. Gratton, H. Jess, W.W. Mantulin, M. Seeber, P.M. Schlag, M. Kaschke, Frequency-domain techniques enhance optical mammography: initial clinical results, Proceedings of the National Academy of Sciences of the United States of America 94 (1997) 6468–6473.
- [34] J.A. Parker, *Image Reconstruction in Radiology*, CRC Press, Boca Raton, FL, 1990.
- [35] S.A. Walker, S. Fantini, E. Gratton, Image reconstruction by backprojection from frequency domain optical measurements in highly scattering media, *Applied Optics* 36 (1997) 170–179.
- [36] S.B. Colak, D.G. Papaioannou, G.W. 't Hooft, M.B. van der Mark, H. Schomberg, J.C.J. Paasschens, J.B.M. Melissen, N.A.A.J. van Asten, Tomographic image reconstruction from optical projections in light-diffusing media, *Applied Optics* 36 (1997) 180–213.
- [37] A.H. Hielscher, A.D. Klose, K.M. Hanson, Gradient-based iterative image reconstruction scheme for time resolved optical tomography, *IEEE Transactions on Medical Imaging* 18 (3) (1999) 262–271.
- [38] S.R. Arridge, Optical tomography in medical imaging, *Inverse Problems* 15 (2) (1999) R41–R93.
- [39] A.D. Klose, A.H. Hielscher, Iterative reconstruction scheme for optical tomography based on the equation of radiative transfer, *Medical Physics* 26 (8) (1999) 1698–1707.
- [40] R. Roy, E.M. Sevick-Muraca, Truncated Newton's optimization scheme for absorption and fluorescence optical tomography: Part I theory and formulation, *Optics Express* 4 (10) (1999) 353–371.
- [41] O. Dorn, A transport–backtransport method for optical tomography, *Inverse Problems* 14 (1998) 1107–1130.
- [42] Y.Q. Yao, Y. Wang, Y.L. Pei, W.W. Zhu, R.L. Barbour, Frequency-domain optical imaging of absorption and scattering distributions by Born iterative method, *Journal of the Optical Society of America A* 14 (1997) 325–342.
- [43] H. Jiang, K.D. Paulsen, U.L. Österberg, Optical image reconstruction using DC data: simulations and experiments, *Physics in Medicine and Biology* 41 (1996) 1483–1498.
- [44] H.B. Jiang, K.D. Paulsen, U.L. Osterberg, P.W. Pogue, M.S. Patterson, Optical image reconstruction using frequency domain data: simulations and experiments, *Journal of the Optical Society of America A* 13 (1996) 253–266.
- [45] R.L. Barbour, H.L. Graber, J.W. Chang, S.L.S. Barbour, P.C. Koo, R. Aronson, MRI-guided optical tomography: prospects and computation for a new imaging method, *IEEE Computational Science and Engineering* 2 (1995) 63–77.
- [46] K.D. Paulsen, H. Jiang, Spatially varying optical property reconstruction using a finite element diffusion equation approximation, *Medical Physics* 22 (1995) 691–701.
- [47] M.A. O'Leary, D.A. Boas, B. Chance, A.G. Yodh, Experimental images of heterogeneous turbid media by frequency-domain diffusion-photon tomography, *Optics Letters* 20 (1995) 426–428.
- [48] D.Y. Paithankar, A.U. Chen, B.W. Pogue, M.S. Patterson, E.M. Sevick-Muraca, Imaging of fluorescent yield and lifetime from multiply scattered light reemitted from random media, *Applied Optics* 36 (1997) 2260–2272.
- [49] M.V. Klibanov, T.R. Lucas, R.M. Frank, A fast and accurate imaging algorithm in optical diffusion tomography, *Inverse Problems* 13 (1997) 1341–1361.
- [50] S. Bartel, G. Abdoulaev, A.H. Hielscher, Parallelization of gradient-based image reconstruction schemes, in: *Biomedical Topical Meetings, OSA Technical Digests, Optical Society of America, Washington, DC, 2000*, pp. 433–435.
- [51] M.J. Holboke, B.J. Tromberg, X. Li, N. Shah, J. Fishkin, D. Kidney, J. Butler, B. Chance, A.G. Yodh, Three-dimensional diffuse optical mammography with ultrasound localization in a human subject, *Journal of Biomedical Optics* 5 (2) (2000) 237–247.
- [52] S.R. Arridge, M.A. Schweiger, A gradient-based optimization scheme for optical tomography, *Optics Express* 2 (6) (1998) 213–226.
- [53] A.J. Davies, D.B. Christianson, L.C.W. Dixon, R. Roy, P. van der Zee, Reverse differentiation and the inverse diffusion problem, *Advances in Engineering Software* 28 (1997) 217–221.
- [54] R. Roy, E.M. Sevick-Muraca, Truncated Newton's optimization scheme for absorption and fluorescence optical tomography: Part I theory and formulation, *Optics Express* 4 (10) (1999) 353–371.
- [55] A.D. Klose, U. Netz, J. Beuthan, A.H. Hielscher, Optical tomography using the time-independent equation of radiative transfer. Part 1: forward model, *Journal of Quantitative Spectroscopy and Radiative Transfer* 72 (5) (2002) 691–713.
- [56] A.D. Klose, A.H. Hielscher, Optical tomography using the time-independent equation of radiative transfer. Part 2: inverse model, *Journal of Quantitative Spectroscopy and Radiative Transfer* 72 (5) (2002) 715–732.
- [57] D.G. Luenberger, *Linear and Nonlinear Programming*, Addison-Wesley, Boston, MA, 1984.
- [58] P.S. Pacheco, *Parallel Programming with MPI*, Morgan Kaufmann Publishers, San Francisco, CA, 1997.
- [59] Special Issue: MPI—A Message Passing Interface Standard. *International Journal of Supercomputer Applications and High Performance Computing* 8 (1994).
- [60] D.W. Stoller, *Magnetic Resonance Imaging in Orthopaedics and Sports Medicine*, second ed, Lippincott-Raven, New York, 1997.