A Framework for Automatic Tuning of System Parameters and Its Use in Image Registration

Ren Hui Gong[a] and Ziv Yaniv[a]

[a]Sheikh Zayed Institute for Pediatric Surgical Innovation, Children's National Medical Center, Washington, DC 20010, USA

ABSTRACT

Traditionally, internal parameters of a system are most often obtained empirically, through trial-and-error, if intrinsic knowledge about the parameters is not available. In this paper, we present a more *intuitive* and *systematic* framework for this type of problems, and use it to refine the system parameters of a common registration problem. We formulate the performance of the registration problem as a function of its internal parameters, and use optimization techniques to search for an optimal value for the parameters. Registration quality is evaluated using a set of training images in which the anatomy of interest was segmented and comparing the overlap between the segmentations as induced by the registration. As a large number of computationally complex training registrations are performed during the optimization, a cluster of MPI-enabled computers are used collaboratively to reduce the computation time. We evaluated the proposed method using ten CT images of liver from five patients, and examined three optimization algorithms. The results showed that, compared with the empirical values suggested in the published literature, our new technique was able to obtain better system parameters that are tuned for particular applications in a more intuitive and systematic way. In addition, the proposed framework can be potentially extended to solve similar problems.

Keywords: parameters tuning, framework, meta-optimization, image registration, MPI-based computation

1. MOTIVATION

Nearly every intelligent system in medicine, software or hardware, has internal parameters, and the settings of the parameter values are critical to the system's overall performance. While there are well established methods to determine the parameter values for systems with analytical models (e.g. calibration parameters for imaging devices), empirical approaches through trial-and-error are commonly adopted for the majority of other parameters (e.g. special parameters in image registration that are involved in image pre-processing, similarity metric definition, function optimization, and so on). One main disadvantage of empirical approaches is that the obtained parameters may only work well for typical or general cases, and we may not even know whether the obtained parameters are optimal or not. In clinical practice, ideal parameter values can differ greatly even for similar tasks such as registration of different bony structures. Fine-tuned parameters can potentially improve the performance of these applications.

In this paper, we propose a framework to automate the process of parameters tuning, and use it to refine several numerical* parameters of a common registration problem. This approach uses optimization techniques to search for an optimal solution, and intuitive user inputs to drive the search; thus it provides a more systematic and resilient approach for a large number of similar problems. However, in most cases the cost function is not necessarily smooth, making optimization a challenging task. An additional potential challenge has to do with the computational complexity of the function evaluation, in some cases this can be prohibitive and requires the use of distributed computing.

Our approach to optimizing the parameters involved in registration can be viewed as a variant of meta-optimization.[1] Meta-optimization is the use of optimization techniques to tune parameters of other optimization methods and has been previously applied to tuning neural networks, and various evolutionary algorithms.[2] In this work, we aim to provide a general framework for problems in the medical domain and address the associated challenges.

*A system can have non-numerical or discrete parameters that are critical; however, they are beyond the scope of this work.

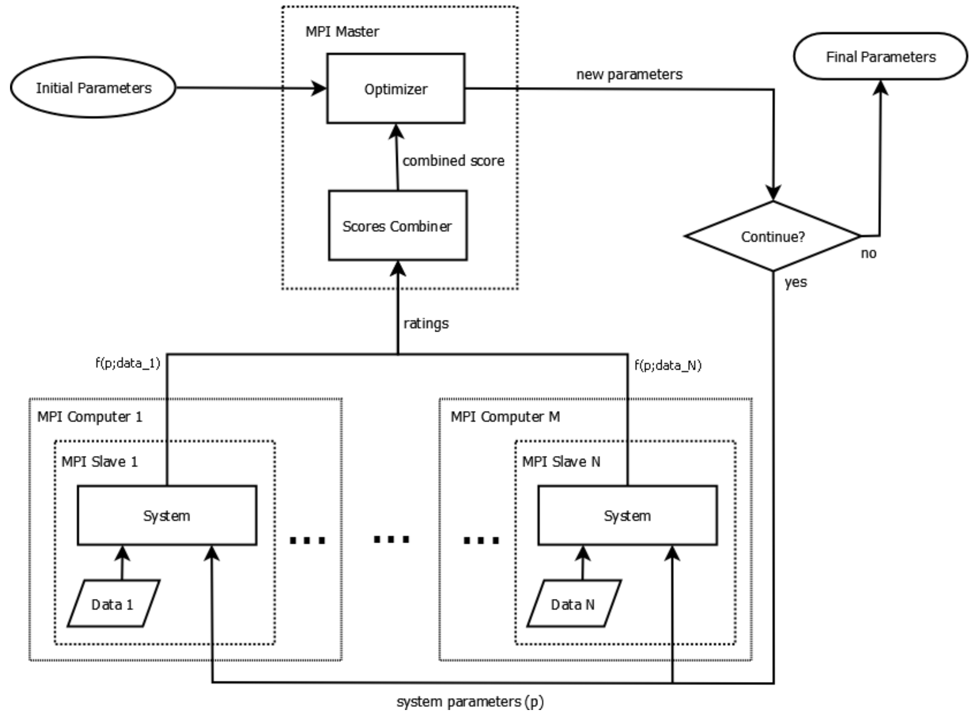


Figure 1. General framework for system parameters tuning.

2. METHOD

2.1 General Framework for System Parameters Tuning

Most systems have *internal settings* that are used to process some *user-supplied input*. We generalize these two types of information with a vector of parameters p, and an abstract data structure *data*. Usually the internal parameters are fixed or user-configurable within a predefined range when the system is deployed, while the user data being processed during production can have a large variability and the outcome is highly dependent on the settings of the internal parameters. In this work, we only consider parameters in $\mathbb{R}$ that can be numerically optimized, although other types of parameters may also be important to the system.

Assume that, for any given setting of the parameters and any given user data, we can find a function f to evaluate the performance of the system, then the problem of parameters tuning can be solved by optimizing the function $f(p, data)$ with respect to parameters p, taking into account all possible variability of the user data. As it is difficult to predict the user data to be processed, we assume that a set of training data, denoted $\{data_i\}, i = 1...N$, that captures the distribution of the expected inputs is available, then we propose a general framework to automatically solve this type of problems. Fig. 1 illustrates the main components as well as the flowchart of the framework.

In Fig. 1, an initial value of the parameters is provided to start the optimization process. In each optimization iteration, the parameters are fed into a set of N instances of the system, each processes one training data set and is rated under the current setting of the parameters. The individual ratings are then collected and aggregated by a score combiner, which is in turn fed into the optimizer to produce an improved setting of the parameters.

In practice, the computation cost of this framework can be very expensive. This is mainly because it is often difficult to find a well-behaved function between the parameters and the system's overall performance, thus f can be highly non-linear and very noisy, requiring the use of population-based algorithms which are more appropriate for searching these types of functions. In addition, a large N may be needed to represent the distribution of the user data, and each single evaluation of the system can have a high computational complexity too. To overcome these issues, we propose to implement the framework on a cluster of workstations using the Message Passing

Interface (MPI).[3] As illustrated in Fig. 1, the problem is solved using a master-slave computing style on a set of M computers. The function evaluation for each training data set is evaluated in a separate slave process, and the master process is responsible for combining the individual ratings and performing the optimization. Each participating computer can simultaneously run a number of slave processes that are specified by the user.

The modalities of the user data, definition of the function f, choices of the optimizer, and so on, are application specific. In the next subsection, we present how this framework fits into the context of a common registration problem in medical image processing.

2.2 Automatic Tuning of Registration Parameters

2.2.1 Parameters to be optimized

Deformable 3D-3D image registration is a fundamental task in medical image processing. It is commonly used for constructing anatomical atlases, and enabling image-guided interventions in soft tissue structures. The registration task is usually solved using a multi-level scheme: in the first level, the global alignment between the images is estimated using a transformation of lower degrees-of-freedom (DOFs) such as rigid, similarity or affine; in the second level, registration is locally refined with a transformation of higher DOFs such as Free Form Deformations (FFD). Gradient-descent (GD) is commonly used as the optimization algorithm in both levels because of its high efficiency. However, the settings of several internal parameters in GD can significantly affect the overall performance of the registration, which includes failure rate, accuracy, and computation time.

We use our framework to determine four critical parameters of GD for a typical deformable registration problem, which consists of a similarity registration in the first level, and a B-spline FFD registration in the second level. The similarity registration determines three types of transformation parameters, which are rotation in the form of a versor, translation, and isotropic scaling. As the parameters are not commensurate it is common to scale their values so that changes in all values have similar effects on the change in the optimized function value. These scaling factors have a significant effect on the registration optimization process. If not appropriately set, the optimization may be biased to a subset of the transformation parameters, which yields premature termination or higher failure rate. In the B-spline deformable registration, one important GD parameter is the relaxation factor, which determines how much or how fast to relax the step length when the current step length no longer drives the registration. Too large or too small step sizes will affect the registration performance, especially the convergence speed. We use the following vector to represent the GD parameters to be optimized:

$$p = \{s_v, s_t, s_s, r\}, \tag{1}$$

where s_v, s_t and s_s are the scaling factors for the components of rotation (i.e. versor), translation and isotropic scaling, respectively, in the similarity registration, and r is the relaxation factor for the step length in the B-spline deformable registration.

We use the Insight Registration and Segmentation Toolkit (ITK)[4] to implement this registration. There are suggested values for p given in the ITK manual and examples; however, no reasoning is given with regard to the specific value choices, and we do not know whether they are optimal.

2.2.2 Evaluation of registration performance

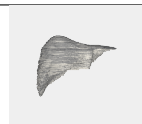
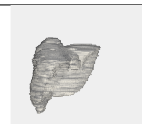
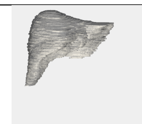
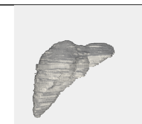
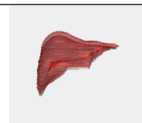
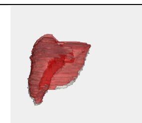
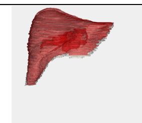
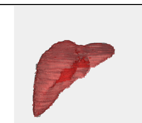
The performance of the described registration is evaluated by performing a number of registrations with a set of training images and evaluating the overall final alignment.

For each training image, an anatomy of interest is pre-segmented. The final alignment of a registration is calculated as the complement of the Dice Similarity Coefficient (DSC),[5] which computes the ratio of overlap between the two segmentations. The complemented DSC for a pair of images $(I_i^A, I_i^B) = data_i$ is defined as follows:

$$f(p; I_i^A, I_i^B) = 1.0 - \frac{2.0 \times |S_i^A \bigcap R(p, S_i^B)|}{|S_i^A| + |R(p, S_i^B)|}, \tag{2}$$

where $(S_i^A,\ S_i^B)$ are the corresponding segmentations, $R(p, .)$ denotes the registered image under parameters p, $\bigcap$ is the overlap between two segmentations, and $|.|$ denotes the number of voxels inside a segmentation or the overlap.

Table 1. Image specifications and coronal views of the segmented liver. The last row shows the overlaps between T0 (end-inhale) and T50 (end-exhale), which illustrates the largest motions between the two respiratory phases).

	Patient 0	Patient 1	Patient 2	Patient 3	Patient 4
Resolution	$512^2 \times 136$	$512^2 \times 120$	$512^2 \times 151$	$512^2 \times 120$	$512^2 \times 160$
Spacing (mm^3)	$0.976562^2 \times 2.5$				
T0					
T50 over T0					

The overall final alignment is computed as the average alignment across all registrations:

$$f(p; \{I_i^A, I_i^B\}_{i=1 \ldots N}) = \frac{1}{N} \sum_{i=1}^{N} f(p; I_i^A, I_i^B). \tag{3}$$

2.2.3 Optimization algorithms

The function defined in Eq. 3 is highly non-linear and very noisy, rendering local optimization techniques such as Amoeba and GD unreliable for this problem. We propose to use population-based optimization algorithms to search for an optimal value for p. We evaluate three algorithms, Particle Swarm Optimization (PSO),[6] Covariance Matrix Adaptation Evolution Strategy (CMA-ES),[7] and an adapted Brute Force (BF) search. The first two algorithms use a number of sample points in the parameter's space to drive the search process, thus they are more robust to noise and local minima. For the brute force approach, we first evaluate the function at points of a regular grid sampled from a user-defined region in the parameter's space; then, starting from the best value, a second search with the Amoeba local optimization algorithm is performed.

3. RESULTS

3.1 Data

We used 4D CT abdominal images to evaluate the proposed method. The images came from five patient data sets, each consisting of ten CT images corresponding to different respiratory phases. We used two images from each patient, end-inhale[†] and end-exhale. Each image also had an associated segmentation of the liver that was performed by an attending interventional radiologist.

Table 1 lists the specifications of the images and renderings of the segmentations from the coronal view.

3.2 Evaluation Scheme

We used five images to train our proposed method, and the other five images to verify the results obtained from the training. For each of the training and verification studies, ten different registrations were formed from five images. Fig. 2 illustrates the registration schemes used for training and verification.

We used Mattes Mutual Information[8] as the similarity metric in both registration levels. The similarity registrations used about 1% of voxels from the liver region for metric calculation, and were executed for 150 iterations. The B-spline deformation registrations used about 3% of voxels, and were executed for 50 iterations. A $7 \times 7 \times 7$ B-spline control grid over the liver region was used to model the tissue deformations.

[†]For patient 3, we used the first respiratory phase after end-inhale instead of end-inhale as the reconstruction of end-inhale exhibited large motion artifacts.

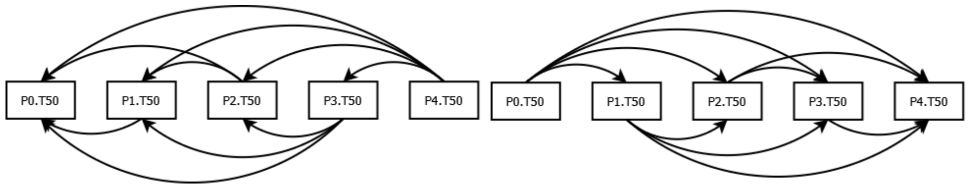


Figure 2. Registration schemes used for training (left) and verification (right). An arrow from PA.TX to PB.TY means registering the image of patient A at time X to the image of patient B at time Y.

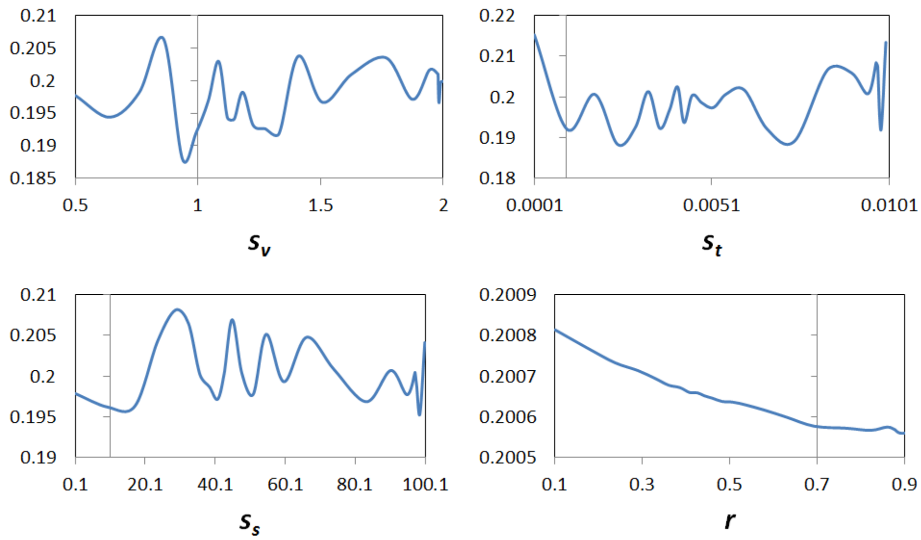


Figure 3. Function plots along each parameter axis in the bounded feasible region used for the PSO algorithm. Vertical lines in the middle show the locations of individual components of the ITK recommended value.

We studied the behaviors of the proposed method under each of the three described optimization algorithms, and used the parameters suggested in the ITK examples, which were $(1.0, 0.001, 10, 0.7)$, as a reference for comparison. For PSO, 25 particles were used and 30 generations were executed. It does not need an initial value of the parameters; instead, the bounding region $(0.5, 0.0001, 0.1, 0.1) - (2.0, 0.01, 100, 0.9)$ was used to create the particles as well as to constrain the parameters during the optimization. For CMA-ES, $(1.0, 0.01, 50, 0.5)$ was used as the initial value, 7 particles were used in each iteration, and 100 total iterations were performed. For adapted BF, we placed a $5\times5\times5\times5$ grid over the same bounding region used for the PSO and performed function evaluations at all grid points, then an additional optimization with Amoeba was performed for 50 iterations. Each of the above experiments performed about 700 total registrations, which were executed in parallel using 10 slave processes.

We used a cluster of four computers and the MPICH2 implementation of the MPI protocol to perform the experiments. Each computer was equipped with an Intel i7 CPU, 12GB RAM and 64-bit Windows 7. One computer executed the master process plus one slave process, and the rest of the workstations executed three slave processes each. For each optimization algorithm being studied, the computation took about 8 hours.

3.3 Results and Discussion

We first look at the function plots at the ITK suggested value and along each parameter axis. From Fig. 3, it is obvious that we were dealing with a function that has many local minima, and searching for an optimal solution is challenging.

Fig. 4 shows the overall alignments over time for the three optimization algorithms being studied, and Table 2 shows the final alignments when the obtained parameters were used to register the testing images.

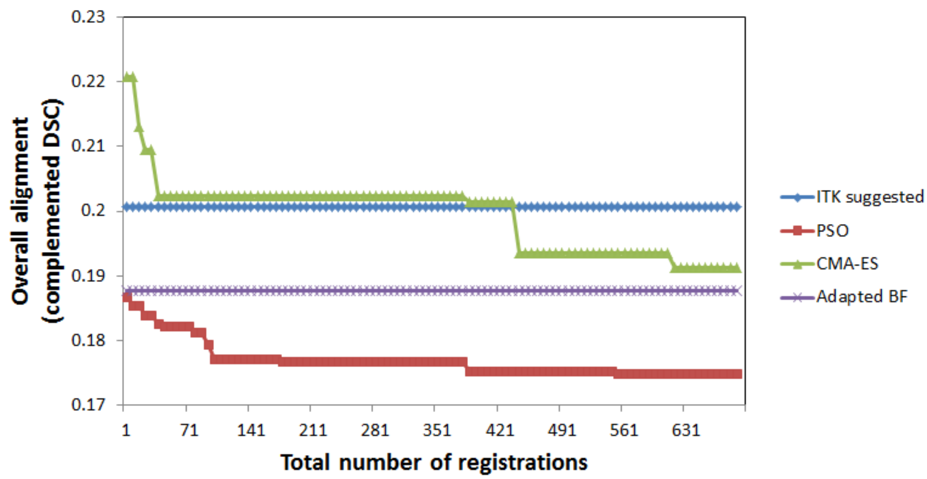


Figure 4. Overall alignments over time for the three optimization algorithms. The value for the adapted BF was available only at the end of the optimization. This value as well as the ITK suggested one is displayed for illustration purposes.

Table 2. Results with testing images: final alignments (complemented DSC) obtained from parameters suggested by ITK and our proposed method. An arrow from PA.TX to PB.TY means registering the image of patient A at time X to the image of patient B at time Y.

Registration	ITK suggested	PSO	CMA-ES	Adapted BF
P0.T50 $\rightarrow$ P1.T50	0.24	0.24	0.26	0.24
P0.T50 $\rightarrow$ P2.T50	0.13	0.11	0.15	0.12
P0.T50 $\rightarrow$ P3.T50	0.13	0.10	0.13	0.12
P0.T50 $\rightarrow$ P4.T50	0.22	0.22	0.21	0.20
P1.T50 $\rightarrow$ P2.T50	0.21	0.16	0.20	0.15
P1.T50 $\rightarrow$ P3.T50	0.14	0.14	0.17	0.17
P1.T50 $\rightarrow$ P4.T50	0.22	0.24	0.22	0.22
P2.T50 $\rightarrow$ P3.T50	0.19	0.14	0.21	0.16
P2.T50 $\rightarrow$ P4.T50	0.16	0.15	0.18	0.16
P3.T50 $\rightarrow$ P4.T50	0.23	0.23	0.24	0.23
mean $\pm$ std.dev.	0.19 ± 0.04	0.17 ± 0.05	0.20 ± 0.04	0.18 ± 0.04

In Fig. 4, all three studied optimization algorithms were able to improve the overall alignments at the end of optimizations for the training images, with the best result being produced by PSO. From Table 2, we observed an overall 2% improvement in final alignment for PSO, and 1% for adapted BF when the obtained parameters were applied to the testing images. No improvement was observed for CMA-ES, but the results were still comparable to those obtained using the ITK suggested value. We believe that the CMA-ES results can be further improved if we allow more iterations and use the improved CMA-ES implementations that incorporate anisotropic initialization for different parameters.

We also examined the parameters produced by the three algorithms. We observed that the optimal value produced by PSO, $(0.5, 0.001, 0.1, 0.1)$, was quite different from the ITK suggested one. This is reasonable as we can see from Fig. 3 that the function was very noisy and multiple local minima existed.

4. CONCLUSION

In this paper we presented a framework to automate the process of system parameters tuning, and used it to refine the internal parameters of a common registration problem. The proposed approach is more intuitive and systematic than the conventional trial-and-error approach, and was able to produce improved system parameters. We also proposed an MPI-based implementation such that the expensive tuning task can be performed using a cluster of workstations.

In the future, we will perform additional testing with different data modalities and data with more variability. We will also improve the MPI implementation to enable automatic load balancing to further improve computation performance.

REFERENCES

[1] Karpenko, A. P. and Svianadze, Z. O., "Meta-optimization based on self-organizing map and genetic algorithm," *Optical Memory & Neural Networks* **20**, 279–283 (Dec. 2011).

[2] Smit, S. K. and Eiben, A. E., "Comparing parameter tuning methods for evolutionary algorithms," in [*IEEE Congress on Evolutionary Computation*], 399–406 (May 2009).

[3] Skjellum, A., Doss, N. E., Viswanathan, K., Chowdappa, A., and Bangalore, P. V., "Extending the message passing interface (MPI)," in [*Proceedings of Scalable Parallel Libraries, IEEE Conference II*], (October 1994).

[4] Ibanez, L., Schroeder, W., Ng, L., and Cates, J., [*The ITK Software Guide Second Edition (Updated for ITK version 2.4)*], Kitware (2005).

[5] Dice, L. R., "Measures of the amount of ecologic association between species," *Ecology* **26**, 297–302 (July 1945).

[6] Kennedy, J. and Eberhart, R., "Particle swarm optimization," in [*IEEE Conference on Neural Networks*], **4**, 1942–1948 (Nov. 1995).

[7] Hansen, N., "The CMA evolution strategy: A comparing review," in [*Advances on estimation of distribution algorithms*], 75–102, Springer (2006).

[8] Mattes, D., Haynor, D. R., Vesselle, H., Lewellen, T. K., and Eubank, W., "Nonrigid multimodality image registration," in [*Proceedings of SPIE*], **4322**, 1609–1620 (2001).