sciendo

TATRA MOUNTAINS
Mathematical Publications

DOI: 10.2478/tmmp-2020-0006
Tatra Mt. Math. Publ. **75** (2020), 81–102

EFFICIENT 3D SHAPE REGISTRATION BY USING DISTANCE MAPS AND STOCHASTIC GRADIENT DESCENT METHOD

Polycarp Omondi Okock[1,2] — Jozef Urbán[2] — Karol Mikula[1]

[1]Slovak University of Technology, Bratislava, SLOVAKIA

[2]TatraMed Software s.r.o , Bratislava, SLOVAKIA

ABSTRACT. This paper presents an efficient 3D shape registration by using distance maps and stochastic gradient descent method. The proposed algorithm aims to find the optimal affine transformation parameters (translation, scaling and rotation) that maps two distance maps to each other. These distance maps represent the shapes as an interface and we apply level sets methods to calculate the signed distance to these interfaces. To maximize the similarity between the two distance maps, we apply sum of squared difference (SSD) optimization and gradient descent methods to minimize it. To address the shortcomings of the standard gradient descent method, i.e., many iterations to compute the minimum, we implemented the stochastic gradient descent method. The outcome of these two methods are compared to show the advantages of using stochastic gradient descent method. In addition, we implement computational optimization's such as parallelization to speed up the registration process.

1. Introduction

Registration of two shapes means finding a geometrical transformation that aligns them. This is done so that the same features overlap and differences are highlighted. Computing this geometrical transformation is a fundamental problem in computer vision which then can be used for higher level shape processing methods or analysis.

2010 Mathematics Subject Classification: 65K10, 35Q68, 65Y05, 65Y20.
Keywords: distance map, stochastic gradient method, registration, parallelization, affine transformation, optimization.
This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 721537. The third author was also supported by the grant APVV-15-0522.

We follow the general registration formulation which is as follows: let $\mathbf{F}, \mathbf{M} \subset \mathcal{R}^3$ be the fixed and moving shapes, respectively. Our objective is to find a 3D affine transformation matrix $\mathbf{A}$ that maps $\mathbf{M}$ to $\mathbf{F}$ while minimizing the dissimilarity measure between the transformed shape $\mathbf{A}(\mathbf{M})$ and the fixed shape $\mathbf{F}$. We use a sum of squared differences (SSD) as the dissimilarity measure. This dissimilarity measure determines the quality of the registration process. In addition, the shape representation has a significant impact on the registration performance. Some methods are based on explicit representations such as active contours [6], Fourier descriptors [24], active shapes models [2] while others are based on implicit representations such as signed distance functions [14]. These implicit representations have the advantage of natural extension to higher dimensions over their explicit counterparts [21].

Techniques for registration include those that are contour-based [9, 10, 12] registration methods which are known to be robust and efficient. But these techniques require point correspondence for the boundary of the shapes. Other techniques are gradient descent based registration methods. These are iterative in nature and converge to the minimum by moving iteratively in the direction of the steepest gradient. This direction is defined by the negative of the gradient.

This paper proposes usage of distance maps to represent the 3D shapes. Then we perform registration of the two distance maps. Through this approach, we can use the level sets method [21]. These methods avoid any explicit reconstruction of the boundary, which can introduce errors or slow down the process. We adopt sum of squared difference optimization procedure. This accounts for the distance maps corresponding to each other and use the gradient descent methods for minimization. In addition, the standard gradient descent method can take many iterations to compute a global minimum with required accuracy. This is true for large domains. Because of this, we have also implemented stochastic gradient descent [19] method, also known as incremental gradient descent method. This method performs a stochastic approximation of the gradient descent optimization.

Our approach is similar to the methods described in [14,22]. But we go further and apply successful registration to 3D shapes and implement stochastic gradient descent method. We also added computational optimization routines to speed up the registration process.

This paper is organized as follows. In section 2, we present the underlying transformation model that maps one shape to another. In section 3, we present a way to describe shapes using the level set methods [21] and use the signed distance function to such level set. In section 4, we present the optimization procedure used to measure the similarity between two distance maps being registered and computational optimizations. In section 5, we present some numerical experiments and discuss their results.

2. Transformation model

An important concept in mapping one shape to another shape is the underlying transformation model. Goal of the registration is finding the transformation that maps one space to another space. Usually the shape we are mapping to is defined as the fixed shape whereas the shape that we are mapping to the fixed shape is referred to as the moving shape.

Complexity is determined by what transformation model we are using. In our case, complexity stands for the number of parameters that have to be solved for. We chose to use the *affine* transformation model. This model is invariant to *translation*, *rotation* and *scaling* transformations. We wish to find the *affine* transformation that minimizes the differences between the moving and fixed shape defined in $\mathcal{R}^3$.

Let $\mathbf{F}, \mathbf{M} \subset \mathcal{R}^3$ be the fixed and moving shapes, respectively. The objective is to find the 3D affine transformation matrix, $\mathbf{A}$, that maps $\mathbf{M}$ to $\mathbf{F}$.

2.1. 3D Affine Transformation

The transformation matrix $\mathbf{A}$ is defined as

$$\mathbf{A} = \mathbf{T} \cdot \mathbf{S} \cdot \mathbf{R}\,.$$

$\mathbf{R}$ is the 3D rotation matrix defined as

$$\begin{bmatrix} \cos\psi\cos\theta & -\cos\theta\sin\psi & \sin\theta & 0 \\ \cos\phi\sin\psi + \cos\psi\sin\phi\sin\theta & \cos\phi\cos\psi - \sin\phi\sin\psi\sin\theta & -\cos\theta\sin\phi & 0 \\ \sin\phi\sin\psi - \cos\phi\cos\psi\sin\theta & \cos\psi\sin\phi + \cos\phi\sin\psi\sin\theta & \cos\phi\cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The rotation direction is anticlockwise with the angles ϕ →x direction, θ →y direction, ψ →z direction. Rotation in 3D is not commutative and the order of rotation is important. We chose the following **XYZ** order. The x rotation, r_x, y rotation, r_y are defined as:

$$r_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi & 0 \\ 0 & \sin\phi & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad r_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

and z rotation, r_z is defined as:

$$r_z = \begin{bmatrix} \cos\psi & -\sin\psi & 0 & 0 \\ \sin\psi & \cos\psi & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Therefore $\mathbf{R} = r_x \cdot r_y \cdot r_z$. $\mathbf{S}$, the scaling matrix and $\mathbf{T}$, the translation matrix, are given by:

$$\mathbf{S} = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 & 0 \\ 0 & 0 & \frac{1}{s_z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad \mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & -t_x \\ 0 & 1 & 0 & -t_y \\ 0 & 0 & 1 & -t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Therefore, the transformation matrix $\mathbf{A}$ is given as

$$\begin{pmatrix} \frac{\cos(\psi)\cos(\theta)}{s_x} & -\frac{\cos(\theta)\sin(\psi)}{s_x} & \frac{\sin(\theta)}{s_x} & -t_x \\ \frac{\cos(\phi)\sin(\psi)+\cos(\psi)\sin(\phi)\sin(\theta)}{s_y} & \frac{\cos(\phi)\cos(\psi)-\sin(\phi)\sin(\psi)\sin(\theta)}{s_y} & -\frac{\cos(\theta)\sin(\phi)}{s_y} & -t_y \\ \frac{\sin(\phi)\sin(\psi)-\cos(\phi)\cos(\psi)\sin(\theta)}{s_z} & \frac{\cos(\psi)\sin(\phi)+\cos(\phi)\sin(\psi)\sin(\theta)}{s_z} & \frac{\cos(\phi)\cos(\theta)}{s_z} & -t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The goal is to find the optimal set of parameters ϕ, θ, ψ, $s_x > 0, s_y > 0, s_z > 0$, t_x, t_y, t_z that minimizes the difference between the fixed shape $\mathbf{F}$ and moving shape $\mathbf{A}(\mathbf{M})$. We therefore have to solve a problem with 9 degrees of freedom.

3. Shape representation

To represent the shape, we view it as a curve or surface and use the level set methods [21] to describe it. The shape representation is given as a zero level set (interface) of some function. We use signed distance function to such zero level set in the registration procedure. The information is not only defined on discrete curve given by small number of points but also in the whole computational domain.

3.1. Signed distance map for shapes

Let $\Phi : \mathbf{D} \subset \mathbb{R}^3 \to R$ be a Lipschitz function that refers to signed distance representation for a given closed interface, $\mathbf{\Gamma}$, contained in the domain $\mathbf{D}$. The domain enclosed by $\mathbf{\Gamma}$ will be denoted by Ω, i.e., $\mathbf{\Gamma} = \partial\Omega$. We suppose that the interface is given implicitly as the zero level set such that:

$$\Phi(x,y,z) = \begin{cases} 0 & (x,y,z) \in \mathbf{\Gamma} = \partial\Omega, \\ +d\big((x,y,z),\mathbf{\Gamma}\big) > 0 & (x,y,z) \in \mathbf{D} - \Omega, \\ -d\big((x,y,z),\mathbf{\Gamma}\big) < 0 & (x,y,z) \in \Omega, \end{cases} \tag{1}$$

where $d((x,y,z),\mathbf{\Gamma})$ is the min. Euclidean distance between the point (x,y,z) and the closed interface $\mathbf{\Gamma} = \partial\Omega$.

- The level set representation (1) is invariant to translation and rotation, see [15].

We implemented the Fast Sweeping method [25], an iterative method that is based on non-linear Gauss-Seidel method with eight (3D case) different orderings ("sweeps"). The detailed description is given in [25]. This was used in the standard gradient descent method described in Section 4.1. On the other hand, in Section 4.2, for the stochastic gradient descent method, we used the so-called "brute force" method to calculate the Euclidean distance between couple of points.

Other popular numerical algorithms include the fast marching method [21], the linearization of Eikonal equation [3]. We do not aim to compare the available numerical algorithms for the computation of distance maps or for a solution of stationary linear advection equation with respect to efficiency and accuracy. Each one is preferable in some special situations, see [4].

4. Optimization procedures

For functional defined on a parameter space, we attempt to quantify the similarity between two distance maps. For our case we have chosen to use sum of squared differences (SSD). The optimization criterion $E(\mathbf{A})$ is defined as:

$$E(\mathbf{A}) = \int_{\mathbf{D}} \left[\Phi_F\left(x,y,z\right) - \Phi_M\big(\mathbf{A}(x,y,z)\big)\right]^2 \mathrm{d}x\mathrm{d}y\mathrm{d}z, \tag{2}$$

where

$$\Phi_{\mathbf{F}}\left(x,y,z\right), \qquad \Phi_{\mathbf{M}}\left(x,y,z\right)$$

are the fixed and moving distance maps, respectively. $\mathbf{A}$ is the 3D affine transformation matrix and $E(\mathbf{A})$ is the **SSD/Energy** we want to minimize by applying the optimal $\mathbf{A}$. Similarly to [15], we reduce the calculation to a narrow band in the distance δ around the inputs. We define the narrow band as:

$$N_\delta(\Phi_1,\Phi_{2)}) = \begin{cases} 1, & \min(|\Phi_1|,|\Phi_2|) \le \delta, \\ \\ 0, & \min(|\Phi_1|,|\Phi_2|) > \delta. \end{cases} \tag{3}$$

The constrained optimization (2) criterion becomes

$$E(\mathbf{A}) = \int_{\mathbf{D}} N_\delta(\Phi_F,\Phi_M)\left(\Phi_F\left(x,y,z\right) - \Phi_M\left(\left(\mathbf{A}(x,y,z)\right)\right)\right)^2 \mathrm{d}x\mathrm{d}y\mathrm{d}z. \tag{4}$$

For minimization and calculation of particular components we chose gradient descent method for $\nabla E(\mathbf{A})$

$$\begin{aligned}
\partial_{\mathbf{R}} E(\mathbf{A}) &= 2\int_{\mathbf{D}} N_\delta(\Phi_F, \Phi_M)\big(\nabla\Phi_M \cdot \nabla_R(\mathbf{Ax})\big)\big(\Phi_F - \Phi_M(\mathbf{A})\big)\mathrm{d}x\mathrm{d}y\mathrm{d}z,\\
\partial_{\mathbf{S}} E(\mathbf{A}) &= 2\int_{\mathbf{D}} N_\delta(\Phi_F, \Phi_M)\big(\nabla\Phi_M \cdot \nabla_S(\mathbf{Ax})\big)\big(\Phi_F - \Phi_M(\mathbf{A})\big)\mathrm{d}x\mathrm{d}y\mathrm{d}z,\\
\partial_{\mathbf{T}} E(\mathbf{A}) &= 2\int_{\mathbf{D}} N_\delta(\Phi_F, \Phi_M)\big(\nabla\Phi_M \cdot \nabla_T(\mathbf{Ax})\big)\big(\Phi_F - \Phi_M(\mathbf{A})\big)\mathrm{d}x\mathrm{d}y\mathrm{d}z. \qquad (5)
\end{aligned}$$

For better understanding of the formula $\nabla E(\mathbf{A})$, the $\mathbf{x}$ in $(\mathbf{Ax})$ is defined as

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix},$$

and $\mathbf{Ax}$ is

$$\begin{pmatrix}
\frac{x\cos(\psi)\cos(\theta)}{s_x} - \frac{y\sin(\psi)\cos(\theta)}{s_x} + \frac{z\sin(\theta)}{s_x} - t_x \\
\frac{x(\sin(\phi)\cos(\psi)\sin(\theta)+\cos(\phi)\sin(\psi))}{s_y} + \frac{y(\cos(\phi)\cos(\psi)-\sin(\phi)\sin(\psi)\sin(\theta))}{s_y} - \frac{z\sin(\phi)\cos(\theta)}{s_y} - t_y \\
\frac{x(\sin(\phi)\sin(\psi)-\cos(\phi)\cos(\psi)\sin(\theta))}{s_z} + \frac{y(\cos(\phi)\sin(\psi)\sin(\theta)+\sin(\phi)\cos(\psi))}{s_z} + \frac{z\cos(\phi)\cos(\theta)}{s_z} - t_z \\
1
\end{pmatrix}.$$

The grad of Φ_M, $\nabla\Phi_M$ is the row vector defined as

$$\nabla\Phi_M = (\partial_x\Phi_M, \partial_y\Phi_M, \partial_z\Phi_M, 0)$$

and gradient for the components are the following column vectors

$$\begin{aligned}
\nabla_R(\mathbf{Ax}) &= \big(\partial_{r_x}\mathbf{Ax}, \partial_{r_y}\mathbf{Ax}, \partial_{r_z}\mathbf{Ax}, 0\big)^T,\\
\nabla_S(\mathbf{Ax}) &= \big(\partial_{s_x}\mathbf{Ax}, \partial_{s_y}\mathbf{Ax}, \partial_{s_z}\mathbf{Ax}, 0\big)^T,\\
\nabla_T(\mathbf{Ax}) &= \big(\partial_{t_x}\mathbf{Ax}, \partial_{t_y}\mathbf{Ax}, \partial_{t_z}\mathbf{Ax}, 0\big)^T.
\end{aligned}$$

In discrete case, we can obtain optimal affine transformation similarly. Let n be the number of points in the narrow band and $n \ll N$, where N is the total number of points. Functional (4) can be rewritten in discrete formulation as

$$E_D(\mathbf{A}) = \frac{1}{n}\sum_{i=1}^{n}\big(\Phi_{F,i} - \Phi_{M,i}(\mathbf{A})\big)^2. \qquad (6)$$

Approximation of gradient components in (5) are given as

$$\partial_R E_D(\mathbf{A}) = 2\frac{1}{n}\sum_{i=1}^{n}\Big(\nabla\Phi_{M,i}\cdot\nabla_R(\mathbf{Ax})\big(\Phi_{F,i}-\Phi_{M,i}(\mathbf{A})\big)\Big),$$
$$\partial_S E_D(\mathbf{A}) = 2\frac{1}{n}\sum_{i=1}^{n}\Big(\nabla\Phi_{M,i}\cdot\nabla_S(\mathbf{Ax})\big(\Phi_{F,i}-\Phi_{M,i}(\mathbf{A})\big)\Big),$$
$$\partial_T E_D(\mathbf{A}) = 2\frac{1}{n}\sum_{i=1}^{n}\Big(\nabla\Phi_{M,i}\cdot\nabla_T(\mathbf{Ax})\big(\Phi_{F,i}-\Phi_{M,i}(\mathbf{A})\big)\Big). \tag{7}$$

Optimization using gradient methods is highly dependent on the initial conditions. For our case it is the initial position, rotation and scaling of two processed shapes. These methods seek a local minimum of the functional. Thus, result can be a transformation that is not what we are looking for.

In addition, gradient descent method can take many iterations to compute a local minimum with required accuracy. For larger domains, this process can be significantly slow. Because of this we have also implemented the stochastic gradient descent [19] method, also known as the incremental gradient descent. This method performs a stochastic approximation of the gradient descent optimization.

4.1. Standard gradient descent method

The standard gradient descent method computes the gradient of the optimization criterion function with respect to the particular component for the entire domain. For our case, it is the narrow band as it was done in [15]. We can do it because just the zero isosurface is interesting for us and gradient vector field of a distance function is well defined. All the samples are selected in the order that they appear in the domain.

Let $\mathbf{w}$ be the vector of parameters of $\mathbf{A}$, the transformation parameters, that minimizes $E(\mathbf{A})$, the optimization criterion. The standard gradient descent method performs the following parameter update

$$\mathbf{w}^{\tau} := \mathbf{w}^{\tau-1} - \lambda\frac{1}{n}\sum_{i=1}^{n}\frac{\partial E_i(\mathbf{A}^{(\tau-1)})}{\partial\mathbf{w}},$$

where λ is the step size, n is the number of points in the narrow band, i is the grid point and τ is the iteration. Note that $n \ll N$, where N is the total number of data points. The standard algorithm has the following procedure

(1) Choose initial component $\mathbf{w}^0$ and the step size λ
(2) Repeat until max no. of iterations, $\tau_{\max}$, or accepted tolerance for $E(\mathbf{A})$ is reached

(a) **For each** $\mathbf{w}^{\tau} : \tau = \{1, 2, \ldots, \tau_{\max}\}$ **do**:

$$\mathbf{w}^{\tau} := \mathbf{w}^{\tau-1} - \lambda \frac{1}{n} \sum_{i=1}^{n} \frac{\partial E_i(\mathbf{A}^{(\tau-1)})}{\partial \mathbf{w}}.$$

In calculation of the constrained optimization criterion, $E(\mathbf{A})$, we used the fast sweeping method to calculate the distance maps $\Phi_{\mathbf{F}}$ and $\Phi_{\mathbf{M}}$. This is because we are calculating distance for all the points in the dataset.

4.2. Stochastic gradient descent method

As previously mentioned, it performs a stochastic approximation [17] of the gradient descent optimization. The gradient is approximated from a single randomly chosen point. The algorithm repeats this process over the data points, with each iteration randomly picking a single point at each step. This randomly chosen point is used to approximate the gradient, until convergence is achieved. Approximation of the true gradient from a single point is referred to as the "true" stochastic gradient descent.

In this method, the trade-off is between computational cost versus accuracy of the component update. This means that the computational costs are significantly less when compared to the standard gradient descent method. For the standard gradient descent method, the accuracy of component update is better when compared to the stochastic gradient decent method but at a slower computational speed. These effects are significantly observable as the data size becomes large.

A compromise between computing the "true" gradient and the gradient at a single point is to compute the gradient using more than one point, also referred to as a "mini-batch" [8], at each step. The number of points in the mini-batch can be in the range 10 to 1000 points. This can perform significantly better than the "true" stochastic gradient descent method described since it is less affected by noise [11]. It may also result in a smoother convergence, as the gradient computed at each step is averaged over more points.

Let $\mathbf{w}$ be the vector of parameters of $\mathbf{A}$, the transformation parameters that minimizes $E(\mathbf{A})$, the optimization criterion. The stochastic gradient descent method performs the following parameter update

$$\mathbf{w}^{\tau} := \mathbf{w}^{\tau-1} - \lambda \frac{1}{M} \sum_{i=1}^{M} \frac{\partial E_i(\mathbf{A}^{(\tau-1)})}{\partial \mathbf{w}},$$

where λ is the step size, M is the mini-batch of random points and τ is the iteration. $M \ll n \ll N$, where n is the number of points in the narrow band and N is the total number of data points, respectively. In the case when $M = 1$, we have the original "true" stochastic gradient descent approach. The stochastic algorithm is as follows:

(1) Choose an initial component $\mathbf{w}^0$ and step size λ;
(2) Repeat until max no. of iterations $\tau_{\max}$ or accepted tolerance for $E(\mathbf{A})$ is reached
   (a) **Choose a set M_τ of random points:** $M_\tau = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$ **and for each** $\mathbf{w}^\tau : \tau = \{1, 2, \ldots, \tau_{\max}\}$ **do**:

$$\mathbf{w}^\tau := \mathbf{w}^{\tau-1} - \lambda \frac{1}{M} \sum_{i=1}^{M} \frac{\partial E_i(\mathbf{A}^{(\tau-1)})}{\partial \mathbf{w}}$$

where $M = \mathrm{card}(M_\tau)$ is the total number of points in M_τ set.

We have chosen mini-batch sizes of 100 and 1000 points for our numerical experiments. In calculation of the constrained optimization criterion (4), we have used the brute force method to calculate the distance values Φ_F and Φ_M, since we are calculating distances only for the few points in the mini-batch. The brute force method calculates the distance for every point in the shape to the selected random point and then select the shortest distance, see [23]. And due its simplicity, we can easily parallelize it.

Minimization of the objective function $E(\mathbf{A})$ using the stochastic gradient descent method fluctuates causing, e.g., jump to new and potentially local minima. From practical experience, reducing the step size by an appropriate rate, reduces the fluctuation behaviour of the objective function. The convergence of stochastic gradient descent method has been analysed using the theories of convex minimization and of stochastic approximation, see [18, 20].

4.3. Computational optimization

We carried out additional computational optimization routines that led to the speed-up of the registration process. They should not affect the outcome of the registration results except for time speed-up gains. Our optimization procedures are performed using gradient descent methods, which are the first order iterative optimization algorithms. These methods can take many iterations to compute the minimum with a required accuracy. This can be more time-consuming when computing the minimum over a large domain. Because of these reasons, we decided to apply the additional computational optimization routines.

The list of areas addressed by computational optimization includes:

(1) Selection of the narrow band size δ.
(2) Bounding box around the region of interest.
(3) Narrow band updated along with transformation update.
(4) Calculation of distance values to the interface only in points required by stochastic gradient descent method.
(5) Identification and parallelization of resource intensive sections.

These routines are explained in the follow-up subsections.

4.3.1. Selection of the narrow band size δ

The goal is to have enough input data points to successfully perform the optimization procedure. For a very small δ value, there is not enough data points to measure the similarity between two distance maps. For a large δ value, there is an excess of input data points and this slows down the optimization process since it has to find the minimum over more points.

In addition, when choosing the suitable δ value, we also considered the relative positions of the two processed shapes. The value chosen was large enough to cover the overlapping regions of the processed shapes.

4.3.2. Bounding box around the region of interest

Searching through the whole domain for data points that are inside the narrow band can be time consuming. Instead, we proposed a defining the region of interest which is called the bounding box and is created from the regions inside the narrow band of size δ. Then we search for data points inside this smaller region instead of the whole domain.

Two bounding boxes are created for the fixed and moving shapes, respectively. Then we create a best fit bounding box that encompasses the two created bounded regions. This becomes the new region of interest, where we calculate inside the distance map and the optimization procedures. See Figure 1 for illustration of the procedure.

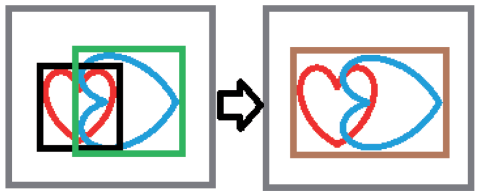

FIGURE 1. Two 2D heart shapes (red and blue) wrapped by their respective bounding rectangles, black and green, respectively. On the right, we have the best fitting box represented by the brown rectangle. The grey box represents the whole domain.

In every iteration, we evolve the bounding box as we seek a local minimum with the new transformation parameter updates and recalculate the best fit region bounding box. See Figure 2 for illustration of this process.

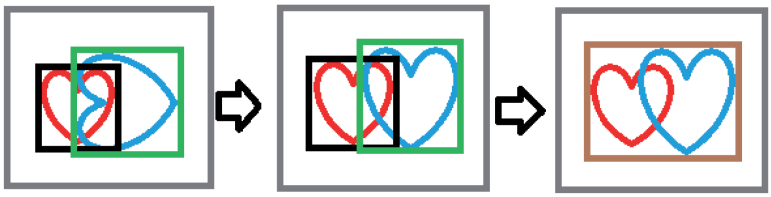

FIGURE 2. On the left, we have the original heart shapes (red and blue) with their respective bounding rectangle boxes. In the middle, we have the transformed (rotated) heart (blue) and unchanged heart (red) images. On the right, we have the best fit bounding box covering the new transformed heart (blue) and original heart (red) images.

4.3.3. Narrow band updated along with transformation update

We proposed a new approach of calculating the narrow band area inside a binary mask representing the data. The binary mask is defined as follows

$$\mathbf{B}\left(x,y,z\right)=\begin{cases}\mathfrak{f}, & \left|\Phi\left(x,y,z\right)\right|\ \leq\ \delta,\\ \mathfrak{b}, & \left|\Phi\left(x,y,z\right)\right|\ >\ \delta,\end{cases} \tag{8}$$

where $\Phi\left(x,y,z\right)$ is the distance map, δ is the narrow band size, $\mathfrak{b}$ and $\mathfrak{f}$ are the background and foreground values, respectively.

We define two binary regions of interest from the fixed Φ_F and moving Φ_M distance maps, respectively. Similarly to (3), we reduce the calculation to the narrow band defined by the foreground value, $\mathfrak{f}$. We define

$$N_{\mathfrak{f}}(B_1,B_2)=\begin{cases}1, & B_1\ \textbf{or}\ B_2\ =\ \mathfrak{f},\\ 0, & \textbf{else}.\end{cases} \tag{9}$$

The constrained optimization (4) is now defined as

$$E(\mathbf{A})=\int_{\mathbf{D}} N_{\mathfrak{f}}(B_F,B_M)\Big(\Phi_F(x,y,z)-\Phi_M\big(\mathbf{A}(x,y,z)\big)\Big)^2\,\mathrm{d}x\mathrm{d}y\mathrm{d}z. \tag{10}$$

In this approach, we only need to calculate the distance maps in the first iteration. In the subsequent iterations, the binary mask $B_{\mathbf{M}}$ is updated using the new transformation $\mathbf{A}$ parameters update corresponding to the moving image.

4.3.4. Calculation of distance values to the interface only in points required by stochastic gradient descent

The distance map to a shape is usually the distance calculated to the interface from all other points in the domain. In the stochastic gradient descent method,

we do not need to evaluate distance values everywhere but just in points considered in the optimization process. We exploit this advantage and calculate distance values using the simple brute force method.

4.3.5. Identification and parallelization of resource intensive sections

From our experience, distance map calculation is the most time-consuming task. In our implementation, we are using brute force method for stochastic gradient method. Its cost is proportional to the size of M random points. This cost grows as we increase the size of M random points.

Due the simplicity of the brute force method, we decided to parallelize the method using the OpenMP [7]. Other parallelization tools would be using **CUDA** [1].

5. Numerical experiments

We present the experiments carried out and discuss their results in this section. The purpose of the experiments is to illustrate successful registration of 3D shapes using the standard and stochastic gradient descent methods. The main focus will be using few randomly selected points to perform a successful registration. In addition, we plan to show the overall improvement in the registration total time with the addition of parallelization to the process.

The setup is as follows: the data used was 3 human bladder cases each having the dimension 100x100x40 voxels, see Figures 3 and 4. A narrow band of size $\delta = 10$, step size $\lambda = 0.1$ for the rotation and translation and $\lambda = 0.025$ for the scaling components, respectively, belonging to the affine transformation $\mathbf{A}$. The tests are performed on a known affine transformation $\mathbf{A}$ given in Table 1.

Table 1. The affine transformation we are interested to find.

Rotation			Scaling			Translation		
ϕ	θ	ψ	s_x	s_y	s_z	t_x	t_y	t_z
-0.17453	0.17453	0.349066	0.8	1.1	1.25	3.0	5.0	-7.0

The first experiment was to test how successful the registration was when applying either the standard or stochastic gradient descent methods. The stochastic gradient descent (Sgd.) method was tested with mini-batch sizes $M = 100$ and $M = 1000$ random points. In both methods, the step size λ was set the same as described previously. From Figure 5, the comparison of minimized $E(\mathbf{A})$, **SSD/Energy** is done for the standard and stochastic descent methods.

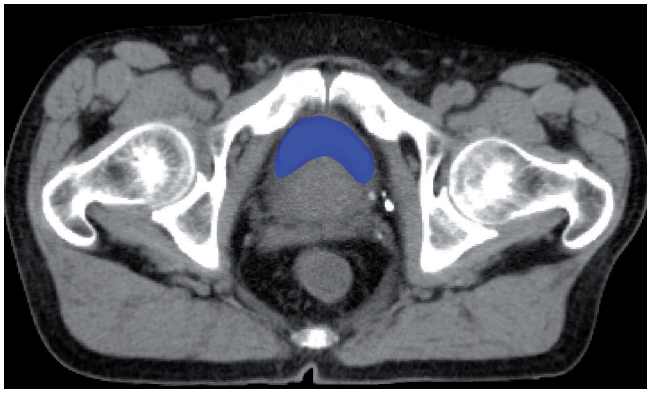

FIGURE 3. The blue region represents a section through the bladder in a man's pelvis region. The anterior view.

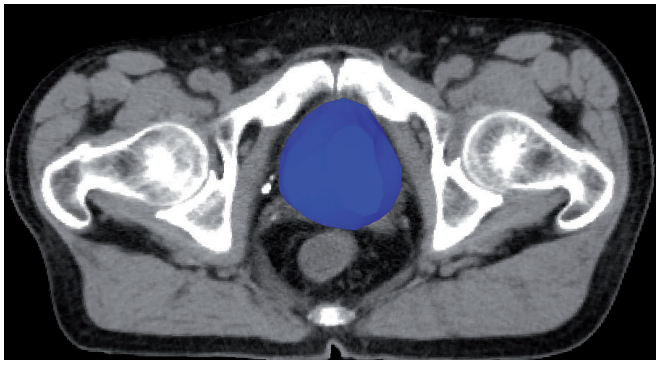

FIGURE 4. The blue region represents a bladder shown in Figure 3 of a man's pelvis region. The posterior view.

As shown in the figure, both curves converge similarly for the case of bladder 1. Similar convergence is observed for cases of bladders 2 and 3, see Figures 6 and 7.

The resultant transformations found after carrying out registration are given in Tables 2, 3 and 4. The results are comparable to the affine transformation that we seek to find, see Table 1. It is clear that with fewer points, i.e., mini-batch size $M = 100$ and $M = 1000$, the registration was still successful.

In the next experiment, we compare two registration solutions. The first solution computes the distance map using the fast sweeping approach whereas the second solution uses the brute force method to compute the distance values for each point in M. We used $M = 100$ but the same one is also applicable to $M = 1000$. The results in Tables 5, 6 and 6 show that our new approach

gives comparable results to the affine transformation that we seek to find, see Table 1. They justify the use of the brute force method. We have only used few points, $M = 100$, instead of computing the distance map in the whole best fitting bounding box.

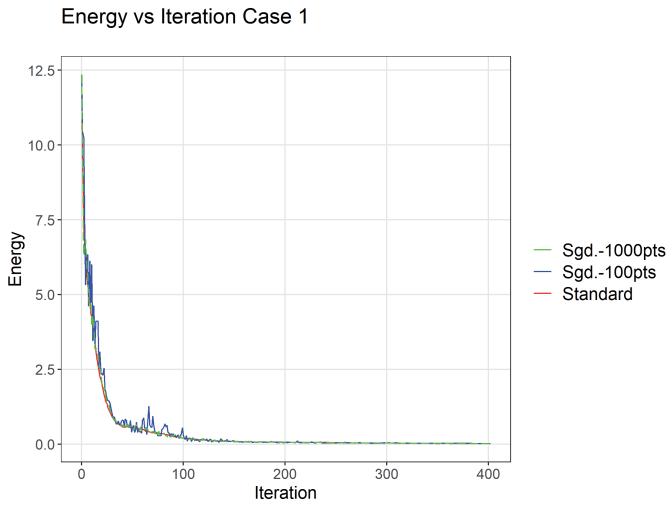

FIGURE 5. The case of bladder 1. Sgd. is an acronym for the stochastic gradient descent.

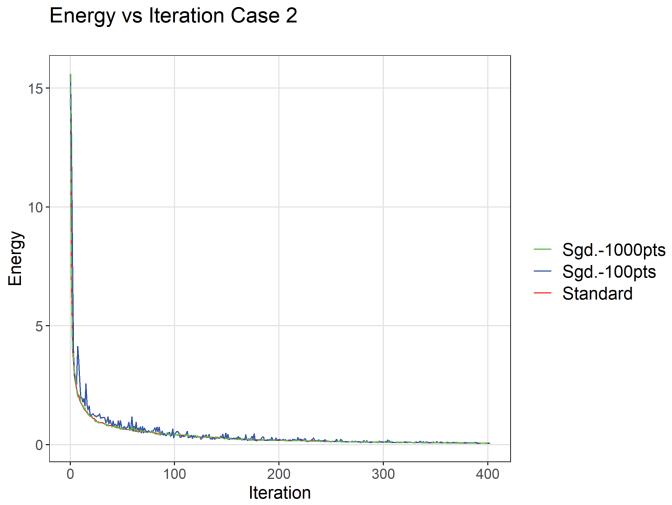

FIGURE 6. The case of bladder 2. Sgd. is an acronym for the stochastic gradient descent.

Similar to Figures 5, 6 and 7, we have the comparison of minimized $E(\mathbf{A})$, **SSD/Energy** done for both approaches. The results of Sgd.+BF and Sgd.+FSM methods are comparable. See Figures 8, 9 and 10 comparing the convergence of Sgd.+BF and Sgd.+FSM methods. Sgd.+BF refers to our new approach using brute force method to calculate the distance values for the M points to the interface. Sgd. + FSM refers to the original implementation which calculate the distance map inside the best fitting bounding box.

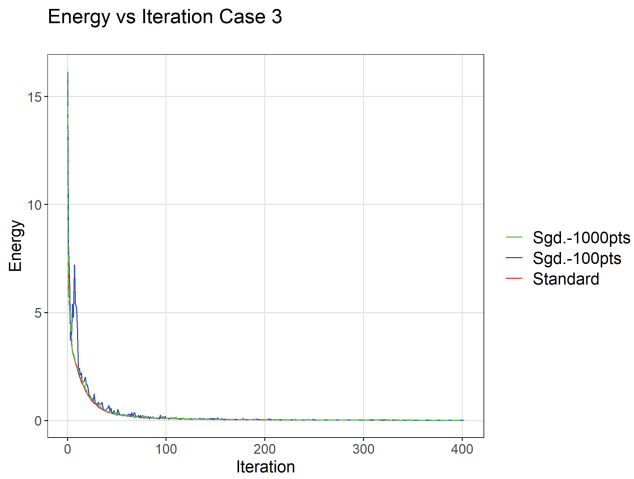

FIGURE 7. The case of bladder 3. Sgd. is an acronym for the stochastic gradient descent.

TABLE 2. Resultant transformation parameters after registration: The case of bladder 1. Sgd. is an acronym for the stochastic gradient descent.

	Rotation			Scaling			Translation		
	ϕ	θ	ψ	s_x	s_y	s_z	t_x	t_y	t_z
Standard	-0.175	0.173	0.352	0.799	1.103	1.253	2.996	5.039	-6.980
Sgd. - 100pts	-0.176	0.175	0.349	0.799	1.103	1.252	2.998	5.038	-6.985
Sgd. - 1000pts	-0.175	0.173	0.351	0.799	1.102	1.252	3.003	5.039	-6.980

TABLE 3. Resultant transformation parameters after registration: The case of bladder 2. Sgd. is an acronym for the stochastic gradient descent.

	Rotation			Scaling			Translation		
	ϕ	θ	ψ	s_x	s_y	s_z	t_x	t_y	t_z
Standard	-0.175	0.181	0.340	0.807	1.110	1.237	3.015	4.883	-7.067
Sgd. - 100pts	-0.174	0.180	0.345	0.809	1.097	1.239	3.014	4.899	-7.060
Sgd. - 1000pts	-0.175	0.182	0.339	0.808	1.098	1.238	3.009	4.883	-7.063

TABLE 4. Resultant transformation parameters after registration: The case of bladder 3. Sgd. is an acronym for the stochastic gradient descent.

	Rotation			Scaling			Translation		
	ϕ	θ	ψ	s_x	s_y	s_z	t_x	t_y	t_z
Standard	-0.175	0.175	0.349	0.800	1.101	1.251	3.001	5.003	-7.000
Sgd. - 100pts	-0.173	0.174	0.348	0.800	1.102	1.251	3.011	5.007	-6.996
Sgd. - 1000pts	-0.174	0.174	0.349	0.800	1.101	1.251	3.011	5.007	-6.994

TABLE 5. Resultant transformation parameters after registration: Case of bladder 1.

	Rotation			Scaling			Translation		
Sgd. - 100 pts	ϕ	θ	ψ	s_x	s_y	s_z	t_x	t_y	t_z
Fast Sweeping	-0.176	0.175	0.349	0.799	1.103	1.252	2.999	5.038	-6.985
Brute Force	-0.177	0.173	0.351	0.783	1.086	1.198	3.072	5.097	-7.285

TABLE 6. Resultant transformation parameters after registration: Case of bladder 2.

	Rotation			Scaling			Translation		
Sgd. - 100 pts	ϕ	θ	ψ	s_x	s_y	s_z	t_x	t_y	t_z
Fast Sweeping	-0.174	0.180	0.345	0.809	1.097	1.239	3.014	4.899	-7.060
Brute Force	-0.186	0.190	0.353	0.796	1.082	1.165	2.786	4.876	-7.482

TABLE 7. Resultant transformation parameters after registration: Case of bladder 3.

	Rotation			Scaling			Translation		
Sgd. - 100 pts	ϕ	θ	ψ	s_x	s_y	s_z	t_x	t_y	t_z
Fast Sweeping	-0.173	0.174	0.348	0.800	1.102	1.251	3.011	5.007	-6.996
Brute Force	-0.185	0.189	0.362	0.786	1.087	1.164	2.908	5.032	-7.497

TABLE 8. Running times (s) of critical registration parts: Case of bladder 1.

	Distance (s)	Transformation (s)	Others (s)	Total Reg. Time (s)
test 0	0.353	1.95699	0.17401	2.484
test 1	0.098	2.06000	0.18300	2.341
test 2	0.106	1.11500	0.18700	1.408

In the final experiment, we tested the improvements brought by parallelization. We chose to parallelize using the OpenMP [7] application programming interface (API). The computer we tested on had 6 cores (12 Threads), Intel® Core™ i7-5820K [5]. We implemented the following:

(1) Calculation of the distance value for each M point was performed using the sequential brute force method. The result when using this brute force method have been shown in Tables 5, 6 and 6. The recorded times are saved under **test 0** row in Table 8 and represented by **Sgd. + Bf** bar in Figure 11.

(2) The calculation of the distance value for each M point was done using the sequential brute force method as the previous step (1). The improvement was that we parallelize the process, so that each of the calculation was running on a separate thread. Using 4 threads, we were able to reduce the calculation from M times to approximately $M/4$ times. The recorded times are saved under **test 1** row in Table 8 and represented by **test 1** bar in Figure 11.

(3) Building on the previous step (2) improvements, we also parallelized the transformation function with an additional 2 threads. The recorded times are saved under **test 2** row in Table 8 and represented by **test 2** bar in Figure 11.

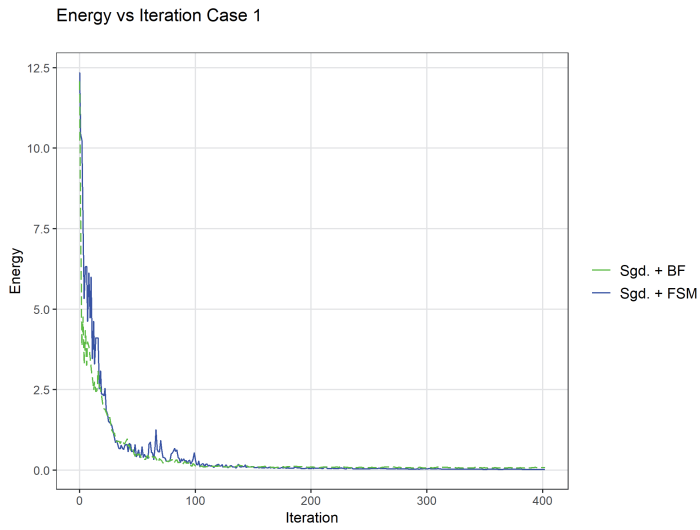

FIGURE 8. The case of bladder 1, where Sgd. is an acronym for the stochastic gradient descent.

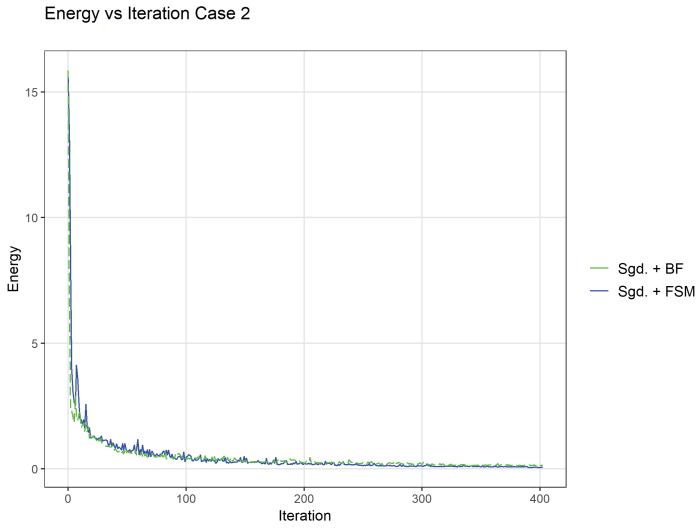

FIGURE 9. The case of bladder 2, where Sgd. is an acronym for the stochastic gradient descent.

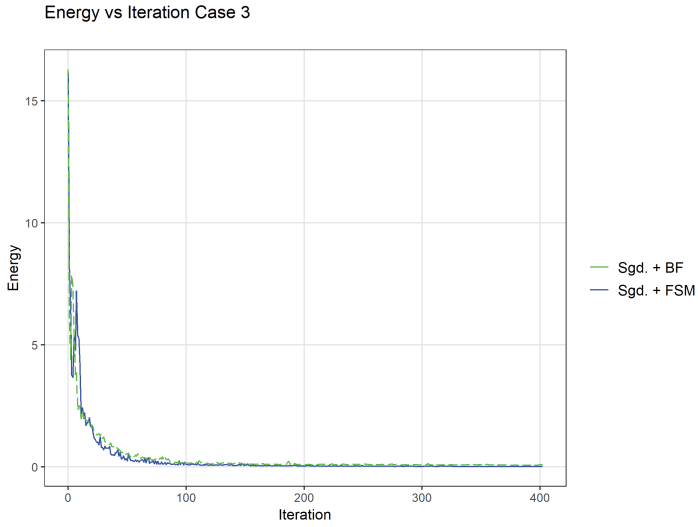

FIGURE 10. The case of bladder 3, where Sgd. is an acronym for the stochastic gradient descent.

A total of 4 threads was used for parallelization of the brute force method and 2 threads were used in parallelizing affine transformation of the moving shape. It was not possible to use all the threads since there were other running system processes. In addition, creating or using more threads can lead to an increase in overhead.

From Table 8, it is clear that the total registration time reduced after parallelization of the process. The other columns represent the calculation time for running other tasks during the registration process. The total height of the each bar in Figure 11 represents the total registration time. The optimal time was when we performed **test 2**, see (3) for the description. With just 4 threads for distance computation and 2 threads for applying transformation, we were able to achieve lower computation time. We expect that these times will reduce significantly if more threads are used when using a computer with more cores/threads.

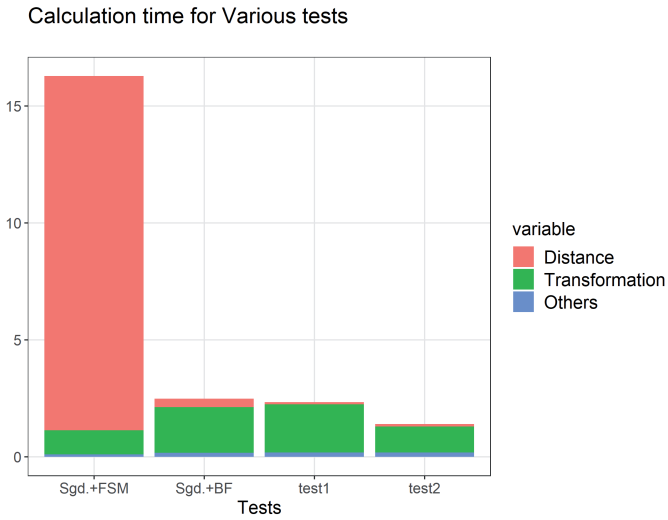

FIGURE 11. Case of bladder 1.

We compared the above results to the original implementation. In the original implementation, fast sweeping approach was used to calculate the distance map inside the whole best fit bounding box. The registration results are shown in Tables 5, 6 and 6. The improvements in registration times are significantly higher when compared to this approach. This original implementation is represented by the **Sgd. + FSM** bar in Figure 11.

Figure 11 shows the difference between our original approach and the approach utilizing the brute force method which was also parallelized. Similar results are obtained for the other human bladders cases.

6. Conclusion

In this article, we used the gradient descent methods for the optimization to perform registration of shapes, $\mathbf{F}, \mathbf{M} \subset \mathcal{R}^3$ fixed and moving shapes, respectively. Our objective was to find a 3D affine transformation matrix $\mathbf{A}$ that maps $\mathbf{M}$ to $\mathbf{F}$.

In our experiments, we have shown that we can perform successful registration using both the standard and stochastic gradient descent methods. In addition, our new approach of using the brute force method to compute distance values for M points gave comparable results. We also performed computational optimization by applying parallelization and achieved significantly shorter the total registration time.

In future release, we plan to parallelize the methods using CUDA [1]. In addition, other computational optimization techniques to be considered include downsampling of the shapes and performing registration to the downsampled shapes. Also, we plan to consider using some of the interface points instead of all points in the interface. The selection of interface points will be random.

We used the ImageInLib [13], an open source image processing library that also contain the Wikis explaining how to use the it. This library is completely free and supports 3D image processing functionality. The article wiki page [16] is also available.

Acknowledgement. We thank TatraMed Software s.r.o. for technical support and to ImageInLife for funding.

REFERENCES

[1] COOK, S.: *CUDA Programming: A Developer's Guide to Parallel Computing with GPUs*. Applications of GPU Computing Series. Elsevier Science, London, 2012.

[2] COOTES, T.—TAYLOR, C.—COOPER, D.—GRAHAM, J.: *Active Shape Models-Their Training and Application*, Computer Vision and Image Understanding **61** (1995), 38–59.

[3] FOMEL, S.: *Traveltime Computation with the Linearized Eikonal Equation*, Report, Sep-94, 1997, 123–131.

[4] HYSING, S.—TUREK, S.: *The Eikonal equation: numerical efficiency vs. algorithmic complexity on quadrilateral grids*. In: *Proceedings of the Algorythmy 2005*, pp. 22–31.

[5] INTEL®: . *Intel® Core^{TM} i7-5820K Processor (15M Cache, up to 3.60 GHz) Product Specifications*. `https://ark.intel.com/content/www/us/en/ark/products/82932/intel-core-i7-5820k-processor-15m-cache-up-to-3-60-ghz.html`.

[6] KASS, M.—WITKIN, A.—TERZOPOULOS, D.: *Snakes: Active contour models*, Int. J. Comput. Vision **1** (1988), 321–331.

[7] KLEMM, M.—DE SUPINSKI, B.—BOARD, T.: *OpenMP Application Programming Interface Specification Version 5.0*. Independently Published, 2018.

[8] LI, M.—ZHANG, T.—CHEN, Y.—SMOLA, A. J.: *Efficient mini-batch training for stochastic optimization*. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, ACCM, New York, NY, 2014. pp. 661–670.

[9] MARKOVSKY, I.—MAHMOODI, S.: *Least-squares contour alignment*, IEEE Signal Processing Letters **16** (2009), 41–44.

[10] MARQUES, J. S.—ABRANTES, A. J.: *Shape alignment — optimal initial point and pose estimation*, Pattern Recognition Letters **18** (1997), 49–53.

[11] METEL, M. R.: *Mini-batch stochastic gradient descent with dynamic sample sizes*, arXiv e-prints (2017), arXiv:1708.00555.

[12] MIKULA, K.—URBÁN, J.: *Fully automatic affine registration of planar parametric curves*, In: *Proceedings of the Conference Algoritmy 2016*, pp. 343–352.

[13] OKOCK, P. O.—JOZEF, U.—UBA, M. O.: *ImageInLib v1.0.0 Release*, February 15, 2019.

[14] PARAGIOS, N.—ROUSSON, M.—RAMESH, V.: *Matching Distance Functions: A Shape-to-Area Variational Approach for Global-to-Local Registration*. ECCV, Copenhangen, Denmark, 2002.

[15] PARAGIOS, N.—ROUSSON, M.—RAMESH, V.: *Matching Distance Functions: A Shape-to-Area Variational Approach for Global-to-Local Registration*. In: *Computer Vision––ECCV 2002* (A. Heyden, G. Sparr, M. Nielsen, P. Johansen, eds.), Springer-Verlag, Berlin, 2002, pp. 775–789,

[16] OKOCK, P.: *Efficient 3D shape registration using distance maps and stochastic gradient descent method*. `http://bit.ly/2LXmVLK`.

[17] ROBBINS, H.—MONRO, S.: *A stochastic approximation method*, Ann. Math. Stat. **22** (1951), 400–407.

[18] ROBBINS, H.—SIEGMUND, D.: *A Convergence Theorem For Non Negative Almost Supermartingales And Some Applications**Research supported by NIH Grant 5–R01–GM-16895–03 and ONR Grant N 00014-67-A-0108-0018.*. In: *Optimizing Methods in Statistics* (J. S. Rustagi, ed.), Academic Press, 1971. pp. 233–257.

[19] ROBBINS, H.—MONRO, S.: *A Stochastic Approximation Method*, Ann. Math. Statist. **22** (1951), 400–407.

[20] RUSTAGI, J.: *Optimizing Methods in Statistics: Proceedings*. Academic Press,1971.

[21] SETHIAN, J. A.: *Level Set Methods and Fast Marching Methods*. In:*Cambridge Monographs on Appl. Comput. Math. Vol. 3*, Cambridge: Cambridge University Press. xx, 1999.

[22] URBÁN, J.: *The New Improvements of Atlas Based Image Segmentations*. PhD Thesis, Slovak University of Technology in Bratislava, jozef.urban@gmail.com, 7, 2016.

[23] WIKIPEDIA CONTRIBUTORS: *Brute-force search — Wikipedia, The Free Encyclopedia*. 2019. [Online; accessed 9-June-2019]
`https://en.wikipedia.org/w/index.php?title=Brute-force_search&oldid=890487309`

[24] ZAHN, C. T.—ROSKIES, R. Z.: *Fourier descriptors for plane closed curves*, IEEE Trans. Comput. **C-21** (1972), 269–281.

[25] ZHAO, H.: *A fast sweeping method for eikonal equations*, Math. Comput. **74** (2005), no. 250, 603–627.

Received June 26, 2019

Omondi Polycarp Okock
TatraMed Software s.r.o
Líščie údolie 9
841 04 Bratislava
SLOVAKIA
E-mail: polycarp.okock@tatramed.sk
polycarp.okock@stuba.sk

Jozef Urbán
TatraMed Software s.r.o
Líščie údolie 9
841 04 Bratislava
SLOVAKIA
E-mail: jozef.urban@tatramed.sk

Karol Mikula
Department of Mathematics and
Descriptive Geometry
Faculty of Civil Engineering
Slovak University of Technology
Radlinského 11
810 05 Bratislava
SLOVAKIA
E-mail: karol.mikula@stuba.sk