2D-3D motion registration of rigid objects within a soft tissue structure

Nargiza Djurabekova[a], Andrew Goldberg[a,b], David Hawkes[a], Guy Long[c], Felix Lucka[a,d], and Marta M. Betcke[a]

[a]Centre for Medical Image Computing (CMIC) at the Department of Computer Science, University College London (UCL)
[b]Royal National Orthopaedic Hospital, UCL
[c]CurveBeam Europe, Ltd
[d]Computational Imaging (CI) Group at the Centrum Wiskunde & Informatica (CWI)

ABSTRACT

The study of rigid body dynamics in soft tissue is an essential part of orthopaedic imaging. Our focus is the foot and ankle structure, which consists of 28 bones surrounded by a variety of soft tissue such as 30 muscles, numerous ligaments, bursae and nerves. The importance of understanding the involved dynamics is evident by the frequent need to find functional joint replacements to fight diseases such as hindfoot arthritis. We study a simplified problem by constructing a phantom with two "bones" submerged in silicone and a remotely controlled LEGO robot that moves one of the two bones.

We perform motion registration by manipulating the bone segmented from an initial static scan and matching its digitally reconstructed radiographs to a sequence of scanned X-ray projections. The registration of the 3D volume to 2D projections requires the right combination of optimization and interpolation to achieve an optimal result. We test four different approaches with the help of the Flexible Algorithms for Image Registration (FAIR) toolbox.

Keywords: 2D3D registration, dynamic X-ray CT, cone beam CT, rigid motion

1. INTRODUCTION

This project covers the approach to dynamic imaging of the foot and ankle through the lens of image registration, in particular volume to projection registration. We explore the parametric, bone-by-bone approach as we assume that the movement of the individual bones can be modelled through rigid transformation. Each bone is segmented out of the initial static scan and the volume containing one of the bones is matched to a sequence of fluoroscopic projections. We use the term fluoroscopy here to mean an "X-ray movie", i.e. photons shooting in quick succession to track the movement of the scanned object while the source, detector and the sample stage remain stationary. The registration is performed with the help of the Flexible Algorithms for Image Registration (FAIR) package[1] for MATLAB and to test this approach we use the phantom described in section 1.1.

1.1 Data acquisition

The data for this project was acquired at the CWI research institute in Amsterdam, using the state-of-the-art X-ray scanner called FleXray. During the acquisition, the FleXray is completely sealed and can only scan small (up to 10 cm tall) objects, so we constructed a remotely controlled base and a simple phantom consisting of two "bones" made from gypsum plaster submerged in some silicone. Silicone was chosen to reproduce the effect of cartilage and other soft tissue between and surrounding the bones.

The data was collected at 78 kV with maximum power due to the high attenuation of gypsum and silicone. To remove lower energy photons, layers of thin copper plates were attached to the source as can be seen in fig. 1. Static frame-by-frame data of 40 individual positions was acquired as well as data from continuously shooting X-ray beams (i.e. fluoroscopic data) from a few selected angles.

Further author information: (Send correspondence to N.D.)
N.D.: E-mail: nargiza.djurabekova.17@ucl.ac.uk

7th International Conference on Image Formation in X-Ray Computed Tomography,
edited by Joseph Webster Stayman, Proc. of SPIE Vol. 12304, 123042C
© 2022 SPIE · 0277-786X · doi: 10.1117/12.2647250

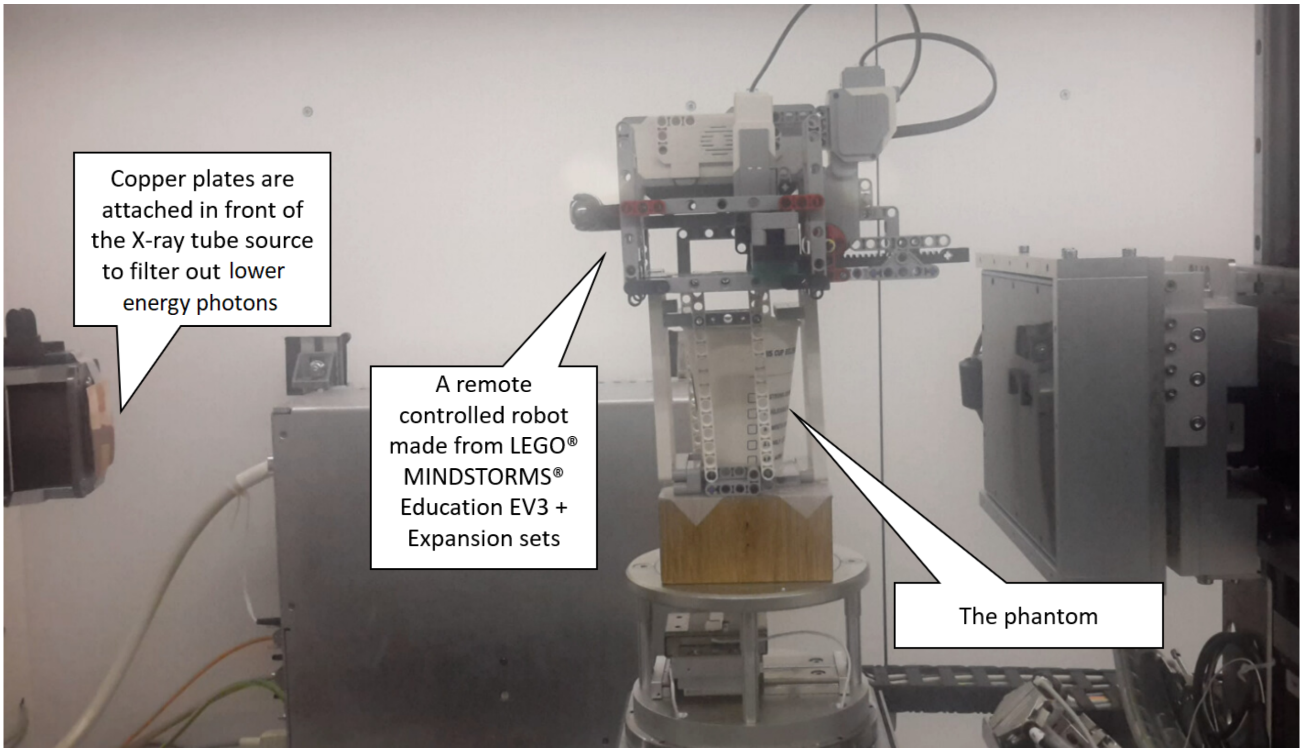
Copper plates are attached in front of the X-ray tube source to filter out lower energy photons
A remote controlled robot made from LEGO® MINDSTORMS® Education EV3 + Expansion sets
The phantom

Figure 1. The FleXray scanner set-up with the CWI phantom.

1.2 Image registration

Image registration is an umbrella term for aligning two or more sets of images. It is often used in image processing for combining images to either extend the view, such as in panoramic images, or to enhance an existing view and reduce signal-to-noise ratio. The purpose of image registration is often to identify similar features in different images in order to acquire further information about the features. This has a very clear application in medical imaging, where scans are often performed multiple times during treatment and sometimes even using different devices (e.g. MR and X-ray systems). To find out the exact differences between the images, it is crucial to align the images correctly, i.e. perform registration.

To avoid the need for experts to identify correct features in order to perform landmark-based registration, we instead choose to rely on intensity based registration. Intensity based methods align images by comparing their intensity values and contrasts, and often require little to no preparation in terms of marker placement etc., although they can be quite time consuming during the registration process. An example of a fully intensity based method can be found in[2] where individual vertebrae in a 3D CT volume were robustly registered to the corresponding vertebrae in a 2D X-ray image. The registration was performed by optimising for one of the six degrees of freedom of the 3D bone at a time. We propose a method in which we use the FAIR toolbox to optimise for all six transformation parameters simultaneously.

1.3 Practical background

Image registration can occur when both images are in the same domain. That is, both are in $\Omega \subset \mathbb{R}^d$, where d is the common dimension, usually 2 or 3. So for some reference image $u_R : \Omega \to \mathbb{R}$, we try to transform a template image $u_T : \Omega \to \mathbb{R}$ using the transformation $\rho : \mathbb{R}^d \to \mathbb{R}^d$ in a way that minimizes the following objective function:

$$\arg\min_w F(w) = dist(u_T(\rho(w,x)), u_R(x)) + \mathcal{R}(w). \tag{1}$$

Here, $dist$ is some distance measure function and $\mathcal{R}$ is a regularizer, $x \in \Omega$ is the set of coordinates and w is the vector of parameters for the transformation ρ. In our case, the template is a 3D bone s segmented out of the whole image u_T and the reference is a projection (or a set of projections) g_R. Since the registration is to be performed between a 3D volume s and 2D measurement data g_R, one must first generate digitally reconstructed radiographs (DRRs) from the 3D volume, and then perform the registration between the DRR(s) and given X-ray measurement data. Let $s : \Omega \to \mathbb{R}$ be the template image and $g_R : \Omega' \to \mathbb{R}$ be a reference image. Domain Ω' is a set of one or more stacked X-ray projections from different angles, $\Omega' \subset \mathbb{R}^{d-1} \times [0, 2\pi)$. Let also A_θ be the X-ray cone-beam forward operator, where θ is the set of indices representing the present projection angles. Then for 2D3D registration, the eq. 1 becomes

$$\arg\min_w F(w) = dist(A_\theta(s(\rho(w,x))), g_R(x)) + \mathcal{R}(w).$$

As we are working with rigid bodies, our parameter vector w is going to consist of 6 elements corresponding to 6 degrees of freedom, 3 for rotation w_1, w_2, w_3 and 3 for translation w_4, w_5, w_6.

2. NUMERICAL REALIZATION AND RESULTS

Recall that the objective function we want to minimize is

$$\arg\min_w F(w) = dist(A_\theta(s(\rho(w,x))), g_R(x)) + \mathcal{R}(w). \tag{2}$$

We set the distance function to be the squared ℓ^2 norm, also known as the sum of squared differences, which we refer to as ψ. So if the residual $r = A_\theta(s(\rho(w,x))) - g_R(x)$, then the $dist$ function is

$$\begin{aligned} dist(A_\theta(s(\rho(w,x))), g_R(x)) = \psi(r) &= \frac{1}{2}||r||_2^2 \\ &= \frac{1}{2}||A_\theta(s(\rho(w,x))) - g_R(x)||_2^2. \end{aligned}$$

In the discrete setting, where the image is defined on a grid of cells, $\psi(r)$ is multiplied by a scaling factor h which is defined as the cell width. This is particularly useful for the multiscale optimization, where different levels/scales correspond to different grid densities.

As a constraint on transformation parameters w, we use Tikhonov's regularization, i.e. given a reference set of parameters w^{ref},

$$\mathcal{R}(w) = \frac{1}{2}(w - w^{ref})^\intercal M (w - w^{ref}),$$

where M is a diagonal matrix of weights for the transformation parameters.

2.1 Pre-processing the data

For the two-dimensional reference images, one or two projection images are used for each of the 41 static scans (one initial position + 40 poses along a programmed motion path). The template image is obtained by reconstructing the static scan in initial position and segmenting it with K-means[3] into 3 separate masks - air, silicone (as well as lego) and gypsum, corresponding to air, soft tissue and bone. Noise from the reconstruction u_T is smoothed by a convolution with a Gaussian filter. The bone mask is split into two masks corresponding to the two shapes coded as the m_{tibia} (top) and the m_{talus} (bottom). These bones are then segmented out of the reconstruction

$$s_{bones} = (m_{tibia} + m_{talus}) \odot u_T$$

by means of element-wise multiplication ($\odot$) to obtain a background volume which can be projected with the forward operator A_θ into the data domain

$$g_{bg} = A_\theta(u_T - s_{bones}).$$

These background projections can then be removed from all following reference images to give the best chance to our intensity based registration algorithm,

$$g_{bones} = g - g_{bg}.$$

The masks, segmented bones and an example of a projected background for the static initial position can be found in fig. 2.

As the bones move further away from the initial position, the background extracted from it becomes a worse approximation to the true background due to decreasing overlap of the occlusion in the initial position with the occlusion in the present frame. However, as this only affects a small part of the background volume, there is still a benefit to subtracting the background.

2.2 Results

We run the registration problem on all 40 frames + the first initial position using 4 different strategies. The registration itself is performed using Gauss-Newton method and the template image s_1 is always the bone (in our case tibia, or s_{tibia}) in the initial position, while w_1^{ref} is set to $w_0 = [0, 0, 0, 0, 0, 0]^\intercal$.

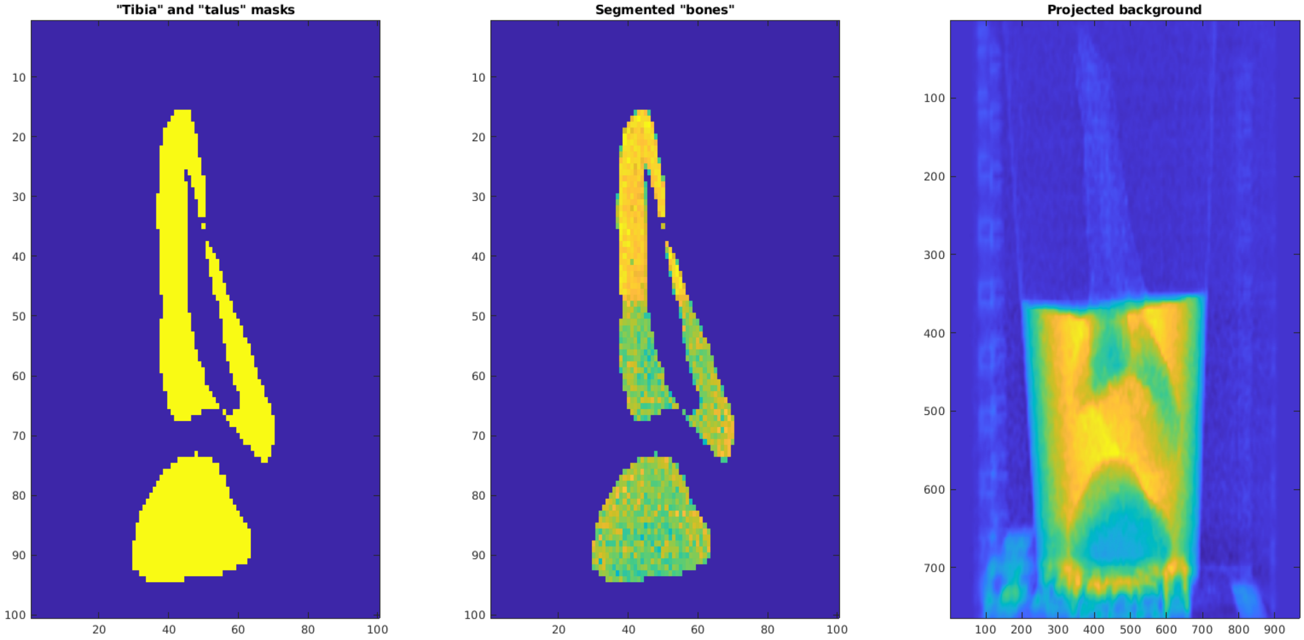

Figure 2. Demonstration of data pre-processing. *Left:* A slice through the bone masks volume, top: m_{tibia}, bottom: m_{talus}; *Middle:* A slice through the segmented bones volume s_{bones} acquired by applying the bone masks to the full reconstruction of the initial pose; *Right:* A simulated projection of the "background", g_{bg}.

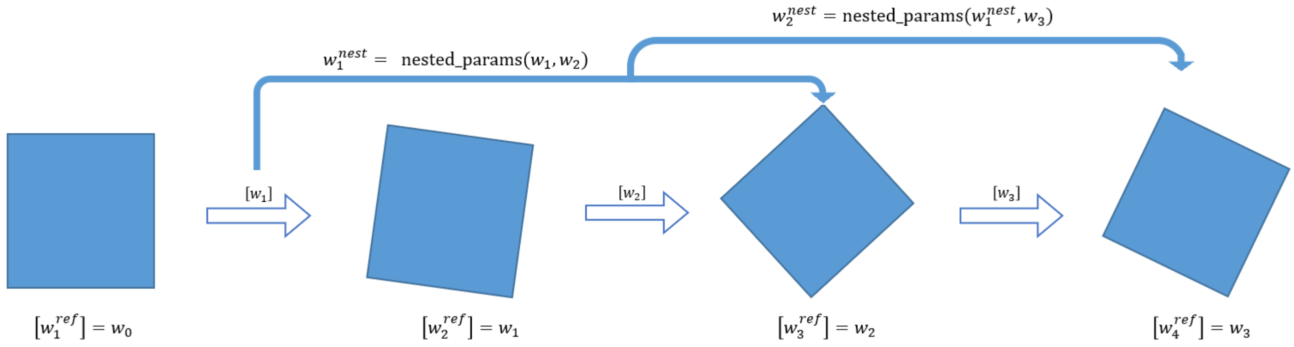

Figure 3. Diagram of parameter aggregation via the `nested_parameters` function.

Setup 1.

For each consecutive frame to be registered (iteration $k+1$), the GN method is warm started by setting w^{ref} to the parameters w_k obtained from the registration of the previous frame k

$$w^{ref}_{k+1} = w_k$$

Meanwhile the template image s for a consecutive frame registration is updated based on the previous iteration's output parameters, using rigid transformation T_{rigid} and spline interpolation $\mathcal{T}_{SI}$, so

$$s_{k+1} = \mathcal{T}_{SI}(T_{rigid}(s_k, w_{k+1})).$$

To simplify notation, we combine the rigid transformation and the interpolation into one transformation operator T. The updated s_{k+1} is used as a template image for the following iteration. Results for setup 1 are visualized in fig. 4(a).

Setup 2.

In this setup, we obtain the template image s_k (for $k > 1$) directly from s_1 via rigid transformation with parameters w_k, while the reference parameters w^{ref} for the optimization stay the same for all iterations. Then in every iteration we obtain global transformation parameters as opposed to setup 1, where the parameters are computed locally. For every iteration $k = 1, 2, ..., 40$,

$$s_{k+1} = T(s_1, w_{k+1}).$$

Note that w^{ref} and s_1 remain unchanged through all iterations. Results for setup 2 are visualized in fig. 4(b).

Setup 3.

As in setup 2, except that the registration is now performed using the coarse-to-fine multiscale approach for registration with the goal to register over larger distances. Results for setup 3 are visualized in fig. 4(c).

Setup 4.

In this setup we propose a scheme to aggregate the optimized parameters and use those along with the s_1. As can be seen in fig. 3 this scheme aggregates incremental transformations $w_1, ..., w_k$. In practice, this is performed stepwise using the `nested_parameters` function described in appendix A. This way, interpolation is applied only once to s_1 to move it into position of frame k. The variables for this setup are updated as follows:

For $k = 0$,

$$w^{ref}_{k+1} = w^{ref}_1$$

$$w_{k+1} = \mathrm{GN}(w^{ref}_{k+1})$$

$$w^{nest}_{k+1} = w_{k+1}$$

$$s_{k+1} = T(s_1, w^{nest}_{k+1})$$

For $k = 1, 2, 3, ..., 40$

$$w^{ref}_{k+1} = w_k$$

$$w_{k+1} = \mathrm{GN}(w^{ref}_{k+1})$$

$$w^{nest}_{k+1} = \texttt{nested\_parameters}(w^{nest}_k, w_{k+1})$$

$$s_{k+1} = T(s_1, w^{nest}_{k+1})$$

The updated template s is used as initialization for the following iterations. Results for setup 4 are visualized in fig. 4(d).

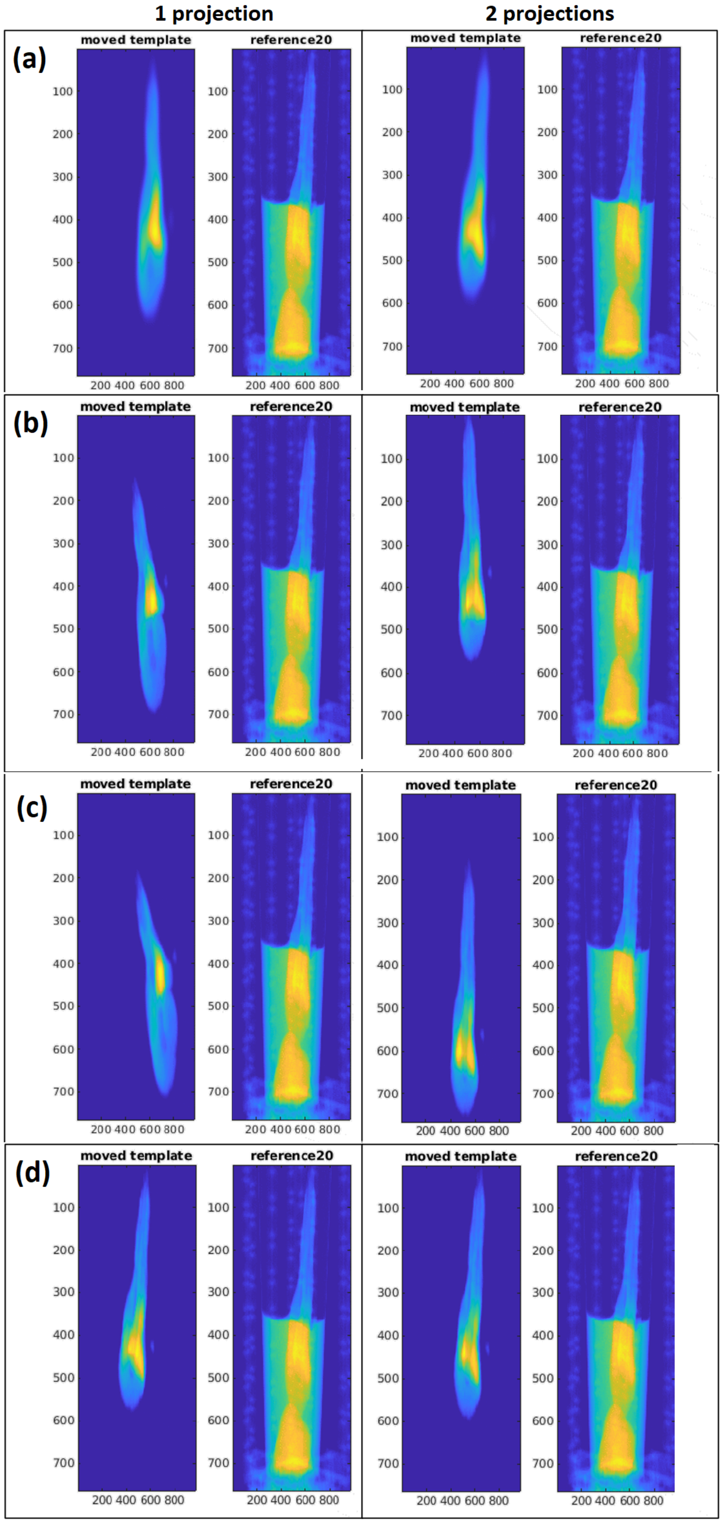

Figure 4. Demonstration of the four different setups. Registration performed with 1 projection at angle 45° (Left) and with 2 projections at angles 0° and 90° (Right). Rows (a) to (d) correspond to registration results for frame 20 for setups 1 to 4 respectively.

3. DISCUSSION

In this work we consider the 2D3D parametric image registration between fluoroscopic X-ray data and a segmented volume. The registration is performed with spline interpolation and Gauss-Newton optimization as implemented in the FAIR toolbox,[1] adjusted for our 2D3D problem. We create an objective function eq. 2, which we optimize for rigid transformation parameters that would align the template bone with the reference projections. Gauss-Newton algorithm functions as the optimizer. The main difficulty arises with the initialization, so we test out four different setups.

The first setup involves initializing the optimization of a consecutive frame with a previous frame's transformed template and optimal parameters. This leads to very blurry results due to what amounts to recursive interpolation (see fig. 4(a)). To avoid this, the second setup initializes the rigid transformation with all parameters w set to 0 to directly compute the transformation from the initial position to the currently processed timestep. However, as the bone moves further away from its initial position, the results become consistently worse. And then as the bone moves back, closer to the initial position, the algorithm is able to recover and realign the bones (see fig. 4(b)). To try and make up for the second setup's inability to account for large displacements, we repeat its initialization in setup 3 but now the registration is performed with a coarse-to-fine grid approach. As can be seen from fig. 4(c), though, this multiscale approach does not lead to better results. This is likely due to the presence of background elements in the reference image(s). In particular, the presence of the other bone causes the template to be aligned with it, as can be most clearly seen in fig. 4(c) with 2 projections. Finally, for setup 4, we propose initializing the Gauss-Newton optimization with the previously optimised parameters and updating the template image from the initial position with aggregated parameters. These aggregated parameters come from nesting previous transformations together to achieve one single transformation (using `nested_parameters` function, appendix A). In fig. 4(d), we can see that this approach does not manage to perform correct registration with just one projection, since if it wrongly estimates in one frame, the mistake only gets amplified with each frame and the algorithm can no longer recover. The moving bone can even end up completely out of the visible region of interest in the final few frames. However, two orthogonally directed projections (at 0° and 90°) constrain the problem enough to eliminate the ambiguity between motion and scaling, leading to promising results depicted in fig. 4(d) on the right, with the transformed template bone tracking the true motion quite accurately.

ACKNOWLEDGEMENTS

The authors would like to thank NVIDIA Corporation for the GeForce Titan Xp GPU used to create the reconstructions presented here.

APPENDIX A. NESTED_PARAMETERS FUNCTION

The `nested_parameters` function is a function that aggregates previous registration parameters to move a rigid object from the initial position at $t = 0$ to all the positions in the following timeframes. The nesting or aggregation of parameters is derived as follows. Recall that rigid transformation ρ on x can be expressed as

$$\rho = R(x) + \tau_{ext} = R(x) + (I - R)c + \tau, \tag{3}$$

where $\tau_{ext} = [\tau^1_{ext}, \tau^2_{ext}, \tau^3_{ext}]^\intercal$ is the extended translation vector that contains the shift to the centre of the domain c, rotation Rc, and shift back, as well as the actual translation vector $\tau = [\tau^1, \tau^2, \tau^3]^\intercal$. This extended translation vector $\tau_{ext} = (I - R)c + \tau$ corresponds to the last three elements of the rigid transformation parameters vector w. The first three parameters of w are the angles of rotation α, β and γ (with respect to the axes x, y and z), which are used to calculate the corresponding rotation matrix R. We want to be able to perform the transformation 3 recursively to compute the effective rotation R_{nest} and translation τ_{nest} of two timeframes together. Let R_1 and τ_{ext} refer to the rotation and translation performed on the first timeframe and R_2 and τ'_{ext} on the second one. Then the effective transformation ρ' can be computed with the nested rotation matrix R_{nest} and nested translation vector τ_{nest} as follows

$$\rho' = R_{nest}(\rho) + \tau_{nest} = R_2(R_1(x) + \tau_{ext}) + \tau'_{ext}.$$

Computing the translation parameters of τ_{nest} is straightforward

$$\tau_{nest} = R_2(\tau_{ext}) + \tau'_{ext},$$

while finding the rotation angles a, b and c of R_{nest} is a little more involved. We know that as a 3D rotation matrix, R_{nest} has the form

$$R_{nest} = \begin{bmatrix} \cos a \cos b & \cos a \sin b \sin c - \sin a \cos c & \cos a \sin b \cos c + \sin a \sin c \\ \sin a \cos b & \sin a \sin b \sin c + \cos a \cos c & \sin a \sin b \cos c - \cos a \sin c \\ -\sin b & \cos b \sin c & \cos b \cos c \end{bmatrix}.$$

To solve for the rotation parameters (a, b, and c), we consider the following equation

$$\begin{aligned} R_{nest} &= R_2 R_1 \\ &= \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix} \times \\ &\begin{bmatrix} \cos\alpha'\cos\beta' & \cos\alpha'\sin\beta'\sin\gamma' - \sin\alpha'\cos\gamma' & \cos\alpha'\sin\beta'\cos\gamma' + \sin\alpha'\sin\gamma' \\ \sin\alpha'\cos\beta' & \sin\alpha'\sin\beta'\sin\gamma' + \cos\alpha'\cos\gamma' & \sin\alpha'\sin\beta'\cos\gamma' - \cos\alpha'\sin\gamma' \\ -\sin\beta' & \cos\beta'\sin\gamma' & \cos\beta'\cos\gamma' \end{bmatrix}. \end{aligned}$$

To make the calculation simpler to follow, we shall replace $\cos\alpha, \cos\beta$ and $\cos\gamma$ by $c(1)$, $c(2)$ and $c(3)$. For $\sin\alpha$, $\sin\beta$, $\sin\gamma$ we have $s(1)$, $s(2)$, $s(3)$ and analogously for the α', β', γ' parameters, their sine and cosine functions turn into $s'(1)$, $s'(2)$, $s'(3)$, and $c'(1), c'(2), c'(3)$. This gives us

$$\begin{aligned} R_{nest} = R_2 R_1 &= \begin{bmatrix} c(1)c(2) & c(1)s(2)s(3) - s(1)c(3) & c(1)s(2)c(3) + s(1)s(3) \\ s(1)c(2) & s(1)s(2)s(3) + c(1)c(3) & s(1)s(2)c(3) - c(1)s(3) \\ -s(2) & c(2)s(3) & c(2)c(3) \end{bmatrix} \\ &\times \begin{bmatrix} c'(1)c'(2) & c'(1)s'(2)s'(3) - s'(1)c'(3) & c'(1)s'(2)c'(3) + s'(1)s'(3) \\ s'(1)c'(2) & s'(1)s'(2)s'(3) + c'(1)c'(3) & s'(1)s'(2)c'(3) - c'(1)s'(3) \\ -s'(2) & c'(2)s'(3) & c'(2)c'(3) \end{bmatrix} \\ &= \begin{bmatrix} R_{11} & R_{21} & R_{31} \\ R_{12} & R_{22} & R_{32} \\ R_{13} & R_{23} & R_{33} \end{bmatrix}, \end{aligned}$$

where

$$
\begin{aligned}
R_{11} = {} & c(1)c(2)(c'(1)c'(2)) + (c(1)s(2)s(3) - s(1)c(3))(s'(1)c'(2) \\
& - (c(1)s(2)c(3) + s(1)s(3))s'(2), \\
R_{21} = {} & c(1)c(2)(c'(1)s'(2)s'(3) - s'(1)c'(3)) + (c(1)s(2)s(3) - s(1)c(3))(s'(1)s'(2)s'(3) \\
& + c'(1)c'(3)) + (c(1)s(2)c(3) + s(1)s(3))(c'(2)s'(3)), \\
R_{31} = {} & c(1)c(2)(c'(1)s'(2)c'(3) + s'(1)s'(3)) + (c(1)s(2)s(3) \\
& - s(1)c(3))(s'(1)s'(2)c'(3) - c'(1)s'(3)) + (c(1)s(2)c(3) + s(1)s(3))(c'(2)c'(3)), \\
R_{12} = {} & s(1)c(2)(c'(1)c'(2)) + (s(1)s(2)s(3) + c(1)c(3))(s'(1)c'(2)) \\
& - (s(1)s(2)c(3) - c(1)s(3))s'(2), \\
R_{22} = {} & s(1)c(2)(c'(1)s'(2)s'(3) - s'(1)c'(3)) + (s(1)s(2)s(3) \\
& + c(1)c(3))(s'(1)s'(2)s'(3) + c'(1)c'(3)) + (s(1)s(2)c(3) - c(1)s(3))(c'(2)s'(3)), \\
R_{32} = {} & s(1)c(2)(c'(1)s'(2)c'(3) + s'(1)s'(3)) + (s(1)s(2)s(3) \\
& + c(1)c(3))(s'(1)s'(2)c'(3) - c'(1)s'(3)) + (s(1)s(2)c(3) - c(1)s(3))(c'(2)c'(3)), \\
R_{13} = {} & -s(2)(c'(1)c'(2)) + (c(2)s(3))(s'(1)c'(2)) - (c(2)c(3))s'(2), \\
R_{23} = {} & -s(2)(c'(1)s'(2)s'(3) - s'(1)c'(3)) + (c(2)s(3))(s'(1)s'(2)s'(3) \\
& + c'(1)c'(3)) + (c(2)c(3))(c'(2)s'(3)), \\
R_{33} = {} & -s(2)(c'(1)s'(2)c'(3) + s'(1)s'(3)) + (c(2)s(3))(s'(1)s'(2)c'(3) \\
& - c'(1)s'(3)) + (c(2)c(3))(c'(2)c'(3)).
\end{aligned}
$$

Then the rotation parameters a, b, c are:

$$
\begin{aligned}
b = \arcsin(-R_{13}) = {} & \arcsin(s(2)(c'(1)c'(2)) - (c(2)s(3))(s'(1)c'(2)) \\
& + (c(2)c(3))s'(2)),
\end{aligned}
$$

$$
\begin{aligned}
a = \arcsin(R_{12}/\cos b) = {} & \arcsin((s(1)c(2)(c'(1)c'(2)) + (s(1)s(2)s(3) \\
& + c(1)c(3))(s'(1)c'(2)) - (s(1)s(2)c(3) - c(1)s(3))s'(2))/\cos b),
\end{aligned}
$$

$$
\begin{aligned}
c = \arcsin(R_{23}/\cos b) = {} & \arcsin((-s(2)(c'(1)s'(2)s'(3) - s'(1)c'(3)) \\
& + (c(2)s(3))(s'(1)s'(2)s'(3) + c'(1)c'(3)) + (c(2)c(3))(c'(2)s'(3)))/\cos b).
\end{aligned}
$$

So the whole nested transformation vector w_{nest} is then defined as

$$
w_{nest} = [w^1_{nest}, w^2_{nest}, w^3_{nest}, w^4_{nest}, w^5_{nest}, w^6_{nest}]^\intercal = [a, b, c, \tau^1_{ext}, \tau^2_{ext}, \tau^3_{ext}]^\intercal,
$$

and serves as the output to the `nested_parameters`(w, w') function, where w is the vector of transformation parameters of the first timeframe and w' of the second. The process can be repeated to compute the effective rotation and transformation for any number of steps, assuming w_{nest} is a good estimate of all the previous timeframes' transformations.

REFERENCES

[1] Modersitzki, J., [*FAIR: Flexible Algorithms for Image Registration*], SIAM, Philadelphia (2009).

[2] Penney, G., Batchelor, P., Hill, D., Hawkes, D., and Weese, J., "Validation of a two- to three-dimensional registration algorithm for aligning preoperative ct images and intraoperative fluroscopy images," *Medical physics* **28**, 1024–32 (07 2001).

[3] Jin, X. and Han, J., [*K-Means Clustering*], 563–564, Springer US, Boston, MA (2010).