HHS Public Access

Author manuscript

*Conf Comput Vis Pattern Recognit Workshops*. Author manuscript; available in PMC 2017 December 19.

Published in final edited form as:

*Conf Comput Vis Pattern Recognit Workshops*. 2016 ; 2016: 583–591. doi:10.1109/CVPRW.2016.79.

# Effects of Resolution and Registration Algorithm on the Accuracy of EPI vNavs for Real Time Head Motion Correction in MRI

**Yingzhuo Zhang**

Institute for Applied Computational Science and Engineering, Harvard John A. Paulson School of Engineering and Applied Sciences, Cambridge, Massachusetts, USA

**Iman Aganj**, **André J. W. van der Kouwe**, and **M. Dylan Tisdall**

Athinoula A. Martinos Center for Biomedical Imaging, Massachusetts General Hospital, Charlestown, Massachusetts, USA, Department of Radiology, Harvard Medical School, Boston, Massachusetts, USA

## Abstract

Low-resolution, EPI-based Volumetric Navigators (vNavs) have been used as a prospective motion-correction system in a variety of MRI neuroimaging pulse sequences. The use of low-resolution volumes represents a trade-off between motion tracking accuracy and acquisition time. However, this means that registration must be accurate on the order of 0.2 voxels or less to be effective for motion correction. While vNavs have shown promising results in clinical and research use, the choice of navigator and registration algorithm have not previously been systematically evaluated. In this work we experimentally evaluate the accuracy of vNavs, and possible design choices for future improvements to the system, using real human data. We acquired navigator volumes at three isotropic resolutions (6.4 mm, 8 mm, and 10 mm) with known rotations and translations. The vNavs were then rigidly registered using trilinear, tricubic, and cubic B-spline interpolation. We demonstrate a novel refactoring of the cubic B-spline algorithm that stores pre-computed coefficients to reduce the per-interpolation time to be identical to tricubic interpolation. Our results show that increasing vNav resolution improves registration accuracy, and that cubic B-splines provide the highest registration accuracy at all vNav resolutions. Our results also suggest that the time required by vNavs may be reduced by imaging at 10 mm resolution, without substantial cost in registration accuracy.

## 1. Introduction

Prospective motion detection and correction during an MRI scan has been shown to allow the acquisition of clinically useful images, even with substantial subject movement [7, 15]. While many methods have been developed to reduce the artifacts caused by motion during MRI scanning, Volumetric Navigators (vNavs) use the MRI scanner to track subject motion, requiring no additional equipment or setup be added to the scanner or workflow to enable motion correction [4, 12]. vNavs are low-resolution, whole-head volumes, acquired rapidly (~ 300 ms), and interspersed over the several minutes of a longer neuroimaging sequence. Head motion information is recovered from these volumes via registration, and the resulting estimates of head position are used to update the MRI scanner's imaging coordinates,

following the subject's head to compensate. vNavs are used in both clinical and neuroscientific studies to measure and correct motion.

While vNavs can be acquired frequently, they cost both acquisition and processing time which must be found in the MRI sequence being corrected (e.g. during pre-existing dead-times). The effectiveness of the vNavs system relies on the accuracy of the volume registrations, which can be affected both by the choice of vNav resolution, and the interpolation method used in the registration cost function. The lowest-resolution navigator that provides acceptable motion tracking would be preferred in practice due to it having the shortest duration. Previous work with vNavs [4, 12] has mostly used 8 mm isotropic resolution volumes, and relied on the PACE algorithm [10] for image registration on the scanner. However, there has been no systematic evaluation of the choice of resolution or registration algorithm.

In this work we address this gap, evaluating the registration accuracy of different interpolation methods at various resolutions of vNavs. Three interpolation methods are implemented and tested on the volumetric data: trilinear, tricubic [6], and cubic B-spline interpolation [14, 11], and a Gauss-Newton search algorithm was implemented to perform rigid registration using the 2-norm cost function. We have acquired data in a human volunteer at three feasible vNav resolutions: 6.4 mm, 8 mm, and 10 mm. Motions detected from these low-resolution vNavs need to be accurate enough for correcting high resolution MR imaging with voxels on the order of 1 mm. Comparing this with the resolutions of our vNavs, we need registration accuracy on the order of $\frac{1}{10}$ of a vNav voxel.

## 2. Methods

### 2.1. Data Acquisition

Imaging was performed on a 3 T TIM Trio (Siemens Healthcare, Erlangen, Germany) with all data acquired using the body coil to reduce spatial variations in signal intensity. One human volunteer, having given informed consent, was scanned with a custom pulse sequence that acquires a series of vNavs. Data was acquired at three isotropic resolutions: 6.4 mm, 8 mm, and 10 mm (acquisition parameters are shown in Table 1). To ensure the subject remained as still as possible during the scan, the acquisitions were broken into ~ 30 s sets, during which the volunteer was instructed to hold his breath to minimize respiratory motion. Each set consisted of a volume at iso-center and on-axis, followed by volumes with a range of rotations from either 0.5° to 2.5° or 3° to 5° at 0.5° increments, and at each rotation a series of 5 translations from 1 mm to 5 mm at 1 mm increments. Rotations were performed around x, y, and z axes, with z translations, and additional rotations around the oblique x/y, x/z, and y/z axes with x/y translations. Volumes within each set are registered to the first volume of the set. A total of 432 volumes per resolution were acquired, from which 420 pairs (one reference and one moved volume) can be extracted for registration.

Our previous experiments have shown that larger rotations/translations are easy to detect with any choice of resolution or interpolation method because there are enough change in the volume. We chose to consider small rotations/translations only on the assumption that

gross registration can be performed by a variety of methods. We instead are interested in the accuracy of registration near the ground truth.

## 2.2. Masking

When an image is rotated, the higher frequency components in the corners of the Fourier domain are aliased into lower-frequency regions. Applying a circular mask to filter out higher frequency components can reduce this effect [1]. In our work, masking was performed in both the Fourier and spatial domains to remove corners in the cubic volumes that cannot be extrapolated during rotation operations. The volumes were preprocessed before registration by applying a smoothed spherical mask in the Fourier domain. The mask was defined using the window function

$$m\left(r'\right) = \begin{cases} 1 & r' < \frac{3}{4} \\ w_{\cos}\left(\frac{8}{3}r' - 2\right) & \frac{3}{4} \leq r' \leq 1, \\ 0 & r' > 1 \end{cases} \quad (1)$$

where $r' = r/R$, $r$ is the radius from the image center, $R$ is the radius of the volume, and $w_{\cos}$ is the cosine window function, defined as

$$w_{\cos}(t) = \begin{cases} \cos(\pi t) & -\frac{1}{2} \leq t \leq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases}. \quad (2)$$

The same mask was applied again in the spatial domain, after the volumes were interpolated. Figure 1 shows sample slices prior to and after masking in three different resolutions.

## 2.3. Registration

**2.3.1 Cost function—**To perform prospective motion correction, navigator volumes acquired throughout the longer MRI scan would be registered to the first volume of the sequence, referred to as the *reference volume* $\mathbf{V}_{\text{ref}} : \mathbb{R}^3 \to \mathbb{R}$. Given a new *moving volume* $\mathbf{V}_{\text{mov}} : \mathbb{R}^3 \to \mathbb{R}$, we can describe our registration as minimizing the error function

$$\epsilon = \sum_i \left(\mathbf{v}_{\text{ref}}(\mathbf{x_i}) - \mathbf{V}_{\text{mov}}\left(\mathbf{x}_i + \mathbf{d}_i\left(\mathbf{P}^{-1}\right)\right)\right)^2, \quad (3)$$

where $\mathbf{v}_{\text{ref}}$ is the vector that represents the reference volume sampled at Cartesian grid points. $\mathbf{V}_{\text{mov}}(\mathbf{x}_i + \mathbf{d}_i(\mathbf{P}^{-1}))$ is the function describing the moving volume evaluated at an unmeasured point $\mathbf{x}_i + \mathbf{d}_i(\mathbf{P}^{-1})$, and $\mathbf{x}_i = (x_i, y_i, z_i)$ is the $i^{\text{th}}$ grid point of the volume. $\mathbf{d}_i(\mathbf{P}^{-1})$ is the estimated displacement of $\mathbf{x}_i$ from $\mathbf{V}_{\text{mov}}$ to $\mathbf{V}_{\text{ref}}$ given the set of transformation parameters $\mathbf{P}^{-1}$.

Our goal is real time registration, with total run-times on the order of tens of milliseconds. Given that we expect our volumes to have identical contrast and our noise variation to be

negligible (due to the large voxels being used), we have chosen to use the 2-norm for our cost function.

We note that (3) requires resampling every incoming volume during the registration step, which can be computationally expensive. Registering the reference volume to the incoming volumes produces the opposite transformation, but only the reference volume needs to be resampled every time [10]. This changes our error function to,

$$\epsilon = \sum_i \left(\mathbf{V}_{\text{ref}}\left(\mathbf{x}_i + \mathbf{d}_i\left(\mathbf{P}\right)\right) - \mathbf{v}_{\text{mov}}\left(\mathbf{x}_i\right)\right)^2, \quad (4)$$

where $\mathbf{P}$ denotes the parameters for the inverse transformation of $\mathbf{P}^{-1}$ and $\mathbf{v}_{\text{mov}}$ is the vector that represents the moving volume sampled at Cartesian grid points. With (4), we can register the reference volume to incoming volumes instead and apply the opposite transformation for motion correction, to reduce computation.

**2.3.2 Interpolators—**Voxels located on a Cartesian grid are measured by the scanner, but to evaluate (4) we also need to evaluate the reference volume at points that have not been measured. We approximate the values that would have been measured off-grid using interpolators which have the linear form

$$\mathbf{V}_{\text{ref}}\left(\mathbf{x}_i + \mathbf{d}_i\left(\mathbf{P}\right)\right) \approx \mathbf{I}\left(\mathbf{x}_i + \mathbf{d}_i\left(\mathbf{P}\right), \mathbf{v}_{\text{ref}}\right), \quad (5)$$

where $\mathbf{I}(\mathbf{x}_i + \mathbf{d}_i(\mathbf{P}), \mathbf{v}_{\text{ref}})$ is an interpolation operator that takes $\mathbf{v}_{\text{ref}}$ as input to estimate values of the reference volume at transformed point $\mathbf{x}_i + \mathbf{d}_i(\mathbf{P})$. The choice of interpolator can significantly affect registration accuracy. Unsmooth cost functions, which can result from interpolation artifacts, can cause the minimization to be trapped in local minima [1]. We have evaluated the accuracy of three interpolators: trilinear, tricubic [6], and cubic B-spline [14, 11].

**Trilinear interpolation** approximates the value of a volume $\mathbf{V}$ at an unknown point $\mathbf{x} = (x, y, z)$ (with coordinates expressed in units of voxels) using the eight grid points around $(x, y, z)$ from $\mathbf{v}$, which is sampled on Cartesian grid points. The algorithm first finds the nearest "base" grid point $\mathbf{x}\_ = (x\_, y\_, z\_)$, whose indices are the floor of $(x, y, z)$. From the "base" grid point, the relative off-set $\Delta\mathbf{p} = (\Delta p_x, \Delta p_y, \Delta p_z) = (x - x\_, y - y\_, z - z\_)$ between the two points is computed. Since voxels are isotropic in our data, the interpolation equation is then

$$\begin{aligned}&\mathbf{I}_{\text{trilinear}}\left(\mathbf{x}, \mathbf{v}\right)\\ &= \sum_{i=0}^{1}\sum_{j=0}^{1}\sum_{k=0}^{1}\left[\mathbf{v}\left(x\_ + i, y\_ + j, z\_ + k\right) \times \right.\\ &\qquad \left.|1 - i - \Delta p_x||1 - j - \Delta p_y||1 - k - \Delta p_z|\right], \end{aligned} \quad (6)$$

**Tricubic interpolation** can be written in the form

$$\begin{aligned}&\mathbf{I}_{\text{tricubic}}(\mathbf{x},\mathbf{v})\\&= \sum_{i=0}^{3}\sum_{j=0}^{3}\sum_{k=0}^{3} a_{ijk}(\mathbf{x}_{-})(\Delta p_x)^i(\Delta p_y)^j(\Delta p_z)^k,\end{aligned} \quad (7)$$

where $a_{ijk}(\mathbf{x}\_)$ are the 64 coefficients of tricubic interpolation at the "base" grid point. The choice of $a_{ijk}$ is generally defined by imposing continuity constraints on the interpolated function at the grid points, and for which we have followed a derivation by Lekien *et al.* [6]

First, we note that if we define a 64-vector $\mathbf{a}_{\text{tricubic}}(\mathbf{x}\_)$ that contains all the $a_{ijk}(x\_, y\_, z\_)$ and similarly a 64-vector $\mathbf{k}(\mathbf{\Delta p})$ that contains all the powers of $(\Delta p_x)^i(\Delta p_y)^j(\Delta p_z)^k$ that appear in (7), we can rewrite our interpolation equation as

$$\mathbf{I}_{\text{tricubic}}(\mathbf{x},\mathbf{v}) = \mathbf{a}_{\text{tricubic}}(\mathbf{x}_{-})^T\mathbf{k}(\Delta\mathbf{p}). \quad (8)$$

Lekien *et al.* showed that $\mathbf{a}_{\text{tricubic}}(\mathbf{x}\_)$ can be computed as a linear combination of values from the point $\mathbf{x}\_$ and the the additional 7 grid points that can be reached by adding 1 to each of its indices. The values needed at each point are the image, its three first derivatives, three second cross derivatives, and one third cross derivative; we have used central finite differences to compute the derivatives. These source values can be put in a 64-vector $\mathbf{b}_{\text{tricubic}}(\mathbf{x}\_)$, which can then by multiplied by a fixed matrix $\mathbf{B}_{\text{tricubic}}$ giving [6]

$$\mathbf{a}_{\text{tricubic}}(\mathbf{x}_{-}) = \mathbf{B}_{\text{tricubic}}\mathbf{b}_{\text{tricubic}}(\mathbf{x}_{-}). \quad (9)$$

From an efficiency perspective, Lekien *et al.* observed that, if many sample points with the same "base" grid point are going to be interpolated, $\mathbf{a}_{\text{tricubic}}(\mathbf{x}\_)$ is heavily reused. These vectors can then be precomputed and saved for every point of the volume [6]. Using the reversed order of registration described in (4), the reference volume needs to be resampled many times during the registration process, but doesn't change as each new moving volume arrives. Therefore, the tricubic coefficients can be precomputed and saved for the reference volume, reducing registration time at the cost of a 64-fold increase in the memory required to store the reference volume.

**Cubic B-spline interpolation** is an example of a generalized interpolator, as defined by Unser *et al.* [13] These methods are "generalized" in that they first compute a volume of coefficients from the input image volume, and then perform a linear operation on the coefficients, with the resulting output being an interpolation of the original input volume [11]. They have shown that cubic B-spline, among many other families of interpolators, are an example of this generalized formulation and demonstrated efficient algorithms for the calculation of the coefficient volumes in the cubic B-spline case [14].

Unser *et al.* introduced an efficient two-step procedure for computing coefficient volumes by first computing an intermediate volume, $\mathbf{c}^+$, and then from this computing the desired

coefficients, **c** [14]. However, this algorithm assumed mirror boundaries, and we have assumed circular boundaries in all our algorithms (in part, because circular wrap-around is expected to occur in MRI scans) giving the following algorithm (applied first along the x, then y, and finally z axes, with each of these three stages taking as input the previous step's the output). With $z_0 = -2+\sqrt{3}$, compute $g_i$ from $f_i$ (the signal, of length $n$) using the recursion

$$\begin{array}{rcll} g_0 & = & 6f_0 & \\ g_i & = & z_0 g_{i-1} + 6f_i & (i \neq 0, i \neq n-1) \\ g_{n-1} & = & \frac{z_0 g_{n-2} + 6f_{n-1}}{1-z_0^n}. & \end{array} \quad (10)$$

Then compute $\mathbf{c}^+$ from **g** using

$$\begin{array}{rcll} c^+_{n-1} & = & g_{n-1} & \\ c^+_i & = & g_i + z_0^{i+1} g_{n-1} & (i \neq n-1). \end{array} \quad (11)$$

Now, we can compute $h_i$ from the $c_i^+$ using the backwards recursion

$$h_{n-1} = -z_0 c^+_{n-1} \quad (12)$$

$$h_i = z_0 \left( h_{i+1} - c_i^+ \right) \quad (i \neq 0, i \neq n-1) \quad (13)$$

$$h_0 = \frac{z_0 \left( h_1 + c_0^+ \right)}{1 - z_0^n}. \quad (14)$$

Finally, from **h** we can compute **c** using

$$c_0 = h_0 \quad (15)$$

$$c_i = h_i - z_0^{n-i} h_1 \quad (i \neq 0). \quad (16)$$

After computing the coefficient volume, we interpolate a point in 3D using the 64 surrounding values in the coefficient volume via the equation

$$\mathbf{I}_{\text{B- spline}}(\mathbf{x},\mathbf{v}) = \sum_{i=-1}^{2}\sum_{j=-1}^{2}\sum_{k=-1}^{2}\beta(\Delta p_x - i, \Delta p_y - j, \Delta p_z - k)\,\mathbf{c}(x_- + i, y_- + j, z_- + k), \quad (17)$$

where $\beta(.)$ is the 3D cubic B-spline interpolator kernel, which, being separable, can be written as the product of three, 1D cubic B-spline interpolator kernels, as defined in [11]

$$\beta(x) = \begin{cases} \frac{2}{3} - \frac{1}{2}|x|^2(2-|x|), & 0 \le |x| < 1 \\ \frac{1}{6}(2-|x|)^3, & 1 \le |x| < 2. \\ 0, & 2 \le |x| \end{cases} \quad (18)$$

Equation (17) is structurally very similar to (7), in that, with some rearrangement, it can be written as the inner product of a 64-vector of coefficients, $\mathbf{a}_{\text{B-spline}}(\mathbf{x}\_)$ and the 64-vector of powers of $\Delta p_x$, $\Delta p_y$, and $\Delta p_z$, which we have previously called $\mathbf{k}(\Delta\mathbf{p})$:

$$\mathbf{I}_{\text{B- spline}}(\mathbf{x}) = \mathbf{a}_{\text{B- spline}}(\mathbf{x}_-)^T\mathbf{k}(\Delta\mathbf{p}). \quad (19)$$

Mirroring the derivation for tricubic interpolation, we can combine the 64 values of **c** that are used in interpolation from "base" point **x_** into a vector $\mathbf{b}_{\text{B-spline}}$, and, from the structure of (17), define a fixed matrix $\mathbf{B}_{\text{B-spline}}$ such that

$$\mathbf{a}_{\text{B- spline}}(\mathbf{x}) = \mathbf{B}_{\text{B- spline}}\mathbf{b}_{\text{B- spline}}(\mathbf{x}_-). \quad (20)$$

We can thus use the same trade-off of memory for efficiency in repeated interpolations as demonstrated in [6]. As in tricubic interpolation, $\mathbf{a}_{\text{B-spline}}(\mathbf{x}\_)$ can be precomputed and saved for the reference volume. Given the "base" grid point and the relative offset, evaluation of $\mathbf{I}_{\text{B-spline}}(\mathbf{x})$ consists of computing $\mathbf{k}(\Delta\mathbf{p})$, which takes 62 operations, and performing a dot product with the appropriate precomputed vector, requiring 127 operations. Thus a total of 189 floating point operations are required to interpolate at an unknown point. A naive implementation of the algorithm proposed by Unser *et al.* requires at least 10 floating point operations at each of the 64 neighboring point to evaluate $\beta(x)$ and multiply it with the coefficients. Hence our implementation is at least three times faster than the naive implementation.

**2.3.3 Minimization—**Given the time constraints of real time motion correction, we have opted to use a Gauss-Newton minimization algorithm, which requires only the first derivatives of the residual in our cost function. Formally, the Gauss-Newton algorithm descends towards the minimum of the cost function (4) with respect to the set of six rigid transformation parameters **P** through an iterative process.

$$\mathbf{P}^{s+1} = \mathbf{P}^{s} - \left(\mathbf{J_r}^{\mathrm{T}}\mathbf{J_r}\right)^{-1}\mathbf{J_r}^{\mathrm{T}}\mathbf{r}\left(\mathbf{P}^{s}\right), \quad (21)$$

where $\mathbf{r}(\mathbf{P}^s)$ is the residual between the moving volume and the reference volume interpolated with parameters $\mathbf{P}^s$ and $\mathbf{J_r}$ is the Jacobian with respect to the transformation parameters. Residual $\mathbf{r}(\mathbf{P})$ is defined at all grid points $i$ via,

$$\begin{aligned} r_i\left(\mathbf{P}\right) &= \mathbf{V}_{\mathrm{ref}}\left(\mathbf{x}_i+\mathbf{d}_i\left(\mathbf{P}\right)\right) - \mathbf{v}_{\mathrm{mov}}\left(\mathbf{x}_i\right) \\ &\approx \mathbf{I}\left(\mathbf{x}_i+\mathbf{d}_i\left(\mathbf{P}\right), \mathbf{v}_{\mathrm{ref}}\right) - \mathbf{v}_{\mathrm{mov}}\left(\mathbf{x}_i\right). \end{aligned} \quad (22)$$

We approximate the Jacobian of the residual using a Taylor series approximation of the residual evaluated at the (unknown) true value of the parameters $\mathbf{P}_0$ as follows

$$\begin{aligned} \left(\mathbf{J_r}\right)_i &= \nabla_{\mathbf{P}} r_i\left(\mathbf{P}_0+\mathbf{P}\right) \\ &\approx \nabla_{\mathbf{P}} \mathbf{V}_{\mathrm{ref}}\left(\mathbf{x}_i+\mathbf{d}_i\left(\mathbf{P}_0\right)+\mathbf{d}_i\left(\mathbf{P}\right)\right) \\ &\approx \nabla_{\mathbf{P}}\left[\mathbf{V}_{\mathrm{ref}}\left(\mathbf{x}_i+\mathbf{d}_i\left(\mathbf{P}_0\right)\right)\right. \\ &\quad + \left.\nabla_{\mathbf{x}_i}\mathbf{V}_{\mathrm{ref}}\left(\mathbf{x}_i+\mathbf{d}_i\left(\mathbf{P}_0\right)\right)\mathbf{d}_i\left(\mathbf{P}\right)\right] \\ &= \nabla_{\mathbf{x}_i}\mathbf{V}_{\mathrm{ref}}\left(\mathbf{x}_i+\mathbf{d}_i\left(\mathbf{P}_0\right)\right)\nabla_{\mathbf{P}}\mathbf{d}_i\left(\mathbf{P}\right) \\ &= \nabla_{\mathbf{x}_i}\mathbf{V}_{\mathrm{mov}}\left(\mathbf{x}_i\right)\nabla_{\mathbf{P}}\mathbf{d}_i\left(\mathbf{P}\right) \\ &\approx \nabla_{\mathbf{x}_i}\mathbf{V}_{\mathrm{mov}}\left(\mathbf{x}_i\right)\nabla_{\mathbf{P}}\mathbf{d}_i\left(\mathbf{P}\right) \end{aligned} \quad (23)$$

where we have used the fact that, by definition, $\mathbf{V}_{\mathrm{ref}}(\mathbf{x}_i + \mathbf{d}_i(\mathbf{P}_0)) = \mathbf{V}_{\mathrm{mov}}(\mathbf{x}_i)$ and used $\nabla_{\mathbf{x}i}$ $\mathbf{v}_{\mathrm{mov}}(\mathbf{x}_i)$ to denote the discrete approximation to the partial derivatives at voxel $i$ with respect to the three coordinate axes (we used central differences in our implementation).

The rigid transformation parameter $\mathbf{P}$ is stored as a 6-vector where the first three elements express translations along the three coordinate axes and the next three elements forms a vector that points along the axis of rotation whose magnitude is the angle of rotation in radians [9]. With such parametrization, we can use the fact that small rotations are effectively translations to approximate $\nabla_{\mathbf{P}}\mathbf{d}_i(\mathbf{P})$ with the the linear relationship

$$\begin{aligned} \nabla_{\mathbf{P}}\mathbf{d}_i\left(\mathbf{P}\right) &\approx \nabla_{\mathbf{P}}\mathbf{M}_i\mathbf{P} \\ &= \mathbf{M}_i, \end{aligned} \quad (24)$$

where

$$\mathbf{M}_i = \begin{bmatrix} 1 & 0 & 0 & 0 & z_i & -y_i \\ 0 & 1 & 0 & -z_i & 0 & x_i \\ 0 & 0 & 1 & y_i & -x_i & 0 \end{bmatrix}. \quad (25)$$

Substituting this approximation back into equation (23) gives us,

$$(\mathbf{J_r})_i \approx \nabla_{\mathbf{x}_i} \mathbf{v}_{\text{mov}}(\mathbf{x_i})\,\mathbf{M}_i. \quad (26)$$

Including all of our approximations, the steps defined in equation (21) can sometimes increase the residual error. To address this, we use a scaling factor to shrink our update $\Delta\mathbf{P}$; the factor starts at 1.0 and whenever a step increases error we undo the update to $\mathbf{P}$ and multiply the scaling factor by 0.25. The minimization is terminated when either the absolute difference between $\mathbf{P}^s$ and $\mathbf{P}^{s-1}$ becomes negligible or a maximum number of iterations is reached. In our implementation, the former criterion is specified such that iteration stops when the every element of $|\mathbf{P}^s - \mathbf{P}^{s-1}|$ is less than $10^{-5}$. The maximum number of iteration can be set according to the time constraint of real time motion correction. In our implementation for experimental results, this stopping criterion is never reached. Since we are registering reference volume to source volume, when the iterative process ends, $\mathbf{P}^{-1}$ gives the desired transformation from the source volume to the reference volume.

### 2.4. Quantifying Registration Accuracy

Registration accuracy can be measured by comparing the result from Gauss-Newton algorithm and the "ground truth" transformation that was applied to the acquisition. Multiplying the inverse of the estimated rigid transform by the known true rigid transform results in a rigid transformation describing the error. The six parameters of this rigid transformation consist of three values describing translations, and three describing rotations, hence the error transformation can be measured separately for translations and rotations. Let $\mathbf{R}$ denote the rotation matrix of the difference in rotation between the approximated transformation and "ground truth", and $\mathbf{t}$ be the difference in translation between the two (in mm). The natural measure of translation error is simply the 2-norm

$$E_{\text{trans}} = \sqrt{\mathbf{t}^T\mathbf{t}}. \quad (27)$$

The error in rotation can be summarized by the magnitude of its angle in degrees, which can be derived from $\mathbf{R}$ via,

$$E_{\text{rot}} = \arccos\left(\frac{\mathbf{Tr}(\mathbf{R}) - 1}{2}\right) \quad (28)$$

As a single summary metric, we have chosen to use the RMS displacement that the error transformation would produce in a sphere that is centered at the middle of the reference volume. We are using the assumption, as in Maurer *et al.*, that the head is a sphere of radius 100 mm [8]. The target registration error for volume defined in [8] is a discrete form of the RMS measure derived by Jenkinson [5] as,

$$E_{\text{RMS}} = \sqrt{\frac{1}{5}r^2 Tr\left[(\mathbf{R}-\mathbf{I})^T(\mathbf{R}-\mathbf{I}) + \mathbf{t}^T\mathbf{t}\right]} \quad (29)$$

giving an error displacement in mm, where $r$ is the radius (in mm) of the sphere over which we average. In our comparisons, we have set $r$ = 100 mm to represent an approximate radius for a human head. Note that this naturally defines a measure of rotation error, by taking just the first term in (29), giving the RMS translation inside the sphere due to error in rotation

$$t_{\text{rot- RMS}} = \sqrt{\frac{1}{5}r^2 Tr\left[(\mathbf{R}-\mathbf{I})^T(\mathbf{R}-\mathbf{I})\right]} \quad (30)$$

In addition to the average error, we are also interested in the worst-case error in a sphere of the same size. The maximum translation on the sphere imposed by the rotation error has magnitude (in mm)

$$t_{\text{rot- max}} = r\sqrt{3-\text{Tr}(\mathbf{R})}. \quad (31)$$

Combining this with the displacement from translations gives the maximum displacement (in mm) in the sphere,

$$E_{\max} = \sqrt{t^2_{\text{rot- max}} + 2t_{\text{rot- max}}|\mathbf{t}-(\mathbf{t}^T\mathbf{u}_\mathbf{R})\,\mathbf{u}_\mathbf{R}|_2 + \mathbf{t}^T\mathbf{t}}, \quad (32)$$

where $\mathbf{u_R}$ is a unit vector that represents the axis of rotation in angle-axis representation, which can be obtained by normalizing the last three elements of the six rigid transformation parameters.

## 3. Results

In all of the data we processed, the Gauss-Newton algorithm reached the stopping criterion roughly in 10 iterations and all produced plausible estimation of rigid motion.

Rotation error and translation error, from equations (28) and (27) respectively, are plotted in Figure (2). Cubic B-splines performed marginally better than tricubic, and significantly better than trilinear, for translations. However, as expected, trilinear interpolation suffers significantly in estimating rotations.

Figure (3) shows the RMS displacement, defined in (29), from these registrations. The quantile values of the RMS displacement are listed in Table 2. At all resolutions, we find that cubic B-spline interpolation performs the best, followed by tricubic interpolation, and then trilinear interpolation. However, the results across resolutions are more mixed. With tricubic interpolation, the median registration accuracy is similar with 8 mm and 10 mm resolution, the variability in accuracy is smaller with 10 mm, the lower resolution. For cubic B-spline interpolation, the registration accuracy is reversed in that 10 mm resolution had the smallest RMS displacement in most quantiles.

Figure (4) displays the maximum displacement of error transformation. The quantile values of the maximum displacement are also listed in Table 2. The trends in accuracy with resolution and algorithm are the same with this error metric as in Figure (3).

To better understand these effects, we examined the contribution of rotations and translations to the error transformations separately, to see whether one is dominating the other. Figure (2) already shows the effect of translations, and Figure (5) shows maximum and RMS displacements due to just the rotation component of the error, derived from equations (31) and (30) respectively. Comparing Figure (2) and Figure (5), it can be observed that the relative scale of the RMS displacements in mm due to rotation and translation error are roughly the same, where as the scale of maximum displacements from rotation error is slightly larger than the translation error.

We tested improvements across different resolutions and interpolation methods using the Wilcoxon signed-rank test on the RMS errors of the registrations. The test was performed on each pair of resolutions or interpolation methods, and the test statistic from all 420 registrations in each condition was converted to a z-score. These z-scores from the test are summarized in Tables (3) and (4).

These results support the observation that both tricubic and cubic B-spline interpolation outperforms trilinear interpolation, and cubic B-spline interpolation produces more accurate registration than tricubic interpolation at all resolutions. The results for comparison across resolutions are perhaps counter-intuitive; 10 mm resolution showed higher accuracy with both trilinear and cubic B-spline interpolation than 8 mm resolutions. However, there is an improvement in registration accuracy from 6.4 mm resolution to both 8 mm and 10 mm resolutions.

## 4. Discussion

Our accuracy goal was 1 mm error, based on the target application of real time MRI motion correction. Our results show that registration at all three vNav resolutions achieves this goal when using cubic B-spline interpolation or tricubic interpolation. In particular, the registration accuracy with cubic B-spline interpolation at 10 mm resolution is acceptable, which can be helpful in real time MRI motion correction since it is significantly faster to acquire and register this data compared to the 8 mm navigators currently being used in practice. We are working to acquire more data in order to explore the counter-intuitive result that 10 mm resolution were of higher accuracy than 8 mm resolution. There is also measurable improvement in accuracy from using 6.4 mm that may be worth the additional cost in acquisition and registration time in certain MRI applications, *e.g.*, single-voxel spectroscopy where there is more available dead-time [4].

Our results also indicate that, at all resolutions, cubic B-splines provide more accurate results than either trilinear or tricubic interpolation. This is consistent with the literature, showing fewer artifacts in cubic B-spline interpolation compared to the other two algorithms [11]. We have demonstrated that, by paying a cost in memory, cubic B-spline interpolation can be made as computationally efficient as tricubic interpolation in our application.

Interestingly, we observed that registration accuracy decreased with rotations involving the head foot (HF) axis. This may be due to our scanning protocol's axes (HF/readout/z, AP/phase/y, LR/partition/x), where rotations around the HF and LR axes cause the phase-encoded axis to rotate. This will, in turn, cause the susceptibility distortions around the sinuses to shift direction, potentially influencing the accuracy of our results, as noted in previous work registering higher-resolution EPI volumes [2]. Figure 6 shows an axial slice of the difference between a volume with no rotation and known true 5° rotations around each imaging axis. Comparing rotation around HF with that around AP, we see the sinus much more prominently in the HF rotation. Further work is needed to verify whether this hypothesis is correct. If so, it may be possible to mask out these regions, since they are localized to the sinuses and other internal air/tissue interfaces.

## 5. Conclusion

We have evaluated the accuracy of vNavs for tracking the human head using data with known ground-truth rigid transformations. This application demands accurate sub-voxel registration be performed as efficiently as possible on the vNav data as it arrives. We have explored choices of both vNav resolution, and derived and evaluated three efficient registration algorithms, to understand the trade-offs between acquisition and registration time and accuracy of motion tracking.

We demonstrated registration algorithms that perform accurate sub-voxel volume registration on vNavs. Median tracking errors from 6.4 mm, 8 mm and 10 mm vNavs using either cubic b-spline or tricubic interpolation achieved our goal of 1 mm. Our results show that improving vNav resolution provides measurable improvement in registration accuracy at the cost of acquisition time. Our algorithmic comparison shows that cubic B-spline interpolation provides the best results at all vNav resolutions, in particular cubic B-spline interpolation can achieve high accuracy, even at 10 mm resolution.

We have also presented a refactoring of the cubic B-splines interpolation algorithm that reduces the perinterpolation time by storing pre-computed interpolation coefficients. This makes the computational cost of cubic B-spline interpolation the same as tricubic interpolation in our application.

We identified distortions in our MRI data due to field inhomogeneity as a potential limitation to our current method and plan to acquire further data to explore masking strategies to mitigate their effects.

All image processing and analysis algorithms were implemented in Python. However, our eventual goal is to develop a new registration algorithm for use on-scanner with vNavs. In the derivations of the algorithms explored in this work, we have focused on the time-efficiency of our algorithms, with the goal that the remaining speed-up in the on-scanner implementation will come from low-level memory management, *etc.* in C++ and the potential for parallelizing work, not fundamentally different algorithms. Our initial prototype, single-threaded C++ implementation, using the Eigen [3] library, has an average registration time of 25 ms for an 8 mm vNav volume. Future work will involve

parallelization of the algorithm and further optimization, along with integration to the scanner environment.

Although PACE is currently implemented on the scanner, the full algorithm is not publicly available. Additionally, the on-scanner implementation cannot be used with all of the resolutions we are evaluating. For these reasons, we have not compared our results directly with PACE. However, a comparison of the current vNav+PACE system with any new navigator/registration combination will be important once the new system is implemented on the scanner.

## 6. Acknowledgements

This work was funded by NIBIB P41EB015896; NIH Shared Instrument Grants S10RR021110, S10RR023043, and S10RR023401; NICHD 4R00HD074649-03 and R01HD071664; NIA R21AG046657; NIDDK K01DK101631

## References

[1]. Aganj I, Yeo BT, Sabuncu MR, Fischl B. On removing interpolation and resampling artifacts in rigid image registration. Image Processing, IEEE Transactions on. 2013; 22(2):816–827. 2, 3.

[2]. Andersson JL, Hutton C, Ashburner J, Turner R, Friston K. Modeling geometric deformations in epi time series. Neuroimage. 2001; 13(5):903–919. 8. [PubMed: 11304086]

[3]. Guennebaud G, Jacob B, et al. Eigen v3. 2010 http://eigen.tuxfamily.org. 8.

[4]. Hess AT, Dylan Tisdall M, Andronesi OC, Meintjes EM, van der Kouwe AJ. Real-time motion and b0 corrected single voxel spectroscopy using volumetric navigators. Magnetic resonance in medicine. 2011; 66(2):314–323. 1, 8. [PubMed: 21381101]

[5]. Jenkinson M. Measuring transformation error by RMS deviation. Technical Report TR99MJ1, FMRIB, Oxford. 1999 5.

[6]. Lekien F, Marsden J. Tricubic interpolation in three dimensions. International Journal for Numerical Methods in Engineering. 2005; 63(3):455–471. 1, 3, 4.

[7]. Maclaren J, Herbst M, Speck O, Zaitsev M. Prospective motion correction in brain imaging: a review. Magnetic resonance in medicine. 2013; 69(3):621–636. 1. [PubMed: 22570274]

[8]. Maurer CR Jr, Fitzpatrick JM, Wang MY, Galloway RL, Maciunas RJ, Allen GS. Registration of head volume images using implantable fiducial markers. Medical Imaging, IEEE Transactions on. 1997; 16(4):447–462. 5.

[9]. Reuter M, Rosas HD, Fischl B. Highly accurate inverse consistent registration: a robust approach. Neuroimage. 2010; 53(4):1181–1196. 5. [PubMed: 20637289]

[10]. Thesen S, Heid O, Mueller E, Schad LR. Prospective acquisition correction for head motion with image-based tracking for real-time fmri. Magnetic Resonance in Medicine. 2000; 44(3):457–465. 1, 3. [PubMed: 10975899]

[11]. Thévenaz P, Blu T, Unser M. Interpolation revisited [medical images application]. Medical Imaging, IEEE Transactions on. 2000; 19(7):739–758. 1, 3, 4, 8.

[12]. Tisdall MD, Hess AT, Reuter M, Meintjes EM, Fischl B, van der Kouwe AJ. Volumetric navigators for prospective motion correction and selective reacquisition in neuroanatomical mri. Magnetic resonance in medicine. 2012; 68(2):389–399. 1. [PubMed: 22213578]

[13]. Unser M, Aldroubi A, Eden M. B-spline signal processing. I. Theory. Signal Processing, IEEE Transactions on. 1993; 41(2):821–833. 3.

[14]. Unser M, Aldroubi A, Eden M. B-spline signal processing. II. Efficiency design and applications. Signal Processing, IEEE Transactions on. 1993; 41(2):834–848. 1, 3, 4.

[15]. Zaitsev M, Maclaren J, Herbst M. Motion artifacts in mri: A complex problem with many partial solutions. Journal of Magnetic Resonance Imaging. 2015; 42(4):887–901. 1. [PubMed: 25630632]

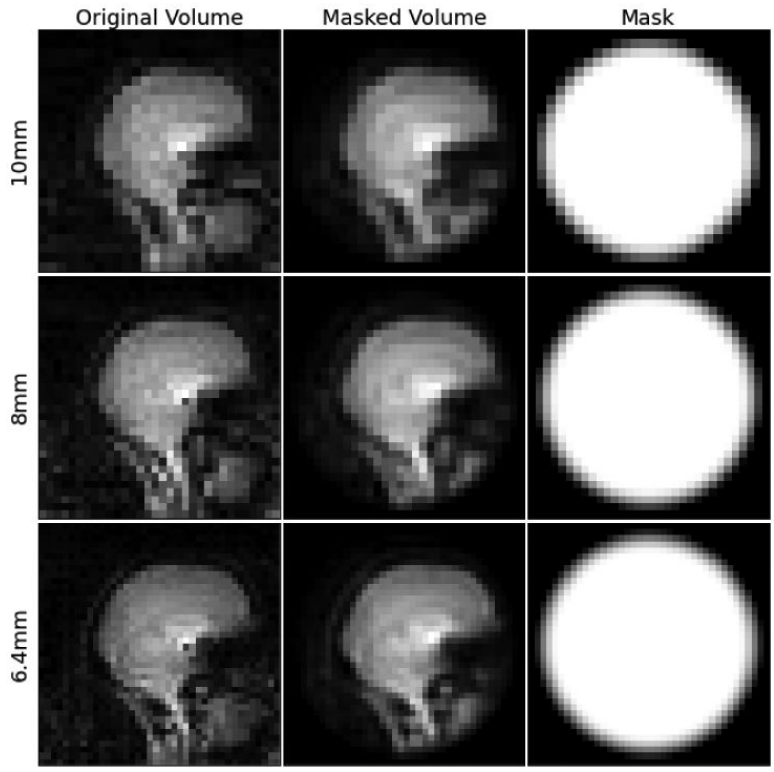


**Figure 1.**
Original and masked slices from vNavs, and the masks used at different resolutions

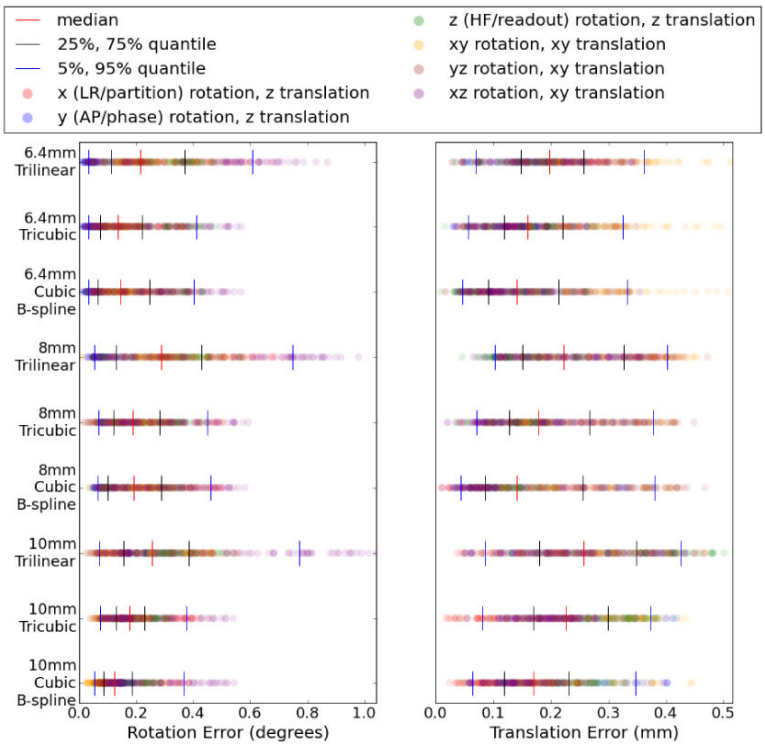


**Figure 2.**

Rotation (left, degrees) and translation (right, mm) errors with cubic B-spline, tricubic and trilinear interpolation, with each row representing a separate algorithm/resolution combination. Each of the 420 results is plotted as a dot in each row, colored based on the ground truth axis of rotation. Quantiles are displayed with vertical lines.

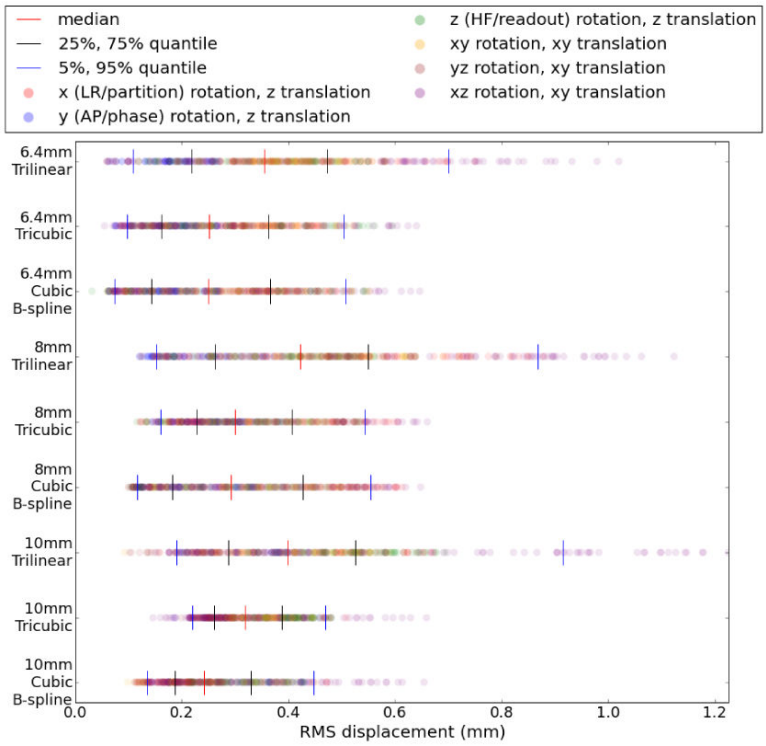


**Figure 3.**

RMS error displacement from cubic B-spline, tricubic and trilinear interpolation, with each row representing a separate algorithm/resolution combination. Each of the 420 results is plotted as a dot in each row, colored based on the ground truth axis of rotation. Quantiles are displayed with vertical lines.

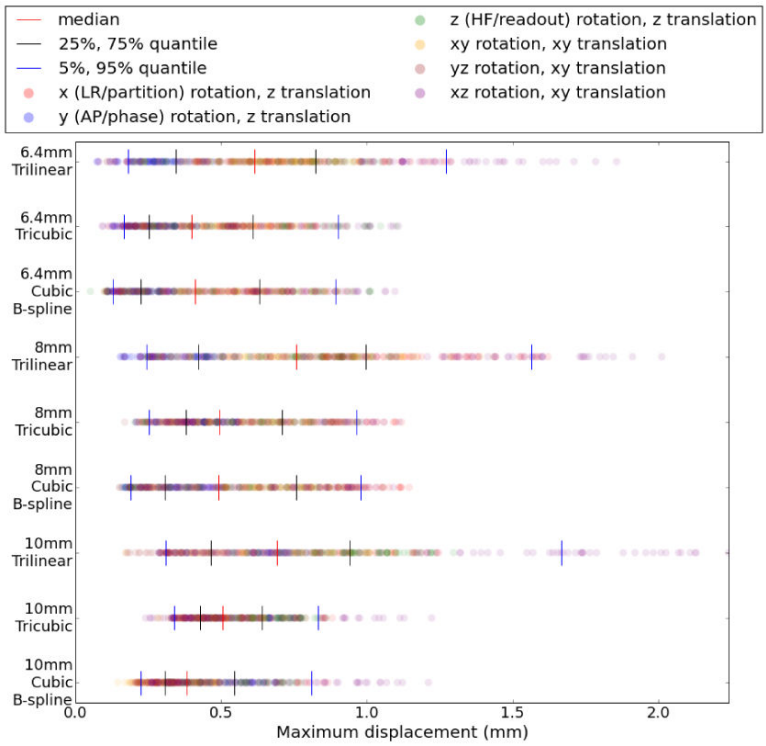


**Figure 4.**

Maximum error displacement from cubic B-spline, tricubic and trilinear interpolation, with each row representing a separate algorithm/resolution combination. Each of the 420 results is plotted as a dot in each row, colored based on the ground truth axis of rotation. Quantiles are displayed with vertical lines.

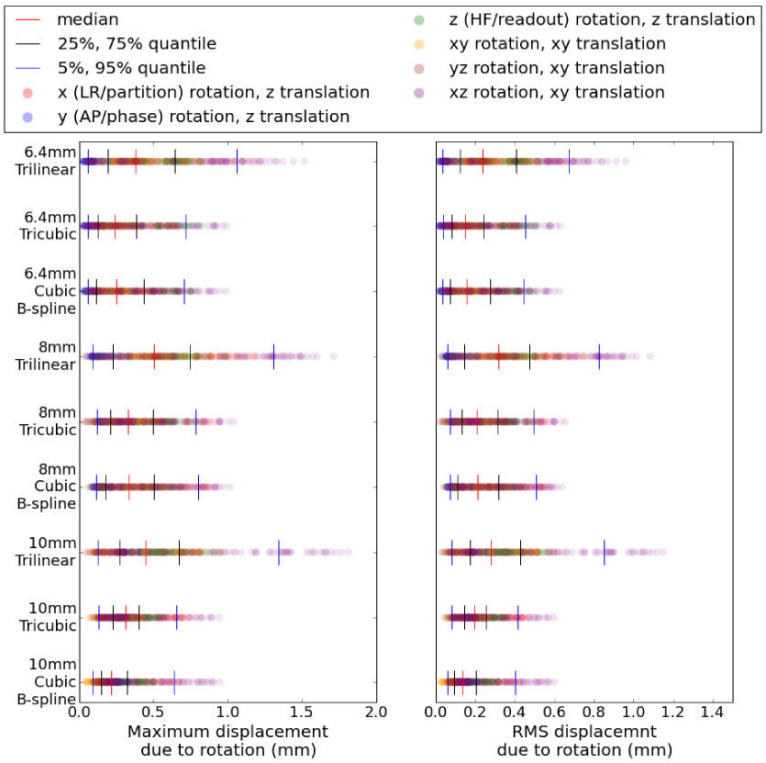


**Figure 5.**

Maximum (left) and RMS (right) displacement from rotation error for cubic B-spline, tricubic, and trilinear interpolation, with each row representing a separate algorithm/resolution combination. Each of the 420 results is plotted as a dot in each row, colored based on the ground truth axis of rotation. Quantiles are displayed with vertical lines.

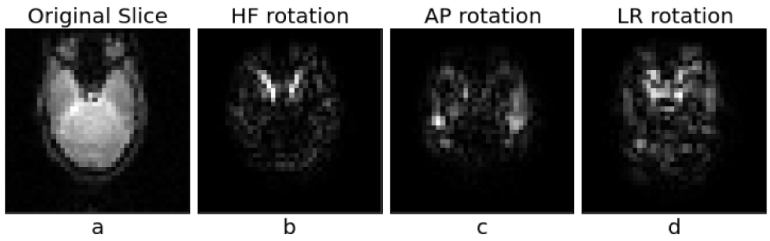


**Figure 6.**

(a) Sample sagittal slice in the HF direction at 6.4 mm resolution, with mask applied in the Fourier domain only. (b)-(d) Difference between a volume with no rotation and volumes acquired with 5° rotations around the HF, AP and LR axes respectively, masked in both Fourier and image domains.

**Table 1**

Sequence parameters for vNavs

| | **6.4 mm** | **8 mm** | **10 mm** |
|---|---|---|---|
| TR | 15 ms | 11 ms | 10 ms |
| TE | 6.7 ms | 5.0 ms | 4.1 ms |
| FA | 3° | 3° | 3° |
| BW | 4310 Hz/Px | 4596 Hz/Px | 4578 Hz/Px |
| FOV | 256 mm | 256 mm | 260 mm |
| Total Scan Time | 600 ms | 352 ms | 260 ms |

**Table 2**

Quantiles of RMS displacement and maximum displacement (in mm) computed with trilinear, tricubic and cubic B-spline interpolations across three different resolutions. These values are depicted with vertical bars in Figures (3) and (4).

| quantile | resolution (mm) | trilinear | | tricubic | | cubic B-spline | |
|---|---|---|---|---|---|---|---|
| | | RMS | Max | RMS | Max | RMS | Max |
| 5% | 6.4 | 0.11 | 0.16 | 0.10 | 0.17 | 0.07 | 0.13 |
| | 8 | 0.15 | 0.24 | 0.16 | 0.25 | 0.12 | 0.19 |
| | 10 | 0.19 | 0.31 | 0.22 | 0.34 | 0.13 | 0.22 |
| 25% | 6.4 | 0.22 | 0.34 | 0.16 | 0.25 | 0.14 | 0.22 |
| | 8 | 0.26 | 0.42 | 0.23 | 0.38 | 0.18 | 0.31 |
| | 10 | 0.29 | 0.47 | 0.26 | 0.43 | 0.19 | 0.31 |
| median | 6.4 | 0.35 | 0.61 | 0.25 | 0.40 | 0.25 | 0.41 |
| | 8 | 0.42 | 0.76 | 0.30 | 0.49 | 0.29 | 0.49 |
| | 10 | 0.40 | 0.69 | 0.32 | 0.51 | 0.24 | 0.38 |
| 75% | 6.4 | 0.47 | 0.82 | 0.36 | 0.61 | 0.37 | 0.63 |
| | 8 | 0.55 | 1.00 | 0.41 | 0.71 | 0.43 | 0.76 |
| | 10 | 0.53 | 0.94 | 0.39 | 0.64 | 0.33 | 0.54 |
| 95% | 6.4 | 0.7 | 1.27 | 0.5 | 0.90 | 0.51 | 0.89 |
| | 8 | 0.87 | 1.56 | 0.54 | 0.96 | 0.55 | 0.98 |
| | 10 | 0.91 | 1.67 | 0.47 | 0.83 | 0.45 | 0.81 |

**Table 3**

Wilcoxon signed-rank test for comparing registration accuracy across interpolation methods. A negative score indicates that there is an improvement in the pair from the first interpolation method to the second, and a positive score indicates the opposite. Bolded values in the table are z-scores that are not significant for a one-sided difference at $p < 0.01$.

| | **6.4 mm** | **8 mm** | **10 mm** |
|---|---|---|---|
| Trilinear to Tricubic | −15.52 | −12.84 | −12.34 |
| Trilinear to Cubic B-spline | −15.55 | −15.93 | −16.93 |
| Tricubic to Cubic B-spline | − **2.61** | −5.68 | −15.63 |

**Table 4**

Wilcoxon signed-rank test for comparing registration accuracy across resolutions. A negative score indicates that there is an improvement in the pair from the first resolution to the second, and a positive score indicates the opposite. Bolded values in the table are z-scores that are not significant for a one-sided difference at $p < 0.01$.

| | **10 to 8 mm** | **8 to 6.4 mm** | **10 to 6.4 mm** |
|---|---|---|---|
| Trilinear | **0.49** | −11.62 | −7.74 |
| Tricubic | −3.09 | −12.11 | −10.79 |
| Cubic B-spline | 5.54 | −11.29 | − **0.71** |