A General Method for Motion Compensation in X-ray Computed Tomography

Ander Biguri, Manjit Dosanjh, Steven Hancock, Manuchehr Soleimani
Engineering Tomography Lab(ETL), University of Bath, Bath, UK
CERN, Geneva, Switzerland
E-mail: a.biguri@bath.ac.uk
December 2016

Abstract. Motion during data acquisition is a known source of error in medical tomography, resulting in blur artefacts in the regions that move. It is critical to reduce these artefacts in applications such as image-guided radiation therapy as a clearer image translates into a more accurate treatment and the sparing of healthy tissue close to a tumour site. Most research in 4D X-ray tomography involving the thorax relies on respiratory phase binning of the acquired data and reconstructing each of a set of images using the limited subset of data per phase. In this work, we demonstrate a motion-compensation method to reconstruct images from the complete dataset taken during breathing without recourse to phase-binning or breath-hold techniques. As long as the motion is sufficiently well known, the new method can accurately reconstruct an image at any time during the acquisition time span. It can be applied to any iterative reconstruction algorithm.

Keywords: Tomography, 4D-CT, motion compensation

1. Introduction

Organ movement can be a serious problem in X-ray imaging as the inconsistency between data taken at different phases of the motion leads to blurring and makes the boundaries between different regions hard to distinguish. This effect is particularly important in image-guided radiation therapy (IGRT), especially for tumours located in the thorax, such as lung or liver tumours. Respiratory motion is always present and can limit the quality of the image to the point where establishing the precise size, position and boundary of a tumour becomes difficult. Nowadays the principal clinical solution to ensure the irradiation of all the tumour is to irradiate the entire region where it is estimated to be located during the full respiratory cycle, thus necessarily damaging some healthy tissue. This is even more critical in hadron therapy because, unlike photons, charged particles deposit most of their energy at the Bragg peak, which means a tumour
could be missed completely and only healthy tissue irradiated if targeting is carried out on the basis of inaccurate treatment planning. Currently one of the most promising real-time imaging technologies is the MRI-Linac, which can acquire some limited image data at a high frame rate. However, the use of a magnetic resonance imaging (MRI) machine is not compatible with hadron therapy because the magnetic field of the former would perturb the particle beam of the latter.

The most common device for IGRT imaging is a cone-beam computed tomography (CBCT) machine. This is a low-dose, low-cost, 3D X-ray modality. It is used each time a patient undergoes a stage of radiation therapy in order to correct for anatomical changes, such as tumour shrinkage and patient weight loss, that typically occur during the course of treatment. Due to its lower energy than a conventional CT scan and to its slow data acquisition rate, a CBCT image is generally riddled with noise and motion artefacts. Research into the removal of motion artefacts in CBCT is widespread and numerous articles have been published on the subject. The most studied method to deal with motion is phase-correlated CBCT, also called 4D-CBCT[1][2][3][4][5]. In 4D-CBCT, projection data are binned according to respiratory phase and then the data from each bin are reconstructed separately to produce a series of images. This approach has several drawbacks. Even though the amount of data per reconstructed image is smaller than usual, the total number of projections increases which means a longer irradiation time and a higher dose for the patient, limiting its clinical use. In addition, the image quality of each 4D-CBCT reconstruction is inferior to a 3D-CBCT one due to its reduced dataset and to small inconsistencies resulting from binning inaccuracies.

Due to the limitations of standard 4D-CBCT imaging, extensive research has been conducted to improve the quality of the images. This work can be divided into two main groups: algorithmic approaches and deformation vector field (DVF) optimization methods. Methods in the first group rely on regularization and other similar approaches. An example is the work by Jia et al[6], who implemented a non-local means of reconstruction to improve the temporal similarity between images. Total variation methods (TV)[7], which minimize gradients within an image, have been also proposed with a temporal dimension included in the gradient[8]. Another method based on TV minimization is the so-called PICCS algorithm[9][10][11], which minimizes the TV and the difference between the reconstructed image and a prior image. This prior image is generally a CBCT reconstructed with motion artefacts. PICCS can reconstruct 4D-CBCT images from highly undersampled datasets. More complex algorithms have also been proposed, such as ROOSTER[12], where a series of regularizations and minimizations are performed inside a region of interest to create clear 4D images in that area.

The methods of the second group generally (but not always) rely on a previous high-quality 4D-CT treatment planning scan as the basis from which to compute the DVFs. As breathing motion is neither truly periodic nor reproducible in a given patient over time, the DVFs are corrected by matching real projections with simulated ones. Finally, when the best DVF is computed, a synthetic image is generated by deforming
A General Method for Motion Compensation in X-ray Computed Tomography

the prior high-quality CT scan. Examples include the work of Brock et al[13] and Ren et al[14], who managed to reduce the number of projections required to about 60 using non-linear conjugate-gradient methods. In order to improve robustness and reduce the dimensionality of the problem, DVF principal component analysis (PCA) methods have also been proposed[15]. Li et al[16][17] demonstrated that good accuracy can be achieved using only a single projection for the DVF optimization.

Hybrids between DVF-based and algorithmic approaches also exist, such as using TV regularization methods to improve convergence by initializing the DVFs[18] or using temporal regularization with DVFs to improve the ROOSTER algorithm[19]. Hybrid methods can lead to highly complex optimization strategies. Examples include segmented mesh-based 4D-CBCT[20] and the separation of static and moving images using TV, tight frame regularization and DVF optimization[21]. In addition, Christoffersen et al[22] have proposed a multi-step algorithm using TV and optical flow for motion estimation.

Finally, some special mathematical algorithms have also been suggested that are unique in their approach. These include the cine-CBCT algorithm[23] and the 5D motion modelling approach[24], which does not use phase-correlated binning.

The literature is made of these and many other approaches, ranging from the computationally and mathematically complex to those that sacrifice accuracy for simplicity and speed. Most have been shown to yield good 4D-CBCT reconstructions, some in clinical scenarios. But they all have drawbacks. CBCT is a severely ill-posed problem where the amount of data is key for a good reconstruction. The simplest methods that rely on binning will always suffer to some extent from a lack of data, even if temporal coherence is enforced with mathematical norms. Additionally, they involve the reconstruction of several images, which is very expensive both computationally and in terms of memory. The DVF-based approaches also have computational limitations. Those that employ PCA do, indeed, reduce the computational cost by the removal of principal components, but these could describe effects that then are lost. And most of them ultimately use the DVFs to deform a prior image rather than using the acquired data directly to produce a reconstruction. Further, they assume that a DVF can describe every possible anatomical change with respect to that prior image and this does not necessarily hold.

Here, we propose a completely different approach to motion compensation in 4D-CT imaging. We will focus on thorax CBCT in this article, but the method is generalizable to any X-ray absorption CT modality and to arbitrary motion. The method requires no binning, but instead uses all projections to reconstruct an image at any respiratory phase. It does require a sufficiently accurate description of the motion in terms of DVFs, but the approach is a modelling one so it can be used to introduce motion compensation into any iterative reconstruction algorithm.
2. Methods

The method we propose is based on iterative reconstruction algorithms for CT, ideas pioneered in phase space tomography and on GPU acceleration techniques. In this section the principles of iterative reconstruction, GPU acceleration and then the motion-compensation algorithm itself are described.

2.1. Image reconstruction in CT

We briefly introduce the image reconstruction problem in X-ray tomography. While the most commonly used[25] algorithm in CBCT reconstruction in hospitals is FDK[26], it has been repeatedly demonstrated that iterative algorithms perform better[27][28][29][30]. Specifically, they can produce better images with fewer artifacts from the same data, or the same quality of image from less data. This article will only focus on iterative reconstruction methods.

If we denote the attenuation of photons inside a 3D image by \( f(\vec{r}) \) with \( \vec{r} = (x, y, z) \), we can describe the function \( p(\theta, u, v) \) measured in the detector as

\[
p(\theta, u, v) = \int_0^\infty f(\vec{r}_0(\theta) + \alpha \hat{\gamma}) d\alpha,
\]

where \( \vec{r}_0(\theta) = (R \sin \theta, R \cos \theta, 0) \) is the source location, \( \alpha \in [0, \sqrt{D^2 + u^2 + v^2}] \), \( R \) the distance between the source and the centre of rotation and \( D \) the source to detector distance. \( \hat{\gamma} \) denotes the direction of the line between the source and the detector coordinates \( u \) and \( v \), as in

\[
\hat{\gamma} = \frac{\vec{d}(u, v, \theta) - \vec{r}_0(\theta)}{\left| \vec{d}(u, v, \theta) - \vec{r}_0(\theta) \right|},
\]

with \( \vec{d}(u, v, \theta) \) the detector pixel coordinates.

Equation 1 is Beer’s law for CBCT geometry and states that the measured projection is the integral of the image attenuation over the X-ray path. It can be approximated by a linear system of equations,

\[
Ax = b,
\]

where \( x \) is a 1D vector of all image voxels, \( b \) are the detector pixels also expressed as a 1D vector and \( A \) is the so-called system matrix describing the X-ray path length through each of the voxels in the image. Equation 3 can be tackled by a minimization approach,

\[
\hat{x} = \arg \min_x \|Ax - b\|^2 + R(x),
\]

where \( \hat{x} \) is the resultant image and \( R(\cdot) \) is an optional regularization function. This minimization can be performed using a wide variety of iterative algorithms. These algorithms all have one thing in common, the use of \( A \cdot x \) (the projection operator) and \( A^T \cdot b \) (the back projection operator) at least once per iteration. Such operations
are computationally very expensive because the matrix $A$ is huge and, as a way to overcome this, the projection and back projection operators may be accelerated on graphical processing units (GPUs).

As the methods to perform the projection and back projection operations on a GPU are relevant to the motion-correction algorithm we propose, a further analysis of them is given.

### 2.2. GPU methods in tomography

While iterative algorithms do out-perform the standard FDK one, their computational cost is very high. Each row of the matrix $A$ describes the intersection length of a single X-ray through each of the voxels of the image (see figure 1(a)). This translates into a matrix that has as many columns as the image has voxels, and as many rows as the product of detector pixels and the number of projections. A standard medical image of $512^3$ voxels with a detector of $512^2$ and 360 projections would result in a matrix of the order of $10^8 \times 10^9$ in size. Even with a typical sparsity of roughly 0.0017%, this results in a memory requirement of about 320Gb. Consequently, the projection and back projection operators are usually computed on the fly without ever explicitly saving $A$, only the result of $A \cdot x$ and $A^T \cdot b$ directly. However, even with the fastest algorithms to compute the intersection length (Siddon’s algorithm[31] with improvements from Jacob et al[32]), this computation still takes an excessive amount of time in a CPU.

The relative independence of the computations involved in tomography make it an ideal candidate for GPU acceleration and many researchers have studied the best way of parallelizing the projection and back projection operations on GPUs[33][34][35][36]. GPUs provide access to different paradigms and hardware-based optimization methods, one of which is the texture memory cache. Texture memory mode (not really a separate memory in hardware, but a memory accessing system) allows the contents of an image $I$ to be sampled as $I(i,j,k)$, where $i, j, k \in \mathbb{R}^3$ (i.e., any real number can be used as an index), and the value intermediate to the values in the integer voxel mesh will be computed by interpolation. The advantage of the texture memory cache is that the GPU processors are not performing the interpolation themselves, instead the memory cache computes it before passing the result to each of the processors. This mechanism can be thought of as computation free‡. It can be used to construct a new projection operator based on equidistant sampling of the X-ray path (see figure 1(b)). This approach reduces computation time by removing the need for mathematical operations to compute distance inside voxels. However, the sampling rate is generally best chosen to be half a voxel or lower to increase accuracy[37], so a larger number of samples is needed which takes longer. Overall, the computation times of both methods sketched in figure 1 are similar[38].

‡ Not an accurate description as memory access is the most computationally expensive operation on a GPU. However, if one already needs to access memory, accessing with interpolated values will not add to the computation time.
A General Method for Motion Compensation in X-ray Computed Tomography

The GPU acceleration software used in this work is freely available from the Tomographic Iterative GPU-based Reconstruction (TIGRE) Toolbox\cite{39}. TIGRE is an open-source software repository which it is hoped will act as a platform to help bridge the gap between academics and clinicians and lead to the technology being adopted more widely.

2.3. Alternative motion modelling approach

Motion in tomography is a problem not only in X-ray modalities. Phase space tomography\cite{40} is a hybrid algorithm that combines particle tracking in a computer model of a synchrotron with iterative ART to reconstruct an image of the population of a bunch of particles circulating in the accelerator. The particle motion involves non-linear rotation and is non-cyclic, but a 1D projection of the distribution can be completely acquired as a single snapshot on one turn of the machine. By tracking test particles to gain a knowledge of how the geometry of the 2D image plane (longitudinal phase space) deforms, the information in all the discrete time slices acquired over many turns can be translated back to the same instant and tomographically combined in a single image. Conceptually this means adding the motion information to the geometry of the model – in the $A$ matrix – with which the problem is posed rather than inserting it somehow into the mathematics of the tomography by which a solution is found.

The concept can be transferred to standard absorption tomography, but the idea of following the motion of test points in a 3D image volume simply does not scale from the 2D tracking used for the modest number of pixels typical in phase space tomography. It would lead to unreasonable computing times and memory requirements. Instead, motion is modelled as a different effect with the same mathematical result. Thus a shift upwards of the voxels in a region of the image is modelled as a local shift downwards of...
the X-ray paths through a regular voxel mesh that remains frozen in the state at which the reconstruction is made. The motion can be arbitrary provided it does not send any voxels out of the image or add new ones to it.

The idea is illustrated in figure 2 where two different states of motion are sketched. In order to reconstruct at the initial time (a), the measurement at detector element \(d_k\) at later time (b) is back projected along the deformed line of response in (a) and so combined with the measurement at element \(d_j\) made directly at the earlier time. Likewise, if the later time is chosen for the reference state at which to reconstruct, the integral over the deformed line of response in (b) provides the attenuation figure needed to project onto \(d_j\) in order to iterate the measurement at that element which was actually made at time (a). Note that the paths are not only bent (or ”warped”) but also stretched in some places and compressed in others.

2.4. Warped projection operator in a GPU

The warped X-ray paths cannot be easily translated into classic projection operators. Evaluating the length of a curvilinear path accurately inside every voxel it traverses would require a set of complex numerical methods which would inevitably increase the computation time significantly. Instead, the uniformly sampled projection method is used, for which the ray-warping operation becomes a straightforward modification of the code. Rather than sampling at each image coordinate along a straight line, the vector field at that coordinate is first added and then the image is sampled.

The pseudocode in a GPU is outlined in algorithm 1. One thread per X-ray is launched to compute \(N_{ray}\) threads organized in \(\text{divV} \times \text{divU}\) blocks. This means that instead of computing each of the path integrals in lexicographical order, small subsets of blocks (in the detector) are computed together. This decreases the memory latency.
and increases the overall speed of the kernels by up to 300% in our tests. For more information about GPU memory access and optimal X-ray indexing we refer the reader to the work by Chou et al \cite{41}. As information about the texture cache is proprietary, empirical tests were made to find the best size for \text{divV} \text{ and divU}. These showed \(32 \times 32\) to be fastest on an NVIDIA Tesla 40k. Note that there are reportedly faster structures for GPU kernels \cite{33}, but our tests showed no such improvement so we have stuck to the simplest approach of one thread per ray.

**Algorithm 1** Motion interpolated X-ray projection

1. Precompute geometric constants

**Launch:** \(N_{ray}\) threads organized in \(\text{divU} \times \text{divV}\) blocks

2. for X-ray path do
3. \hspace{1em} Compute \([x, y, z]\) sample position
4. \hspace{1em} Sample \([\text{DVF}_x, \text{DVF}_y, \text{DVF}_z] = \text{DVF}(x, y, z)\)
5. \hspace{1em} \text{Sum} += \Delta l \cdot \text{Image}(x + \text{DVF}_x, y + \text{DVF}_y, z + \text{DVF}_z)
6. end for

End Kernel

Once an X-ray is selected, it is sampled over its path at every user-provided \(\Delta l\) step length. As previously mentioned, any real-valued coordinates, \(p = [x, y, z]\), will yield a sample value using texture memory. The point is first sampled over the relevant DVF, yielding the change in coordinates of that specific point, then the image is sampled at the new displaced coordinates, \(q = p + \text{DVF}\). The DVFs needed are those that describe the deformation from all the shifted states back to the reference one of the reconstruction. Note that this description must be provided in the coordinate system of the reference state, so that \(q - p\) is the extent of the inter-phase motion arriving at \(p\) rather than originating from it. This makes it more complicated than the forward mappings from the reference state to each of the others.

### 2.5. Warped back projection operator in a GPU

Warped back projection is simpler to compute as shown in the pseudocode outlined in algorithm \[2\]. First, the standard back projection is computed using memory latency aware voxel ordering. Then, a second GPU kernel is launched with the same thread and block sizes and, for each voxel, a sample of the relevant shifted image is taken at \((x + \text{DVF}_x, y + \text{DVF}_y, z + \text{DVF}_z)\). This last step is basically a 3D interpolation. It is important to note that the DVFs used here are not the same as those for projection. And, although they are the inverse of each other, that inversion is not nearly as mathematically straightforward as a change of sign.

Several reportedly faster back projection operator structures exist in the literature. We found that the one by Zinsser et al \cite{34} can lead to execution speeds up to four times faster, but only when multiple back projections are used at the same time in the kernel. Since the vector fields of the motion-compensated algorithm would generally be different
**Algorithm 2** Motion X-ray back projection

1: Precompute geometric constants

**Launch:** $N_{\text{voxel}}$ threads organized in $\text{divX} \times \text{divY} \times \text{divZ}$ blocks

2: Compute $[u, v]$ detector position in line with a source-voxel direction

3: Sample Detector $(u, v)$

4: Compute corresponding weight $w$

5: WarpedImage$= w \times \text{Detector}(u, v)$

**End Kernel**

**Launch:** $N_{\text{voxel}}$ threads, organized in $\text{divX} \times \text{divY} \times \text{divZ}$ blocks

6: Sample $[\text{DVF}_x, \text{DVF}_y, \text{DVF}_z] = \text{DVF}(x, y, z)$

7: Image$= \text{WarpedImage}(x + \text{DVF}_x, y + \text{DVF}_y, z + \text{DVF}_z)$

**End Kernel**

for each back projection, this kernel structure will not accelerate the computation. However, as the drawbacks of using the more complex structure are negligible, it has nevertheless been implemented in our code. Thus, if a case is treated in which there is no motion to compensate, it will run more quickly.

### 2.6. Motion-compensated algorithm

Using any iterative CT reconstruction algorithm with warped projection and back projection is simple once the DVFs are known. But first, a reference image is needed and the DVFs from this reference state to the shifted states must be computed, together with the inverse DVFs back to the reference. Once the DVFs are known, the only modifications to a given algorithm are minor. Whenever the projection operator is used, the warped projection operator with the inverse DVFs should be used instead. Likewise, for back projection, the warped version with the forward DVFs should replace the standard code. This allows motion to be included in both operators inside any algorithm independently of the mathematics that invokes those operators.

### 3. Results

In order to validate the motion-compensation algorithm, two different tests were performed. The results are presented in this section. First, a proof of principle is established by subjecting a digital thorax phantom to a well-defined, if somewhat contrived deformation with time. In this case, the expected image at any instant is perfectly known, but despite this the inverse motion map must still be computed numerically and is necessarily only approximate. The motion moves all voxels within the image (but not the image boundaries) and the amplitude of the deformation is made substantially larger than any real movement in a breathing patient. A second test is performed using clinical 4D-CT images, where the motion is only approximately known. Nevertheless, even an approximate motion model can be exploited to significant
A General Method for Motion Compensation in X-ray Computed Tomography

3.1. Arbitrary deformation of a digital phantom

The phantom used is a digital representation\textsuperscript{42} of a human thorax comprising $256^3$ cubic voxels. Motion is simulated according to equation 5 using 100 equidistant discrete steps of an arbitrary time scale $t$, which runs from 0 to 1, and using $L = 128$. This creates a steadily increasing sinusoidal deformation in all three spatial dimensions, displacing all voxels throughout the volume of the phantom. Only the boundaries at the faces of the cube and the three perpendicular mid-planes that intersect at its centre remain unshifted. Figure 3 shows a cross-section of the undeformed reference image at $t = 0$ and of the deformed image at $t = 1$. Not only are there no static regions, but the deformation is huge compared with real breathing\textsuperscript{43}, locally approaching three times that for a typical size of thorax.

$$V(x, y, z) = (8 \times t \times \sin(x \cdot \pi/L) \sin(y \cdot \pi/L) \sin(z \cdot \pi/L),$$

$$8 \times t \times \sin(x \cdot \pi/L) \sin(y \cdot \pi/L) \sin(z \cdot \pi/L),$$

$$8 \times t \times \sin(x \cdot \pi/L) \sin(y \cdot \pi/L) \sin(z \cdot \pi/L)) \quad (5)$$

CBCT data are generated comprising 100 projections, one for each time step and covering a full circle. In order to benchmark the results, 100 CBCT projections are also generated from the undeformed $t = 0$ data alone, providing a comparable dataset for
reconstruction but from which motion is entirely absent. The images are reconstructed using 100 iterations of the SART algorithm.

Figure 4 shows cuts of three different CBCT reconstructions of the reference state. Motion is not included in the first and motion compensation is applied only in the last. The latter reconstruction is qualitatively almost identical to the static one despite the necessarily approximate inverse deformation map. However, the error in the inverse DVF, which is computed using a kernel splatting technique, is very small with more than 95% of the errors less than 0.05 voxels in absolute distance. This is typical of the numerical error that one can expect starting from a forward DVF that is well-known.

The error between the original phantom and each of the reconstructions is shown in figure 5. The image in the uncompensated dynamic case is highly saturated in various places, whereas the motion-corrected image has only slightly higher error overall than the static reconstruction. One would expect more iterations to reduce the error further. Cuts in the other planes are found to be qualitatively very similar.

Figure 4. Transverse cross-section of the CBCT reconstruction made (a) using SART in the absence of motion; (b) with motion using uncompensated SART; (c) with motion using compensated SART. The colour scale is in arbitrary units.
A General Method for Motion Compensation in X-ray Computed Tomography

Figure 5. Transverse cross-section of the difference between the known phantom (figure 3(a)) and the CBCT reconstruction made (a) using SART in the absence of motion; (b) with motion using uncompensated SART; (c) with motion using compensated SART. The colour scale is in arbitrary units and is significantly enhanced with respect to that of figure 4.

Tellingly, the difference between the static reconstruction and the motion-compensated one, as shown in figure 6, is quasi-uniform with no large differences at the boundaries between tissue types. This means that, while the error may be larger in the motion-compensated case, it will not prevent the correct delineation of an organ or a tumour.

Figure 6. Transverse cross-section of the difference between the static reconstruction of figure 5(a) and the motion-compensated one of figure 5(c). The colour scale is in arbitrary units.
This test demonstrates that the new method can handle arbitrary, non-cyclic motion and that it works well even when the inverse deformation map is not perfectly known.

3.2. Real patient data

The second test is performed using clinical data with precomputed DVFs, which are not entirely accurate. It is important to note that no real CBCT data are used, only 4D-CT image data. These are taken from the so-called POPI-model[44] and are publicly available[45]. The data comprise ten 3D-CT images (labelled from “0” to “9”) of the thorax equally spaced during the breathing cycle of a single patient. Additionally, motion maps generated by two different methods are provided by the authors describing the inter-phase motion of the voxels. We choose to use the maps that are generated by the parametric method for no specific reason as, statistically, both methods are reported to have similar errors. And we choose to use the 3D-CT image labelled “1” as the reference state to be reconstructed because the authors provide the motion vectors from this state to all the others. The reference state of the thorax can be seen in figure 7. The particular feature of a tumour is highlighted.

![Figure 7](image)

**Figure 7.** 3D-CT scan of a lung radiation therapy patient at breathing phase “1” cut to show the tumour (located inside the green rectangle) in the (a) transverse, (b) coronal and (c) sagittal planes. The colour scale is in arbitrary units.

In order to simulate CBCT data, projections are generated from the phase-binned 3D-CT images. No extra noise is added as the images themselves are already noisy. Ten equally spaced projections covering a full circle are generated for each of the ten states. Note that the DVFs are not used to approximate continuous movement as this would compromise the independence of the test that the quality of any subsequent motion-compensated reconstruction employing those DVFs affords. In order to benchmark the results, 100 CBCT projections are simulated from the state “1” data alone, providing a comparable dataset for reconstruction but from which motion is essentially absent.

There are four significant error sources inherent in the original 4D-CT data before a CBCT reconstruction is even attempted. There is that due to phase binning, which is
particularly significant in the regions that move the most. This is visible in figure [7] for example in the lower boundary of the lungs. Another error source lies at the top and bottom (in the cranial-caudal direction) of the 3D-CT images. Due to the original data acquisition and reconstruction techniques, the images have increased noise-like errors in their extrema and, because of the randomness of these errors, the images are not entirely consistent with each other in these regions. The third main error in the source data is the inaccuracy of the DVFs in some areas. Finally, the inverse of the DVFs will have additional errors due to the numerical method used to invert them.

Given these numerous sources of error, one can expect streaking artifacts in addition to the usual random noise exhibited in any reconstruction. As previously mentioned, it is a strength of the new motion-compensation method that it can be applied to any iterative algorithm, so one can be employed that reduces such artifacts by, for example, minimizing the total variation (TV). We elect to use both the well-known SART and the TV algorithm ASD-POCS[7] in this test.

Figure [8] shows cuts of four different CBCT reconstructions of the reference state. Motion is included in all except the first, but motion compensation is applied in only the last two. The SART algorithm is used in all cases except the last, which is processed with ASD-POCS. The second, uncompensated image has lost much of the detail inside the lungs, while the corrected algorithms, even with all the errors in the DVFs and data, reconstruct the tissue boundaries inside the thorax with higher accuracy. The last, TV case is particularly good. This is even more evident in figure [9] where the difference between the original 3D-CT image and each reconstruction is shown. One can see that the error is smaller overall in the motion-compensated cases, for which the discrepancy where the tumour is located is barely visible. Cuts in the coronal plane, which is the one containing the largest movement of the lungs, underscore the remarkable improvement in the motion-compensated images (see figures [10] and [11]).

![Figure 8](image_url)
Figure 9. Transverse cross-section of the difference between the original 3D-CT image (figure 7(a)) and the CBCT reconstruction made (a) using SART in the absence of motion; (b) with motion using uncompensated SART; (c) with motion using compensated SART; (d) with motion using compensated ASD-POCS. The colour scale is in arbitrary units and is significantly enhanced with respect to that of figure 8.

Figure 10. Zoom on the region where the tumour is located in a coronal cross-section of the CBCT reconstruction made (a) using SART in the absence of motion; (b) with motion using uncompensated SART; (c) with motion using compensated SART; (d) with motion using compensated ASD-POCS. The colour scale is in arbitrary units.
A General Method for Motion Compensation in X-ray Computed Tomography

Figure 11. Zoom on the region where the tumour is located in a coronal cross-section of the difference between the original 3D-CT image (figure 7) and the CBCT reconstruction made (a) using SART in the absence of motion; (b) with motion using uncompensated SART; (c) with motion using compensated SART; (d) with motion using compensated ASD-POCS. The colour scale is in arbitrary units and is significantly enhanced with respect to that of figure 10.

For a more quantitative assessment, the resultant images are cropped around the tumour taking a $36 \times 36 \times 26$ subset of voxels in the anterior-posterior, lateral and cranial-caudal directions (as indicated by the green rectangles in figure 7). Then, in order to evaluate the quality of the reconstruction inside this box, three different indices are used to compare the original 3D-CT image with the four CBCT reconstructions of this test.

- **Root Mean Square Error (RMSE)** is defined
  \[
  \text{RMSE} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (\hat{p}_n - p_n)^2},
  \]
  where $\hat{p}_n$ is a voxel in the original image, $p_n$ a voxel in the reconstructed one and $N$ is the number of voxels. A larger value means more difference.

- **Universal Quality Image (UQI)**\cite{46} is a widely used index and is defined
  \[
  \text{UQI} = \frac{2\text{cov}(\hat{\mu}, \mu)}{\hat{\sigma}^2 + \sigma^2} \cdot \frac{2\hat{\mu} \mu}{\hat{\mu}^2 + \mu^2},
  \]
  where \text{cov} is the covariance function and $\hat{\mu}, \mu$ are the means and $\hat{\sigma}^2, \sigma^2$ the variances of the original and reconstructed images, respectively. UQI yields a value between 0 and 1, increasing with increasing similarity.

- **Segmentation mismatch.** A segmentation value using Otsu’s method\cite{47} is computed for the original image and all voxels in the reconstructed image are identified as lying inside or outside the tumour according to that value. Then the number of voxels that are mislabelled by that segmentation is counted. A larger value means more difference.
The results for each index applied to the subset of voxels in the region of the tumour is shown in table 1. As expected, the SART reconstruction even in the absence of motion in the data does not reproduce the original image with any great accuracy as the data are still not perfect and CBCT reconstruction has its limitations. Nevertheless, it is a sufficiently good reconstruction to take as a benchmark for the others. Indeed, it should be stressed that, although the CBCT data here are artificial, in a practical scenario the equivalent static dataset would require a full order of magnitude more radiation dose to acquire than the dynamic one because of phase binning. In comparison with this static CBCT case, uncompensated SART applied to the dynamic data has considerably worse reconstruction quality, missing almost 10% of the tumour by segmentation. Motion-compensated SART performs significantly better, getting closer to the static SART values. Finally, the motion-compensated ASD-POCS results are very similar to those of the reconstruction without any motion. One would expect more advanced TV algorithms to perform even better.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>UQI</th>
<th>Seg. mismatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>SART without motion</td>
<td>67.18</td>
<td>0.9656</td>
<td>1108 (3.288%)</td>
</tr>
<tr>
<td>SART with motion</td>
<td>172.12</td>
<td>0.7617</td>
<td>3315 (9.838%)</td>
</tr>
<tr>
<td>SART motion-compensated</td>
<td>109.84</td>
<td>0.9077</td>
<td>1505 (4.466%)</td>
</tr>
<tr>
<td>ASD-POCS motion-compensated</td>
<td>82.72</td>
<td>0.9451</td>
<td>1284 (3.811%)</td>
</tr>
</tbody>
</table>

This test demonstrates that the new method can be used in a clinical context even if the motion due to breathing is only approximately known.

4. Discussion

We have demonstrated a significant improvement in CBCT image quality by removing motion artifacts using a novel approach to motion compensation. The resultant images still have some error compared with a static reconstruction, but critically, tissue boundaries are resolved with much greater accuracy than when motion is ignored.

One of the greatest strengths of the new method is that it employs all the projection data to reconstruct an image, reducing the X-ray dose to the patient. It is also completely algorithm independent; its novelty lies in a modelling approach, which in principle can be applied in conjunction with any static reconstruction algorithm. In fact, most of the motion-compensation ideas present in the literature and reviewed at the beginning of this work could incorporate the method. As some of these rely on refining DVFs, then, instead of using those DVFs to generate a deformed image from a prior high-quality one, they could be used to reconstruct images from the real acquired data. Others rely on temporal reconstruction constraints, where the images at successive time steps are regularized to look similar to their neighbours. Again, such techniques can
be used in combination with the new motion compensation because the latter permits any state of the motion to be reconstructed. Indeed, one could generate an X-ray video of the patient breathing if enough time steps are reconstructed and, as these frames are computationally independent, they could be processed in parallel. And none of these extra images would require any extra dose for the patient.

One drawback of the method is the need for an accurate DVF for each of the projections, which can lead to big memory requirements and an increased preprocessing time. But it has been established in phase space tomography that it is possible to trade off the accuracy of the maps against an increased number of iterations and that some parameters in the motion model itself can be refined by their influence on convergence\[48]\[49]. More speculative is the idea of ”bootstrapping” the DVFs without starting from any high-resolution images. One can imagine repeatedly subdividing the CBCT data into more and more motion phases, and thus iterating both the images and the DVFs themselves at each subdivision, whilst still using all the data for each reconstruction by interpolating between DVFs until there are enough of these to describe the motion in sufficient detail. This would be very heavy computationally and there is no guarantee of convergence.

As presented here, the method takes only 2 to 3 times longer than a standard iterative reconstruction algorithm due to the use of GPUs for the motion-compensated projection and back projection operators. So the computational penalty is not large. It is important to note that we used DVFs of the same size as the images (they are generally smaller) and the code was not highly optimized. Careful tuning should lead to appreciably faster execution.

We consider that this motion correction method could have a genuine impact in image-guided radiation therapy even though it is not yet at a clinical stage (mainly due to computation time). Better diagnostic imaging offers the prospect of less collateral damage to healthy tissue and increased survival rates. Ultimately, it might be possible to steer a particle therapy beam in real time to follow a moving target.

Future work will include testing with more data and in combination with other algorithms and evaluating the DVF error tolerance. Improvements in computation time can be expected.

5. Conclusions

A novel approach to motion-compensated CT imaging has been demonstrated that uses the entire dynamic set of acquired data to reconstruct an image that is effectively rendered static, thereby considerably reducing motion artefacts. This introduces the possibility of a significant reduction in radiation dose during imaging and treatment. The method requires a knowledge of the deformation vector fields describing the motion between the reconstructed state and all others during the acquisition time span. It removes much of the unwanted artefacts due to motion even when that knowledge is imperfect. The method can be used in combination with most, if not all, existing static
and 4D iterative reconstruction techniques. Motion compensation based on these ideas is being rolled out to the algorithms available from the open-source TIGRE Toolbox[39]. The potential clinical benefits are clear.

6. Acknowledgements

The authors would like to thank EPSRC for helping to fund this PhD work and NVIDIA for the hardware donation of a GPU. Special thanks go to Robert Bryll and Imanol Luengo for interesting discussions and for speeding up the GPU code.
Bibliography


A General Method for Motion Compensation in X-ray Computed Tomography


A General Method for Motion Compensation in X-ray Computed Tomography

Radiology and Nuclear Medicine, 2011.


