Published in final edited form as:

Ann Biomed Eng. 2008 December ; 36(12): 1942–1960. doi:10.1007/s10439-008-9564-2.

MUlti-Dimensional Spline-Based Estimator (MUSE) for Motion Estimation: Algorithm Development and Initial Results

Francesco Viola, Ryan L. Coe, Kevin Owen, Drake A. Guenther, and William F. Walker
Department of Biomedical Engineering, University of Virginia, 415 Lane Rd., MR5 room 2125, Charlottesville, VA 22908, USA

Abstract

Image registration and motion estimation play central roles in many fields, including RADAR, SONAR, light microscopy, and medical imaging. Because of its central significance, estimator accuracy, precision, and computational cost are of critical importance. We have previously presented a highly accurate, spline-based time delay estimator that directly determines sub-sample time delay estimates from sampled data. The algorithm uses cubic splines to produce a continuous representation of a reference signal and then computes an analytical matching function between this reference and a delayed signal. The location of the minima of this function yields estimates of the time delay. In this paper we describe the MUlti-dimensional Spline-based Estimator (MUSE) that allows accurate and precise estimation of multidimensional displacements/strain components from multidimensional data sets. We describe the mathematical formulation for two- and three-dimensional motion/strain estimation and present simulation results to assess the intrinsic bias and standard deviation of this algorithm and compare it to currently available multi-dimensional estimators. In 1000 noise-free simulations of ultrasound data we found that 2D MUSE exhibits maximum bias of 2.6×10^{-4} samples in range and 2.2×10^{-3} samples in azimuth (corresponding to 4.8 and 297 nm, respectively). The maximum simulated standard deviation of estimates in both dimensions was comparable at roughly 2.8×10^{-3} samples (corresponding to 54 nm axially and 378 nm laterally). These results are between two and three orders of magnitude better than currently used 2D tracking methods. Simulation of performance in 3D yielded similar results to those observed in 2D. We also present experimental results obtained using 2D MUSE on data acquired by an Ultrasonix Sonix RP imaging system with an L14-5/38 linear array transducer operating at 6.6 MHz. While our validation of the algorithm was performed using ultrasound data, MUSE is broadly applicable across imaging applications.

Keywords

Motion estimation; Image registration; Strain estimation; Elasticity imaging; Splines

Introduction

Motion estimation is critical to many modern signal-processing algorithms. In medical ultrasound, for example, it is employed in blood flow estimation,^{2,4,9} phase aberration correction,^{11,20,24} tissue elasticity estimation,^{21,32} and a number of other algorithms. In cell biology, it has been applied to provide better understanding of cellular and intracellular systems, by tracking movement of proteins across cell membrane,^{15,18} myosin-driven movement of actin filament,³⁵ cytoskeleton network formation, etc.⁸ Because of its central

Address correspondence to Francesco Viola, Department of Biomedical Engineering, University of Virginia, 415 Lane Rd., MR5 room 2125, Charlottesville, VA 22908, USA. Electronic mail: fv7d@virginia.edu.

significance, estimator accuracy, precision, and computational cost are of critical importance to these and numerous other applications.

While early tracking was performed manually using tracing paper, the advent of computer assisted techniques provided more rapid and accurate results. A plethora of algorithms have been developed, mostly drawing from RADAR, SONAR, geophysics, and medical imaging applications. In this paper we will focus on the algorithms, also called pattern-matching algorithms, that estimate changes in positions by comparing a sequence of images to a reference image.

The most straightforward approach is shown in Fig. 1a, where two-dimensional (2D) data sets are used as examples. In this approach a pattern matching method, indicated in the “Compute Pattern Matching Function” block, is used to test each possible overlap between I_1 and I_2 to produce c_{12} . This data set indicates the quality of the match between the two data sets at various relative shifts. This function is then searched to locate the position of the best match, which indicates the 2D motion $[m_0, n_0]$. A great number of pattern matching techniques are currently employed, each offering tradeoffs between computation complexity, hardware implementation cost, precision, and accuracy.^{10,12,16,17,27,29} Some of the algorithms commonly used are Normalized Correlation, Non-Normalized Correlation, One-bit Correlation, Hybrid-Sign Correlation, Minimum Sum Squared Differences (SSD), and Minimum Sum Absolute Differences (SAD). While the strategy shown in Fig. 1a has the advantage of being straightforward, its performance is fundamentally limited by the sampling interval. The motion estimates will be quantized at this interval, resulting in estimator bias and variance that are significantly higher than theoretically achievable limits.

The impact of the finite sampling interval can be reduced by employing data interpolation before computing the pattern matching function. This approach, shown in Fig. 1b, allows finer estimation, although the result will still be quantized at a sampling interval determined by the interpolation rate. The computational cost of this approach can be significant. In fact, we must not only perform two interpolation operations, but the pattern matching function must also be computed at a higher sampling rate. The computational costs of pre-pattern matching interpolation can be mitigated by performing post-pattern match curve fitting, as shown in Fig. 1c. In this approach the discrete pattern matching function is used to determine parameters describing a continuous pattern matching function. Analytical methods can then be applied to this continuous-time function to determine the shift producing the best match. Because this approach works with a continuous function, the resulting motion estimate is no longer forced to lie on discrete samples. Several curve-fitting methods have been described. The easiest and most widely used method consists of fitting a parabola around the peak of the pattern matching curve.^{6,14,16} Geiman *et al.* have proposed another curve-fitting algorithm called Grid Slopes, which has been used to estimate sub-sample shifts for two-dimensional blood velocity estimation.^{3,13} Although the computational cost of these methods is often less than that of pre-correlation interpolation, they usually suffer relatively high bias and variance because they use the coarsely sampled pattern matching function as the basis of the continuous “correlation” function. The strengths of the previously described algorithms can be combined by utilizing a combination of pre-pattern match and post-pattern match interpolation, as shown in Fig. 1d. This approach can achieve a relatively low bias and variance, in part because it produces a continuous estimate. This method typically exhibits a high computational cost however because it requires multiple interpolation operations and the pattern matching function must be evaluated for highly sampled signals. Thus, while the performance of this technique is attractive, it is precluded from use in some applications because of its high computational cost.

Our laboratory has recently described a novel one-dimensional displacement and strain estimator with very low bias and variance, which is described schematically in Fig. 1e.³⁰ In

this method we use cubic splines to produce a continuous representation of the reference signal. This continuous time representation is then input to the pattern matching function along with the discrete samples of the second signal. This yields an analytical pattern matching function that can easily be processed analytically to yield a continuous motion estimate. A spline-spline method whereby both the reference and delayed signals are represented analytically with splines has also been proposed^{22,30} and is depicted in Fig. 1f for the multi-dimensional case. In this paper we extend our original algorithm to yield a general multi-dimensional displacement and strain estimator, which we call MUlti-dimensional Spline-based Estimator, or MUSE. Note that while spline-based image registration has previously been described, published techniques are limited by the use of a separable spline model, and use of purely numerical approaches to motion estimation.^{25,26,28}

Multi-Dimensional Spline-Based Estimator (MUSE)

MUSE begins by forming a multi-dimensional (typically 2D or 3D) cubic spline representation of the reference data set utilizing a novel strategy to reduce computational cost and memory requirements. For simple displacement estimation the algorithm then minimizes an analytical error function between the reference (splined) and displaced (sampled) data set. The location of this minimum indicates the displacement between the two data sets. In the following sections we describe the mathematical formulation of a 2D and 3D spline and the development of MUSE. Further, we will show how the algorithm may be altered to incorporate estimation of companding coefficients for compression, stretching, and shear.

Formulation of the 2D Spline Representation

A 2D non-separable spline is a piece-wise continuous representation of a surface. Each patch within the surface at index (i, j) can be described by the following 16 coefficient bicubic polynomial:

$$\begin{aligned} f_{i,j}(x,y) = & 0K_{i,j} + 1K_{i,j}x + 2K_{i,j}y + 3K_{i,j}x^2 + 4K_{i,j}y^2 \\ & + 5K_{i,j}xy + 6K_{i,j}x^3 + 7K_{i,j}y^3 + 8K_{i,j}x^2y \\ & + 9K_{i,j}xy^2 + 10K_{i,j}x^3y + 11K_{i,j}xy^3 + 12K_{i,j}x^2y^2 \\ & + 13K_{i,j}x^3y^2 + 14K_{i,j}x^2y^3 + 15K_{i,j}x^3y^3 \end{aligned} \quad (1)$$

Although there is substantial literature describing the use of splines for signal and image interpolation, there is relatively little published work describing the formulation of non-separable 2D splines (or multi-dimensional splines in general). The most intuitive approach is to form a large set of simultaneous equations describing requirements for continuity with the sampled data, data and derivative continuity between spline patches, and appropriate boundary conditions.²³ Unfortunately such a thorough approach requires solving a large set of coupled equations for a total of $(M-1) \times (N-1) \times 16$ coefficients, where M and N are the number of samples in the two image dimensions. In our experience testing a variety of approaches to this problem, direct formulation and solution of such systems of equations is too costly with regard to memory and computing power to be amenable to most applications.

Unser's group previously proposed a 2D interpolation method based on the formulation of 1D cubic B-splines.^{25,28} Although this approach is limited to forming a separable multi-dimensional spline, the cubic B-spline coefficients can be determined very efficiently using simple infinite impulse response (IIR) filters.

Another approach, originally presented in the computer graphics literature, is based on the 2D Catmull-Rom spline.⁵ This is a non-separable bicubic polynomial formulated such that the tangent at each data point is calculated using the previous and next point on the spline. These

splines are “locally” defined since every patch of the spline is dependent upon a small sub-set of data samples.

We have developed a method of formulating a non-separable 2D spline from a series of simple 1D splines. Although the full non-separable 2D spline is a 16-parameter polynomial, as shown in Eq. (1), the approach proposed here yields a 15-parameter polynomial where the highest order term (i.e., the coefficient multiplying x^3y^3) is forced to zero, as shown in Table 1. We believe this is a valid assumption since the x^3y^3 term describes rapid oscillations in the data, which are typically not present due to smoothness and sampling frequency constraints.

This proposed algorithm consists of the following steps and it is described mathematically in the appendix:

1. Form 1D splines along the x direction of the image, as shown in Fig. 2b.
2. Form 1D splines along the y direction of the image, as shown in Fig. 2c.
3. Evaluate 1D splines in the y direction to obtain all midpoints.
4. Form 1D splines through all the midpoints computed under step 3, as shown in Fig. 2d.
5. Evaluate 1D splines in the x direction to obtain all midpoints.
6. Form 1D splines through all the midpoints computed under step 5, as shown in Fig. 2e.
7. Determine 2D spline coefficients by forcing equality to the 1D splines determined under steps 1, 2, 4, and 6.

Each of the steps outlined above is modest in terms of computation and memory costs. The 1D spline along the x direction can be expressed mathematically by:

$$p_j(x) = \left\{ \begin{array}{ll} 0 \leq x < \chi & {}_0\alpha_{1,j} + {}_1\alpha_{1,j}x + {}_2\alpha_{1,j}x^2 + {}_3\alpha_{1,j}x^3 \\ \vdots & \vdots \\ (i-1)\chi \leq x < i\chi & {}_0\alpha_{i,j} + {}_1\alpha_{i,j}x + {}_2\alpha_{i,j}x^2 + {}_3\alpha_{i,j}x^3 \\ \vdots & \vdots \\ (M-2)\chi \leq x < (M-1)\chi & {}_0\alpha_{M-i,j} + {}_1\alpha_{M-1,j}x + {}_2\alpha_{M-1,j}x^2 + {}_3\alpha_{M-1,j}x^3 \end{array} \right\} \quad (2)$$

where χ is the sample interval along the x direction. Note that this expression is simply a 1D cubic spline representation, and as normally determined, will pass through each of the original image points. A total of $2N-1$ 1D spline representations will be formed along the x direction. In this paper, the cubic spline representations were computed using the *spline* command in MATLAB (MathWorks Inc., Natick, MA). Alternatively, finite impulse response (FIR) filters or IIR filters could be used to obtain the 1D spline coefficients with significantly reduced computational costs.^{28,31}

Similar expressions can be found for the 1D spline along the y direction:

$$q_i(y) = \left\{ \begin{array}{ll} 0 \leq y < \delta & {}_0\beta_{i,1} + {}_1\beta_{i,1}y + {}_2\beta_{i,1}y^2 + {}_3\beta_{i,1}y^3 \\ \vdots & \vdots \\ (j-1)\delta \leq y < j\delta & {}_0\beta_{i,j} + {}_1\beta_{i,j}y + {}_2\beta_{i,j}y^2 + {}_3\beta_{i,j}y^3 \\ \vdots & \vdots \\ (N-2)\delta \leq y < (N-1)\delta & {}_0\beta_{i,N-1} + {}_1\beta_{i,N-1}y + {}_2\beta_{i,N-1}y^2 + {}_3\beta_{i,N-1}y^3 \end{array} \right\} \quad (3)$$

where δ is the sample interval in the y direction. A total of $2M - 1$ 1D spline representations will be formed along the y direction. The 1D splines are then used to determine the 2D spline coefficients as described in the appendix.

An alternative approach sets four of the original 16 coefficients to zero, thus formulating a 12-parameter non-separable spline. This can be accomplished using only steps 1, 2, and 7 above. The coefficients set to zero are highlighted in Table 2.

In Fig. 3, we analyzed the performance of the proposed 15- and 12-parameter spline methods in reconstructing a decimated data set. We also compared the two algorithms to other well-established spline-based methods, such as the 2D B-spline and the non-separable 2D Catmull-Rom spline. For each condition analyzed, 1000 images of band-limited noise were formed, decimated by a factor of 16 along each direction, and then reconstructed using each of the four methods. The original images were formed by convolving a 2D field of Gaussian random noise with a 2D Nuttall smoothing window of varying size, so as to achieve a range of Nyquist ratios (i.e., $F_s/F_{\max}$ where F_s is the sampling frequency and $F_{\max}$ is the highest frequency component in the image). The images were normalized to their root mean square (RMS) value before decimation. The sum-squared error (SSE) between the original and reconstructed data sets was used to evaluate performance. The edges of the images were removed as to minimized edge effects. As shown in Fig. 3, the spline methods developed in this paper achieve better performance than the other two methods.

Formulation of the 3D Spline Representation

A 3D non-separable spline is a piece-wise continuous representation of a volume data set. Each region within the volume at index (i, j, k) can be described by the following 64 coefficient tricubic polynomial:

$$\begin{aligned} f_{i,j,k}(x,y,z) = & 0K_{i,j,k} + 1K_{i,j,k}x + 2K_{i,j,k}y + 3K_{i,j,k}z + 4K_{i,j,k}xy \\ & + 5K_{i,j,k}xz + 6K_{i,j,k}yz + 7K_{i,j,k}x^2 + 8K_{i,j,k}y^2 \\ & + 9K_{i,j,k}z^2 + 10K_{i,j,k}xyz + 11K_{i,j,k}x^2y \\ & + 12K_{i,j,k}x^2z + 13K_{i,j,k}xy^2 + 14K_{i,j,k}y^2z + \dots \\ & + 59K_{i,j,k}x^2y^2z^3 + 60K_{i,j,k}x^3y^3z^2 \\ & + 61K_{i,j,k}x^3y^2z^3 + 62K_{i,j,k}x^2y^3z^3 \\ & + 63K_{i,j,k}x^3y^3z^3. \end{aligned} \quad (4)$$

As with 2D, we can use a set of simple 1D splines to obtain an approximate 3D spline throughout the volume of interest (consisting of $M \times N \times P$ samples in the x , y , and z dimensions, respectively). This approach yields 54 of the 64 terms of a 3D cubic polynomial, where coefficients multiplying each of the terms $x^3y^3z^n$, $x^3y^n z^3$, $x^n y^3 z^3$ were set to zero (where $n = 0, 1, 2, 3$). This algorithm consists of the following steps:

1. Form 1D splines along the x direction of the volume, as shown in Fig. 4b.
2. Form 1D splines along the y direction of the volume, as shown in Fig. 4c.
3. Form 1D splines along the z direction of the volume, as shown in Fig. 4d.
4. Evaluate 1D splines in the x direction to obtain all midpoints.
5. Form 1D splines through all the midpoints computed under step 4, as shown in Fig. 4e.
6. Evaluate 1D splines in the y direction to obtain all midpoints.
7. Form 1D splines through all the midpoints computed under step 6, as shown in Fig. 4f.

8. Evaluate 1D splines in the z direction to obtain all midpoints.
9. Form 1D splines through all the midpoints computed under step 8, as shown in Fig. 4g.
10. Evaluate 1D splines through the midpoints of the splines defined in steps 4, 6, and 8, as shown in Fig. 4h.
11. Determine 3D spline coefficients by forcing equality to the 1D splines determined above.

The 1D splines along the x direction can be expressed mathematically by:

$$p_{j,k}(x) = \begin{cases} 0 \leq x < \chi & {}_0\alpha_{1,j,k} + {}_1\alpha_{1,j,k}x + {}_2\alpha_{1,j,k}x^2 + {}_3\alpha_{1,j,k}x^3 \\ \vdots & \vdots \\ (i-1)\chi \leq x < i\chi & {}_0\alpha_{i,j,k} + {}_1\alpha_{i,j,k}x + {}_2\alpha_{i,j,k}x^2 + {}_3\alpha_{i,j,k}x^3 \\ \vdots & \vdots \\ (M-2)\chi \leq x < (M-1)\chi & {}_0\alpha_{M-1,j,k} + {}_1\alpha_{M-1,j,k}x + {}_2\alpha_{M-1,j,k}x^2 + {}_3\alpha_{M-1,j,k}x^3 \end{cases} \quad (5)$$

where χ is the sample interval along the x direction. A total of $4NP - 2P - 2N + 1$ 1D spline representations will be formed along the x direction to account for the whole volume. Similar expressions can be found for the 1D splines along the y and z directions:

$$q_{i,k}(y) = \begin{cases} 0 \leq y < \delta & {}_0\beta_{i,1,k} + {}_1\beta_{i,1,k}y + {}_2\beta_{i,1,k}y^2 + {}_3\beta_{i,1,k}y^3 \\ \vdots & \vdots \\ (j-1)\delta \leq y < j\delta & {}_0\beta_{i,j,k} + {}_1\beta_{i,j,k}y + {}_2\beta_{i,j,k}y^2 + {}_3\beta_{i,j,k}y^3 \\ \vdots & \vdots \\ (N-2)\delta \leq y < (N-1)\delta & {}_0\beta_{i,N-1,k} + {}_1\beta_{i,N-1,k}y + {}_2\beta_{i,N-1,k}y^2 + {}_3\beta_{i,N-1,k}y^3 \end{cases} \quad (6)$$

$$r_{i,j}(z) = \begin{cases} 0 \leq z < \zeta & {}_0\gamma_{i,j,1} + {}_1\gamma_{i,j,1}z + {}_2\gamma_{i,j,1}z^2 + {}_3\gamma_{i,j,1}z^3 \\ \vdots & \vdots \\ (k-1)\zeta \leq z < k\zeta & {}_0\gamma_{i,j,k} + {}_1\gamma_{i,j,k}z + {}_2\gamma_{i,j,k}z^2 + {}_3\gamma_{i,j,k}z^3 \\ \vdots & \vdots \\ (P-2)\zeta \leq z < (P-1)\zeta & {}_0\gamma_{i,j,P-1} + {}_1\gamma_{i,j,P-1}z + {}_2\gamma_{i,j,P-1}z^2 + {}_3\gamma_{i,j,P-1}z^3 \end{cases} \quad (7)$$

where δ and ζ are the sample intervals in the y and z directions, respectively. A total of $4MP - 2M - 2P + 1$ splines are formed in the y directions and $4MN - 2M - 2N + 1$ splines in the z direction. Also note that, similarly to the 2D case, it is possible to generate a further reduced 3D spline using only steps 1, 2, 3, and 11 described above, for a total of 32 coefficients.

Motion Estimation

MUSE estimates multi-dimensional displacements between reference and displaced multi-dimensional data sets by localizing the minima of a pattern matching function. In the work presented here we utilize the SSE as a metric of the match quality. The key to the performance of MUSE is the use of the piece-wise continuous polynomial functions described above to analytically represent the reference data set. Considering the 3D case, a sampled reference volume of dimensions $M \times N \times P$ is represented using a series of tricubic polynomial functions. A subset of the displaced data set of dimensions $Q \times R \times S$ is also selected to track multi-dimensional motion, where $Q < M$, $R < N$, and $S < P$. Similarly to the 1D case presented in Viola and Walker,³⁰ for every possible overlap between the reference and displaced data sets the sum squared difference error function can be described mathematically as:

$$E_{l,m,n}(x,y,z) = \sum_{i=1}^Q \sum_{j=1}^R \sum_{k=1}^S (f_{i+l,j+m,k+n}(x,y,z) - s_2[i,j,k])^2 \quad \begin{matrix} l=0 \dots (M-Q)-1, \\ m=0 \dots (N-R)-1, \\ n=0 \dots (P-S)-1 \end{matrix} \quad (8)$$

where $f_{i,j,k}(x, y, z)$ is the polynomial representation of the reference volume located at indices i, j, k as described in (4), $s_2[i, j, k]$ is the displaced volume sample at indices (i, j, k) , and (l, m, n) are whole sample offsets. Note that many other error functions can be used, depending upon the computational constraints and physical challenges of the problem. The set of values $(\hat{x}, \hat{y}, \hat{z})$ that minimizes the error function in Eq. (8) is referred as a local delay estimate. While this estimate can be found by finding the roots of the gradient of the error function, in this paper we use a gradient descent method. This estimation process is repeated for the $M-Q$, $N-R$, and $P-S$ possible overlaps along the x, y, z axes, yielding a series of local displacement estimates. The local displacement estimate with overall minimum error and value between 0 and 1 for each component of the displacement vector is chosen as the global estimate, representing the displacement between the two original data sets. A schematic representation of displacement estimation using MUSE is shown in Fig. 5 for a 2D case.

Non-Rigid Transformation Estimation

An important advantage of the MUSE algorithm is its adaptability. For example, by modifying Eq. (8) as shown below, the error minimization process also estimates local 3D compression and stretching. This equation could be easily modified to constrain the aggregate lateral, axial, and elevation compression to account for an incompressible material. Similar modifications can be made to estimate shear or other complex deformations.

$$E_{l,m,n}(x,y,z,\varepsilon_x,\varepsilon_y,\varepsilon_z) = \sum_{i=1}^M \sum_{j=1}^N \sum_{k=1}^P (f_{i+l,j+m,k+n}(x-i \cdot \varepsilon_x \cdot \chi, y-j \cdot \varepsilon_y \cdot \delta, z-k \cdot \varepsilon_z \cdot \zeta) - s_2[i,j,k])^2 \quad \begin{matrix} l=0 \dots (M-Q)-1, \\ m=0 \dots (N-R)-1, \quad n=0 \dots (P-S)-1 \end{matrix} \quad (9)$$

where ε_x , ε_y , and ε_z are the companding coefficients in the three orthogonal directions. The proposed algorithm is thus particularly suited for applications in tissue elasticity estimation, for example. In one of these approaches, a mechanical deformation is applied to the surface of the tissue and ultrasound radio frequency (RF) signals are obtained before and after deformation to estimate internal displacements. Maps of local strain are then obtained from the gradient of the displacement estimates. Several groups have shown that signal decorrelation induced by external mechanical compression represents one of the fundamental limitations of this method.^{1,7,33} A number of groups have proposed companding, or signal stretching, to compensate for strain introduced decorrelation.¹ The advantage of MUSE in this case is that it can search simultaneously for both the “best” displacement estimate and the “best” companding coefficient, thus minimizing the effects of strain-induced decorrelation.

Simulated Performance

Methods for 2D Simulations

Computer simulations were performed to quantify the intrinsic bias and variance of MUSE under noise-free imaging conditions and compare it to currently used 2D motion tracking algorithms such as the Sum-Absolute-Differences (SAD) and the Sum-Squared Differences (SSD) both coupled with parabolic sub-sample estimation. While our validation of the

algorithm was performed using ultrasound data, MUSE is broadly applicable across imaging applications.

Simulated ultrasound speckle patterns were generated by convolving a 2D point spread function (psf) with a field of Gaussian amplitude distributed random scatterers. The psf was modeled as the product of a lateral Nuttall window 1.2 mm wide with an axial pulse defined as a Nuttall windowed sinusoid. The choice of the Nuttall window is arbitrary and other windows could be used, such as Gaussian, Hanning, and Hamming windows, yielding similar results to those presented here. The psf was defined with a center frequency of 7.5 MHz, a 50% fractional bandwidth, and an $f/2$ imaging system, to yield a beamwidth of approximately 400 μm . A total of 1000 unique speckle patterns were generated and sampled at 320 MHz axially and at 16.875 μm laterally. These raw datasets were trimmed to eliminate edge effects. Data sets were sub-sampled to generate shifted image pairs at all sub-sample displacements of 1/8 sample in each dimension. These sub-sampled images had lateral sampling of 135 μm and axial sampling of 40 MHz, making them consistent with current imaging systems. Reference regions were 8 samples laterally and 20 samples axially (roughly 1.1×0.4 mm). Displaced image regions were 4 samples laterally and 16 samples axially (roughly 0.54×0.3 mm). These sub-sampled images were processed by each of the three algorithms to estimate the 2D sub-sample displacement. Both the bias and standard deviation were estimated over all 1000 unique speckle patterns, at each sub-sample shift.

We also performed simulations to characterize the performance of MUSE in the presence of additive noise. The simulations were conducted with no motion between the reference and the delayed data sets, i.e. at position (0, 0) in the lateral and axial dimensions. Three independent, noise free data sets were generated as explained above. The first data set was considered as the “noise free” reference image, while the RMS energies of the second and the third were scaled relative to the first one as to achieve signal-to-noise (SNR) levels of 0, 10, 20, 30, 40, and ∞ dB. The first data set was then added to the scaled second and third data set to generate noisy reference and delayed images.¹²

In this section and the remainder of this paper, we only consider the 15-parameter spline formulation, which is described mathematically in the appendix.

2D Simulation Results

Noise free simulation results are shown in Figs. 6–8 for the three estimators. For each figure, the top row depicts the bias errors while the bottom row depicts the standard deviation of displacement estimates. Also, the bias and standard deviation errors are displayed using the same color bar scale for the axial and lateral performances. Figure 6 shows that MUSE has a maximum axial bias of less than 2.6×10^{-4} samples, which at the simulated sampling rate of 40 MHz corresponds to roughly 4.8 nm, whereas the lateral bias is on the order of 2.2×10^{-3} samples, corresponding to roughly 297 nm (the lateral sampling interval is set to 135 μm). The standard deviation of displacement estimates in both dimensions is on the order of 2.8×10^{-3} samples, or 54 nm axially and 378 nm laterally. Figure 7 shows the simulation results for the SSD with parabolic fitting. In this case, the axial and lateral biases are comparable to one another, on the order of 0.14 samples. The standard deviations in both dimensions are on the order of 0.27 samples, corresponding to 5.2 μm axially and 36 μm laterally. Similar results are depicted in Fig. 8 for the case of SAD with parabolic fitting. The bias and standard deviation of SAD estimates are slightly worse than those observed for the SSD, in agreement with previously reported work.^{29,30} The maximum bias and standard deviation values for the three algorithms are summarized in Table 3.

Figure 9 shows the performance of MUSE with additive noise in the case of no motion. The average estimates are represented as open symbols with error-bars depicting the standard

deviation calculated over 1000 independent trials. As expected, performance degrades as the SNR is reduced to 0 dB. Performance in the axial dimension is better than that observed in the lateral dimension. For an SNR value of 30 dB, the axial and lateral biases are 7.75×10^{-4} and 1.7×10^{-3} samples, respectively (corresponding to 15 and 230 nm, respectively). The standard deviations are comparable at roughly 3.6×10^{-2} samples. Although not presented here, we also examined the performance of the other two algorithms tested above and found that their standard deviation, or jitter, was higher than that observed for MUSE, consistent with 1D results presented in Viola and Walker.³⁰

Methods for 3D Simulations

Noise-free computer simulations were also performed to quantify the bias and variance of the 3D version of MUSE. Simulation methods were the same as those described above for the 2D case. Simulated speckle patterns were generated by convolving a 3D psf with a volume of Gaussian amplitude distributed random scatterers. In this case, the psf in elevation direction was modeled as a Nuttall window 1.2 mm wide. A total of 1000 unique speckle patterns were generated and sampled at 320 MHz temporally and $16.875 \mu\text{m}$ both laterally and in elevation. The data sets were then trimmed and sub-sampled along the three directions to generate reference and displaced volumes with known sub-sample shifts.

We also performed simulations in the presence of additive noise with no motion between the data sets. The methods are identical to those presented above for the 2D case.

3D Simulation Results

The results of the 3D simulations are summarized in Fig. 10. In this figure, every other plane in the elevation direction was eliminated prior to display to provide better visualization of the remaining data. The top panel of Fig. 10 shows the bias and standard deviation in the axial dimension. These two slice plots indicate a maximum bias of 2.23×10^{-4} samples and maximum standard deviation of 3×10^{-3} , corresponding to 5.7 and 57 nm respectively. The second row of images shows MUSE performance in the lateral dimension. As expected, the lateral bias oscillated in the direction of the lateral sub-sample shift. In this case the bias and standard deviation correspond to 290 and 378 nm, respectively (i.e., 2.2×10^{-3} samples for the bias and 2.8×10^{-3} samples for the standard deviation). The last series of images depict the bias and standard deviation in the elevation direction. In this case, bias varies along the elevation sub-sample shift, as evidenced by the different grayscale of the four slices. The maximum bias and standard deviation along this direction are 2.3×10^{-3} and 2.5×10^{-3} samples, corresponding to 310 and 337 nm, respectively. The maximum bias and standard deviation values of MUSE are summarized in Table 4.

Figure 11 shows the performance of 3D MUSE with additive noise in the case of no motion. The average estimates are represented as open symbols along with the standard deviation calculated over 1000 independent trials. The results shown in this figure are similar to those observed for the 2D case. For an SNR value of 30 dB, the axial bias corresponds to 3.8 nm, the lateral bias is roughly 19 nm, and the bias in the elevation dimension is 51.6 nm. The standard deviation in the axial dimension is 516 nm, whereas the standard deviations in the other two dimensions are comparable at $3.25 \mu\text{m}$.

Experimental Performance

Methods

Experiments were performed to test MUSE performance by acquiring displaced sets of RF echo data using an MM3000 mechanical translation stage (Newport Corporation, Mountain View, CA) with a stated precision of $0.1 \mu\text{m}$. RF data was acquired at $2.4 \mu\text{m}$ intervals in range

and $37.5\ \mu\text{m}$ intervals in azimuth. The transducer was mounted to a 1 in. thick Plexiglas plate using nylon cable ties. A two-axis tilt and rotation stage (M37 with micrometers, Newport Corporation) was also employed to align the transducer to the movement of the motion stage so as to keep the image plane parallel to the translation plane. The calibration procedure is described in detail below.

Full RF image frames were acquired over an 8×8 grid of positions from a 10% polyacrylamide tissue mimicking gel phantom with $50\ \mu\text{m}$ Sephadex[®] beads (Amersham, Piscataway, NJ) added as acoustic scattering agent.¹⁹ Echo data was obtained from an Ultrasonix Sonix RP imaging system (Ultrasonix Medical Corporation, Richmond, BC, Canada) with a L14-5/38 linear array transducer. The system was configured to perform dynamic receive focusing with aperture sizes selected to yield a $900\ \mu\text{m}$ beamwidth. At a lateral line sampling of $300\ \mu\text{m}$, these settings yield a lateral sampling interval of approximately $1/3$ the beamwidth. Although this beam dimension is much coarser than would be expected for most clinical imagers, it does yield a beam spacing with fractional beamwidth close to that of the simulations. Signals were sampled temporally at 40 MHz. The experimental sample intervals are $19.25\ \mu\text{m}$ /axial sample and $300\ \mu\text{m}$ /lateral sample. Calibration of the alignment system was performed using correlation coefficients obtained before and after lateral and axial translations of 10 samples in each direction. The correlation was computed over a 65 lateral by 400 axial sample region after accounting for the sample shift movement so as to compute the correlation on the same region of the phantom at each location. The axial correlation was maximized to 0.99435 after 10 sample shifts, while the lateral correlation was maximized to 0.97683 after 10 sample shifts. We hypothesize that the discrepancy between these correlations indicates that even with meticulous data acquisition, out of plane motion, not electronic noise, remains the limit on performance.

Prior to processing, received data was bandpass filtered using a 100th order FIR filter with cutoffs at 4.5 and 8.5 MHz. In this experiment, the MUSE algorithm utilized reference regions of 105 samples axially by 7 samples laterally, and displaced image regions of 79 samples axially by 5 samples laterally, corresponding to an area of $1.5 \times 1.5\ \text{mm}$. A total 6 unique speckle patterns and 60 different noise realizations were processed at each displacement, for a total of 360 trials for each of the 64 displacements.

Results

Figure 12 depicts displacement estimates performed by applying MUSE to experimentally acquired speckle data. Each dot represents one displacement estimate while the open circles represent the mean estimate at each displacement. Lateral displacement estimates are roughly $1/8$ sample, as intended by the $37.5\ \mu\text{m}$ programmed displacements. Axial displacement estimates are approximately $1/8$ sample, as would be expected for this speed of sound (i.e., 1540 m/s). The standard deviation of displacement estimates is on the order of 13.5×10^{-3} samples axially and 8.7×10^{-3} samples laterally (i.e., $0.26\ \mu\text{m}$ axially and $2.61\ \mu\text{m}$ laterally). Figure 13 depicts the displacement estimates from Fig. 12, but given in units of distance, rather than as fractions of a sample. Figure 14 shows a typical search (left panel) and kernel (right panel) data sets obtained from the acrylamide phantom.

Discussion and Conclusions

In this paper we presented a novel algorithm termed MUSE, or MULTI-dimensional Spline-based Estimator that allows precise and accurate estimation of motion using sampled data. We also showed that MUSE could be modified to allow estimation of local stretching/compression and shear, or other complex deformations. The performance of MUSE has been characterized both through simulations and experiments.

Figures 6–8 show the simulated bias and standard deviation of MUSE, SSD, and SAD, respectively. Both SSD and SAD were coupled with parabolic fitting to estimate sub-sample shifts. These alternative algorithms were chosen since they represent methods commonly used to track 2D motion in ultrasound applications. The results presented in these figures show that MUSE bias in the axial dimension is a factor of 540 lower than that of SSD and a factor of 700 lower than SAD. In the lateral dimension MUSE outperforms the two algorithms by a roughly a factor of 70. The standard deviation of MUSE is between 80 and 90 times smaller than that of either SSD or SAD in both lateral and axial directions under noise-free conditions. Figure 10 shows the simulated performance of MUSE for 3D tracking. In this case the bias and standard deviation of the estimates are similar to those observed in Fig. 6 for 2D motion.

Figures 9 and 11 depict the performance of MUSE as a function of additive noise for the 2D and 3D cases, respectively. The results obtained for the 2D case are in close agreement with the 3D results. In both figures, the case of infinite SNR corresponds to the noise-free conditions analyzed above. These data suggest that the performance of MUSE remains excellent for SNR levels equal or greater than 20 dB. This SNR level does not represent a significant barrier for most of the potential applications of MUSE. When needed, data averaging can be used to further improve SNR in noisy environments.

Experimental performance is shown in Figs. 12 and 13. In this case the standard deviations of the estimates in both the lateral and axial dimensions are larger than those observed in noise-free simulations. In the axial dimension, the standard deviation, in terms of samples, is increased by a factor of 4.9, whereas in the lateral dimension by a factor of 3. These experimental values are significantly smaller than the standard deviations observed in simulations for both the SSD and SAD. We hypothesize that the observed displacement errors are due to the combination of the limited accuracy of the motion stage, electronic noise, and alignment between image and translation planes (i.e., out of plane motion). The axial standard deviation may have been artificially increased by the weight of the Plexiglas plate causing slipping in the servo-based translation stage, thus introducing bias in the applied displacements. We also hypothesize that a slippage of the motor could have contributed to the result shown for at position (0, 0.625) in the axial and lateral directions, respectively. In these experiments, the transducer aperture was intentionally set to generate a 900 μm beam, which is much larger than that observed in typical, low F-number ultrasound imaging systems. We expect both the bias and the standard deviation of the estimates to significantly improve with improved lateral resolution.

Comparison with the results presented in Fig. 9 indicates that the standard deviations observed during the experiment correspond to an SNR of 35–40 dB, which is typical for beamformed ultrasound data.

As described in section “MUlti-dimensional Spline-Based Estimator (MUSE)”, a central idea of MUSE is the formation of a reduced order multi-dimensional spline using successive 1D splines. This allows for fast and efficient analytical representation of the multidimensional data set. 1D spline coefficients for an N samples signal are determined (in the most general case) by multiplying the N samples with a sparse matrix of dimensions $4N(N-1)$, yielding the $4(N-1)$ spline coefficients. In the 3D case, for example, $4NP + 4MP + 4MN - 4P - 4N - 4M + 3$ 1D splines are necessary to obtain full representation of the $M \times N \times P$ volume, thus incurring in $4M(M-1)(4NP - 2P - 2N + 1) + 4N(N-1)(4MP - 2M - 2P + 1) + 4P(P-1)(4MN - 2M - 2N + 1)$ additions and multiplications. An alternative solution is to determine the 1D spline coefficients using either IIR or FIR filtering. The IIR approach is implemented using two recursive filters with 2 taps each, plus four polynomial summations across each interval.^{28,31} The FIR approach can be implemented using three filters with number of taps determined by the desired level of accuracy. In our experience we have found that a filter length

of 21 taps yields accurate results down to -100 dB.³¹ Also note that while the IIR approach is particularly suitable for a general processor, the FIR approach is more appropriate when the data is pipelined, as it requires no memory storage for recursions.

Once the spline coefficients have been determined, the error function in Eq. (8) is evaluated for every integer sample overlap of the displaced samples within the reference data set. This is currently implemented using the *fminunc* function of MATLAB, which utilizes a quasi-Newton search algorithm. We have performed simulations and found that, on average, the minimum of the multi-dimensional error function is found with 4.6 iterations for the 2D case and 6.5 iterations for the 3D case for each integer overlap. As described in Viola and Walker,³¹ however, an alternate approach would be to approximate the fifth order polynomials resulting from taking the gradient of (8), with third order polynomials for which roots can be found analytically. The results of simulations and experiments in 1D have shown that this approach significantly improves computational complexity while degrading performance only marginally.³¹ Another alternative is to find the roots using the generalized companion matrix, as described in Wallack *et al.*³⁴ This approach utilizes a generalized eigenvalue problem to yield the overall set of possible roots of (8), thus avoiding getting trapped in local minima.

Further computational savings might also be achieved using a sparse search instead of evaluating the error function for every overlap of the two data sets. Since the polynomial coefficients of the multi-dimensional spline are dependent upon each other, every local estimate will carry information about the location of the global estimate. It is thus possible to speed the algorithm by computing only a subset of the available overlaps and adaptively searching for the global estimate based on the previous estimates. In simulations using 1D signal pairs, we observed a fourfold reduction in number of overlaps computed.³¹ In 2D and 3D we expect even greater gains. Finally, the computational cost can be further reduced by using a unique reference data set and comparing it to successive displaced/deformed data sets. In this case, the reference data set needs to be splined only once and used throughout successive estimations.

While we have presented a general framework for 2D and 3D MUSE, extension to higher dimensionality is also possible using the same methodology. In 4D, for example, a complete 4D spline would require the formulation of 256 independent equations for each sub region. Using the method presented in this paper it would be possible to formulate a 189 parameter 4D spline, where each term containing more than two cubic variables is set to zero (i.e., 67 coefficients are set to zero).

Although we have used 1D cubic splines to generate the piece-wise continuous representation of the data, several other polynomials might also be employed for this purpose, including higher order splines. We expect that the use of higher order splines will generate estimates with a higher degree of accuracy and precision than those obtained using cubic splines, but at higher computational costs.

The results presented here show that MUSE has lower bias and intrinsic variance than other published multi-dimensional motion estimators. MUSE has great potential in tissue elasticity imaging and other applications throughout biomedical imaging and motion tracking.

Acknowledgements

We gratefully acknowledge technical support provided by Kris Dickie and Ultrasonix Medical Corporation. This work was supported by US Army CDMRP Grant W81XWH-04-1-590 and by Grant R01 EB005433 from NIBIB and the NIH.

Appendix

This appendix presents the mathematical derivation of the 15 equations necessary to form a 2D spline given a 2D data set. The bicubic 2D spline, for the data patch represented schematically in Fig. 15 is given by equation:

$$\begin{aligned} f_{i,j}(x,y) = & 0K_{i,j} + 1K_{i,j}x + 2K_{i,j}y + 3K_{i,j}x^2 + 4K_{i,j}y^2 \\ & + 5K_{i,j}xy + 6K_{i,j}x^3 + 7K_{i,j}y^3 + 8K_{i,j}x^2y \\ & + 9K_{i,j}xy^2 + 10K_{i,j}x^3y + 11K_{i,j}xy^3 + 12K_{i,j}x^2y^2 \\ & + 13K_{i,j}x^3y^2 + 14K_{i,j}x^2y^3 \end{aligned} \quad (\text{A.1})$$

where the coefficient multiplying the term x^3y^3 has been set to zero in (A.1).

Once the 1D splines have been obtained, a set of equations can be derived to form the 2D spline based on the continuity along lines L1 to L6. Continuity along line L1 of Fig. 15 is enforced by the following equation:

$$\begin{aligned} \text{L1}(x) = & 0\alpha_{i,j} + 1\alpha_{i,j}x + 2\alpha_{i,j}x^2 + 3\alpha_{i,j}x^3 \\ = & f_{i,j}(x,0) = 0K_{i,j} + 1K_{i,j}x + 3K_{i,j}x^2 + 6K_{i,j}x^3 \end{aligned} \quad (\text{A.2})$$

This yields the following linearly independent equations:

$$\begin{aligned} 0K_{i,j} = & 0\alpha_{i,j} \\ 1K_{i,j} = & 1\alpha_{i,j} \\ 3K_{i,j} = & 2\alpha_{i,j} \\ 6K_{i,j} = & 3\alpha_{i,j} \end{aligned} \quad (\text{A.3})$$

Continuity along L2 yields the following linearly independent equations (in the following, we have removed the linearly dependent equations):

$$\begin{aligned} \text{L2}(y) = & 0\beta_{i,j} + 1\beta_{i,j}y + 2\beta_{i,j}y^2 + 3\beta_{i,j}y^3 \\ = & f_{i,j}(0,y) = 0K_{i,j} + 2K_{i,j}y + 4K_{i,j}y^2 + 7K_{i,j}y^3 \end{aligned} \quad (\text{A.4})$$

$$\begin{aligned} 2K_{i,j} = & 1\beta_{i,j} \\ 4K_{i,j} = & 2\beta_{i,j} \\ 7K_{i,j} = & 3\beta_{i,j} \end{aligned} \quad (\text{A.5})$$

Similarly, continuity along lines L3, L4, L5, and L6 yields the following equations, respectively:

$$\begin{aligned} \text{L3}(x) = & 0\alpha_{i+1,j} + 1\alpha_{i+1,j}x + 2\alpha_{i+1,j}x^2 + 3\alpha_{i+1,j}x^3 \\ = & f_{i,j}(x,1) \end{aligned} \quad (\text{A.6})$$

$$\begin{aligned} 1K_{i,j} + 5K_{i,j} + 9K_{i,j} + 11K_{i,j} = & 1\alpha_{i+1,j} \\ 3K_{i,j} + 8K_{i,j} + 12K_{i,j} + 14K_{i,j} = & 2\alpha_{i+1,j} \\ 6K_{i,j} + 10K_{i,j} + 13K_{i,j} = & 3\alpha_{i+1,j} \end{aligned} \quad (\text{A.7})$$

$$\begin{aligned} \text{L4}(y) = & 0\beta_{i,j+1} + 1\beta_{i,j+1}y + 2\beta_{i,j+1}y^2 + 3\beta_{i,j+1}y^3 \\ = & f_{i,j}(1,y) \end{aligned} \quad (\text{A.8})$$

$$\begin{aligned} 4K_{i,j} + 9K_{i,j} + 12K_{i,j} + 13K_{i,j} = & 2\beta_{i,j+1} \\ 7K_{i,j} + 11K_{i,j} + 14K_{i,j} = & 3\beta_{i,j+1} \end{aligned} \quad (\text{A.9})$$

$$L5(x) = {}_0\alpha_{i+0.5,j} + {}_1\alpha_{i+0.5,j}x + {}_2\alpha_{i+0.5,j}x^2 + {}_3\alpha_{i+0.5,j}x^3 \\ = f_{i,j}(x, 0.5) \quad (A.10)$$

$${}_3\kappa_{i,j} + 0.5{}_8\kappa_{i,j} + (0.5)^2{}_2\kappa_{i,j} + (0.5)^3{}_4\kappa_{i,j} = {}_2\alpha_{i+0.5,j} \\ {}_6\kappa_{i,j} + 0.5{}_10\kappa_{i,j} + (0.5)^2{}_13\kappa_{i,j} = {}_3\alpha_{i+0.5,j} \quad (A.11)$$

$$L6(y) = {}_0\beta_{i,j+0.5} + {}_1\beta_{i,j+0.5}y + {}_2\beta_{i,j+0.5}y^2 + {}_3\beta_{i,j+0.5}y^3 \\ = f_{i,j}(0.5, y) \quad (A.12)$$

$${}_7\kappa_{i,j} + 0.5{}_11\kappa_{i,j} + (0.5)^2{}_4\kappa_{i,j} = {}_2\beta_{i,j+0.5} \quad (A.13)$$

Note that even though we have used the mid points of L1, L2, L3, and L4 to obtain L5 and L6, the method presented here can be generalized to use any point along those lines. Combining (A.3), (A.5), (A.7), (A.9), (A.11), and (A.13) yields the following system of equations:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0.25 & 0 & 0.125 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0.25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0.25 \end{bmatrix} \cdot \begin{bmatrix} {}_0\kappa_{i,j} \\ {}_1\kappa_{i,j} \\ {}_2\kappa_{i,j} \\ {}_3\kappa_{i,j} \\ {}_4\kappa_{i,j} \\ {}_5\kappa_{i,j} \\ {}_6\kappa_{i,j} \\ {}_7\kappa_{i,j} \\ {}_8\kappa_{i,j} \\ {}_9\kappa_{i,j} \\ {}_{10}\kappa_{i,j} \\ {}_{11}\kappa_{i,j} \\ {}_{12}\kappa_{i,j} \\ {}_{13}\kappa_{i,j} \\ {}_{14}\kappa_{i,j} \end{bmatrix} = \begin{bmatrix} {}_0\alpha_{i,j} \\ {}_1\alpha_{i,j} \\ {}_2\alpha_{i,j} \\ {}_3\alpha_{i,j} \\ {}_1\beta_{i,j} \\ {}_2\beta_{i,j} \\ {}_3\beta_{i,j} \\ {}_1\alpha_{i+1,j} \\ {}_2\alpha_{i+1,j} \\ {}_3\alpha_{i+1,j} \\ {}_2\beta_{i,j+1} \\ {}_3\beta_{i,j+1} \\ {}_2\alpha_{i+0.5,j} \\ {}_3\alpha_{i+0.5,j} \\ {}_3\beta_{i,j+0.5} \end{bmatrix} \quad (A.14)$$

$$\underline{A}\underline{\kappa}_{i,j} = \underline{c}$$

The vector $\underline{\kappa}_{i,j}$ representing the 2D spline coefficients is easily obtained by:

$$\underline{\kappa}_{i,j} = \underline{A}^{-1} \underline{c} \quad (A.15)$$

The inverse of the sparse matrix $\underline{A}$ in (A.15) can be computed in advance and stored in memory for successive use.

References

1. Alam SK, Ophir J, Konofagou EE. An adaptive strain estimator for elastography. IEEE Trans Ultrason Ferroelect Freq Cont 1998;45(2):461–472.10.1109/58.660156
2. Bohs LN, Friemel BH, McDermott BA, Trahey GE. A real-time system for quantifying and displaying two-dimensional velocities using ultrasound. Ultrasound Med Biol 1993;19(9):751–761.10.1016/0301-5629(93)90092-3 [PubMed: 8134976]
3. Bohs LN, Geiman BJ, Anderson ME, Breit SM, Trahey GE. Ensemble tracking for 2D vector velocity measurement: experimental and initial clinical results. IEEE Trans Ultrason Ferroelect Freq Cont 1998;45(4):912–924.10.1109/58.710557

4. Bonnefous O, Pesque P. Time domain formulation of pulse-doppler ultrasound and blood velocity estimation by cross correlation. *Ultrason Imaging* 1986;8:73–85.10.1016/0161-7346(86)90001-5 [PubMed: 2946098]
5. Catmull, E.; Rom, R. A class of local interpolating splines. In: Barnhill, RE.; Reisenfeld, RF., editors. *Computer Aided Geometric Design*. New York: Academic Press; 1974. p. 317-326.
6. Céspedes I, Huang Y, Ophir J, Spratt S. Methods for estimation of sub-sample time delays of digitized echo signals. *Ultrason Imaging* 1995;17:142–171.10.1006/uimg.1995.1007 [PubMed: 7571208]
7. Chaturvedi P, Insana MF, Hall TJ. 2-D compading for noise reduction in strain imaging. *IEEE Trans Ultrason Ferroelect Freq Cont* 1998;45(1):179–191.10.1109/58.646923
8. Cheezum MK, Walker WF, Guilford WH. Quantitative comparison of algorithms for tracking single fluorescent particles. *Biophys J* 2001;81:2378–2388. [PubMed: 11566807]
9. Embree PM, O'Brien WD. Volumetric blood flow via time-domain correlation: experimental verification. *IEEE Trans Ultrason Ferroelect Freq Cont* 1990;37:176–189.10.1109/58.55307
10. Fertner A, Sjölund A. Comparison of various time delay estimation methods by computer simulation. *IEEE Trans Acoust Speech Signal Process* 1986;34:1329–1330.10.1109/TASSP.1986.1164930
11. Flax SW, O'Donnell M. Phase aberration correction using signals from point reflectors and diffuse scatters: basic principles. *IEEE Trans Ultrason Ferroelect Freq Cont* 1988;35(6):758–767.10.1109/58.9333
12. Friemel, BH.; Bohs, LN.; Trahey, GE. Relative performance of two-dimensional speckle-tracking techniques: normalized correlation, non-normalized correlation, and SAD. *IEEE Ultrasonic Symposium*; 1995. p. 1481-1484.
13. Geiman, BJ.; Bohs, LN.; Anderson, ME.; Breit, SM.; Trahey, GE. A comparison of algorithms for tracking sub-pixel speckle motion. *IEEE Ultrasonic Symposium*; 1997. p. 1239-1242.
14. Giunta G. Fine estimators of two-dimensional parameters and application to spatial shift estimation. *IEEE Trans Signal Process* 1999;47(12):3201–3207.10.1109/78.806066
15. Gosh RN, Webb WW. Automated detection and tracking of individual and clustered cell surface low density lipoprotein receptor molecules. *Biophys J* 1994;66:1301–1318. [PubMed: 8061186]
16. Jacovitti G, Scarano G. Discrete time techniques for time delay estimation. *IEEE Trans Signal Process* 1993;41(2):525–533.10.1109/78.193195
17. Langeland, S.; D'hooge, J.; Torp, H.; Bijmens, B.; Suetens, P. A simulation study on the performance of different estimators for two-dimensional velocity estimation. *IEEE Ultrasonic Symposium*; 2002.
18. Lee GM, Ishihara A, Jacobson KA. Direct observation of Brownian motion of lipids in a membrane. *Proc Natl Acad Sci USA* 1991;88:6274–6278.10.1073/pnas.88.14.6274 [PubMed: 1712486]
19. Negron LA, Viola F, Black EP, Toth CA, Walker WF. Development and characterization of a vitreous equivalent material for radiation force imaging. *IEEE Trans Ultrason Ferroelect Freq Cont* 2002;49(11):1543–1551.10.1109/TUFFC.2002.1049736
20. Ng GC, Worrell SS, Freiburger PD, Trahey GE. A comparative evaluation of several algorithms for phase aberration correction. *IEEE Trans Ultrason Ferroelect Freq Cont* 1994;41(5):631–643.10.1109/58.308498
21. Ophir J, Céspedes I, Ponnekanti H, Yazdi Y, Li X. Elastography: a quantitative method for imaging the elasticity of biological tissues. *Ultrason Imaging* 1991;13:111–134.10.1016/0161-7346(91)90079-W [PubMed: 1858217]
22. Pinton GF, Trahey GE. Continuous delay estimation with polynomial splines. *IEEE Trans Ultrason Ferroelect Freq Cont* 2006;53(11):2025–2035.10.1109/TUFFC.2006.143
23. Spath, H. *Two Dimensional Spline Interpolation Algorithms*. A. K. Peters Ltd.; 1995.
24. Sumino Y, Waag RC. Measurements of ultrasonic pulse arrival time differences produced by abdominal wall specimens. *JASA* 1991;90(6):2924–2930.
25. Thevenaz, P.; Blu, T.; Unser, M. Image interpolation and resampling. In: Bankman, IN., editor. *Handbook of Medical Imaging, Processing and Analysis*. San Diego, CA: Academic Press; 2000. p. 393-420.
26. Thevenaz P, Ruttimann U, Unser M. A pyramid approach to subpixel registration based on intensity. *IEEE Trans Image Process* 1998;7:27–41.10.1109/83.650848 [PubMed: 18267377]

27. Torp, H.; Lai, XM.; Kristoffersen, K. Comparison between cross correlation and auto-correlation technique in color flow imaging. *IEEE Ultrasonic Symposium*; 1993. p. 1039-1042.
28. Unser M. Splines: a perfect fit for signal and image processing. *IEEE Signal Process Mag* 1999;16(6):22–38.10.1109/79.799930
29. Viola F, Walker WF. A comparison of the performance of time delay estimators in medical ultrasound. *IEEE Trans Ultrason Ferroelect Freq Cont* 2003;50(4):392–401.10.1109/TUFFC.2003.1197962
30. Viola F, Walker WF. A spline based algorithm for continuous time delay estimation using sampled data. *IEEE Trans Ultrason Ferroelect Freq Cont* 2005;52:80–93.10.1109/TUFFC.2005.1397352
31. Viola F, Walker WF. Computationally efficient spline-based time delay estimation. *IEEE Trans Ultrason Ferroelect Freq Cont* 2008;55(9):2084–2091.
32. Walker WF, Fernandez FJ, Negron LA. A method of imaging viscoelastic parameters with acoustic radiation force. *Phys Med Biol* 2000;45:1437–1447.10.1088/0031-9155/45/6/303 [PubMed: 10870702]
33. Walker WF, Trahey GE. A fundamental limit on delay estimation using partially correlated speckle signals. *IEEE Trans Ultrason Ferroelect Freq Cont* 1995;42(2):301–308.10.1109/58.365243
34. Wallack, A.; Emiris, IZ.; Manocha, D. MARS: a MAPLE/MATLAB/C resultant-based solver. *Proc. International Symposium on Symbolic and Algebraic Computation*; 1998. p. 244-251.
35. Work SS, Warshaw DM. Computer-assisted tracking of actin filament motility. *Anal Biochem* 1992;202:275–285.10.1016/0003-2697(92)90106-H [PubMed: 1519753]

FIGURE 1.

Common strategies for motion estimation. (a) Motion estimation without interpolation. (b) Motion estimation with pre-pattern matching interpolation. (c) Motion estimation with post-pattern matching interpolation. (d) Motion estimation with pre- and post-pattern matching interpolation. (e) Spline-based motion estimation algorithm. (f) A modified version of the proposed algorithm.

FIGURE 2.

Formulation of the 2D Spline. (a) Subset of the data samples of the region to be splined. (b and c) 1D splines are formed along the x and y dimensions of the region, respectively. (d and e) 1D splines are formed along the midpoints of the 1D splines previously generated (light gray continuous lines). (f) The 2D spline is generated.

FIGURE 3.

Sum-squared error (SSE) as a function of Nyquist ratio (i.e., $F_s/F_{\max}$). For every condition, 1000 realizations of band-limited noise were decimated by a factor of 16, reconstructed using the four methods described in this paper, and compared to the original data set to generate the SSE.

FIGURE 4.

Formulation of the 54 parameter 3D spline. (a) Subset of the data samples of the volume to be splined (only a subset of the samples are shown for clarity). (b–d) 1D splines are formed along the x , y , and z dimensions of the volume, respectively. (e–g) 1D splines are formed along the midpoints of the 1D splines previously generated (light gray continuous lines). (h) 1D splines are formed within the 3D volume (light gray dotted lines). (g) The 3D spline is generated.

FIGURE 5.

Schematic flowchart of MUSE. Given two sampled data sets, one is processed to obtain an analytical representation using multi-dimensional spline (2D in the case depicted here). For every integer sample overlap of the sampled data set (of dimensions $Q \times R$) within the splined data set (of dimensions $N \times M$, with $N > Q$ and $M > R$) the sum-squared error (SSE) function is computed. The value of x and y that minimized the SSE is called local motion estimate. The global estimate (x_0, y_0) is found amongst the local estimates by selecting the estimate with minimum error and value between 0 and 1.

FIGURE 6. Simulated bias and standard deviation for the MUSE algorithm. All images depict errors in samples.

FIGURE 7.

Simulated bias and standard deviation for the Sum-Squared Differences algorithm coupled with parabolic sub-sample estimation. All images depict errors in samples.

FIGURE 8.

Simulated bias and standard deviation for the Sum-Absolute Differences algorithm coupled with parabolic sub-sample estimation. All images depict errors in samples.

FIGURE 9. Motion estimates in the axial (top panel) and lateral (bottom panel) direction as a function of signal-to-noise ratio. The simulations were conducted with no motion between the two data sets.

FIGURE 10.

Simulated bias and standard deviation for the 3D version of MUSE. The three biases and standard deviations are displayed on the same gray scale. Sub-sample shifts of 1/8 of a sample were generated in the axial, lateral, and elevation directions. Note that three planes on the elevation axis have been removed to provide better visualization of the data. All images depict errors in samples.

FIGURE 11. Motion estimates in the axial (top panel), lateral (middle panel), and elevation (bottom panel) direction as a function of signal-to-noise ratio. The simulations were conducted with no motion between the two data sets.

FIGURE 12.

Experimental displacement estimates in units of samples. The dotted line bounds one sample in range and azimuth. Data was acquired at a 40 MHz temporal sampling rate and $300\ \mu\text{m}$ line spacing. Dots represent individual displacement estimates, whereas open circles represent the mean displacement at each location.

FIGURE 13.

Experimental displacement estimates in units of microns. The dotted line bounds one sample in range and azimuth. Data was acquired at a 40 MHz temporal sampling rate and 300 μm line spacing. Dots represent individual displacement estimates, whereas open circles represent the mean displacement at each location.

FIGURE 14.

Typical 2D search (left) and kernel (right) data sets used for the experimental verification of MUSE. The line spacing is $300\ \mu\text{m}$ and the axial sampling corresponds to $19.25\ \mu\text{m}$.

FIGURE 15.
Schematic representation of the 2D region being splined.

TABLE 1**Bicubic spline coefficients**

	1	x	x²	x³
1	0^K	1^K	3^K	6^K
y	2^K	5^K	8^K	10^K
y²	4^K	9^K	12^K	13^K
y³	7^K	11^K	14^K	15^K

The coefficient highlighted in gray is set to zero in the 15-parameter nonseparable spline described here.

TABLE 2**Bicubic spline coefficients**

	1	x	x²	x³
1	0^K	1^K	3^K	6^K
y	2^K	5^K	8^K	10^K
y²	4^K	9^K	12^K	13^K
y³	7^K	11^K	14^K	15^K

The coefficients highlighted in gray are set to zero in the 12-parameter non-separable spline.

TABLE 3**Maximum values of bias and standard deviation obtained from the 2D noise-free simulations**

	MUSE	SSD	SAD
Axial Bias (samples)	0.00026	0.1362	0.1822
Lateral Bias (samples)	0.0022	0.1564	0.1733
Axial Std (samples)	0.0028	0.2139	0.2674
Lateral Std (samples)	0.0029	0.2788	0.3277

Each sample in the axial dimension corresponds to 19.25 μm ; each sample in the lateral dimension corresponds to 135 μm .

TABLE 4**Maximum bias and standard deviation of motion estimates using MUSE in simulated 3D data sets**

	Axial	Lateral	Elevation
Bias (samples)	0.00022	0.0022	0.0023
Std (samples)	0.003	0.0028	0.0025

Each sample in the axial dimension corresponds to 19.25 μm ; each sample in the lateral and elevation dimensions corresponds to 135 μm .