Solid Freeform Fabrication 2021: Proceedings of the 32nd Annual International Solid Freeform Fabrication Symposium – An Additive Manufacturing Conference Reviewed Paper

Image Registration and Matching Error in 2D and 3D for Laser Powder Bed Fusion

Andrew Lang[1,3], Cesar Ortiz Rios[2], Joseph Newkirk[2], Robert G. Landers[3], James Castle[1], and Douglas A. Bristow[3]

[1]The Boeing Company
[2]Materials Science and Engineering, Missouri University of Science and Technology
[3]Mechanical and Aerospace Engineering, Missouri University of Science and Technology

Abstract

This work outlines a method to register 2D and 3D images taken post-process and in situ from 301L stainless steel parts printed by Laser Powder Bed Fusion. The process uses DREAM.3D, an open-source software that provides for data transport in a non-proprietary format. The Robust Automatic Threshold selection technique is used to create a boundary point cloud of the part from each image. The Iterative Closest Point technique is applied to the point clouds for both 2D images and 3D image stacks to create an affine transformation matrix for registration. Multiple 2D SEM images of the same sampled layer are taken under different settings and imaging conditions and registered to a common target. Images from post-process X-ray Computed Tomography and an in situ short-wave infrared camera create 3D image stacks, which are directly registered in 3D space. Registration accuracy is validated by creating a correspondence list of the closest point in the registered point clouds and the matching error is calculated using mean average error. Mean average error is computed using point-to-point and point-to-plane methods; the point-to-plane method is shown to be more reliable. Finally, the registered 3D images are down sampled to the lower resolution image dimensions creating arrays of in situ and post-process data at the same resolution.

1.0 Introduction

Image registration is frequently used in Laser Powder Bed Fusion (LPBF) research to align multiple modalities of data, analyze variation in images, and fuse training datasets for classification. In registration, a reference dataset is transformed to align with a fixed target dataset [1]. There are two broad strategies for registration: 1) manual alignment [2, 3] and 2) automated algorithms, the most mainstream being Iterative Closest Point (ICP) [4]. Extensive research has been conducted on other registration algorithms that optimize a cost function to minimize error between two datasets using methods like defining datums from part features [5], quasi-Newton optimization [6, 7], least squares optimization [8], or mean squared difference [1]. Registration accuracy can be subjectively estimated by manually inspecting alignment [9] or objectively determined by calculating an error metric between points [8], but it is often not discussed in LPBF research.

Fusing in situ data, often multiple modalities, with ex situ X-ray Computed Tomography (XCT) and/or destructive testing (DT) sectioned images has become a common area of research in quality assurance for metal additive manufacturing, both in industry and academia [10, 11, 12].

This data fusion can be used for statistical analysis or machine learning models that predict resulting material characterization based in in situ parameters. There are many existing commercial tools for in situ monitoring of LPBF processes that are integrated into commercial systems, can be purchased as machine add-ons, or are independent of a particular machine concept [13]. XCT has become standard in industrial verification of additively manufactured (AM) parts [12]. XCT is advantageous for quality control because it is non-destructive and, depending on the resolution required, can relatively quickly capture profiles of a whole part. Micrographs from Scanning Electron Microscopes (SEM) can be used for microscopic material characterization but the part must be destroyed to use SEM and only samples of the part can be taken [14, 15].

Many toolboxes have been developed to perform image registration. Some examples are Slicer, ImageJ, Elastix, Amira, DREAM.3D, and VGSTUDIO MAX. The Insight Toolkit (ITK) is an open-source development framework with an extensive suite of software tools for image analysis, including registration [16, 17], and is integrated in several analysis tools like DREAM.3D or Elastix. Many of publicly available toolboxes and registration techniques were initially developed for medical imaging while LBPF imaging research was still in its infancy [18, 19]. There are several image processing libraries, like OpenCV, Pillow, and scikit-image for Python, MATLAB, and other languages that can be utilized for 2D image registration. The main disadvantages with most available toolboxes are that their application to specific needs of LPBF research is limited, they may not work with 3D image sets or triangle geometry, and they are not tightly integrated with a standard visualization tool.

This paper investigates image registration in both 2D and 3D utilizing DREAM.3D, an open-source tool kit that allows for construction of customized workflows to analyze data [20]. DREAM.3D was developed to process digital instances of microstructure, but many of the tools are useful for LPBF data processing and address the disadvantages found in most available toolboxes. ParaView, an open-source analysis and visualization platform, is used to render 2D and 3D visualizations (paraview.org). The main advantages of DREAM.3D are that the reconstruction, manipulation, and visualization of 3D data is simple and straightforward, the program is free and open-source, over 100 filters from the image processing library ITK are included, data is stored as standard HDF5 files by default, additional features are continuously added, and custom filters can be created.

DREAM.3D processes two broad types of data: 1) image geometry and 2) triangle geometry [21]. Image geometry is a structured grid of constant resolution with data values assigned to each cell. The pixel makeup of a computer screen is an example of image geometry. Triangle geometry consists of vertex points, edges forming lines between points, and faces forming a mesh between lines (faces could be triangles, quads, etc). A geodesic dome is a physical example of triangle geometry. Figure 1 shows a 2D example of a triangle geometry structure (a) with vertices, edges, and faces and an image geometry structure (b) with grayscale values in each cell of the grid.

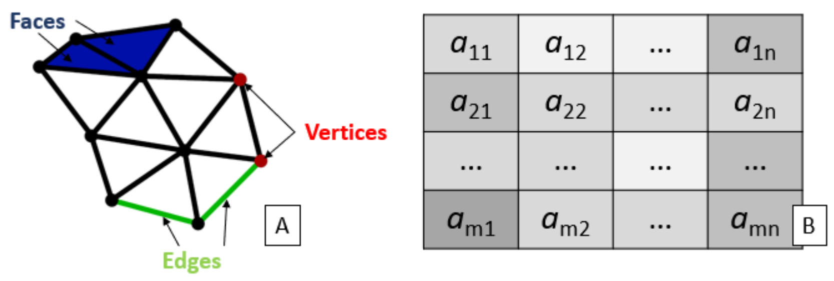

Faces
Vertices
Edges
A
a_{11} a_{12} ... a_{1n}
a_{21} a_{22} ... a_{2n}
... ...
a_{m1} a_{m2} ... a_{mn}
B

Figure 1: Triangle Geometry (a), Image Geometry (b)

Datasets from many LPBF in situ and ex situ platforms are often structured voxel/pixel-based image geometry [13, 12]. The datasets used in this study are structured image datasets in both 2D and 3D. Currently, the DREAM.3D filter *Apply Transformation to Geometry* cannot apply a pre-computed transformation matrix to image-type geometries, so a workaround has been implemented utilizing Python for 2D images and a not-fully-automated process involving manual rotation for 3D images.

2.0 Experimental Setup

For this study, an ASTM E8 tensile specimen [22] has been printed in 304L stainless steel using a Renishaw AM250. The test specimen is 50 mm tall and has a 4 mm diameter neck. The nominal laser power for this build was 200W and a section of lettering ("Missouri S&T") was printed at 100W inside the neck of the part to induce defects. The point distance, d_p, and the hatch spacing, d_h, were held at constant 60 µm and 85 µm, respectively. In-situ data was captured using a Short Wave Infrared (SWIR) camera, XCT imaging was used to generate a 3D array of the part density, and 2D SEM images were taken of extracted sections of the test specimen.

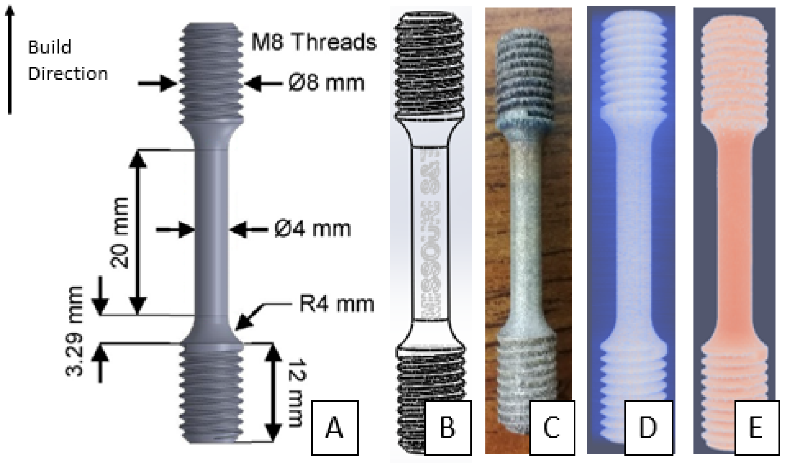


Figure 2: ASTM E8 tensile specimen (a), low power lettering printed in the neck to induce defects (b), the final tensile bar printed by LPBF (c), 3D reconstruction of in situ SWIR imaging (d), and 3D reconstruction of XCT imaging (e).

2.1 SWIR Imaging

The purpose of SWIR imaging for this study was to establish a 3D profile of an in situ dataset to register with an ex situ dataset. The SWIR camera set up and thermal feature extraction has been performed in other work [23]; the process is summarized in this section. An FLIR SC6201 SWIR camera measured visible light emitted from the melt pool. The camera was installed at an observation angle of 15º above the build chamber to observe the build. The camera pixel array was reduced to an 80 × 80 pixel window enabling high frame rate recording (~2500 Hz). The x-direction instantaneous field of view of the SWIR camera was ∼130 µm/pixel. The y-direction instantaneous field of view was ∼135 µm/pixel due to the observation angle (θ = 15°) of the SWIR camera shown in Figure 3. A non-uniformity correction (NUC) was performed to account for differences in the SWIR measurements across the imaging area due to the observation angle. The final corrected pixel dimensions were 125 µm x 125 µm for each layer.

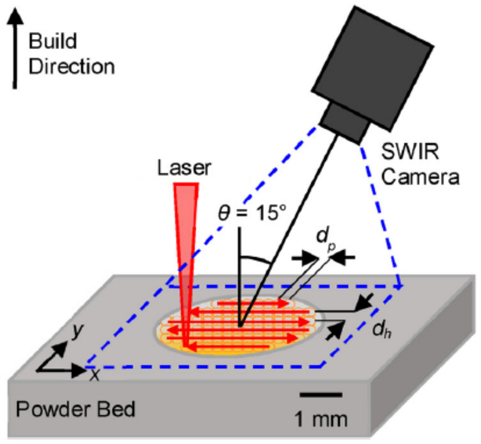


Figure 3: Schematic of SWIR camera observations [23]

Thermal features from a multi-frame time-series recording are extracted to a single image representation of each layer. Each 2D layer is concatenated to create a 3D voxel-based reconstruction of the tensile specimen in HDF5 format. The layer height of this build was 50µm. An image geometry container was created in DREAM.3D with the X, Y, and Z dimensions of the voxels (125 µm, 125 µm, 50 µm) and the HDF5 thermal feature file was imported using the *Import HDF5 Dataset* filter and can be visualized in ParaView.

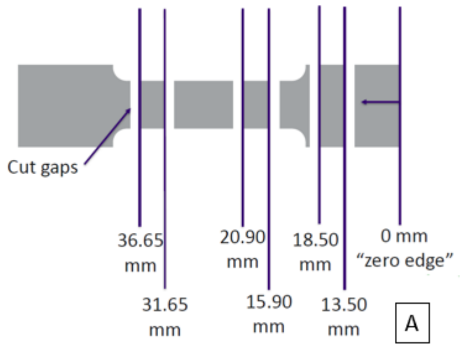
2.2 X-Ray Computed Tomography Imaging

Post-process XCT inspection was performed by NSI using a custom radiograph tool. NSI scanned the test piece with a MeVX6™ High Energy system using a 6 MeV linear accelerator (LINAC) at 450 kV and performed 360º step scan imaging with a focal spot size of 24.15 µm. The full geometry of the tensile piece was captured in 291 x 281 x 1706 voxels. Each voxel is a cube with dimensions 29.96 µm in X, Y, and Z. The XCT array was directly imported into DREAM.3D using the *Import North Star Imaging CT (.sihdr/.nsidat)* filter.

2.3 SEM Imaging

The tensile bar was cut perpendicular to the long axis via wire EDM as shown in the cut plan in Figure 4 (a). The "M" section in the text printed at lower power was extracted between 36.65 and 31.65 mm and mounted in conductive mounting resin then metallographically prepared to a final polish of 0.05 μm for SEM analysis. During the metallographic preparation, 331 μm of material was removed, resulting in a cross section containing the targeted "M" low power region. Imaging parameters were selected to generate good contrast between the polished surface metal and the defect interior to facilitate registration in DREAM.3D. Images were captured with the Through-Lens Detector (TLD) in Backscatter Electron (BSE) mode biased to -150V voltage to eliminate the associated secondary electron (SE) edge effect. Integration of eight frames with a dwell time of 1 μs were used. The orientation of the 0º imaging position was chosen to align to the orientation of the CT data. Images were taken in a 5x5 grid pattern starting from row 3, column 1 in increments of 1000 μm in the X-direction and 800 μm in the Y-direction. Increments were chosen to ensure sufficient overlap for stitching the images together. Once the images were captured, the stage was rotated to establish a new imaging position. The imaging process was repeated until the eight imaging positions, shown in Figure 4 (b), were captured. An image manipulation program was used to stitch the images together. The images were imported as layers and overlayed using the overlapping features. Layers were ordered such that lower rows overlapped higher rows and interior columns overlapped exterior columns. For example, row 1 is overlapped by row 2, and column 3 overlaps both columns 2 and 4, which overlap columns 1 and 5, respectively. Pixel sizes of these images are 1.25 μm x 1.25 μm.

When the images were arranged, the image was flattened and exported as a TIF file. The images were integrated into DREAM.3D using the *ITK:Image Reader* filter. The *Set Origin & Spacing (Image)* filter was used to input the spacing based the pixel dimensions.



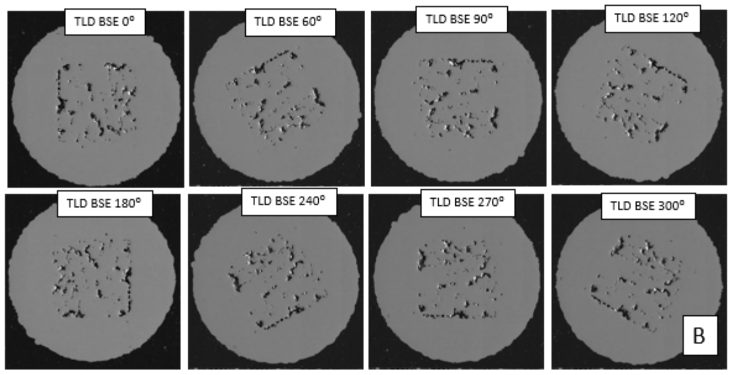


Figure 4 Extracted section for SEM imaging: Cut Plan (a), 8 SEM imaging positions (b)

3.0 Registration

This section outlines the processes used to register multiple modes of data in a single affine space. Boundary point clouds from each image were generated by sampling points along the surface of the part created using the Robust Automatic Threshold Selection (RATS) algorithm. An affine transformation was found to directly register two sets of point clouds using the ICP algorithm. Finally, DREAM.3D, Python, and ImageJ was used to transform the 2D and 3D datasets in a not-fully-automated process.

3.1 Boundary Point Cloud

DREAM.3D was used to create point clouds representing the boundary of each 2D and 3D image array. The Robust Automatic Threshold Selection (RATS) algorithm, an automatic thresholding method based on gradients in the image, was used to define the boundary of the part and point clouds were created by sampling from the boundary. RATS is an automated edge detector algorithm that identifies a threshold for greyscale images based on the image's gradients, often at the maximum gradient [24, 25]. The gradient array was calculated using the *ITK::Gradient Magnitude Image Filter*. The magnitude of the image gradient is:

$$G(x,y) = \|f(x,y)\|_2 = \sqrt{f_x(x,y)^2 + f_y(x,y)^2}$$

The direction of the image gradient is:

$$\theta(x,y) = \tan^{-1}\frac{f_x(x,y)}{f_y(x,y)}$$

Eliminating the square root of the magnitude yields suitable results without the added cost of a final scan across the image to compute the root. In the ITK plugin, the gradient is simply the sum of the squares of the partial derivative operations.

$$g(x,y) = G^2_{(x,y)} = f_x(x,y)^2 + f_y(x,y)^2$$

With the *Robust Automatic Threshold* filter, DREAM.3D creates a new array that is false where the input array is less than the threshold and true otherwise. A regional threshold is computed as the gradient weight sum of the input array, A.

$$T = \frac{\sum_{i=1}^{n}\big(g(x,y)\cdot A(x,y)\big)}{\sum_{i=1}^{n} g(x,y)}$$

This threshold is generally where the gradient magnitude is highest.

The *ITK::Grayscale Fillhole Image* filter, with the FullyConnected parameter set to true, is used to isolate and fill the part mask. To create a point cloud in the 2D images, the *Find Boundary Cells (Image)* filter is applied to the part mask and vertex geometry of each boundary cell is extracted to a point cloud. A visualization of the process is shown in Figure 5 for a 2D array using the 0^{o} TLD BSE SEM image.

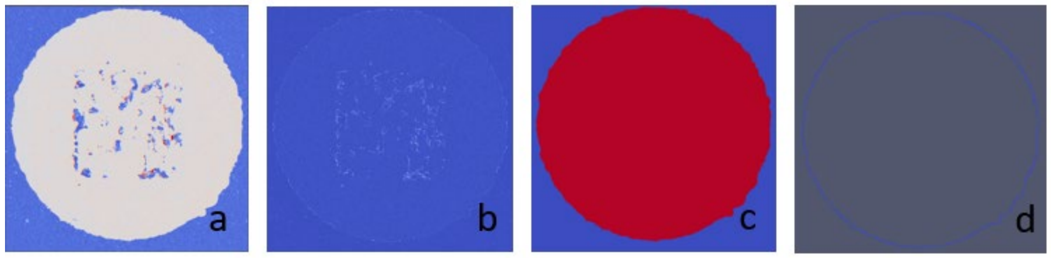
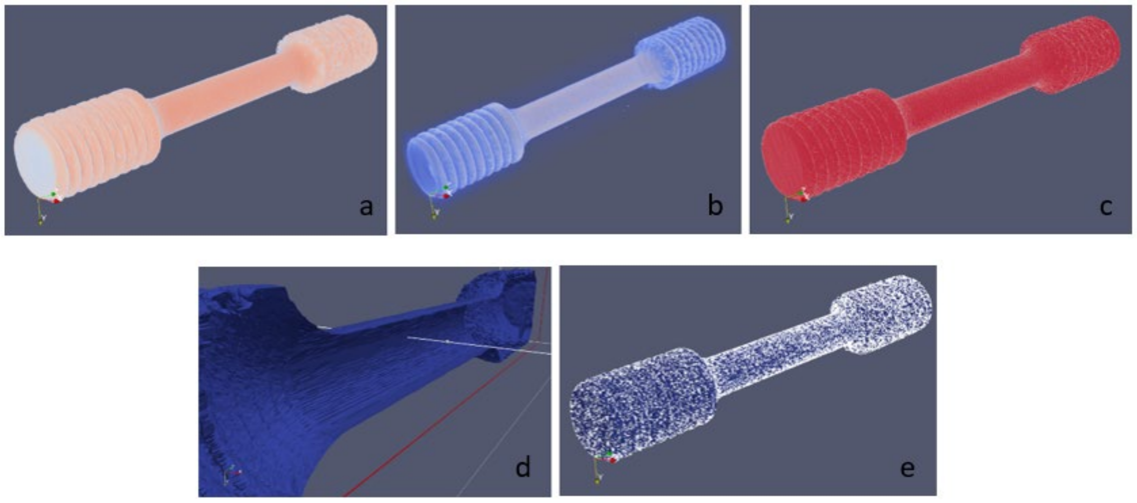


Figure 5: TLD BSE 0º SEM image: Raw image (a), gradient magnitude array (b), part mask after applying RATS and filling holes (c), and boundary point cloud of the image (d).

For both SWIR and SCT 3D datasets, triangle surface arrays were created from the surface of the mask array using the *Quick Surface Mesh* filter. Approximately four and a half million triangles were created for each surface. 3D point clouds were generated by sampling points from the triangle surfaces using the *Point Sample Triangle Geometry* filter. For registration, one hundred thousand sample points were chosen for both SWIR and XCT datasets. A visualization of the process in 3D is shown in Figure 6 using the XCT array.



3.2 Iterative Closest Point

The ICP algorithm iteratively revises a transformation to minimize the sum of squared differences between the coordinates of matched pairs and register two-point clouds to the same affine space [26]. There is a wide body of research to improve the algorithm by minimizing different error metrics, performing point-to-plane registration, or creating various search algorithms to reduce time to identify matched pairs [4, 27, 28]. This study used the DREAM.3D implementation of ICP to find an optimal affine transformation. The *Iterative Closest Point* filter finds a rigid transformation that rotates and transforms the reference data set. The filter does not change the distance between any two points in the transformed point cloud so there is no scale or shear component in the transformation.

An affine transformation is a map between two different affine spaces that preserves lines and parallelism [25]. The transformation matrix created in DREAM.3D is a 4x4 array, T, that transforms each point in an array, A, to a new position, A':

$$[A'] = T\,[A]$$

Or:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = T \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

The optimal transformation, T*, is one that minimizes the Euclidean distance between every pair of points in the reference point cloud, M, to points in the target point cloud, S.

$$T^* = \arg\min_{T} dist(T(M), S)$$

$$dist(T(M), S) = \sum_{m \in T(M)} \|m - s_m\|_2^2 , \; s_m = \arg\min_{s \in S} \|s - m\|_2^2$$

Where T is all possible transformations the optimization algorithm searches for.

For SEM, a ninth image where defects in the image are human labeled, was used as the target and, for the 3D datasets, the XCT array was designated as the target. This implementation of ICP may get stuck at a local minimum so an initial guess of alignment was be provided before implementation. The initial guess for rotation was done using the *Rotate Sample Reference Frame* filter and translation was done using the *Set Origin & Spacing (Image)* filter. After the initial guess was applied, a transformation matrix was found with ICP and the reference point clouds were transformed in DREAM.3D. Figures 7 and 8 show the 2D and 3D point cloud alignment before and after transformation.

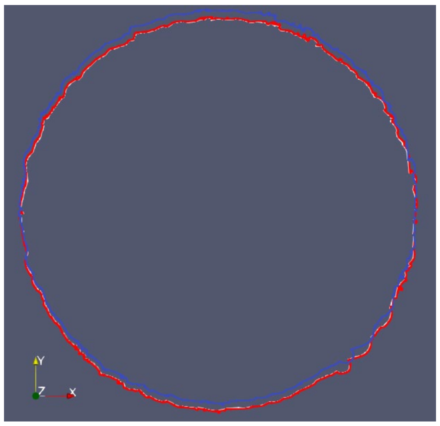


Figure 7: 2D point cloud registration: SEM with TLD BSE 0º before (blue), after (red), and the target point cloud (white)

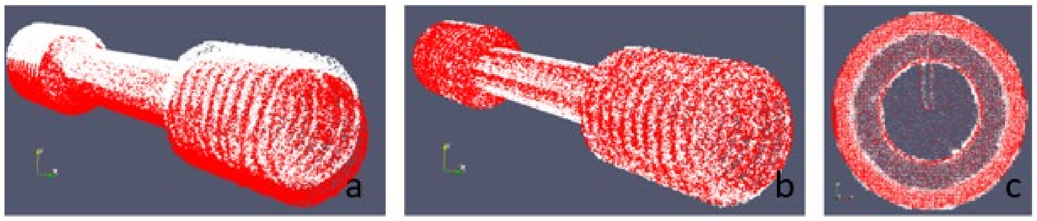


Figure 8: 3D point cloud registration before/after: SWIR (white) and XCT (red) before (a), SWIR and XCT after registration (b and c)

3.3 Apply Transformation to Geometry

The DREAM.3D filter, *Apply Transformation to Geometry,* can apply any 4x4 transformation to triangle geometry but, in this study, all three datasets (SEM, SWIR, XCT) are structured image geometries so the transformations computed in section 3.2 cannot be directly applied in DREAM.3D. Registration for the 2D dataset relied on ImageJ and Python as a patch and registration for the 3D dataset relied on manual rotation before ICP.

Unlike a point cloud, image geometry must be interpolated in the original grid size after it is transformed. Interpolation finds a value between two points on a line so, for this case, the interpolation algorithm determines the value to assign each voxel between two known points. In multidimensional interpolation, an estimate of $y(x_1, x_2, \ldots, x_n)$ is made from an n-dimensional grid of tabulated values y and n one-dimensional vectors giving the tabulated values of each of the independent variables $x_1, x_2, \ldots, x_n$ [29]. Bi-linear interpolation was used for all interpolations in this study.

3.3.1 Apply Transformation to 2D Geometry

The eight SEM imaging positions are transformed with a combination of ImageJ, Python, and DREAM.3D. Before the images were imported into DREAM.3D they were all rotated to roughly align with the 0^o image position. As an example, Figure 9 (a) and (b) show the initial 60^o imaging position before and after the rotation. Bilinear interpolation was used, and the result was saved as a .tiff file which was could be read by DREAM.3D.

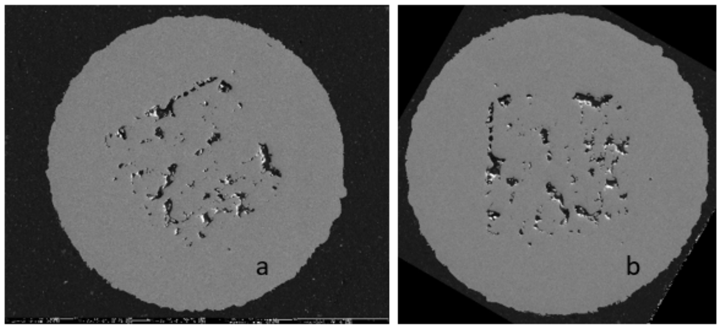


Figure 9: TLD BSE 60^o SEM image: Raw image (a), image rotated to 0^o with ImageJ (b)

Next, the ICP operation in DREAM.3D to determine a transformation matrix, as outlined in section 3.2, was performed. Python was used to apply the transformation to each of the eight

SEM imaging positions. The DREAM.3D file was read into Python using the h5py library (h5py.org) and the applicable transformation matrix was read as a Numpy array (numpy.org). The image was then transformed with skimage.transform.warp using bilinear interpolation (scikit-image.org) and the final image was output as an .h5 file. The final image was read back into DREAM.3D and a surface point cloud was created in the final position.

The rotation components of an affine transformation are:

$$\mathbf{R}_X(\theta)=\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) & 0 \\ 0 & -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \mathbf{R}_Y(\theta)=\begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \mathbf{R}_Z(\theta)=\begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The translation components of an affine transformation are:

$$\mathbf{T}=\begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Table 1 shows the transformation matrix, computed by ICP, for the 60º imaging position. This transformation matrix indicates this is a 2D transformation as the third row and column are each [0, 0, 1, 0]. This transformation has a 2.16º rotation about the Z-axis, as indicated by the four cells in the upper left quadrant of the matrix: $\cos^{-1}(.999289) = \sin^{-1}(0.37712) = 2.16º$ (Equation X). The translation components are $t_x = -0.067751$ mm and $t_y = 0.08299$ (Equation X). There are not scaling or shear components in this transformation matrix. All eight of the transformation matrices computed for the different SEM imaging positions follow the same format of 2D transformation, rotation about the Z-axis, and a small x and y translation.

TLD BSE 60º			
0.999289	0.0377521	0	-0.067751
-0.0377521	0.999289	0	0.08299
0	0	1	0
0	0	0	1

Table 1: The transformation matrix computed by ICP for the TLD BSE 60º SEM image. Dimensions are in mm.

Figure 10 shows the initial location (a) and the transformed final location (b) of a section of the point clouds created from imaging positions. The improvement in registration is clear to see. A technique to measure the error in registration between the point clouds is outlined in Section 4.

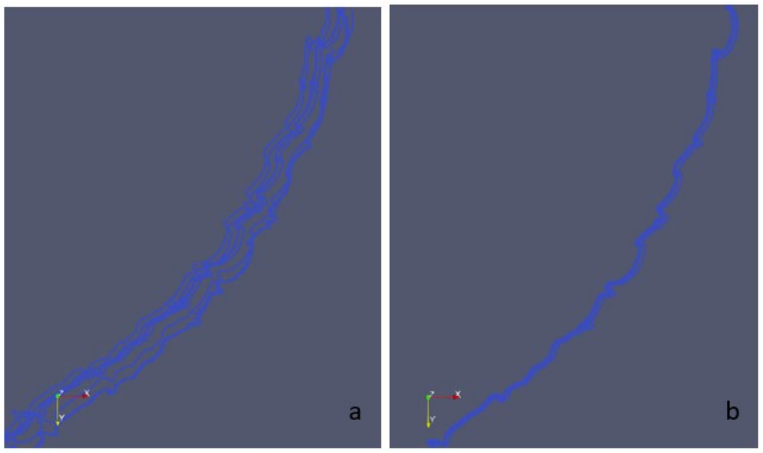


Figure 10: Initial location (a) and final location (b) of a section of the eight SEM imaging positions

3.3.2 Apply Transformation to 3D Geometry

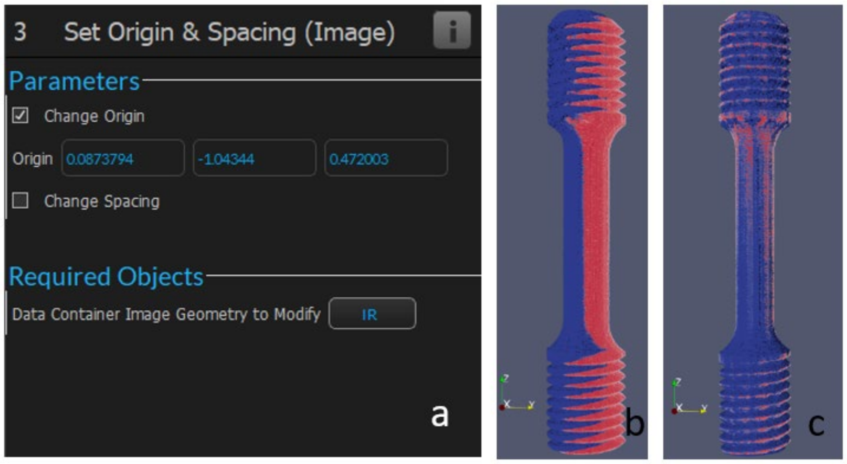
As discussed earlier, the *Apply Transformation to Geometry* filter in DREAM.3D does not work on image-type datasets. ImageJ and the Python libraries that were explored (scikit-image, Pillow, and OpenCV) only work on 2D images and cannot apply 4x4 transformations to 3D image-type datasets like what was done in Section 3.3.1 for the SEM imaging positions.

3D images can, however, be manually translated and rotated in DREAM.3D using the *Set Origin & Spacing (Image)* and the *Rotate Sample Reference Frame* filter. At iterative process where the SWIR array was manually rotated across all three axis, ICP was run to register the resulting point cloud to the XCT point cloud, and the computed transformation matrix compared to the translation-only matrix in equation X. This time-consuming process was stopped when a satisfactorily close solution to the translation-only matrix was achieved. Table 2 shows the resulting transformation matrix from this iterative manual-rotation/ICP process.

SWIR			
0.999995	0.0052380	-0.00070846	0.0873794
-0.00523	0.999988	0.000629	-1.04344
0.000712	-0.00063	1.00001	0.472003
0	0	0	1

Table 2: The transformation matrix computed by ICP for the SWIR image. Dimensions are in mm.

The SWIR image array was translated to its final location with the *Set Origin & Spacing (Image)* filter using the translation components of the transformation, t_x=0.08978 mm, t_y=-1.04344 mm, and t_z=0.472003 mm. Figure 11 (a) shows the application of the translation components in DREAM.3D, (b) shows the SWIR array with the XCT array before the transformation, and (c) shows the SWIR array with the XCT array after the transformation.


3 Set Origin & Spacing (Image)

Parameters

[✓] Change Origin

Origin 0.0873794 -1.04344 0.472003

[ ] Change Spacing

Required Objects

Data Container Image Geometry to Modify IR

Figure 11: Translation components to register SWIR with SCT in DREAM.3D (a), SWIR array (red) before transformation to XCT (blue) (b), and SWIR array (red) after transformation to XCT (blue) (c).

Applying only the translation components of the transformation in Table 2 will introduce some error in the final alignment because the rotation components are not accounted for. Some error, where the two edges are not integrated, is visible at the bottom (-Z) end of the neck of the tensile piece in Figure 11 (c).

The registration accuracy can be interrogated by slicing the registered array in ParaView and examining internal or external reference features between the two arrays (Figure 12). Internal features were imbedded into the tensile specimen by printing specific volume at lower laser power. Slices on the XY plane are shown in (a) at or three different reference features, slices on the YZ plane are shown in (b), and slices on the XZ plane are shown in (c). The internal features are visible in both the SWIR array and the XCT array. Both internal and external features were correlated across the SWIR and XCT arrays showing general alignment between the two datasets.

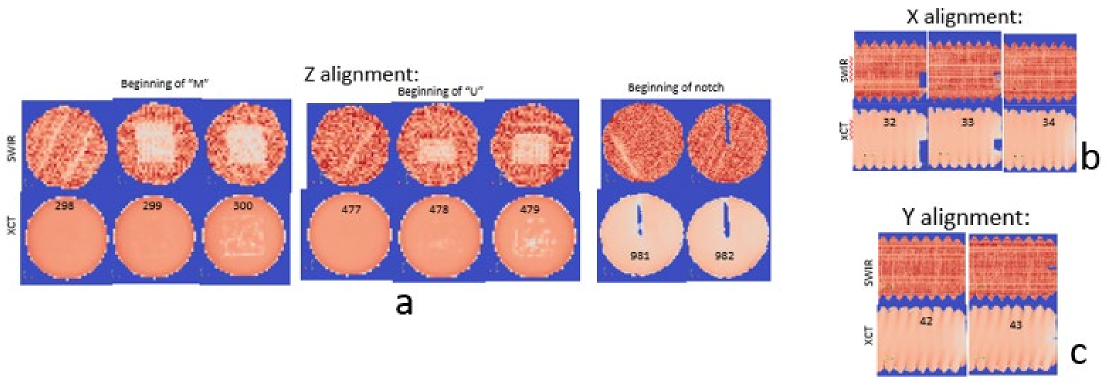


Figure 12: Subjective verification of registration accuracy by comparing internal and external features in both arrays showing alignment along the Z axis (a), the X axis (b), and the Z axis (c). The slice number is shown with each image.

4.0 Registration Accuracy

This study found individual errors by calculating the Euclidian distance between matched points. The point clouds used in this section were derived, following the process outlined in Section 3, from the final position of the transformed image arrays. Individual errors are calculated for both the point the point-to-point error metric and the point-to-plane error metric.

$$e_i = \|m_{position} - s_{position}\| = \sqrt{(m_x - s_x)^2 + \left(m_y - s_y\right)^2 + (m_z - s_z)^2}$$

4.1 Point-to-Point Error

The total error for the point cloud registration is found by calculating the Mean Absolute Error (MAE) for each individual error, e_i. Where n is the number of points in the reference point cloud, M. The standard deviation, σ, for the errors was also calculated.

Traditional ICP algorithms, including the one used in this study to find the transformation, attempt to minimize the Mean Square Error (MSE) of the distance between corresponding points in two point clouds [26, 30]. MSE, as a cost function in ICP algorithms, is advantageous over MAE because larger errors are penalized exponentially. MSE is in squared units which is a problem as an error metric because squared length units are not particularly useful in determining the distance error between point clouds. Root Mean Square Error (RMSE) is in the same unit length as the dataset so it reports a useable error metric. MAE is used in this study because it is in the same units of length as the dataset and is simpler to calculate.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} e_i\,, \qquad MSE = \frac{1}{n}\sum_{1}^{n} {e_i}^2\,, \qquad RMSE = \sqrt{\frac{1}{n}\sum_{1}^{n} {e_i}^2}\,, \qquad \sigma = \sqrt{\frac{\sum(e_i - MAE)^2}{n}}$$

The point-to-point error metric developed in this study first searched each point in the reference point cloud, M, for the closest point in the target point cloud, S. A correspondence array, C, was created to store the matched points from S. Some points in S may appear in C zero or multiple times but the correspondence array always has the same length as the reference point cloud. Python was used to read point clouds from DREAM.3D and implement the error algorithm.

A summary of the point-to-point error process:

1. Create list of correspondence points, C.
    a. For all points in the reference point cloud, M, match closest point in target point cloud, S.
2. Calculate individual errors, e_i.
    a. Find the Euclidian distance between matched pairs in M and C
3. Calculate accuracy metrics
    a. Standard Deviation, MAE, MSE, RMSE

4.1.1 Point-to-Point Error with 2D Images

Point-to-point error was measured for each of the eight SEM imaging positions. Point clouds for the eight SEM imaging positions were the reference point clouds and the ninth human-labeled SEM image was the target point cloud. Table 3 below shows the accuracy metrics, standard deviation, MAE, MSE, RMSE, for the original and final point cloud positions. The table also shows the number of points, n, in each point cloud. MAE is improved in each point cloud using the registration technique from Section 3. The TLD 180º imaging position has the closest final registration to the target, at MAE of 4.18 µm, and the 270º imaging position has the worst final registration to the target, at MAE of 7.99 µm. The point-to-point average MAE for all eight final positions is 6 µm and the length and width of each pixel in the images are 1.25 µm.

	TLD BSE 0º		TLD BSE 60º		TLD BSE 90º		TLD BSE 120º		TLD BSE 180º		TLD BSE 240º		TLD BSE 270º		TLD BSE 300º	
	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final
STDV (µm)	26.38	1.86	11.10	3.75	18.83	4.15	16.56	3.71	26.18	1.74	10.50	3.85	18.59	4.96	10.45	4.03
MAE (µm)	48.51	4.82	13.01	5.75	27.65	7.05	22.39	6.12	49.06	4.18	12.49	5.96	28.91	7.99	12.80	6.42
MSE (µm^2)	3,049.1	26.7	292.4	47.2	1,119.4	67.0	775.6	51.2	3,092.8	20.5	266.3	50.4	1,181.2	88.4	273.0	57.4
RMSE (µm)	55.22	5.16	17.10	6.87	33.46	8.18	27.85	7.16	55.61	4.53	16.32	7.10	34.37	9.40	16.52	7.58
n	22,968		22,595		22,677		22,978		22,934		22,773		22,546		22,718	

Table 3: Point-to-point error results for all eight SEM imaging positions

The point cloud size of each SEM image, n, is between 22 and 23 thousand points. When the target point cloud or the target and the reference point cloud are randomly down sampled, the distance between matched points becomes larger. Error is unreliable if there are not a significant amount of points in the target point cloud. MAE diverges exponentially as the target and reference point cloud are down sampled as seen in Figure 13 (a) shown with a logarithmic scale. Figure 13 (b) shows the effect on MAE of down sampling only the reference point cloud. MAE is effectively unchanged as the reference point cloud is down sampled because the reference points are still able to find matched neighbors in the large target point cloud. As the number of points in the target are increased, the point cloud starts to act as a plane. For a point-to-point error metric to be effective, it is vital to select a high number of points in the target point cloud.

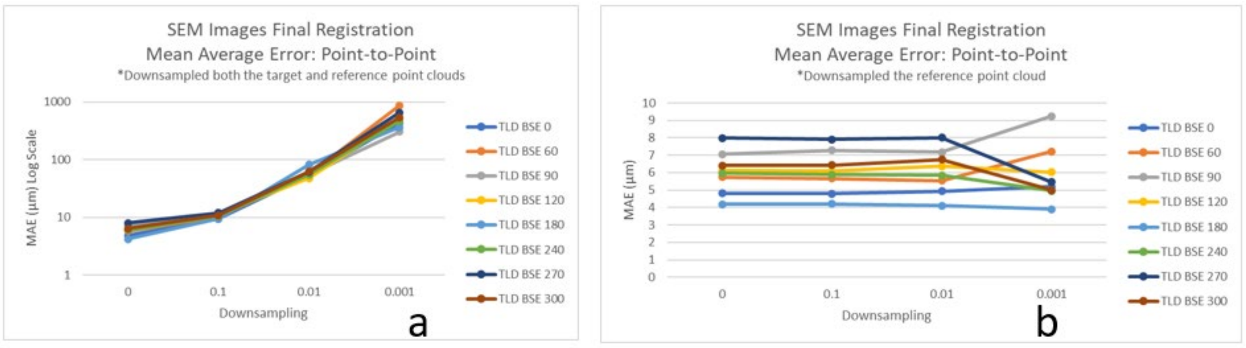
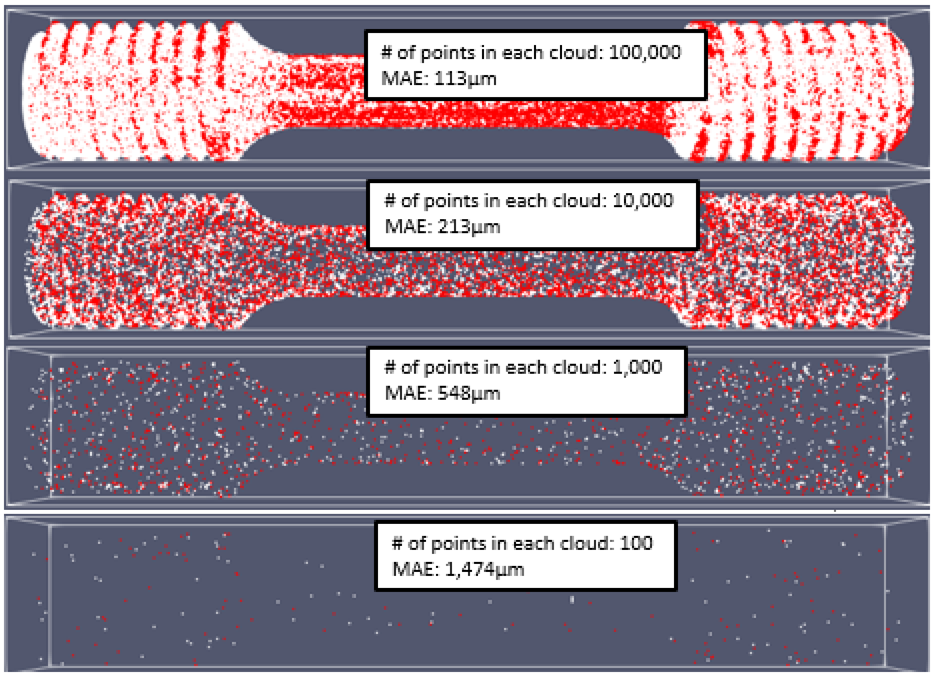


Figure 13: Change in MAE when the target point cloud is down sampled (a) and when the reference point cloud is down sampled (b)

4.1.2 Point-to-Point Error with 3D Images

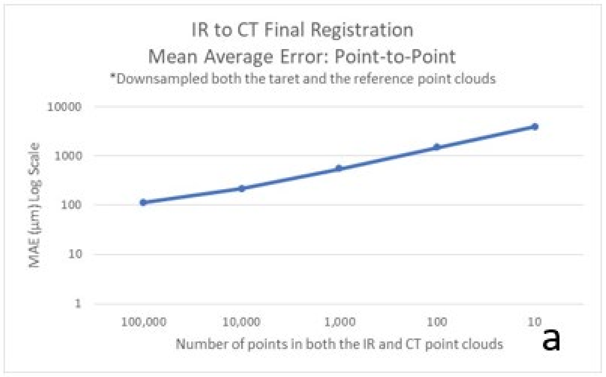
Point-to-point MAE of the 3D data set, where the reference is the SWIR point cloud and the XCT is the target point cloud, was determined in the same way as the 2D SEM images in Section 4.1.1. Each point cloud had 100,000 points that were sampled off the surface as outlined in Section 3.1. Figure 14 shows a ParaView visualization of the two point clouds when the points are randomly down sampled to the same effect as the SEM images.



# of points in each cloud: 100,000
MAE: 113µm
# of points in each cloud: 10,000
MAE: 213µm
# of points in each cloud: 1,000
MAE: 548µm
# of points in each cloud: 100
MAE: 1,474µm

Figure 14: The effect on MAE of down sampling both point clouds. SWIR in red and XCT in white.

When both point clouds are down sampled, MAE increases exponentially as seen in Figure 15 (a) shown with a logarithmic scale. Figure 15 (b) shows the effect of down sampling each point cloud separately. When the reference point cloud (SWIR) is down sampled the error effectively stays the same but when the target point cloud (XCT) is down sampled the error rises. The same effects were seen with the 2D SEM images. The point-to-point MAE for the 3D registration was 112 µm with 100,000 CT points and 86 µm with 1,000,000 CT points.



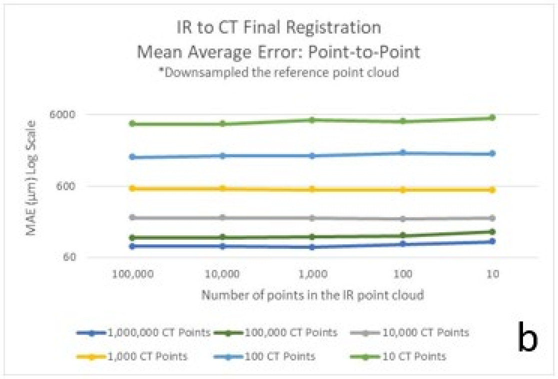


Figure 15: Change in MAE when the target point cloud is down sampled (a) and when the reference point cloud is down sampled (b)

4.2 Point-to-Plane Error

ICP registration of multiple images by point-to-plane has been developed to attempt to improve registration over traditional ICP [30]. The point-to-plane registration method is not used for this study, but the derivation of the error metric for point-to-plane registration has distinct advantages over the point-to-point error metric discussed in Section 4.1. The main advantage is the target is a triangle surface instead of a point cloud so point-to-plane does not depend on the number of points in the target. The second advantage is that DREAM.3D can calculate the individual point-to-plane errors using the *Find Vertex to Triangle Distances* which significantly simplifies the implementation of the algorithm and makes it more repeatable across platforms. Python is only used to calculate the array-level metrics (standard deviation, MAE, MSE, RMSE) which is a very straight forward calculation.

Like point-to-point, the point-to-plane metric matches the closest triangles in the target surface for each point in the reference point cloud. Individual errors are calculated using the distance between each source point and the tangent plane at its corresponding destination point [27, 30].

Unit normals from the face of each triangle in the target surface are added to the triangle surface array in DREAM.3D with the *Generate Triangle Normals* filter. The error between the target surface and sampled points from the reference surface is shown in Figure 16 as e_i.

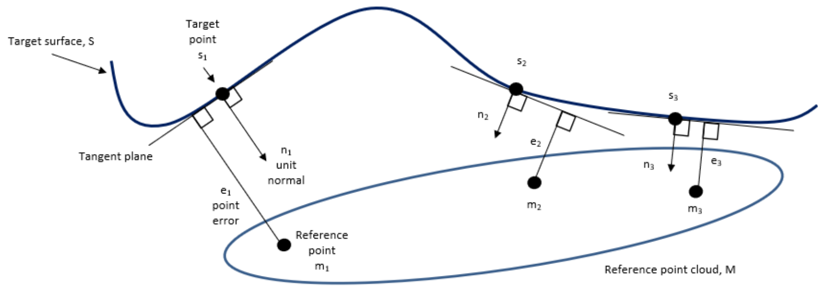


Figure 16: Point-to-plane error from a target surface and points on a source surface.

A summary of the point-to-plane error process

1. Read reference vertex points and the target triangle geometry surface in DREAM.3D
    a. Downsample the reference vertex points.
2. Calculate individual errors, e_i.
    a. For all points in the reference point cloud, M, calculate Euclidian distance to the closest triangle in the target surface
3. Calculate accuracy metrics
    a. Standard Deviation, MAE, MSE, RMSE

4.2.1 Point-to-Plane Error with 2D Images

A triangle surface plane was created in DREAM.3D from the target SEM image. This was done by extrapolating the mask created by the RATS algorithm to several layers and creating a 3D array. The same process in Section 3.1 for 3D images was followed to create a triangle surface of the target image. The mask is shown in Figure 17 (a) and the target surface with a point cloud from one of the SEM imaging positions (red) is shown in (b).

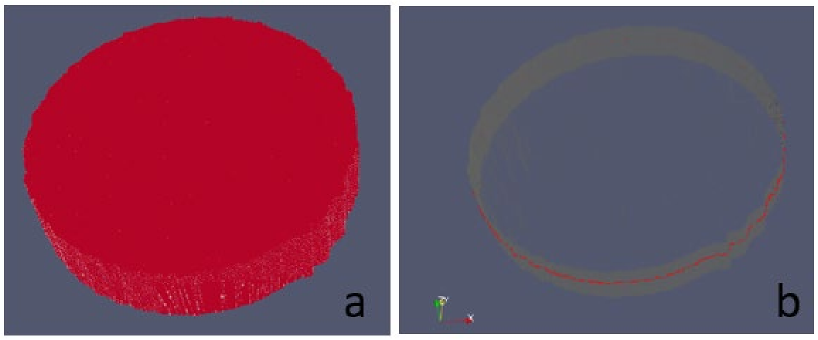


Figure 17: An extruded target image mask (a) and a triangle boundary plane, based on the SEM target image, with one of the SEM imaging position point clouds (b).

The triangle surface plane was the target and the boundary point clouds of each SEM imaging position were the reference point cloud. With this method, there was effectively no change in the MAE when the reference point clouds were down sampled. This is shown in Figure 18. Average MAE across all SEM imaging positions with no down sampling and with all down sampled results are both 2.4 µm.

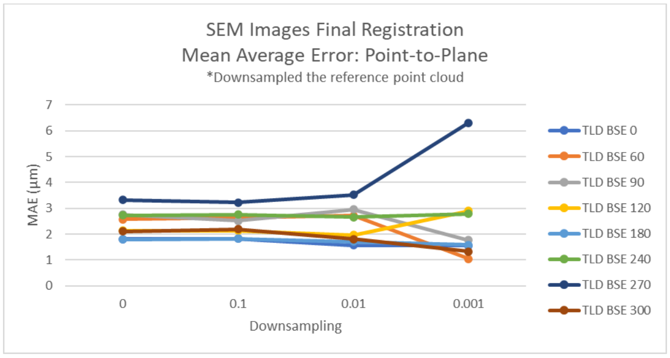


Figure 18: Point-to-plane MAE for SEM images with down sampling

4.2.2 Point-to-Plane Error with 3D Images

The point-to-plane error for the 3D images used the triangle surface created for the XCT array in Section 3.1 to sample the boundary point cloud. The XCT triangle surface was the target and the SWIR point cloud was the reference. Like the 2D images, Figure 19 (a) shows the MAE stays effectively the same when the number of points in the SWIR point cloud were down sampled. The point-to-plane MAE for the 3D registration was, on average with five different levels of down sampling, 75 µm.

Figure 19 (b) highlights the convergence of increasing the points in the target point cloud until it becomes a plane. The scale is logarithmic, making the exponential trend toward the point-to-plane MAE results as the XCT point cloud is increased from 10 points to 1,000,000.

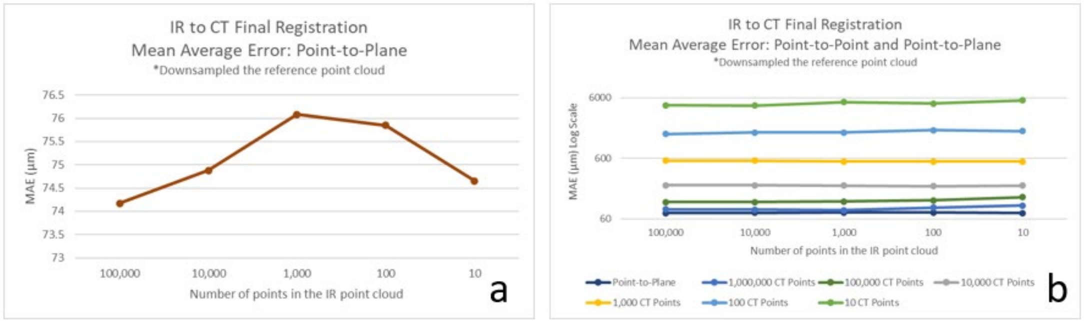


Figure 19: Point-to-plane MAE for 3D images with down sampled SWIR point cloud (a) and comparing point-to-point and point-to-plane MAE

5.0 Data Fusion

To fuse each 3D dataset on a common grid, the DREAM.3D *Fuse Regular Grids (Nearest Point)* is used. The XCT array, along with the final position of the SWIR array, are fused together. The grid of cells in the reference data container (SWIR) is overlaid on the grid of cells in the sampling (XCT) data container. Each cell in the reference container is associated with the nearest point in the sampling container. The values of the nearest point in the sampling container are assigned to that cell in the reference container. This is a nearest neighbor operation as the values of the sampling voxels are not interpolated to the reference voxels. As a result, the sampling array, XCT in this case, is in the same resolution as the reference array, SWIR. Fusing all data on to the same grid allows for any follow-on statistics or machine learning model to directly interpret each voxel of training data discretely.

The 2D SEM images are roughly the same resolution and have the same pixel dimensions so the data fusion process is insignificant in what it changes. The 3D arrays, however, are quite different with the XCT resolution at 29.96 x 29.96 x2 9.96 µm before fusion and 125 x 125 x 50 µm after the fusion process. Figure 20 below shows the difference in resolution of an XCT slice before (a) and after (b) the fusion process with the same layer in the SWIR array (c).

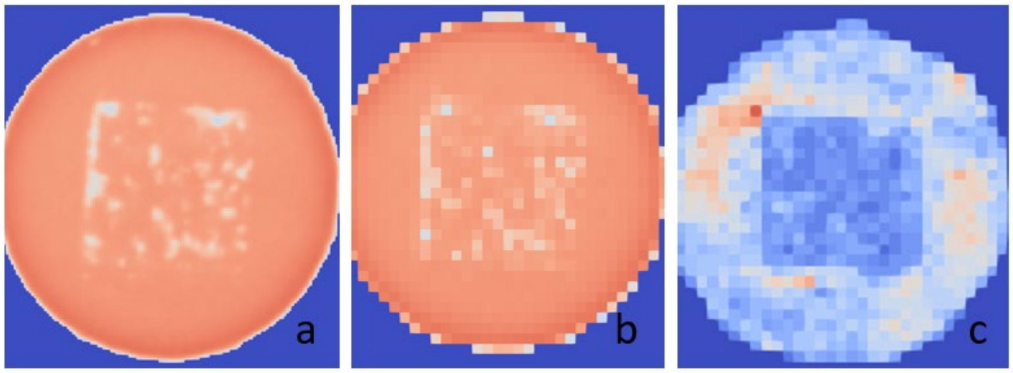


Figure 20: XCT before (a) and after (b) fusion with the same layer in the SWIR array (c).

<u>6.0 Discussion of Results and Conclusions</u>

This study outlined a registration technique that can be used to rapidly register and fuse datasets between multiple sensors and input types and measure the registration accuracy. The process works for both 2D and 3D datasets from multiple data modes and can be extended to other sensor modalities including on- and off-axis optical and thermal images. The publicly available platform, DREAM.3D, provides repeatability and scalability to this technique so the process can be used for analytics in future work.

A point-to-point method has been outlined based on the error distance between matched corresponding points in two-point clouds, and a second point-to-plane method has been outlined based on the error distance between points in one-point clouds to the surface of target dataset. The error metric can be used in DREAM.3D in any type of 2D or 3D dataset, making it easy to objectively judge registration performance in the same units as the dataset.

For a point-to-point error metric to be effective, it is vital to select an extremely high number of points in the target point cloud for an effective point-to-point error metric. The point-to-point error metric converges as the number of points in the target are increased but this study was never able to get enough points in a point cloud to reach the MAE of the point-to-plane method. As more points are added, computing time is exponentially increased. The point-to-plane error metric eliminates this concern because the target is a triangle surface and there is no sampling required.

Figure 21 provides a visualization of the errors at 10,000 randomly chosen SWIR points using point-to-plane method at the final registered position. The values at each point reflect the point-to-plane error for each point in the SWIR array from the XCT surface with blue and red both reflecting larger error, while white reflects small error (error units are in µm). The largest error is seen at the bottom of the part because the because the SWIR camera did not record during the first ten layers of the build. This visualization does indicate that the IR registration has a very slight shift along the negative-Y and positive-Z axis. This may be due to the misalignment of arrays in the first ten layers or the inability to apply a calculated transformation matrix to the image geometries. Future development to apply calculated transformation matrices to image geometries is expected to improve registration accuracy.

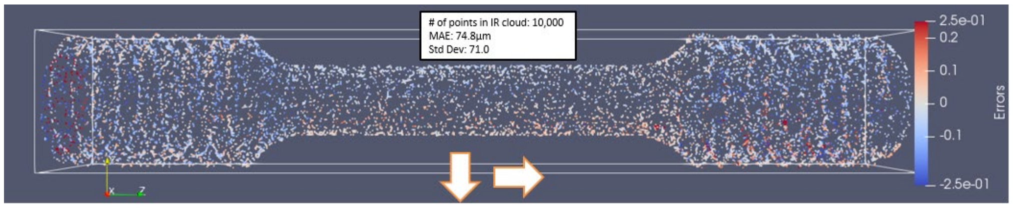


Figure 21: Visual representation of the error in the final 3D registration.

There is always some variation in the MAE because the down sampled point clouds are randomly chosen. Each time a new down sampled point cloud is selected there will be a slight difference in the MAE. This variation increases as the number of points in the point clouds decrease because the impact that each randomly chosen point has on the whole result is increased. This is particularly noticeable in the figures where the down sampling is 0.001 (roughly 23 points) for SEM images or 10 points for the 3D images.

The registration accuracy for all eight final SEM positions is 2.4 µm, found with the point-to-plane error metric. The length and width of each pixel in the images are 1.25 µm meaning that the SEM images were registered within two pixels. Future work will be done to combine the eight SEM imaging positions for an improved defect labeling technique. Registration accuracy is vitally important to this work.

The registration accuracy for the 3D registration was 75 µm. The voxel size of the fused 3D dataset is 125x125x50 µm so the registration is within a voxel. Future work will be done to train a machine learning model on the fused data using SWIR parameters as model. A method to label defects in the XCT array is needed for ground truth labels in supervised machine learning. Registration accuracy is vitally important to this work as well.

DREAM.3D currently cannot apply the transformations that are computed internally. Further development of the *Apply Transformation to Geometry* filter to work with image-type datasets and apply interpolation would fully integrate registration into a single application and the process would be faster. Adding this capability to DREAM.3D should increase registration accuracy because there is known variation between the applied transformation and the calculated optimal transformation.

The 3D images were directly aligned to each other and the 2D images were aligned to a common target. Using this registration technique and error metric future, work can be done to compare the direct alignment of 3D images to aligning to a common target such as a CAD profile.

7.0 Acknowledgements

This work was funded by the Center for Aerospace Manufacturing Technologies and The Boeing Company. The author is particularly grateful to Sean Donegan, at the Air Force Research Lab, for providing guidance on DREAM.3D and Jeremey Eagan, at The Boeing Company, for general computer science support.

8.0 References

[1] S. Klein, M. Staring, K. Murphy, M.A. Viergever, J.P. Pluim, Elastix: a toolbox for intensity-based medical image registration, IEEE Trans. Med. Imaging 2009, 29, 196–205.

[2] J.B. Forien, N. Calta, P. DePond, G. Guss, T. Roehling, M. Matthews, Detecting keyhole pore defects and monitoring process signatures during laser powder bed fusion: A correlation between in situ pyrometry and ex situ X-ray radiography, Additive Manufacturing, Volume 35, 2020, 101336, ISSN 2214-8604, https://doi.org/10.1016/j.addma.2020.101336.

[3] J. Mitchell, T. Ivanoff, D. Dagel, J. Madison, B. Jared, Linking pyrometry to porosity in additively manufactured metals, Additive Manufacturing, Volume 31, 2020, 100946, ISSN 2214-8604, https://doi.org/10.1016/j.addma.2019.100946.

[4] Y. He, B. Liang, J. Yang, S. Li, J. He, An Iterative Closest Points Algorithm for Registration of 3D Laser Scanner Point Clouds with Geometric Features. Sensors. 2017. 17. 1862. 10.3390/s17081862.

[5] S. Feng, Y. Lu, A. Jones, Measured Data Alignments for Monitoring Metal Additive Manufacturing Processes Using Laser Powder Bed Fusion Methods, ASME International Design Engineering Technical Conferences & Computers & Information in Engineering Conference, 2020

[6] G. Mohr, S.J. Altenburg, A. Ulbricht, P. Heinrich, D. Baum, C. Maierhofer, K. Hilgenberg, In-Situ Defect Detection in Laser Powder Bed Fusion by Using Thermography and Optical Tomography—Comparison to Computed Tomography. Metals 2020, 10, 103. https://doi.org/10.3390/met10010103

[7] D. Mattes, D.R. Haynor, H. Vesselle, T.K. Lewellyn, W. Eubank, Nonrigid multimodality image registration. Medical Imaging 2001: Image Processing, pp. 1609–1620.

[8] J. Morgan, J. Morgan Jr., D. Natale, R. Smith, W. Mitchell, A. Dunbar, E. Reutzel, Selection and Installation of High Resolution Imaging to Monitor the PBFAM Process, and Synchronization to Post-Build 3D Computed Tomography. 2017, Solid Freeform Fabrication Symposium, p1382-1399

[9] Z. Snow, B. Diehl, E. Reutzel, A. Nassar, Toward in-situ flaw detection in laser powder bed fusion additive manufacturing through layerwise imagery and machine learning, Journal of Manufacturing Systems, Volume 59, 2021, p12-26, ISSN 0278-6125, https://doi.org/10.1016/j.jmsy.2021.01.008.

[10] S. Everton, M. Hirsch, P. Stavroulakis, R. Leach, A. Clare, Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing. 2016. Materials & Design. 95. 431-445. 10.1016/j.matdes.2016.01.099.

[11] B. M. Colosimo, S. Cavalli, M. Grasso, A cost model for the economic evaluation of in-situ monitoring tools in metal additive manufacturing, International Journal of Production Economics, Volume 223, 2020, 107532, ISSN 0925-5273, https://doi.org/10.1016/j.ijpe.2019.107532.

[12] A. Thompson, I. Maskery, R.K. Leach, X-ray Computed Tomography for Additive Manufacturing: A Review, Meas. Sci. Technol., 2016, 27(7), p 072001–072001. https://doi.org/10.1088/0957-0233/27/7/072001

[13] M. Grasso, B.M. Colosimo, Process defects and in situ monitoring methods in metal powder bed fusion: A review. Measurement Science Technology, 2017, 28, 1–25.

[14] J. Liu, B. Jalalahmadi, YB. Guo, M. Sealy, N. Bolander, Nathan, A review of computational modeling in powder-based additive manufacturing for metallic part qualification. Rapid Prototyping Journal. 2018. 24. 10.1108/RPJ-04-2017-0058.

[15] I. Raffeis, F. Adjei-Kyeremeh, U. Vroomen, E. Westhoff, S. Bremen, A. Hohoi, A. Bührig-Polaczek, Qualification of a Ni-Cu Alloy for the Laser Powder Bed Fusion Process (LPBF): Its Microstructure and Mechanical Properties. 2020. Applied Sciences. 10. 10.3390/app10103401.

[16] T.S. Yoo, M. J. Ackerman, W. E. Lorensen, W. Schroeder, V. Chalana, S. Aylward, D. Metaxas, R. Whitaker, Engineering and Algorithm Design for an Image Processing API: A Technical Report on ITK - The Insight Toolkit. Stud Health Technol Inform. 2002; 85:586-92. PMID: 15458157.

[17] H. Johnson, M. McCormick, L. Ibanez. The ITK Software Guide: Design and Functionality. Fourth Edition.2015. ISBN: 9781-930934-28-3

[18] L. G. Brown, A survey of image registration techniques, ACM Comput. Surv., vol. 24, no. 4, pp. 325–376, 1992.

[19] J. B. A. Maintz and M. A. Viergever, A survey of medical image registration, Medical Image Analysis, vol. 2, no. 1, pp. 1–36, 1998.

[20] M.A. Groeber, M.A. Jackson, DREAM.3D: A Digital Representation Environment for the Analysis of Microstructure in 3D, Integrating Materials 3, 56–72 (2014). https://doi.org/10.1186/2193-9772-3-5

[21] DREAM.3D Data Structure http://www.dream3d.io/1_UsingDREAM3D/%5BA%5D_DataStructure/. Accessed 14 May 2021.

[22] A.S.T.M. E8/E8M, Standard Test Methods for Tension Testing of Metallic Materials, ASTM International, West Conshohocken, PA, 2016.

[23] C.S. Lough, X. Wang, C. Smith, R. Landers, D. Bristow, J. Drallmeier, B. Brown, E. Kinzel, Correlation of SWIR imaging with LPBF 304L stainless steel part properties, Additive Manufacturing, Volume 35, 2020, 101359, ISSN 2214-8604, https://doi.org/10.1016/j.addma.2020.101359.

[24] M.H.F. Wilkinson, F. Schut, Digital Image Analysis of Microbes: Imaging, Morphometry, Fluorometry and Motility Techniques and Applications, 1998, ISBN: 0-471-97440-4

[25] R. Szeliski, Computer Vision: Algorithms and Applications, 2011, ISBN: 978-1-84882-934-3

[26] P. Besl, N.D. McKay, A Method for Registration of 3-D Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1992, 14 (2): 239–256. doi:10.1109/34.121791.

[27] KL. Low, Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration. Technical Report TR04-004, Department of Computer Science, University of North Carolina at Chapel Hill, February 2004.

[28] W. Guan, W. Li, Y. Ren, Point cloud registration based on improved ICP algorithm, 2018 Chinese Control And Decision Conference (CCDC), 2018, pp. 1461-1465, doi: 10.1109/CCDC.2018.8407357.

[29] W. Press, S. Teukolsky, W. Vetterling, B. Flannery, Numerical recipes in C: the art of scientific computing (2nd ed.). 1992, pp. 123-128. ISBN 0-521-43108-5.

[30] Y. Chen, G. Medioni, Object modeling by registration of multiple range images, Proceedings. 1991 IEEE International Conference on Robotics and Automation, 1991, pp. 2724-2729 vol.3, doi: 10.1109/ROBOT.1991.132043.