DIGITAL TWIN VOLUME REGISTRATION FOR VOXEL-BASED CLOSED-LOOP MACHINING SYSTEMS

A Thesis Presented to

The Academic Faculty

By James Samuel Collins

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Mechanical Engineering in the

George W. Woodruff School of Mechanical Engineering

Georgia Institute of Technology

August 2018

DIGITAL TWIN VOLUME REGISTRATION FOR VOXEL-BASED CLOSED-LOOP MACHINING SYSTEMS

Approved by:

Dr. Thomas R. Kurfess, Advisor
Committee Chair & Professor
The George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Thomas M. Tucker,
Principal
Tucker Innovations Inc.

Dr. Christopher J. Saldana
Assistant Professor
The George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Yan Wang
Associate Professor
The George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: 07 / 20 / 2018

ACKNOWLEDGEMENTS

This thesis could not have been completed without the guidance and support of my teachers and fellow students. I would like to thank my academic advisor, Dr. Thomas Kurfess, for his professional guidance and support, as well as the opportunity to have worked on his research team. It has been an incredible experience and I am proud to have been a part of it. I would also like to thank Dr. Tommy Tucker, whose generosity and willingness to share his time and experience has been invaluable in the completion of my research.

For their support an encouragement, I would also like to recognize the contributions of my fellow students and researchers involved with the Precision Machining Research Consortium at Georgia Tech. Specifically, Roman Burkhart, Aoyu Chen, Dongmin Han, Wafa Louhichi, Roby Lynn, John Miers, Vinh Nguyen, Mahmoud Parto, Maxwell Praniewicz, Pierrick Rauby, Kyle Saleeby, Chelsea Silberglied, Ethan Wescoat and Jing Yu. Their professional council and personal friendships have been and continue to be a great privilege.

Most of all, I would like to recognize the support and encouragement of my parents. I could not have completed this without their love and patience.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS & ABBREVIATIONS

CAD	–	Computer-aided design
CAM	–	Computer-aided manufacturing
CLMS	–	Closed-loop machining systems
CNC	–	Computer numerical control
DT	–	Digital twin
GA	–	Genetic algorithm
HDT	–	Hybrid dynamic tree
ICP	–	Iterative closest point
NC	–	Numerical controller
NNS	–	Near-net-shape
PSO	–	Particle swarm optimization
SA	–	Simulated annealing
STL	–	Stereolithography (file)
F	–	Floating image/volume
R	–	Reference image/volume
$H(X)$	–	Entropy of a set X
$H(X,Y)$	–	Joint entropy
$I(X,Y)$	–	Mutual information between X and Y
$p_X(x)$	–	Probability mass function
t	–	Iteration number
$\mathcal{F}(X)$	–	Fitness / objective function
$\mathbb{S}$	–	Point set
$d_{\mathbb{S}}(\rho)$	–	Smallest absolute distance of point in set
ρ	–	Point
$\boldsymbol{R}$	–	Rotation operation
$\boldsymbol{t}$	–	Translation operation
$\mathbb{P}$	–	Point cloud set
$\mathcal{P}$	–	Population set
$\check{X}$	–	best individual solution encountered
$\hat{X}$	–	best global solution encountered
c	–	acceleration constant
r	–	Random variable component
w^t	–	Inertial weight at iteration t
X^t	–	Input variable at iteration t
k_B	–	Boltzmann's constant
ΔE	–	Change in energy
T	–	Temperature
$\mathcal{N}(X^t)$	–	Neighborhood function
α / β	–	User-define parameters

SUMMARY

The goal of representing the physical state of a part throughout the manufacturing process as a digital twin has become a popular topic in recent years. With new advancements in additive-subtractive (hybrid) technologies, the need to gather and process spatial information from inside a CNC system has intensified. Open-loop g-code execution continues to operate with no feedback to describe the current state of the workpiece. It is evident that scanning sensors must be integrated into future machining systems in order to construct a closed-loop architecture whereby the controller can process geometric data to update subsequent commands. This dynamic, closed-loop g-code architecture will revolutionize manufacturing.

In order to advance the research in close-loop machining systems, this thesis presents a simple but novel technique for voxel volume model registration. This is done through the application of registering and machining near-net-shape structures and rough castings. Through the implementation of a Euclidean distance transform and variance calculation, an intensity-based similarity metric is demonstrated over a discrete voxel domain driven by a metaheuristic registration algorithm. Simulation tests conducted over a uniform grid structure show that the technique is successful in positioning a floating volume inside its corresponding near-net-shape. Results for six 0.1mm resolution voxel models are reported followed by the metric’s performance under different starting conditions and registration constraints. Tests indicate that the technique works best for narrow to moderately offset volumes.

The technique is presented as a prototype to demonstrate the viability of the method. Further applications and refinements of this simple technique will provide engineers with an additional method for part registration to be used in future developments of closed-loop machining systems.

CHAPTER 1: INTRODUCTION

1.1 Problem Context

The machining of castings, weldments and irregular structures present several challenges to production. The field of reverse engineering often requires that parts be duplicated when a *computer-aided design* (CAD) model is not readily available. Being able to digitally represent this data in a fast and accurate manner is imperative to compete in a modern, global economy which demands rapid production and short process workflow times. Similarly, machine or robotic toolpaths generated through a *computer-aided manufacturing* (CAM) software require the user to define a starting volume or stock material. Many such starting volumes or rough castings are complex and vary in their dimensional compliance. Even if an accurate computer model of the starting volume can be obtained, the programmer must position the CAD model properly relative to this starting material.

Uncertainties about the actual dimensions of the starting cast material when machining rough castings is a topic of common concern. The process of assessing a workpiece prior to machining is referred to as “marking-out” and typically involves the manual process of leveling on a special plate [1]. There is a danger that a rough casting received on the shop floor is not within tolerance due to the dimensional inaccuracies from thermal deformation in the manufacturing process. Industry needs to quickly determine whether a part is within tolerance to ensure a necessary allowance for all to-be-machined surfaces or if it should be scrapped. Optical scanners and computer vison programs have been used to scan and calculate whether a casting is adequate for machining and what adjustments should be made for proper alignment. For many of these computer vison techniques, datum planes or points are needed and are created by either machining the stock itself or by integrated markers on a specialized fixture [2]. Various fixture schemes exist [3][4] for specific parts so the pre-programmed g-code can be run with little to no adjustment to the material. However, the processes of marking-out and any subsequent part-fixture adjustment is time-consuming, and therefore costly.

These difficulties stem from the fact that the traditional manufacturing process workflow of progressing from CAD to CAM to execution on a *numerical controller* (NC) is inefficient. Currently, toolpath programming is done separately from the actual *computer numerical controller* (CNC) which executes the commands specified in a g-code file. This puts the burden on the shop floor to conform to machine kinematics that have already been defined. Little flexibility is offered in cases of damaged stock, changing fixtures or swapping tools.

This research argues that the most effective way to machine weldments and irregularly shaped casings, particularly in a low volume production scenario, is through the development of a *closed-loop machining system* (CLMS). That is, a system capable of sensing and processing empirical data about the workpiece and generating g-code directly from the controller. Such a system would be able to update toolpaths in discrete, real-time intervals throughout the manufacturing process. Through the use of optical scanning technology, a CLMS would have the ability to evaluate the material removal error from tool/part deflection and tool wear to make corrections in subsequent g-code commands directly. Moreover, interruptions from tool breaks and minor crashes would not necessarily mandate that the machining process be restarted.

In order for such a system to be implemented, methodologies for part/volume registration need to be explored. After a CAM package embedded in a CLMS acquires positional data describing the workpiece, the software needs to process the relationship between the current stock volume and the desired final part. This *digital twin* (DT) is the key to generating toolpaths from the *near-net-shape* (NNS) stock instead of maneuvering the workpiece position to conform to a static g-code. In other words, the CAD model describing the target part geometry needs to be aligned inside the NNS. Such an alignment should be one that ensures enough material is present to encompass all to-be-machined surfaces. For efficient machining, the amount of material encompassing the final part model should be as uniform as possible to avoid unnecessary toolpaths and maintain consistent feed rates.

1.2 Purpose of Thesis

This thesis presents a process for digital twin part/volume registration within the context of closed-loop machining systems. It is proposed that integrating optical scanning technology into CNC machines will allow for closed-loop feedback of a DT representation. Comparisons between this DT and a desired final part geometry model will allow machine controllers to produce g-code dynamically. This research focuses on the initial task of investigating a technique to register CAD models with volumetric data from point cloud scanners.

A proposed intensity-based optimization metric is demonstrated as a simple but novel approach to NNS voxel volume registration. Unlike methods commonly used in medical imaging and point clouds, which are focused on consolidating separate sets of data into a single, overlaid volume, the registration described in this thesis minimizes the variance of selected distance field values. This allows for an optimization algorithm to drive a floating model to be positioned inside a larger reference volume so that the amount of material present between the two models is as uniform as possible.

In order to demonstrate the utility of this method, registration tests were conducted for voxel-based models over a simple uniform grid structure. The initial results of applying three different metaheuristic optimization techniques are reported to broadly illustrate their behavior.

Additional tests conducted using a genetic algorithm demonstrate the method’s performance under different conditions and constraints. In each case the computation of a 6 degree-of-freedom transformation is directed to fit a target part geometry inside a NNS rough casting volume. A test suite of parts with differing levels of complexity shows that the technique is applicable to a wide range of geometries.

It should be noted that this thesis does not seek to provide a direct implementation of point cloud to voxel part registration. Instead, the background and basic theories are discussed and demonstrated to facilitate further research into CLMS and their machine/computer vision registration algorithms. While a complete development of the ideas presented is beyond the scope of this research, the proposed method’s feasibility is validated by a series of successful test results for NNS volume registrations. Areas where the registration does not perform well are also covered to indicate in what situations the technique should not be applied.

CHAPTER 2: BACKGROUND AND PRIOR WORK

This section covers related research and prior work on four main topics that are included in this thesis: the digital twin concept, closed-loop CNC control, direct point cloud machining and alignment algorithms for machinable castings.

2.1 The Digital Twin

Advancements in model-based design, digital manufacturing, cloud computing and big data analytics is leading industry to what will likely be the next industrial revolution. The ease of acquiring and transferring real-time digital data has made it possible for engineers to consolidate this information into a virtual object. This object is referred to as a *digital twin*. Glaessen and Stargel [5] define the DT as “an integrated multiphysics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying [or physical] twin.” Similarly, Boschert and Rosen [6] refer to it as “a comprehensive physical and functional description of a component, product or system which includes more or less all information which could be useful in all—the current and subsequent—lifecycle phases.”

Although efforts have been made to provide a rigorous, conceptual template for the concept [7], the DT can be simply regarded as the result of advancing information and sensor technology providing a new wealth of data for engineers to make more informed decisions. In general, the DT can be thought of as being comprised of three main sub-categories: (1) the physical product/system or “physical space”, (2) the virtual representation of the product/system or “virtual space” and (3) all the data lines that connect the physical and virtual product/system which facilitate the exchange of information [8][9]. It should be noted that the interaction between the two spaces is a key feature of the DT. For example, a sensor measuring tool chatter (physical to virtual interaction) may send the frequency data to a cloud register where a program can then send commands to adjust the spindle speed and/or feed rate (virtual to physical interaction).

Numerous applications of the DT concept have been published. Hochhalter [10] described the use of embedded sensory particles with a DT model to detect cracks in NASA vehicles parts. The integration of non-geometric data into a CAD model was investigated by Miller et. al. [11] as an advancement in model-based paradigms. This demonstrated data storage of parameters such as heat, pressure and surface roughness in a CAD plugin. Zhang et. al. [12] applied the DT to customizable design through a case study of a glass production line. Finally, steps were taken by Knapp et. al. [13] to build a DT for additive manufacturing. Their model produced accurate predictions for temperature, cooling and solidification parameters.

Although it is equally valid to apply the DT concept to entire systems, we will focus on the DT's role in a single-unit production for the remainder of this research. Specifically, we will consider the interaction between the physical and virtual representations of a part through the machining processes.

2.2 Closed-Loop CNC

As industry reacts to changing conditions in a globally competitive market, machining systems must respond by integrating the separate process workflows of design and NC operation. G-code (ISO 6983) is the low-level language which is still the prevailing machine tool programming standard. STEP-NC (ISO 14649) was developed as an extension of the *Standard for Exchange of Product* data model (STEP) (ISO 10303) to facilitate the bidirectional exchange of geometry data between CAD-CAM systems [14]. Unlike g-code, which consists of sequential tool and motion commands, STEP-NC is feature-based, high-level and object-oriented. This makes the standard much more intuitive. Furthermore, STEP-NC allows for shop floor feedback, making production more flexible and responsive to changes. Whereas traditional g-code maintains a separation between CAD, CAM and the NC controller, STEP-NC integrates the process. This makes the controller the central intelligence capable of generating toolpaths from an embedded CAM system [15].

Naturally, engineers have started examining this new closed-loop CNC architecture capable of processing measurement data to alter toolpath trajectories and parameters accordingly. Dynamic path planning will further automate the manufacturing process and avoid errors from tool wear, chatter and misaligned fixtures. The research from Brecher et. al. investigated the combination of part inspection with data feedback in the STEP-NC workflow. They also stressed that STEP-NC is a passive data model requiring intelligent, closed-loop functionality to be developed independently in order for closed-loop process planning to be realized. Rauch et. al. [16] introduced a STEP-NC platform designed for simulating and optimizing high speed machining operations. A prototype architecture and object-oriented controller was constructed by Hu et. al. [17] with integrated sensors for real-time control. Recently, Lynn et. al. [18] implemented a direct servo control scheme for digital volumetric processing from a 5-axis CAM software. Their servo position feedback loop was demonstrated on a modified, consumer grade 5-axis mill with a *LinuxCNC*© based controller.

2.3 Direct Point Cloud Machining

Traditionally, reverse engineering necessitated the acquisition and conversion of discrete points to a CAD model before proceeding to additive or subtractive operations [19]. However, the meshing process is time consuming and can be susceptible to errors introduced by the file format conversion. The desire to avoid this conversion has led recent investigations into direct point cloud machining for reverse engineering. Much of the prior research in this area has been focused on the development of various path planning strategies for direct 3-axis machining from point cloud data. Barnfather and Abram [20] presented an algorithm which used this data as a means of error compensation for lower cost industrial robotics. The dimensional error in machining was considerably decreased by integrating optical point data into the path planning computations. Feng and Teng [21] applied a piecewise path generation algorithm to point clouds. Zou and Zhao [22] similarly extended traditional iso-parametric toolpath planning to discrete data points through an iterative computation of path parameters and linear interpolation. Zhang et. al. [23] used neighboring points within a bounded search to fit curvature-adaptive toolpaths to point cloud data. A B-spline interpolation was used by Masood et. al. [19] and Liu et. al. [24] to generate toolpaths from scanner data stored in a tessellated format. Xu et. al. [25] proposed a path planning technique which used a least-squares projection of paths from a generic surface about a minimum bounding box containing the point cloud data. Machine tool roughing and finishing strategies were achieved through boundary extraction and 2D curve offsetting operations respectively. Finally, a method for generating toolpaths that yield a constant scallop height was proposed by Liu et. al. [26] which significantly reduced the path length compared to an iso-planar method of path generation.

2.4 Alignment of Castings for Machining

Various improvements to the prevailing methods used in the post-processing and finish machining of rough castings have been proposed. A technique for an automated cast or forged workpiece comparison with a CAD model was introduced by Chatelain and Fortin [27]. This research used a balancing technique to properly align the cast workpiece by offsets incorporated into the fixture. Their algorithm compared a measurement dataset taken from the casting with the published CAD model. Convergence to an optimal placement in the fixture indicated that the casting had enough material to position the CAD model inside the measured dataset. Otherwise, the workpiece would be scrapped for being too small. Both least-squares and logarithmic objective functions were used in a nonlinear, global constrained optimization to compute an optimal 4×4 homogeneous transformation matrix. Gessner et. al. [1] utilized an optical scanner and reference CAD model to automate the marking-out process. The point cloud/CAD alignment was achieved through a minimization of machine allowances (i.e., minimal amount of material removed) derived from the distances between the reference CAD model and the average plane calculated from the workpiece surfaces. Haghighi, Ramnath and Kalish et. al. [28][29][30] proposed a means of calculating rough casting fixture adjustments when machining large castings and weldments. Here again, the parameter search also served the additional purpose of evaluating whether the casting of interest was within tolerance or if the decision should be made to scrap the workpiece. Their research used point cloud data describing the casting to determine the position of the workpiece relative to the machine’s pre-programmed toolpaths.

Each of these techniques are successful since simple fixture adjustments are sufficient to modify the machining process of rough castings and weldments without regenerating g-code for

3-axis CNC machines. However, they are not directly applicable to operations requiring adapted, newly generated g-code such as hybrid manufacturing and direct digital 5-axis machining.

The research conducted by Lei and Zheng [31] is of particular interest since their approach has similarities to the technique proposed in this thesis. Lei and Zheng used a combination of laser trackers and CNC positioners in a closed-loop alignment process for large-scale components. Through this process, a link was established between the laser tracker, positioning system and CAD model. Different coordinate systems were used to compute a proper alignment by minimizing the variation of key (geometric) characteristics [32] (e.g., profile, assembly hole position, etc.). This approach distinguishes datum sets based on parallelism, position and concentricity to establish a metric for a geometric transformation of the CAD model. The CNC system handled both the motion control of the positioners and the machining by communicating with an industrial computer running the alignment process.

The presented thesis research expands this prior work by proposing that a fully integrated control system run directly on a joint CNC-CAM controller. This controller would use point cloud data to produce a DT model of the starting volume as opposed to tracking discrete datum points. Since an entire spatial model of the DT is generated, this allows for distance field calculations to be referenced in aligning CAD models to rough castings and irregularly shaped starting volumes.

CHAPTER 3: OVERVIEW OF RELEVANT THEORY

This chapter briefly covers the technical background and core theoretical concepts presented in this thesis. Voxels and voxel data structures are described followed by an overview of the image registration process. Two popular registration algorithms are explained since they are later used to demonstrate that additional image registration metrics are needed for part/volume registrations in CLMS. The remainder of this chapter presents the ideas behind distance fields and the basics of three different metaheuristic optimization algorithms used in later tests.

3.1 Voxels and Voxel Modeling

Voxels can be defined as 3D image pixels. Thus, voxels can be viewed as the 3D counterparts of pixels more commonly encountered in 2D imaging. That is, pixels are the unit squares that describe an image whereas voxels are the unit cubes that constitute a volume. An additional dimension of depth allows voxels to represent solids in the discrete domain of digital computers. Boolean (True/False) values can be tied to specific voxels indicating the presence of a solid object. Alternatively, intensities, like the grayscale/RGB values in pixels, can also be assigned to voxels to signifying parameters such as distance, density or material.

3.2 Uniform Grid

A uniform grid voxel structure is one of several methods used to store and represent voxels. In a uniform grid, a bounding box about a space is subdivided to cells (i.e., voxels) of equal size. This data structure is simply a 3D array with its matrix lattices aligned to the space’s world coordinate system. Volumes contained inside the bounding box are represented by voxel values at discrete locations. For example, a voxel located at a particular (X, Y, Z) coordinate may be a assigned a binary "0" or "1". In the case of modeling solids, this indicates the cubical cell contains either empty space or an intersection with a surface/solid respectively. Regular, uniform spacing in a grid structure allows for easy implementation and indexing. However, these structures are impractical when attempting to represent high resolution volumes. As with 2D curves, large numbers of voxels are required to approximate surfaces in the discrete domain. In the case of 3D grid arrays, refining the resolution increases the number of voxels by a cubic order of magnitude. The memory allocation required can quickly become too cumbersome for CPUs/GPUs to store and process. Uniform grids are therefore only practical when the target volume can be represented at relatively low resolutions [33]. Figure 1 shows the difference in how resolution affects part representation.

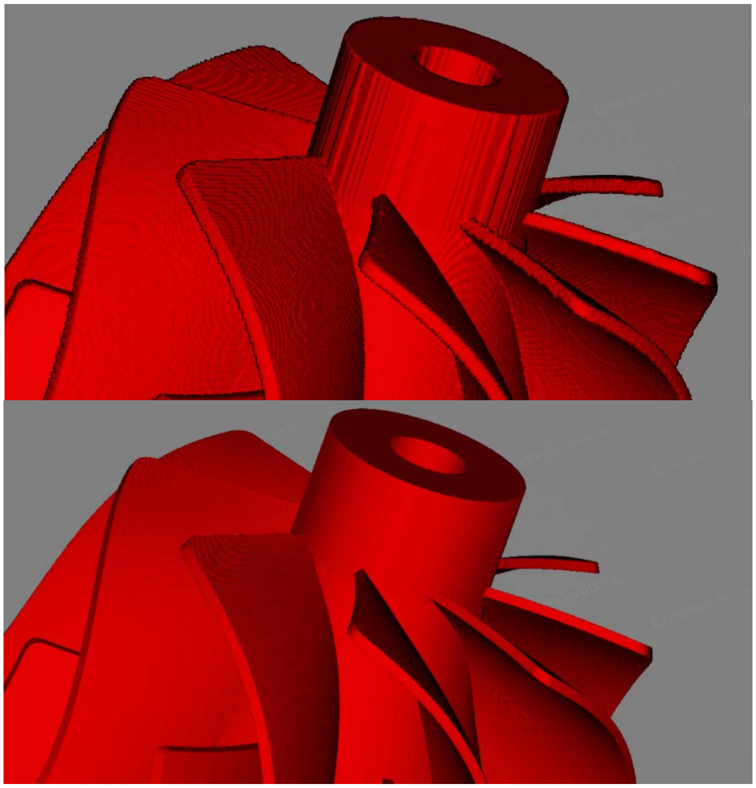


Figure 1. Comparison between voxel resolutions of an impeller model

3.3 Other Voxel Storage Structures

The shortcomings of a direct, single-level uniform grid can be overcome through more efficient spatial descriptions of the volume. An improvement to uniform grid, called a two-level grid, exploits adaptive refinement by making each voxel itself a uniform grid of arbitrary resolution [34]. The first, general grid is the top-level and the collection of grids corresponding to individual voxels is the second-level. However, this hierarchical structure is fixed at two grid levels and therefore still is susceptible to similar problems in memory overhead for high resolutions [33].

An octree is another data structure that utilizes a voxel hierarchy by dynamically subdividing the bounding box space containing a solid. It is regarded as the canonical way of improving grid-based methods [35]. Unlike uniform grids, voxel sizes are not required to be regular or uniform over the entire space containing the volume. Moreover, the depth of the hierarchy is determined by the extent to which the algorithm adaptively subdivides space. The selected resolution then corresponds to the smallest voxel size used in the octree partitioning. Figure 2 is a graphical description of how a quadtree (the 2D equivalent of an octree) hierarchy is constructed.

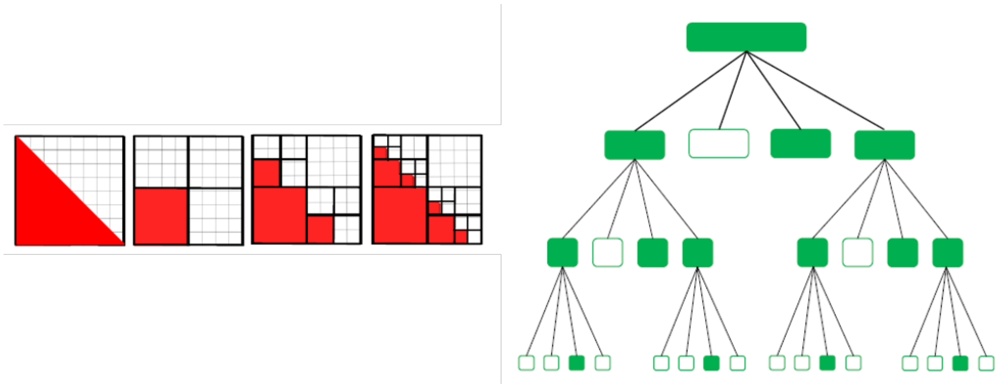


Figure 2. Quadtree of a triangle and resulting hierarchical diagram [36]

Octrees follow a similar logic but sequentially subdivide 3D space into eight octants rather than four quadrants in 2D space. The resulting nodes from the partitioning are evaluated and categorized based on their contents. Full/empty nodes are those that contain a solid or empty space respectively. A partial node signifies an intersection with an object's surface and is marked for further subdivision if the minimal voxel resolution has not yet been reached. Here, octrees are explained in the context of a parametrized CAD model; however, point cloud data can also be stored as an octree where octant subdivisions are governed by whether or not points are contained inside the nodes. The disadvantage to using an octree to represent high resolution parts is that it requires many levels in the overall tree structure. This can become a computational bottleneck in applying transformations and data processing [37][33].

A hybrid voxel representation called the *hybrid dynamic tree* (HDT) uses an adaptive hierarchical (tree-based) data structure which reduces the memory footprint requirement for high resolution 3D data and allows for fast GPU processing [36][33][38][39]. The advantage of HDT structures is that it blends dense grids with octrees to be well suited for parallel execution on a GPU device [40]. The voxel-based CAM software, *SculptPrint*©, used in this research represents part geometries in a HDT data structure. The computational load of 5-axis inverse kinematics, path planning and collision avoidance is made worse by working from complex *boundary representation* (B-Rep) and *non-uniform rational b-splines* (NURBS) surfaces. SculptPrint mitigates this challenge by discretizing parametric curves into a HDT voxel structure. As a result, calculating material removal or model intersections becomes a simple operation of removing boolean voxels from regions of space and executing logical comparators respectively.

Other voxel structures, such as hash table and k-d trees, are not considered here as they are not applicable to the scope of this research or recommendations for future work.

3.4 Image Registration

The general goal of image registration is to obtain a transformation of a floating image *F* that best aligns it with a reference image *R*. This typically means the complete superimposition of *F* over *R*. However, this is not always the case, as when attempting to align partial point cloud scans through a closest-point algorithm. Since these algorithms minimize the overall distance between point sets, segmentation is typically required where structures of interest are selected and isolated for analysis [41]. Therefore, a particular image registration program must be selected with knowledge of the criteria necessary to achieve the desired alignment given a particular demand. In registration terminology, images not only refer to arrays of pixels, but also to 3D volumetric structures. This research will use the terms “image” and “volume” interchangeably when describing a 3D registration. The convention of using the letters *F* and *R* to refer to the respective registration objects will also be followed.

The registration process can be generalized to four main components: a transformation function, cost function, similarity metric and optimization algorithm. Figure 3 shows a generalized sequence in which these four structures interact. The transformation function iteratively takes the parameter(s) to be optimized and uses the current value(s) to describe a particular geometric transformation. This is often in the form of a 4×4 homogeneous transformation matrix. The following flowchart in Figure 3 illustrates the transformation operator *T(x)* acting on the image *F*.

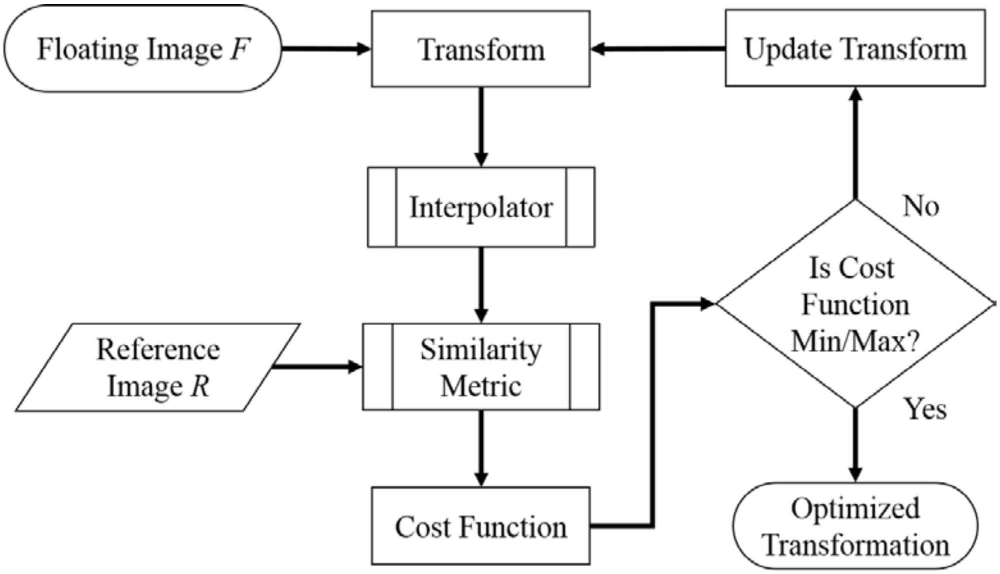


Figure 3. Generalized flowchart for registration optimization

Voxel registration techniques are driven by intensity-based cost functions. This is in contrast to feature or distance-based techniques such as closest-point and pattern search algorithms. An intensity-based cost function is created with an expectation of how an image's voxel intensities should relate to its corresponding image when properly aligned [42]. For example, if the two images have the same or very similar intensity values when aligned, an intensity cost function would attempt to minimize the differences between the intensity values [43]. Cases where the intensities should be nearly identical at corresponding points are referred to as intramodal. In practice, this occurs when attempting to register two images taken from the same sensor. However, when two images are taken from different sensors or imaging study (e.g. magnetic resonance imaging and computer tomography) the intensity values representing the same object are often recorded differently [44]. Intermodal or multimodal intensity cost functions are implemented in such situations where an alignment of slightly dissimilar perspectives is required.

Medical image registration is an area that continues to see increased interest as clinical and research applications demand faster and more efficient algorithms [42]. Although general image registration is applicable to areas of machine vision and robotics, a large portion of the published research in the field is in medical imaging [45][46]. Medical registration is focused on the comparison and alignment of the same area of interest taken from different perspectives or at different times. For example, a qualitative analysis of the progression of multiple sclerosis requires multiple scans to be overlaid [47]. Further applications include new areas of image guided surgery [45][48].

3.5 Information Entropy and Mutual Information

The concept of information entropy is used in medical imaging to quantify the uncertainty contained when comparing image stack files from different measurement machines. Qualitatively, entropy represents the amount of "uncertainty" or "unpredictability" contained in an information source. Considering the discrete case of a random variable X which takes on values $X = \{\, x_1, x_2, \ldots\}$ with associated probabilities of occurring described by the probability mass function $p_X(x) = \ \{p_1, p_2, \ldots\}$, the entropy contained in X may be quantitatively written as

$$H(X) = \ -\sum_{x \in X} p(x)\, log_2\, p(x) \tag{1}$$

Here, $log_2\, p(x)$ is measured in units of bits. It can be seen that the less information known *a priori* about the likelihood of events in the set X occurring, the greater the entropy measure $H(X)$. In other words, $H(X)$ is maximum if all p_i have an equal likelihood of occurring such that $p_i = \frac{1}{i} \forall i$ [44]. For example, a fair die has 6 equally likely outcomes.

$$H(D) = -\sum_{i=1}^{6} p(d)\, log_2\, p(d) = -\frac{1}{6}\sum_{i=1}^{6} log_2\left(\frac{1}{6}\right) = log_2\, 6 \tag{2}$$

If the die in question was loaded (more information is known *a priori*) then the probability mass function $p(d)$ would not be uniformly distributed. Consequentially, $H(D)_{fair} > H(D)_{loaded}$.

Consider the case of two discrete random variables *X* and *Y* defined on the same probability space. The behavior of the two random variables may be simultaneously defined over a joint probability distribution and described by the joint probability mass function $p(x, y)$. This can be visualized as a surface in 3D space. It follows that the joint entropy quantifying the amount of uncertainty of two discrete information sources over the same space is

$$H(X, Y) = -\sum_{x\in X,}\sum_{y\in Y} p(x, y)\, log\, p(x, y) \tag{3}$$

Both of these entropy measures need to be calculated by approximating the probability distribution functions. This is accomplished by constructing histograms for the entropies of *X* and *Y* individually and a joint histogram for the joint entropy. A joint, or bivariate, histogram is analogous to traditional histogram plots on a 2D surface but with an added dimension. Figure 4 and Figure 5 show the different ways of graphically representing histogram and joint histogram data.

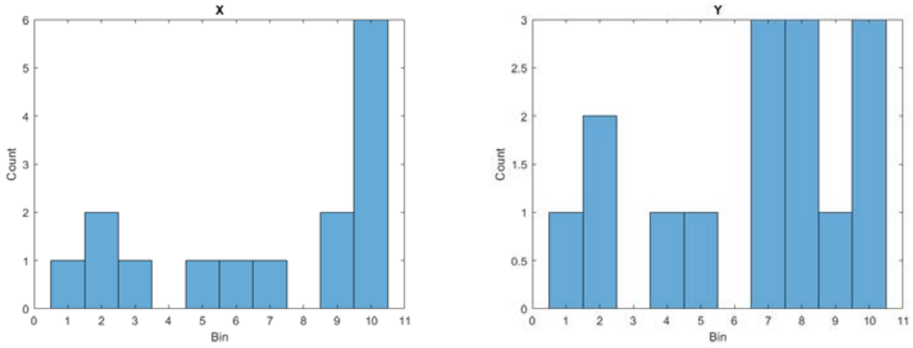
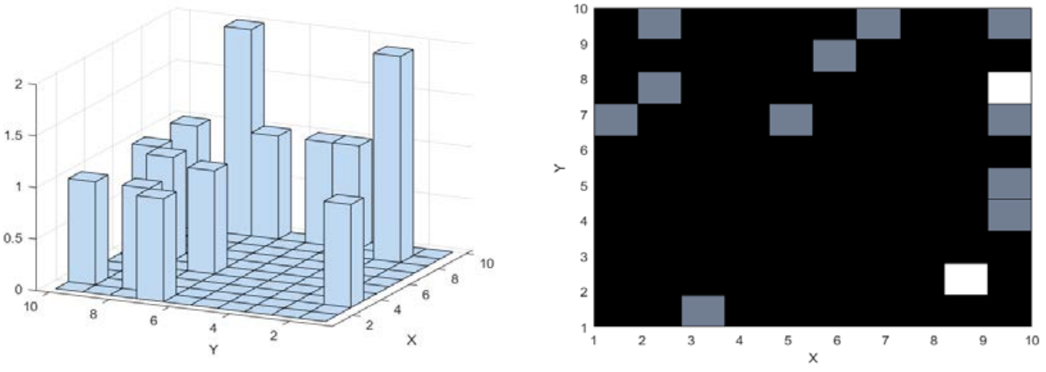


Figure 4. Histogram plots for two random variables *X* and *Y*



(a) (b)

Figure 5. Two representations of a joint histogram
(a) a joint histogram with bin counts indicated incrementing along a Z-axis
(b) a joint histogram with bin counts indicated by pixel intensity (black to white)

A popular registration technique used in medical imaging involves finding the configuration that yields the *maximum mutual information* (MMI) between two images. Derived from information theory, *mutual information* (MI) measures the statistical dependence of two random variables. In terms of voxels and images, MI indicates how much information one image contains about the other based on the intensities of corresponding voxels in each image [49]. The MI between two random variables can be written as

$$I(X,Y) = \sum_{x\in X}\sum_{y\in Y} p_{XY}(x,y)\log\frac{p_{XY}(X,Y)}{p_X(x)\ \cdot p_Y(y)} = H(X) + H(Y) - H(X,Y) \tag{4}$$

Thus, MI depends on the joint entropy of the two variables taken together as well as their marginal entropies. These individual entropy measures are the amounts of information contributed from overlapping regions of X and Y. Rewriting this equation in terms of a translated, floating image *F* and rigid image *R*,

$$I(F,R) = H(F) + H(R) - H(F,R) = H\big(T(F)\big) + H(R) - H(T(F),R) \tag{5}$$

Here, *F* is explicitly shown as undergoing a geometric transformation by the operator *T(X).* Finding a positioning of *F* that most closely aligns it with *R* is equivalent to maximizing the mutual information between the two images. It can be seen in Equation 5 that this corresponds to maximizing the marginal entropies $H(T(F))$ and $H(R)$ while minimizing the contribution of the joint entropy $H(T(F), R)$. Since R remains fixed, $T(F)$ is the only term that determines the amount and quality of overlap between the two volumes.

Minimizing the final term directly decreases the overall amount of uncertainty between the two images. That is, the absolute value of the joint entropy is decreased with transformations that result in regions of *R* superimposed over complex regions of *F* [47]. In other words, minimizing the joint entropy term is analogous to maximizing how well *F* “describes” *R*. The amount of joint entropy contained between the two images is evaluated based on the occurrence of filled voxel space at various uniform grid points.

Since MMI is based on the mutual information as described by voxel intensity values, observing a graphical representation of the joint histogram between two overlapping images can lend a qualitative understanding of the algorithm’s behavior. Figure 6 demonstrates a 2D registration of an early *Apple*® logo and its corresponding joint histogram plots. Similarly, Figure 7 shows a set of colored intensity joint histograms as an image is moved to superimpose its copy. When misaligning two images and tracking the resulting joint histograms, high intensity regions decrease and plots exhibit dispersion characteristics [44]. Assuming the two images are identical, a transformation that perfectly superimposes *F* over *R* is one which has a 1:1 correspondence between all voxel intensities. That is, all voxels with value 2 should be overlaid with voxels also of value 2, and so on. For each 1:1 mapped pixel between *F* and *R*, the count on the joint histogram is increased by one. If two identical images were perfectly aligned, we would expect bin counts only along the main diagonal of the joint histogram. Therefore, qualitatively judging the accuracy of a fit by evaluating the joint histogram alone becomes equivalent to seeing how close or dispersed the bin counts are from the main diagonal. A different similarity metric based on joint histograms called cross-correlation operates under this assumption.

However, it should be noted that minimizing the joint entropy alone is not equivalent to maximizing mutual information. Recall that the marginal entropies $H(T(F))$ and $H(R)$ are not invariant to the amount of overlap between the two images [44].

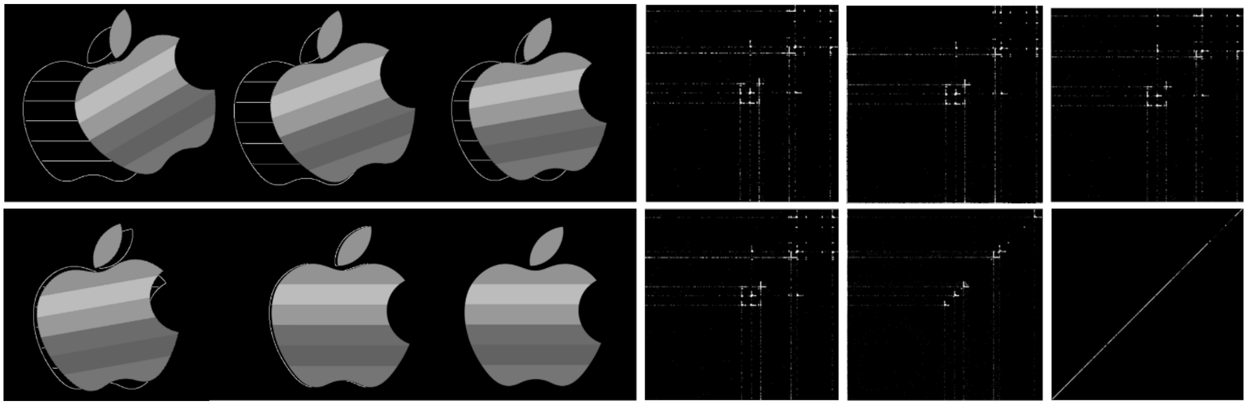


Figure 6. Apple® logo registration with joint histogram plots

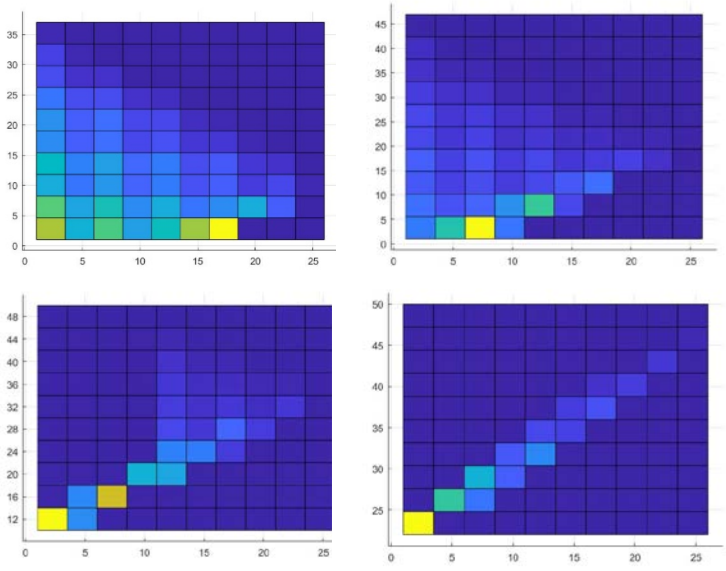


Figure 7. Colored intensity histograms through a superimposition sequence

These terms are capable of changing through each registration iteration. In some situations, this overlap dependence is mitigated through normalization schemes [49]. Since MI depends on both the marginal and joint entropies, this causes the similarity metric to not assume a specific linear nature of dependence on mapped voxel intensities [49].

3.6 Iterative Closest Point

Iterative Closest Point (ICP) is a simple and popular algorithm used in the geometric processing and image registration problems [50]. The method matches two surfaces by minimizing the distance between corresponding points based on an initial displacement. Figure 9 shows the general pseudocode for an ICP algorithm. A point set $\mathbb{P}_1$ is transformed through a rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$ so that each point $\rho_{1i} \in \mathbb{P}_1$ will be closest to its corresponding point $\rho_{2i} \in \mathbb{P}_2$. This is achieved through a squared distance minimization, written mathematically as:

$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_i ((\boldsymbol{R}\rho_{1i} + \boldsymbol{t}) - \rho_{2i})^2 \tag{6}$$

However, like many optimization methods, ICP does not guarantee convergence to the global minimum. This is due to the fact that the method is heavily reliant on a decent initial placement between the set $\mathbb{P}_1$ and $\mathbb{P}_2$. Specifically, the choice of a closest point ρ_{1i} for each ρ_{2i} determines the likelihood of convergence to the desired overlap. Various extensions [51][52] have been proposed to the ICP algorithm to improve its robustness and efficiency.

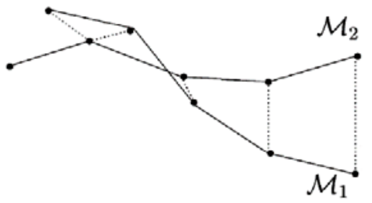


Figure 8. Two point sets with corresponding points in ICP [50]

Algorithm 1. Pseudocode for a standard ICP algorithm

```
load point set $\mathbb{P}_1$ with points $\boldsymbol{\rho_{10}}, \dots \boldsymbol{\rho_{1i}}, \dots \boldsymbol{\rho_{1\mu}}$
load point set $\mathbb{P}_2$ with points $\boldsymbol{\rho_{20}}, \dots \boldsymbol{\rho_{2i}}, \dots \boldsymbol{\rho_{2\mu}}$
repeat
    for $i = 0$ to $\lambda$ do
        find closest point $\rho_{2x} \in \mathbb{P}_2$to $\rho_{1i} \in \mathbb{P}_1$
    end for
    find R and t such that $\min(\sum_i((\boldsymbol{R}\rho_{1i} + \boldsymbol{t}) - \rho_{2i})^2)$
    update $\mathbb{P}_1$ with R and t
until termination condition
```

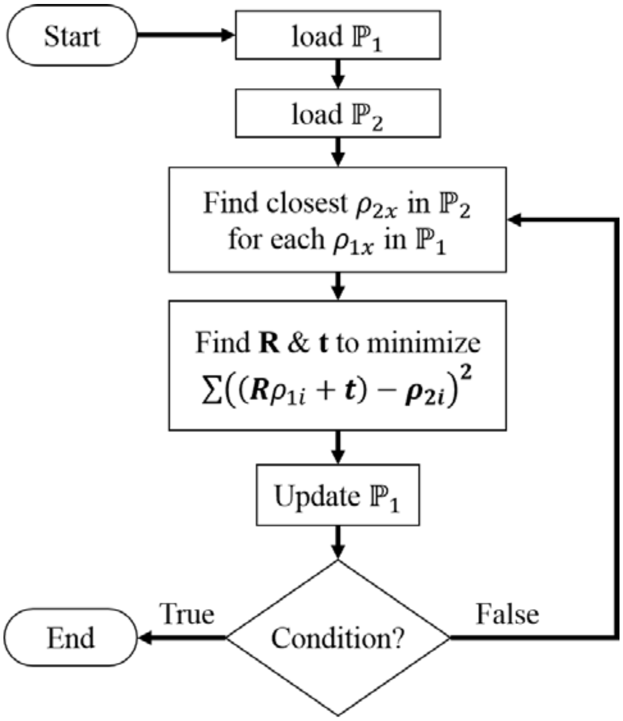

Start
load $\mathbb{P}_1$
load $\mathbb{P}_2$
Find closest ρ_{2x} in $\mathbb{P}_2$ for each ρ_{1x} in $\mathbb{P}_1$
Find **R** & **t** to minimize $\sum((\boldsymbol{R}\rho_{1i} + \boldsymbol{t}) - \boldsymbol{\rho_{2i}})^2$
Update $\mathbb{P}_1$
Condition?
True
False
End

Figure 9. Flowchart for basic ICP algorithm

3.7 Distance Field Transforms

Distance fields within a voxel space are a collection of intensities that increase or decrease as a function of position. Typically, this position is based on a previously defined array of boolean values. For example, a distance field returned by a computer program might reflect the distance of each voxel to a false value contained in an input array. Considering a set $\mathbb{S}$, an unsigned distance function can be defined as:

$$d_{\mathbb{S}}(\rho) = inf\|x - \rho\| \ \forall \ x \in \mathbb{S} \tag{7}$$

Where $d_{\mathbb{S}}(\rho_i)$ is the smallest absolute distance of ρ_i to a point contained in $\mathbb{S}$ [53]. The most direct method of computing a distance field is to apply an unsigned distance function over all voxels in a space. However, this method is impractical due to the high computational cost in iteratively calculating all the distances to $\mathbb{S}$ for all the voxels. Distance transforms operate differently by assigning each pixel/voxel the smallest distance from itself to a subset of pixels/voxels. That is, instead of evaluating all distances in a space, only those distances near a surface boundary condition are calculated and the remaining voxel intensities are estimated by propagating these initial values.

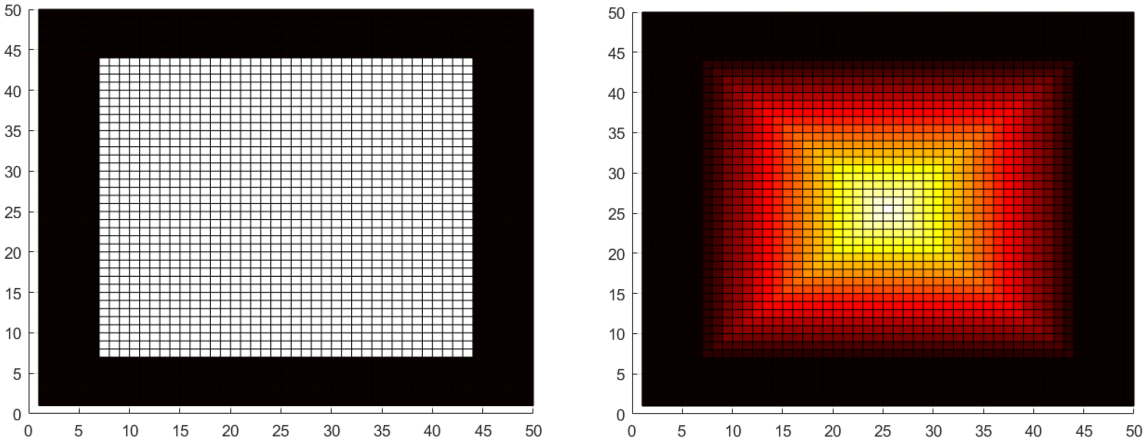


Figure 10. Example of a distance transform over a square in a uniform voxel grid

Various algorithms [53][54] have been developed and are generally classified by the method in which the distance values are estimated from a voxel of known quantity and how these values are used to propagate subsequent values to the remaining voxels [53]. Although different schemes may be chosen to populate the intensities through a distance transform (e.g., Manhattan, Chessboard, etc.) this research uses an exact Euclidean distance transform as provided in MathWorks® Image Processing Library™ [55].

3.8 Genetic Algorithm

The concept of modeling evolutionary processes and applying the mechanics as an optimization technique began with the development of evolutionary computation by computer scientists from 1957 to 1967 [56]. The development of electronic computers allowed for the application of large-scale iterative calculations necessary to simulate and test evolutionary theories [57][58]. However, the application of biological theory to computer science also provided engineers a new, stochastic method for resolving complex engineering problems of multiple parameters [56][59]. Several paradigms have been developed in the field of evolutionary computation which differ in their various representations, methods and operators. *Genetic algorithms* (GAs) are the most popular of these paradigms and are considered in this research [60].

A GA can be categorized as a non-gradient-based, metaheuristic optimization technique. Here, metaheuristic refers to the deliberate introduction of randomness to the search strategy in specific parts of the algorithm. By contrast, a purely heuristic technique would be a random-search procedure [61]. These random elements mimic the stochastic aspects of genetics and mutation observed in evolutionary biology. The goal of a common GA optimization is to determine a set of parameter values that maximize/minimize a complex, multivariate function [56]. Consider such a function, $y = \mathcal{F}(X)$ where $X = \{x_1, x_2, \dots, x_{n-1}, x_n\}$. This function can be optimized by interpreting it as representing a fitness function. There are various ways to define a fitness function but we will assume for this research that the fitness function and the multivariate function to be optimized are the same [62]. The complete range of this fitness function is called the fitness landscape [56]. How low/high the dependent output variable y is understood to reflect how well a handful of chosen input parameters satisfy the function's criteria. From this perspective, there is a clear analogy between the fitness function and a law of natural selection. Therefore, simulating the evolutionary process of a population of candidate solution parameters vying for survival should eventually converge to a set of solutions that optimally satisfy the governing multivariate function. GAs and other evolutionary algorithms make few or no assumptions about the behavior of this fitness landscape. Therefore, the governing fitness function is not required to be continuous differentiable, unlike traditional gradient-based methods of optimization. Instead, the algorithm searches for the global optimum through the population and stochastic elements [63].

Algorithm 2 shows the generalized pseudocode structure for a common GA. The program represents each candidate solution as a binary string chromosome among a population of randomly initialized candidates. Following from the biological analogy, each bit in the string is referred to as a gene and each gene position as a locus. Each iteration, or generation, of the algorithm tracks the performance of the set of chromosomes against the selected fitness function and applies three general operators: selection, crossover and mutation [56]. As seen on Figure 11 and Algorithm 2, the individual chromosomes of each generation are scored and sorted by a selection function which favors higher valued fitness scores.

Algorithm 2. Pseudocode for a standard GA

```
Initialize random population $\mathcal{P}$ of chromosomes $\boldsymbol{X_0}, \dots \boldsymbol{X_i}, \dots \boldsymbol{X_\mu}$
Generate scores for $\boldsymbol{X_i} \in \mathcal{P}$ using fitness function $\mathcal{F}$
repeat
    for $i = 0$ to $\lambda$ do
        Select $\rho$ parents $\in \mathcal{P}$
        Crossover operator $\rightarrow \boldsymbol{X_i'}$
        Mutation operator $\rightarrow \boldsymbol{X_i'}$
        Evaluate $\mathcal{F}(\boldsymbol{X_i'})$ for new scores
        Update $\mathcal{P}$
    end for
until termination condition
```

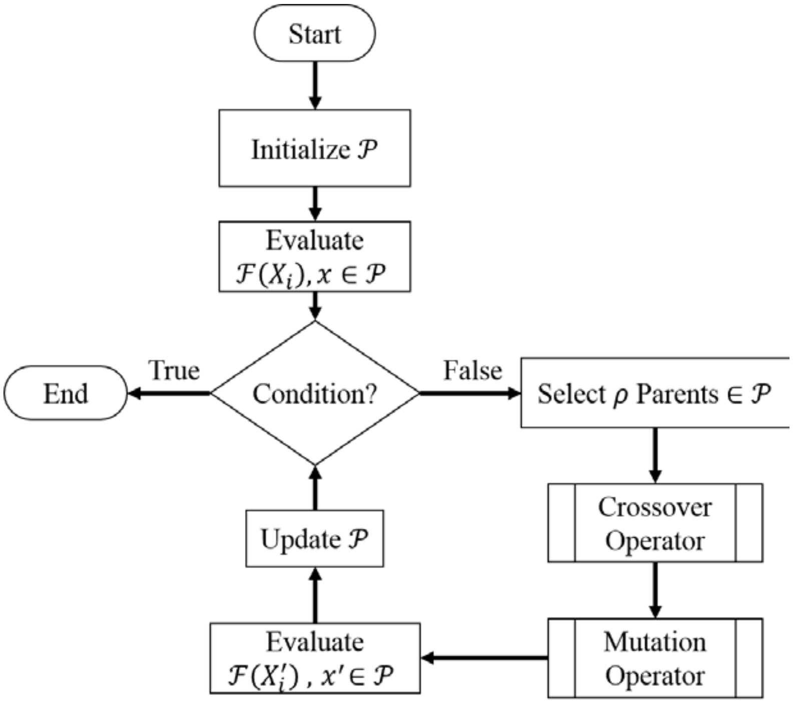


Figure 11. Flowchart for basic GA

Those chromosomes reflecting favorable fitness scores are selected as parents for the next generation. Parents are then further selected to undergo either a crossover or mutation operation. Two chromosome parents are paired for swapping discrete, random set(s) of genes by a crossover operator. After crossover, the two new children chromosomes are passed to the next generation. A chromosome offspring may also be selected for mutation where a randomly selected gene is inverted (bit-flipped). Different schemes have been developed for the crossover and mutation operators but are generally governed by user-defined crossover and mutation rates which reflect the probability involved in determining the locus to swap/mutate. The combined processes of crossover and mutation drives the population toward favorable results but also introduces genetic diversity in successive iterations. Figure 12 shows a general representation of a crossover and mutation sequence on a binary chromosome string. This iterative process of selection, crossover and mutation continues until some terminating criteria selected by the programmer is met.

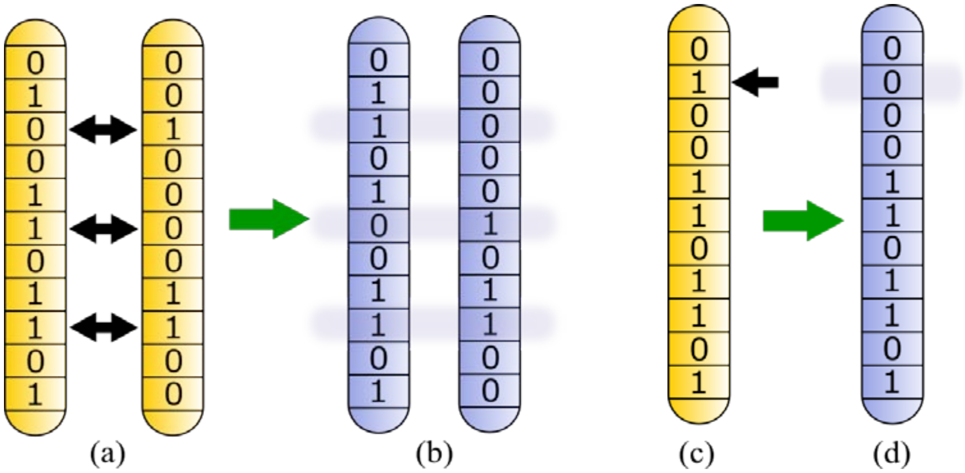


Figure 12. GA chromosome operations crossover (a-b) and mutation (c-d)

3.9 Particle Swarm Optimization

Particle swarm optimization (PSO), like GAs, is a biologically inspired and population-based optimization algorithm. However, PSOs are based on the intelligence of decentralized, organized systems called swarms [64]. Since it does not use crossover or mutation operators it is simpler to implement. Also, PSO techniques have been shown to sometimes outperform GAs when applied to the same problem. This is due to the mechanism that tracks the current global optimum across the population and the fact that the algorithm has fewer parameters for the user to tune [65][66]. The individual agents or particles that comprise the population form multiple, piecewise paths through the fitness landscape by their individual trajectories. Each individual's trajectory consists of a stochastic component and a deterministic or collective component. This causes the particle to experience an attraction toward both the current global best among the entire population $\mathcal{P}$ and the current best encountered along its own path $\check{X}$. At the same time, the stochastic component allows for variation to be introduced into the particle's path [61]. Therefore, these individual, interacting agents optimize the objective function by collectively adapting to the global function landscape.

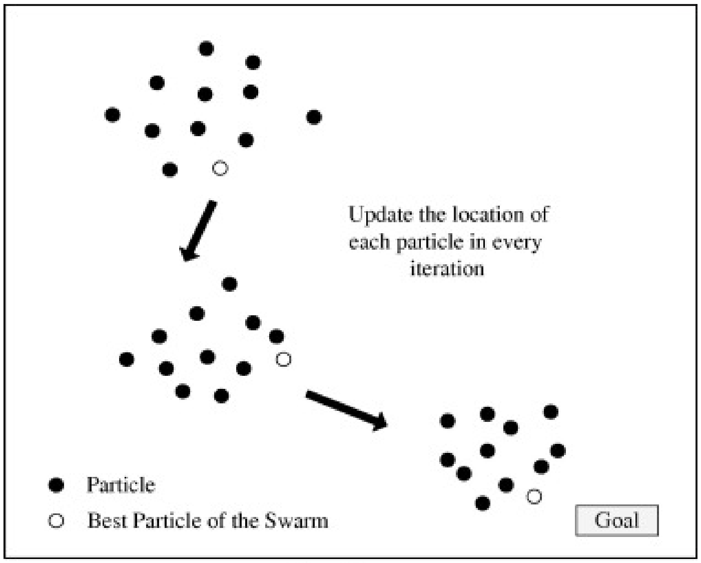
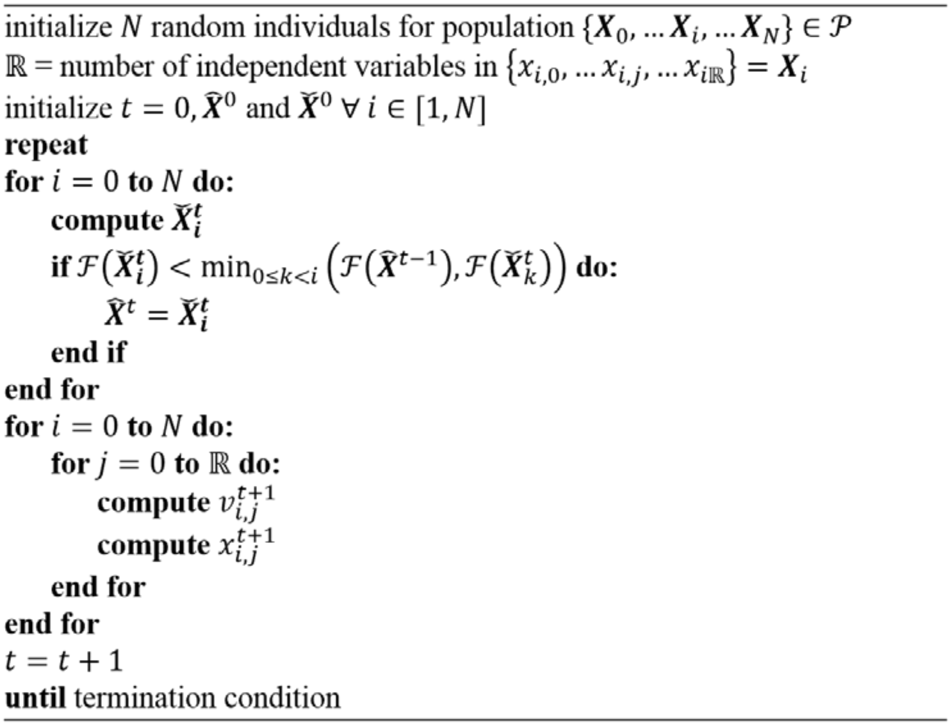

Update the location of each particle in every iteration
● Particle
○ Best Particle of the Swarm
Goal

Figure 13. Illustration of PSO in successive iterations [67]

Through successive iterations, each particle keeps track of the current best solution across the whole population $(\hat{X})$ as well as its own personal best solution $(\check{X})$. These are referred to as social and cognitive components respectively. The velocity and position of each particle in successive iterations is a function of these components. However, the contribution of each term to the iteration is randomly determined [64]. As the particles travel through ℝ dimensional space defined by the rank of the objective function $\mathcal{F}(X)$, they maintain their own position and velocity as well as record the global and personal best positions.

Algorithm 3. Basic PSO pseudocode



```
initialize $N$ random individuals for population $\{\boldsymbol{X}_0, \dots \boldsymbol{X}_i, \dots \boldsymbol{X}_N\} \in \mathcal{P}$
$\mathbb{R}$ = number of independent variables in $\{x_{i,0}, \dots x_{i,j}, \dots x_{i\mathbb{R}}\} = \boldsymbol{X}_i$
initialize $t = 0, \widehat{\boldsymbol{X}}^0$ and $\check{\boldsymbol{X}}^0 \;\forall\; i \in [1, N]$
repeat
for $i = 0$ to $N$ do:
    compute $\check{\boldsymbol{X}}_i^t$
    if $\mathcal{F}\left(\check{\boldsymbol{X}}_i^t\right) < \min_{0 \le k < i}\left(\mathcal{F}\left(\widehat{\boldsymbol{X}}^{t-1}\right), \mathcal{F}\left(\check{\boldsymbol{X}}_k^t\right)\right)$ do:
        $\widehat{\boldsymbol{X}}^t = \check{\boldsymbol{X}}_i^t$
    end if
end for
for $i = 0$ to $N$ do:
    for $j = 0$ to $\mathbb{R}$ do:
        compute $v_{i,j}^{t+1}$
        compute $x_{i,j}^{t+1}$
    end for
end for
$t = t + 1$
until termination condition
```

For a time t, let X_i be the location of a particle in ℝ dimensional space and a member of a population of size N. Mathematically, $X_i = \{x_{i0}, \dots x_{ij}, \dots x_{iN}\} \;\forall\; j \in [1, \mathbb{R}]$. Let v_i be particle i's corresponding velocity. Both the position and velocity can be updated for the subsequent iteration at time $t + 1$ along each dimension $j \in [1, \mathbb{R}]$ from the equations:

$$v_{i,j}^{t+1} = w^t v_{i,j}^t + c_1 r_{1,j}^t (\check{x}_{i,j}^t - x_{i,j}^t) + c_2 r_{2,j}^t (\hat{X}_j^t - x_{i,j}^t) \tag{8}$$

$$x_{i,j}^{t+1} = x_{i,j}^t + v_{i,j}^{t+1} \tag{9}$$

Here, c_1 and c_2 are acceleration constants and r_1 and r_2 are random variables over a uniform distribution [64]. The acceleration constants typically have values $c_1 \approx 0.1 \sim 0.4$ and $c_2 \approx 0.1 \sim 0.7$. The variable w^t is an inertial weight which is a function of the iteration number and used to control the maximum velocity across the population [68]. The first term in Equation 8 is a memory term from the previous iteration. The second and third terms are the cognitive and social terms since they are governed by the personal and collective bests respectively [64]. It should be noted that c_1 can be set to decrease as the algorithm progresses to ignore the stochastic influences in latter iterations [65].

At least 24 different PSO variants exist which employ different approaches and augmentations [61]. The version used in this research and provided by the MathWorks® Optimization Toolbox™ is a classic PSO algorithm [69] with modifications for constrained optimization [70] [71]. Due to the fact that these particles form multiple, interacting Markov chains, analysis on the convergence of PSO algorithms is very difficult or impossible. It follows that the parameters c_1 and c_2 must be fine-tuned for specific applications and function landscapes [65]. In general, as the population size increases, the probability of convergence also increases since only one agent is needed to update the global best $\hat{X}$ [64]. However, convergence is not guaranteed. PSO algorithms have also been demonstrated to be very effective in conjunction with other optimization algorithms to produce hybrid functions [72].

3.10 Simulated Annealing

Simulated annealing (SA) is an optimization technique that mimics the annealing process where a material is heated and slowly cooled. Classic SA is not population-based and starts from an initial solution. This starting point is iteratively modified and updated to a new solution slightly away from the previous solution. However, population-based SA algorithms have been proposed [73][74] to gain a wider scope to search the optimization function landscape.

A conceptual understanding of SA can be gained by considering atoms bound inside a uniform crystalline structure. In the metallurgical annealing process atomic bonds are broken as energy (heat) is increasingly added to the system. This allows the atoms previously bound inside various positions of the crystalline structure to break free and move around. If the metal is cooled, new bonds form to lock the atoms in place. However, if the substance is cooled very slowly, the atoms tend to rearrange into a more regular, uniform distribution throughout the crystal lattice structure [75].

Like PSO, the algorithm works in terms of a Markov chain and accepts changes that are favorable when evaluated by the objective function [76]. However, unfavorable changes in the points are also accepted depending on a probability p. This allows SA to avoid local minima. This probability p may be written based on the Boltzmann distribution:

$$p = exp\left(\frac{-\varDelta E}{k_B T}\right) \tag{10}$$

where T is the temperature, $\varDelta E$ is the change in energy and $\mathrm{k_B}$ is Boltzmann’s constant. If $\varDelta E$ is assumed to be directly proportional to the change in the objective function $\varDelta\mathcal{F}(X)$, the expression can be simplified by assuming $\varDelta\mathcal{F}(X) = \varDelta E$. Equation 10 may be rewritten as in terms of a random variable r:

$$p = exp\left(\frac{-\varDelta E}{T}\right) > r \tag{11}$$

Equation 11 indicates that the algorithm will accept a modification for a candidate solution X^t at time t that is unfavorable when evaluated by $\mathcal{F}(\mathrm{X^t})$ if p is greater than some randomly generated number r [76].

The most important parameter in the SA algorithm is the initial temperature T_o. If T_o is too high, the probability used in evaluating unfavorable changes will also be high. With more and more changes accepted without discrimination, the algorithm approaches the behavior of a random search. Alternatively, if T_o is too low, the probability p will also be low and few unfavorable changes will be accepted. Thus, the algorithm would show more hill-climbing characteristics and be in danger of getting stuck in local minima. This latter case is analogous to quenching a heated material. Figure 14 is a graphical comparison to illustrate how disorder in a viscous material state can be brought into order in a crystalline state through annealing or converge prematurely to an amorphous state.

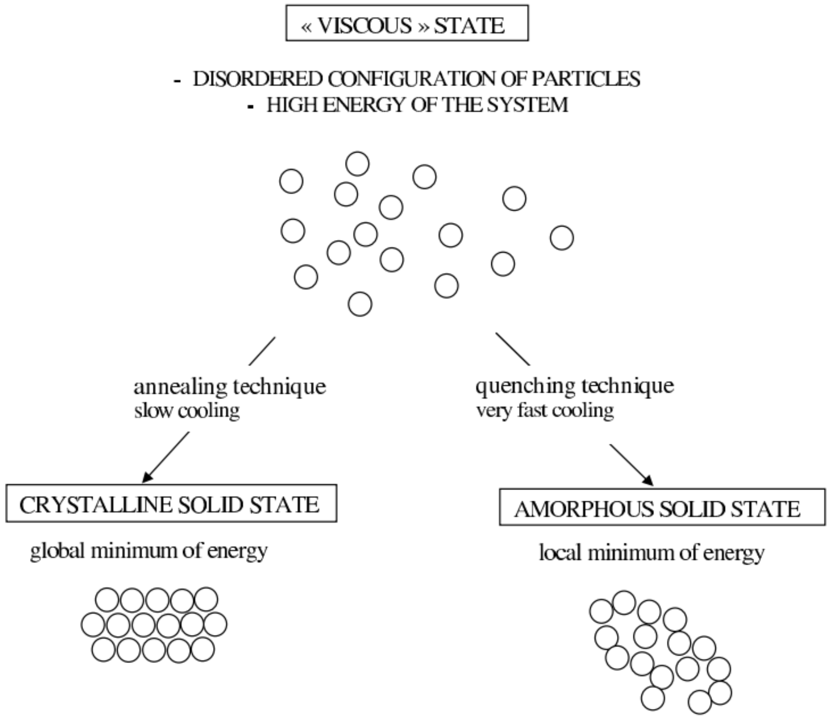


Figure 14. Annealing and quenching processes to reach a lower energy state

Another parameter important to SA is the cooling rate. Equations 12 and 13 are two common cooling distributions that are dependent on user-defined parameters α and β [76].

$$T(t) = T_o - \beta t \tag{12}$$

$$T(t) = T_o \alpha \tag{13}$$

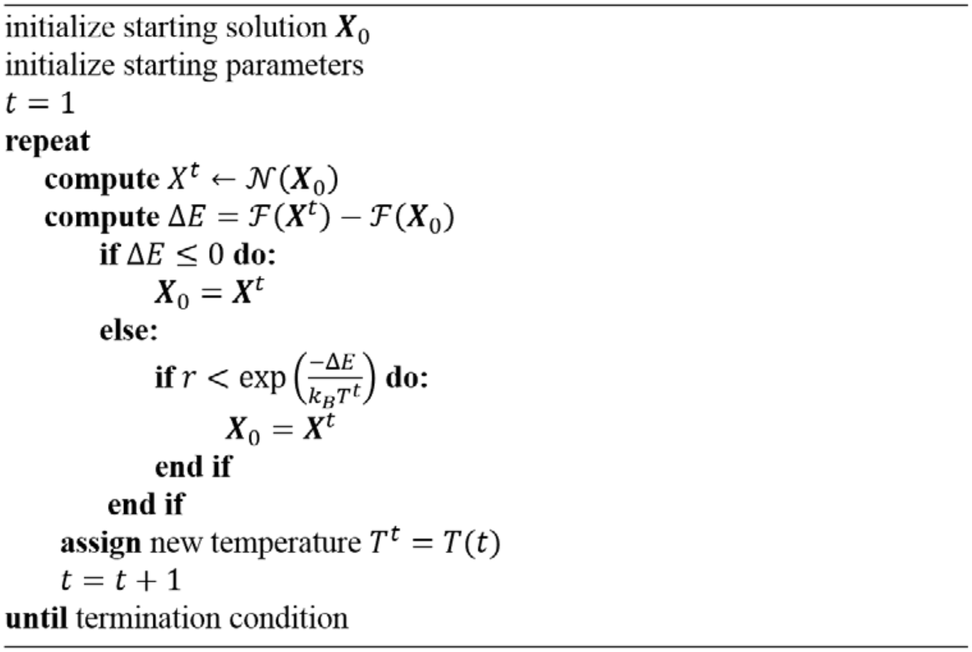
The temperature function selected for this research is in the form:

$$T = T_0 * 0.95^t \tag{14}$$

where t is the current iteration number. The final parametric function of interest is the neighborhood function $\mathcal{N}(X^t)$ which generates a new, modified candidate solution for the $t + 1$ iteration.

Unlike PSO or GA, SA can be proven to eventually converge to a global optimum provided some appropriate conditions are met [72][77]. However, such conditions are governed by the particular optimization problem and the defined parameters so care should be taken in evaluating the result of a SA optimization despite a theoretical guarantee of convergence. SA is easy to implement but has the disadvantage that the programmer must be reasonably experienced to make necessary adjustments in the optimization parameters for particular problem suites [77].

Algorithm 4. Basic pseudocode for a non-population-based SA algorithm



```
initialize starting solution $\boldsymbol{X}_0$
initialize starting parameters
$t = 1$
repeat
    compute $X^t \leftarrow \mathcal{N}(\boldsymbol{X}_0)$
    compute $\Delta E = \mathcal{F}(\boldsymbol{X}^t) - \mathcal{F}(\boldsymbol{X}_0)$
        if $\Delta E \leq 0$ do:
            $\boldsymbol{X}_0 = \boldsymbol{X}^t$
        else:
            if $r < \exp\left(\frac{-\Delta E}{k_B T^t}\right)$ do:
                $\boldsymbol{X}_0 = \boldsymbol{X}^t$
            end if
        end if
    assign new temperature $T^t = T(t)$
    $t = t + 1$
until termination condition
```

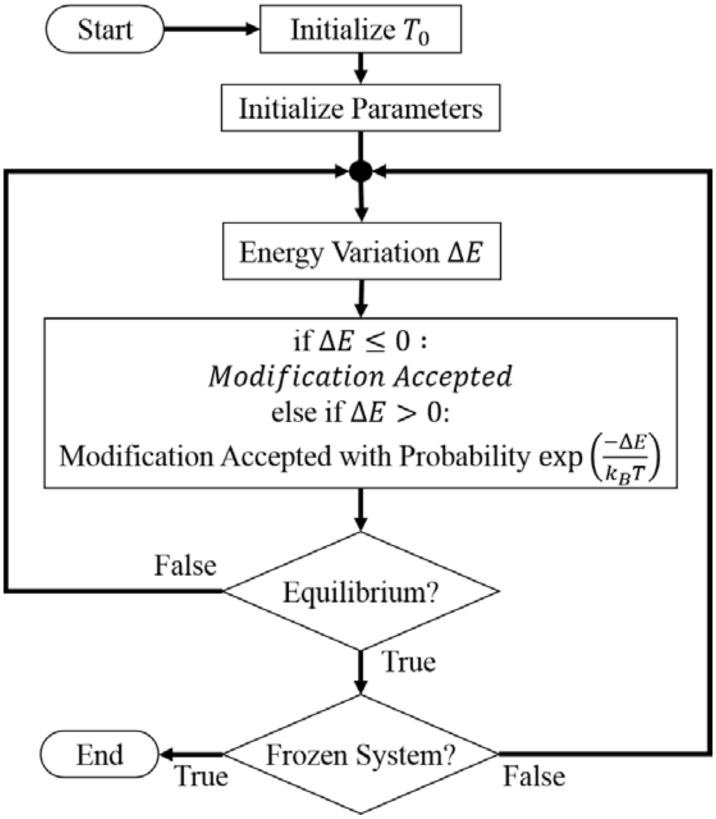


Figure 15. Flowchart for a basic SA algorithm

CHAPTER 4: PROPOSED REGISTRATION TECHNIQUE

The primary objective of this thesis is to suggest a particular use of a distance field similarity metric in image registration. This chapter introduces the proposed method of registering two voxel volumes through minimization of distance field intensity variance. To demonstrate the need for another approach to image registration, the performances of two existing and popular registration techniques are examined. Details of the proposed method are then explained before they are tested in Chapter 5.

4.1 Performance of ICP and MMI

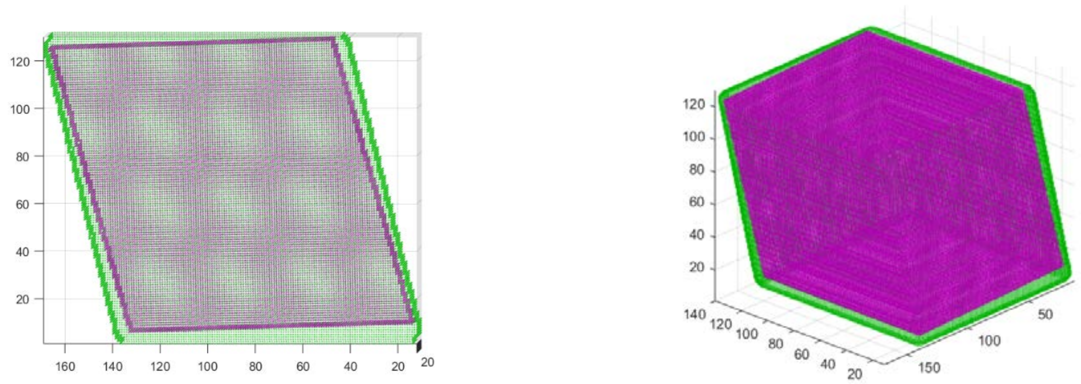
ICP and MMI are widely used methods in image registration. However, both can be shown to have inaccuracies when attempting to match NNS geometries. Software designed for point cloud data acquisition and registration commonly use an ICP algorithm to join partial scans. Similarly, MMI is a very popular multimodal registration technique used to align volumetric images. However, both of these methods hold an underlying assumption that the sets of data being registered describe the same geometry. This assumption can cause problems in attempting to accurately register castings, weldments and NNS geometries.

As covered in section 3.6, ICP determines the transformation of a floating image *F* by minimizing the mean squared error in the Euclidean distance between associated points. This makes sense if *F* is simply a misaligned version of the reference image *R*. If the geometries are the same, we should expect $\left\|\rho_i^R - \rho_i^F\right\| \approx 0$ for corresponding points assuming the error attributed to any measurement device is small. However, this distance will be a finite number in registering rough castings and NNS geometries. That is, $\left\|\rho_i^R - \rho_i^F\right\| = d_i$. In general, it is possible for a transformation to cause enough points in *F* to overlap with *R* that minimize the overall mean squared difference between the images but allow a subset of points to be completely misaligned. This is in contrast to what is desired when aligning parts for programming toolpaths. Material removal should be as consistent as possible along all dimensions.

Figure 16 and Figure 17 show the results of applying ICP to a parallelepiped, impeller and their corresponding offset geometries to simulate a NNS rough casting. An ICP algorithm only considers surface data points so a point cloud was created in the test from a uniform number of voxels defining the outermost layer of the volume. The green point cloud corresponds to the reference geometry with a 1.5mm offset from the purple point cloud's outer surface. All registration tests were conducted with the floating volume *F* translated to be outside *R*.

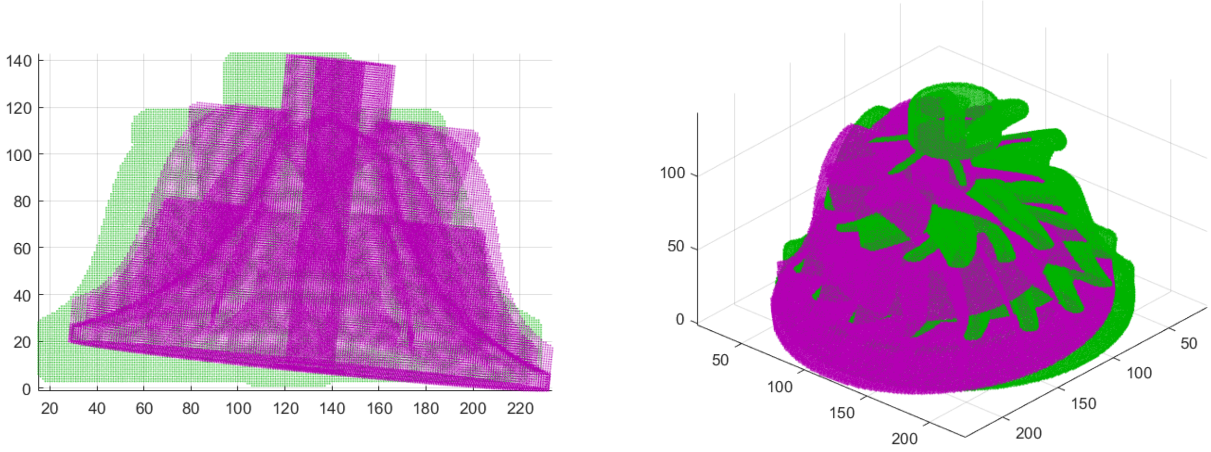
Figure 16 indicates that the final fit of the parallelepiped inside the green point cloud is largely acceptable except when viewed along the X-axis (Figure 16a). As expected, the registration converges to a global minimum by causing some points to be closer to their associated counterpart than others. Despite the inaccuracy caused by the tilt, the part was transformed to be fully inside the green point cloud defining *R*. However, Figure 17 shows a clear failure in applying ICP to the impeller model.

It is reasonable to anticipate failures when applying ICP to NNS registrations. Again, ICP algorithms canonically attempt to drive the error between two images to zero. If the two volumes being registered are not identical, there is no reason to expect that a transformation resulting in a consistent error among all points (i.e., a centrally positioned part inside its NNS) should also be the one that yields minimal error. Instead, ICP will converge to whatever *F* position yields the smallest overall error among points. ICP is therefore definitely not applicable to offset geometries where all points must be guided to remain inside the NNS volume.



(a) (b)

Figure 16. Parallelepiped and NNS point cloud registration results using ICP



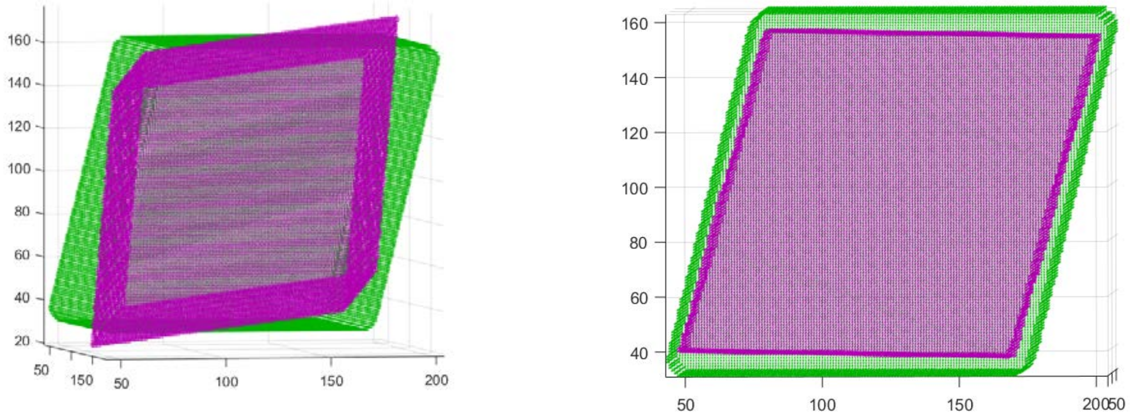
(a) (b)

Figure 17. Impeller and NNS point cloud registration results using ICP

Unlike ICP, MMI takes the pixels throughout the volume into account and does not rely on minimizing distances between surface points. MI-based metrics are popular because they are robust in multimodal registration where voxel intensities between the two images differ in pixel brightness and contrast. However, directly applying a MMI technique to CAD and DT models derived from point cloud data is not straightforward.

The software application used in this research assigns voxel intensities of 0 (for empty space) or 1 (for solid material) by default. Therefore, a standard of assigning values outside of this binary scheme must be chosen. Otherwise, the registration terminates for any orientation where *F* is completely inside *R*. The method proposed in this research is to assign intensities based on a Euclidean distance transform. Voxel intensity values can be derived from a distance field generated for both volumes. However, this results in two slightly different intensity patterns since distance transforms are naturally dependent on the amount of material present.

Figure 18 and Figure 19 show the same registration results of the parallelepiped and impeller but through a MMI registration technique. The intensities are assigned to both volumes through a distance field transform. The figure is shown as if it were a point cloud for clarity. Figure 18a is the registration result which terminated to leave sections of *F* completely lying outside of the green reference volume. Although it might appear that the metric failed, Figure 18b, shows that a local maximum of 0.3344 was missed by the previous registration and corresponds to a proper alignment.



(a) **(b)**

Figure 18. Point cloud representation of parallelepiped registration using MMI (a) Mis-registration MI = 0.2850 (b) Corrected registration MI = 0.3344

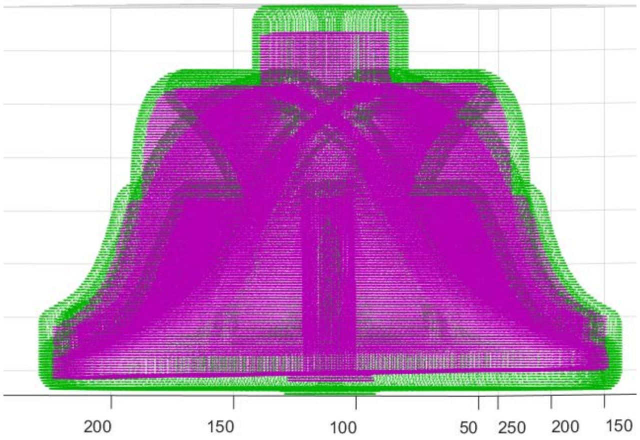


Figure 19. Point cloud representation of impeller registration using MMI

MMI registration appears to perform much better when tested on the impeller geometry. Figure 19 shows that the registration successfully terminated with a MI measure of 0.2297. The registration is very favorable despite the fact that the models being aligned were volumetrically offset by about 1.5mm. This indicates that utilizing a Euclidean distance transform to assign voxel intensities in MMI registration works well for NNS parts. Noise and pixel value differences may exist in multimodal registrations; therefore, this form of MMI should be seriously considered as a viable alternative to certain ICP registrations.

However, MMI is limited to only slightly offset volumes. Figure 20 shows a MMI registration of the impeller to a general stock volume. Clearly, MMI did not successfully align the part model inside the cylindrical stock. This result is reasonable if it is recognized that distance transforms applied to unlike geometries create unexpected minima in the MMI registration landscape.

As the offset reference volume *R* from Figure 20 increases in size, less information is contained in either image to adequately describe the other. MMI produces a clear optimum when the overall geometries being aligned have either matching, distinct features or are similarly shaped. Although MI is valued as a robust similarity metric, because it does not assume the nature of dependence on corresponding voxel intensities [49], it does assume that some relationship exists. The distance transforms applied to images do not constitute a relation between the two parts if the volumes used to generate the intensity values are not similar. A different method of voxel intensity assignment would be required.

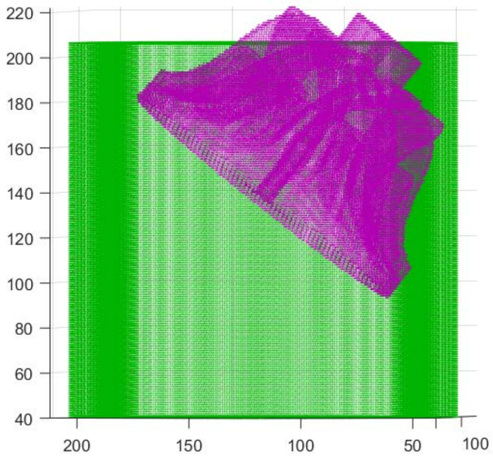


Figure 20. Impeller registered with a non-offset/dissimilar volume

It may be possible to reassign voxel values in *F* and *R* to guide the algorithm to a more central location, but such analytical processing would defeat the need for an automated registration technique. An empirical or *a priori* knowledge of how *F* should fit inside *R* would be necessary to make such adjustments.

Although MMI matches the complex regions of *F* and *R*, it does not incorporate a mechanism to explicitly position *F* with minimal variance of distance to the outer surface of *R* which is necessary for a uniform finishing allowance. Additionally, interpolation artifacts resulting from sub-voxel transformations have the potential to form undesirable local maxima at locations askew from the optimal positioning. This effect is caused by trilinear interpolation created from non-grid alignment orientations which form additional voxels. The increased number of voxels introduces noise or dispersion to the histogram. This translates to an increase in entropy levels. Finally, the computation cost associated with computing the joint histogram is of concern. Despite these shortcomings, the implementation of MMI in computer-aided engineering is an exciting application that deserves closer attention.

This research instead investigates a method that simply and explicitly considers the variance of distance values defined only by the reference volume R. As a result, the technique avoids joint histogram calculations, MMI interpolation artifacts and is slightly more amenable to dissimilar registrations.

4.2 Minimization of Distance Field Intensity Variance

The tests conducted in Section 4.1 show that an alternative registration technique that directly addresses scenarios involving NNS part geometries is needed. This thesis proposes a new method which utilizes only a Euclidean distance transform of the reference volume R to judge the positioning of F. Figure 21 is a flowchart describing the prototype technique.

For a given transformation $T(F)$, those voxel intensities in R that are not overlapped by F are indexed through a material non-implication operation from R to F (i.e., $R \not\supset F$ or $R \nrightarrow F$). The variance calculated over the resulting one dimensional vector of intensities is then used as a similarity metric to drive the optimization algorithm. In other words, those voxels in R that are not overlapped by F are considered. A lower variance among this collection translates to a positioning with consistent distances to the exterior of R in all directions. In other words, there will always be more single-valued voxels on the outermost surface of R being one voxel length minimal distance to empty space as assigned from the Euclidean distance transform.

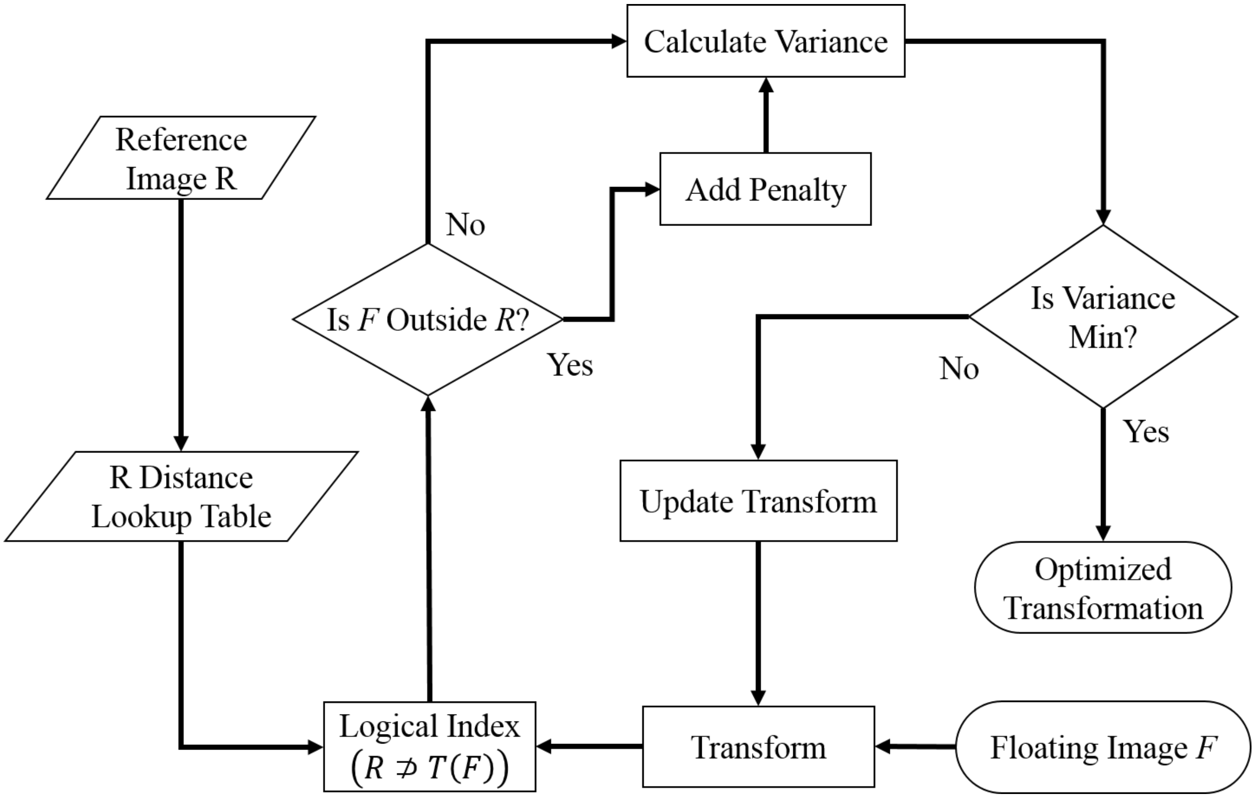
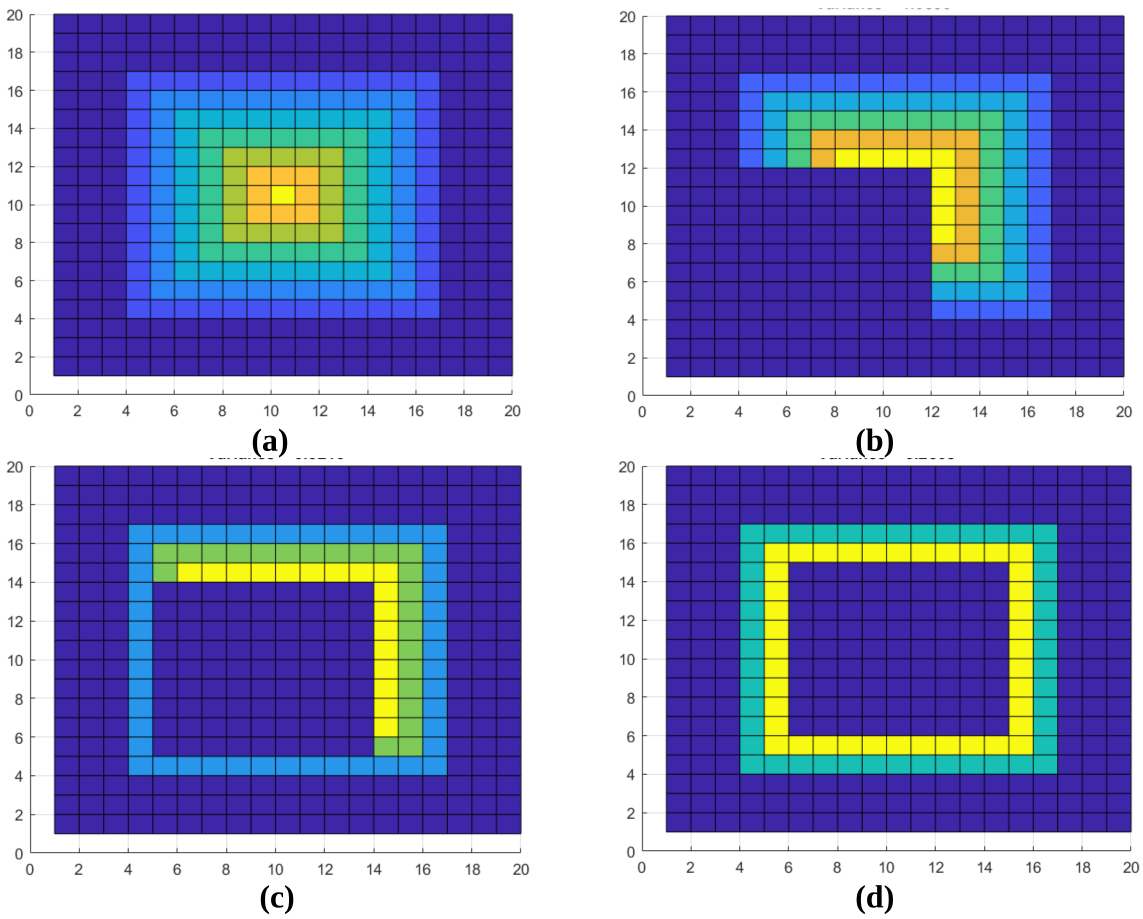


Figure 21. Proposed registration flowchart for minimizing distance field intensity variance (used in uniform voxel grid)

Originally, only those voxels in *R* which were overlapped by the outermost surface of *F* were considered. This gave an intuitive and direct measure of the distance field values as they pertained to the positioning of the target model inside its NNS. However, initial program runs revealed that the final transformations were slightly inaccurate when compared to a material non-implication index. Two primary sources of this error can be identified. First, the variance calculation had a lower sample size when compared to the material non-implication index. As a result, the variance values taken from the positions of *F's* perimeter did not transition as gradually when the volume was iteratively moved to central alignments. This had a more pronounced effect in the relatively coarse resolution uniform grid structure used in the prototype program. Therefore, the optimization had a greater tendency to prematurely terminate at local minima. Second, the number of voxels around the external surface of *F* changed from trilinear interpolation. The values in *R* were determined by first transforming *F* into the space defining *R*. Voxel center points naturally fell between grid points. To change these points, trilinear interpolation was used to assign *F* voxels into the grid of *R*. However, this caused some voxels to be divided between grid points and the increasing total voxel number affected the different variance calculations. Therefore, the choice to use a material non-implication index was made as a practical measure to buffer the fluctuations in variance from operating on a uniform grid. In general, a form of a material non-implication index or direct index method should be usable in a practical, non-uniform grid application of this technique.

The aim of a DT voxel model registration for CLMS is to minimize the variation of the machining allowance (i.e., the distances). This would allow for more consistent feeds and speeds as well as ensuring enough material is present. Since Euclidean distance transforms calculate distances from a known starting location (i.e., the external surface for uniform grid voxel models) any fit that minimizes the variance of certain sampled points is equivalent to minimizing the variance of the distances of the volume *F* to the exterior of *R*. Assuming *R* is larger than *F*, these sampled points can be all those in *R* that are not overlapped by *F*. The location of these points is easily determined through a boolean comparison. Figure 22 shows an example of how a material non-implication index produces a minimal distance field variance when the distance transform is performed from the outer surface of *R*. As progressively more central regions of *R*'s distance field is negated by an overlapping volume, the intensity variance of the non-overlapped area decreases. Note that in Figure 21 there is a penalty component added to the variance calculation in case any portion of *F* is transformed to be entirely outside of *R*.



(a)

(b)

(c)

(d)

Figure 22. Progressive material non-implication operations over a distance field The variance values are: (a) 2.333 (b) 1.689 (c) 0.622 (d) 0.251

While any increasing penalty function may be chosen, the one used in this research takes the form:

$$\eta = x\, ln(x+1) \tag{15}$$

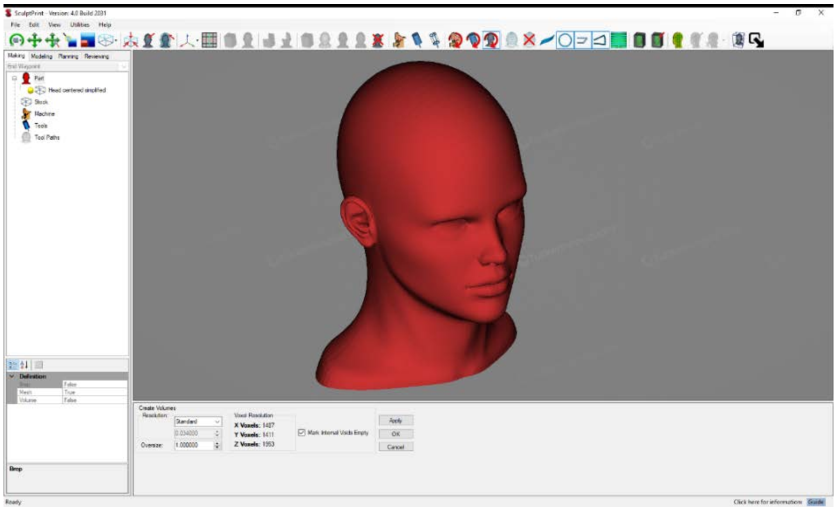
This equation contributes an exponentially increasing penalty as a function of the number of voxels x that lie outside of R. As a result, those candidate solutions encountered through the optimization process which have sections of F outside the reference volume are heavily penalized. These candidate solutions quickly are marked as “poor choices” by the algorithm so regions more likely to yield optima may be searched.

CHAPTER 5: TESTING

This chapter covers the testing and results of registering six voxel part models using a minimized distance field variance technique. An explanation of the general parameters common for all subsequent tests is first presented. An initial set of tests is then reported which compares the “out-of-the-box” performance of different metaheuristic techniques used to drive the registration. An additional set of tests were conducted with a GA optimizer to show the performance in registering a rotated model. Finally, a registration of a dissimilar NNS volume is then conducted to demonstrate the limits and flexibility of the technique.

5.1 Parameters and Equipment Used

All part models used in testing were first voxelized and exported as image stacks from the voxel-based CAM software *SculptPrint*©. Figure 23 shows the user interface in which a *stereolithography* (STL) model is voxelized to a specific resolution. SculptPrint uses a HDT voxel structure to store volumetric data in order to conserve memory and increase efficiently. As a result, it can achieve extraordinarily minute voxel sizes for precision machining while avoiding complex NURBS and B-rep surfaces. Figure 24 shows a comparison between the STL and voxelized models in SculptPrint when a coarse resolution is selected.



Create Volumes
Resolution: Standard
0.034000
Oversize: 1.000000
Voxel Resolution
X Voxels: 1487
Y Voxels: 1411
Z Voxels: 1953
☑ Mark Internal Voids Empty

Brep	False
Mesh	True
Volume	False

Figure 23. User interface for SculptPrint used to voxelize models used in tests

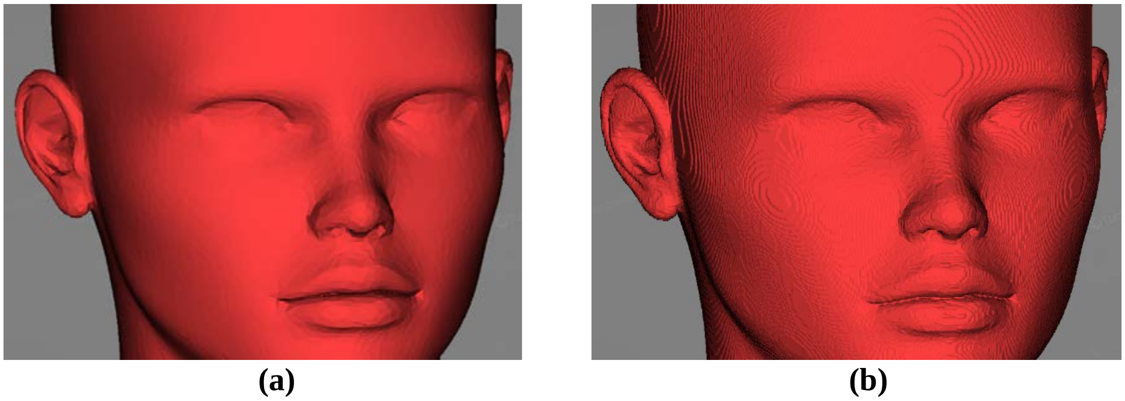


Figure 24. Comparison between the (a) STL and (b) voxelized head model

A uniform grid voxel structure was chosen to demonstrate the registration technique because it was the most direct method to write and test. Since a computer with a powerful processing unit and large memory was available, the general impracticality of large data sets was not as great of a concern. The concepts presented should be equally applicable to professional implementations that use octree or HDT voxel storage schemes. Further considerations of a more in-depth application of the presented technique is covered in the conclusions portion of this thesis. All operations used to produce voxelized models and NNS volumes were performed in SculptPrint using a convolution computation with the offset parameter set to different values.

Due to the practical constraints imposed by using a uniform grid, a voxel resolution of 0.1mm was chosen in SculptPrint for the test models. This size was chosen because it is a common Z-axis resolution in commercial grade fuse deposition modelling 3D printers. Therefore, it is fine enough for basic details but still allows for uniform grid indexing and processing on a powerful computer.

The files used to run optimization and registration were coded in Matlab® R2017a using the MathWorks® Optimization Toolbox™ and Parallel Computing Toolbox™. All tests were run on an Intel® Xeon® 2.40GHz 8-core CPU over a 64bit Windows® operating system. To mitigate the long execution times involved with working with large data sets, tests involving population-based optimizations were performed in parallel on the 8-core CPU. X-Y-Z center point coordinates for the volume *F* were collectively translated and rotated about the volume's centroid and trilinearly interpolated for each iteration. Every new transformation of *F* was used to index the material non-implication voxels from *R* to *F* which were stored in a lookup table. These values were then converted to a vector for a simple variance calculation. A 1×6 vector containing eulerian roatation angles (ZYX) and voxel increment coordinate translations (XYZ) was coded as the input arguments to be optimized.

Figure 25 shows the voxelized parts used in the registration tests. Four simple geometric shapes were selected (a cube, sphere, parallelepiped and torus) as well as two more complicated structures (a bust of a human head and an impeller). Note that the voxel bounding box is slightly larger than the actual image when initially processed from an image stack file source.

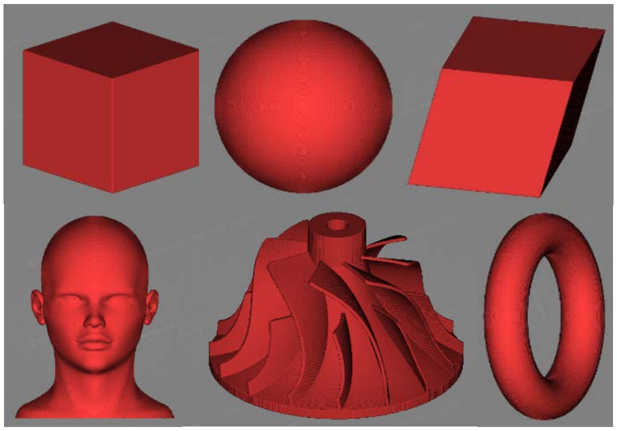


Figure 25. Part models used in voxel volume registration tests
From left to right: cube, head, sphere, impeller, parallelepiped and torus

Table 1. Part list used in registration tests with general measurements

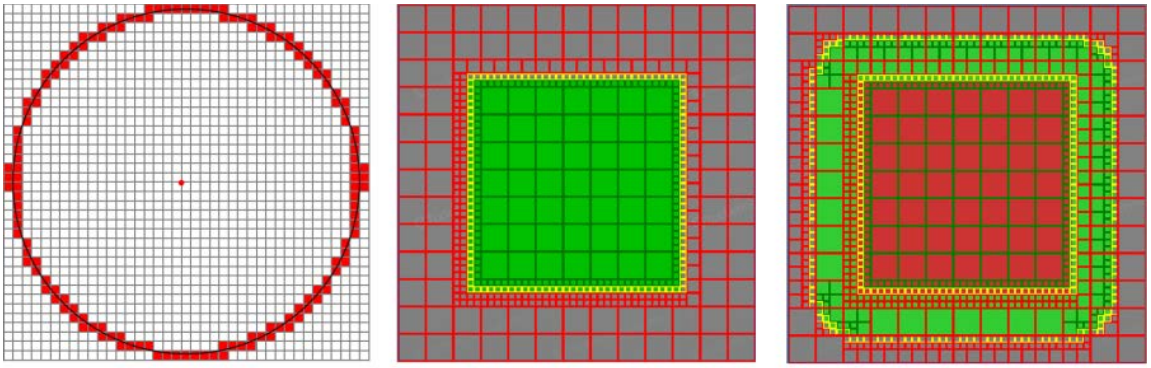
PART NAME	ACTUAL SIZE	RESOLUTION	BOUNDING BOX
Cube	40.0mm × 40.0mm × 40.0mm	0.1mm	512 × 512 × 401
Sphere	40.0mm × 40.0mm × 40.0mm	0.1mm	512 × 512 × 401
Parallelepiped	40.0mm × 50.0 × 40.0mm	0.1mm	576 × 640 × 401
Torus	50.0mm × 50.0 × 10.0mm	0.1mm	704 × 704 × 101
Head	49.2mm × 46.0mm × 64.4mm	0.1mm	640 × 640 × 646
Impeller	68.8mm × 68.8mm × 44.5mm	0.1mm	896 × 896 × 446

As stated previously, rough casting volumes were simulated by a voxel convolution offset performed in SculptPrint with a dilation parameter set to either 1.5mm or 2mm depending on the test. Since the model *F* was used to derive the reference casting *R*, the original, relative positions of the two volumes was be established as the “best” alignment for certain tests in which this alignment could be maintained inside the Matlab development environment. Results from subsequent tests may be compared to this positioning where applicable. Table 2 lists the parameters for each baseline part placement used to judge the quality of registration results.

Table 2. Baseline position metrics by part model

Model	Offset Input	Variance Penalty	Minimum Distance	Maximum Distance	Mean	Standard Deviation
Cube	1.5	19.105	1.25	1.60	1.60	0.207
	2	34.992	1.68	2.20	2.13	0.560
Sphere	1.5	19.860	1.41	1.72	1.63	0.443
	2	34.431	1.95	2.19	2.12	0.291
Parallelepiped	1.5	19.417	1.10	1.70	1.62	0.520
	2	34.253	1.66	2.16	2.11	0.291
Torus	1.5	22.386	1.51	1.93	1.71	1.030
	2	38.185	1.94	2.45	2.21	1.106
Head	1.5	20.012	1.17	3.56	1.62	1.121
	2	35.239	1.63	4.24	2.13	1.473
Impeller	1.5	22.002	1.23	2.65	1.66	1.655
	2	40.054	1.76	3.38	2.20	2.214

Note that the minimum distance for any part is less than the offset parameter of 1.5mm or 2mm. This entry reflects a relatively miniscule number of small voxel lengths between the sharp edges of *F* and the exterior of the surrounding *R* volume. The discrepancy can be traced to a discretization error in the convolution algorithm. Hossain et al. [40] proposed the CUDA accelerated offset algorithm for subtractive 3D printing which is used in this research. Figure 26 shows the 2D representation of how the offset algorithm operates. A circular structuring element is swept across the leaf node voxels in the HDT to trace an offset of a certain radial distance.



(a) 2D Structuring Element

(b) 2D cross-section of HDT pre-sweep

(c) Dilated HDT cross-section after template sweep

Figure 26. Offsetting illustration [40]

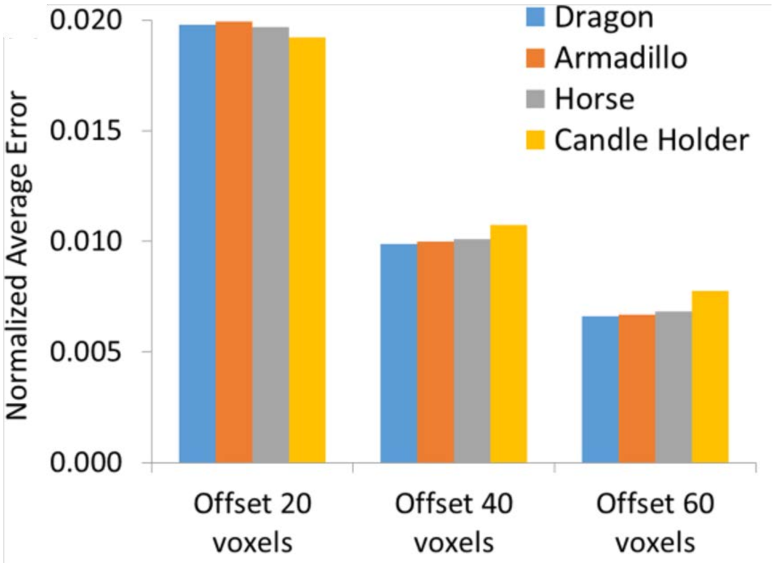


Figure 27. Voxel offset error reporting from research conducted by Hossain et. al. [40]

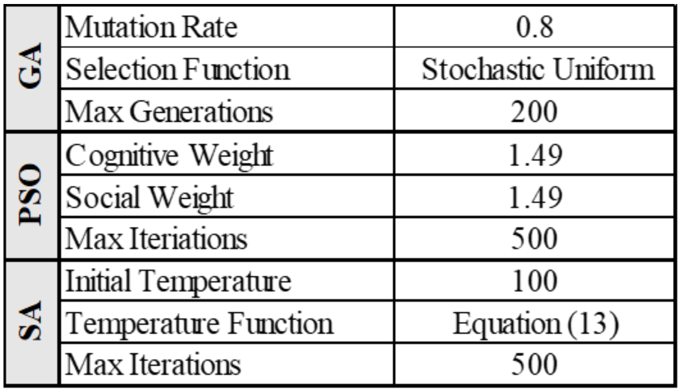
The cited research demonstrated that the technique was very successful but is susceptible to errors when the number of voxels in the offset was low. The minor distance errors contained in this thesis can be attributed to the fact only a 15-20 voxel offset was selected for the low-resolution grid structure. To give a better measure of how well placed *F* is inside *R*, the mean and standard deviation of all minimum distances across the surface of *F* to the exterior of *R* is also reported.

Function tolerances and stall iterations are the primary stopping criteria for all tests. A function tolerance is the specified threshold difference between successive iterations. Every sequential iteration that has a function value difference below this threshold is counted as one stall iteration. For example, if the i^{th} and $(i-1)^{th}$ iterations are evaluated by the function $\mathcal{F}(x)$, and if $\|\mathcal{F}(i) - \mathcal{F}(i-1)\| < \Delta\mathcal{F}_{tolerance}$, then a counter is incremented by one in the program to count the stall iterations. If the next function difference is above the threshold the counter resets to zero. The program terminates once a maximum number of stall iterations has been reached. The maximum number of total iterations is specified as simply a precaution to avoid an infinite loop.

As expected, all registrations in this thesis eventually converged before reaching their assigned maximum iteration number. The number of function evaluations is also reported to give an idea of the computation load required to run each code file to completion. As the name implies, this is the number of individual similarity metric function calls or comparisons that took place for a given registration test.

All Eulerian angles are in units of degrees and all distance measurements are reported in millimeters or voxel units. Sub-voxel transformations and angles are reported to 1/10th accuracy. Different registration program runs were given slightly different optimization settings depending on the initial conditions or focus of the test. Table 3 lists the optimization parameters which are common across all reported tests. Those parameters which were altered, including bounded constraints and optimization settings, are reported in the tabulated results.

Table 3. Optimization parameters common across all registration tests



GA	Mutation Rate	0.8
	Selection Function	Stochastic Uniform
	Max Generations	200
PSO	Cognitive Weight	1.49
	Social Weight	1.49
	Max Iteriations	500
SA	Initial Temperature	100
	Temperature Function	Equation (13)
	Max Iterations	500

5.2 Initial Optimization Tests with GA, PSO and SA

The first series of tests attempted to register the six part models in the test suite to their respective NNS geometries. Figure 28 and Figure 29 are examples of the floating part volumes, NNS volumes and ideal alignments after registration. In order to implement and test the use of a minimal distance field intensity variance similarity metric, three different optimization methods were used for an initial comparison: GA, PSO and SA.

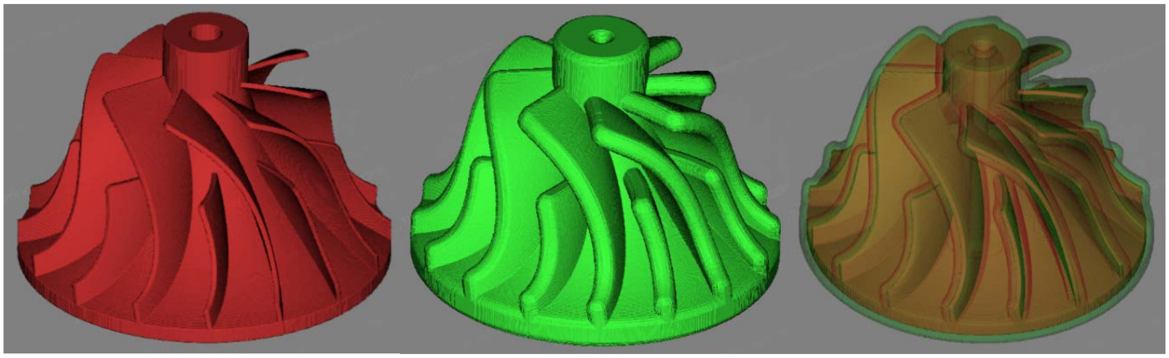


Figure 28. Impeller model with convolution offset
(a) voxelized floating model *F* (b) offset reference model *R* (c) centered *F* inside *R*

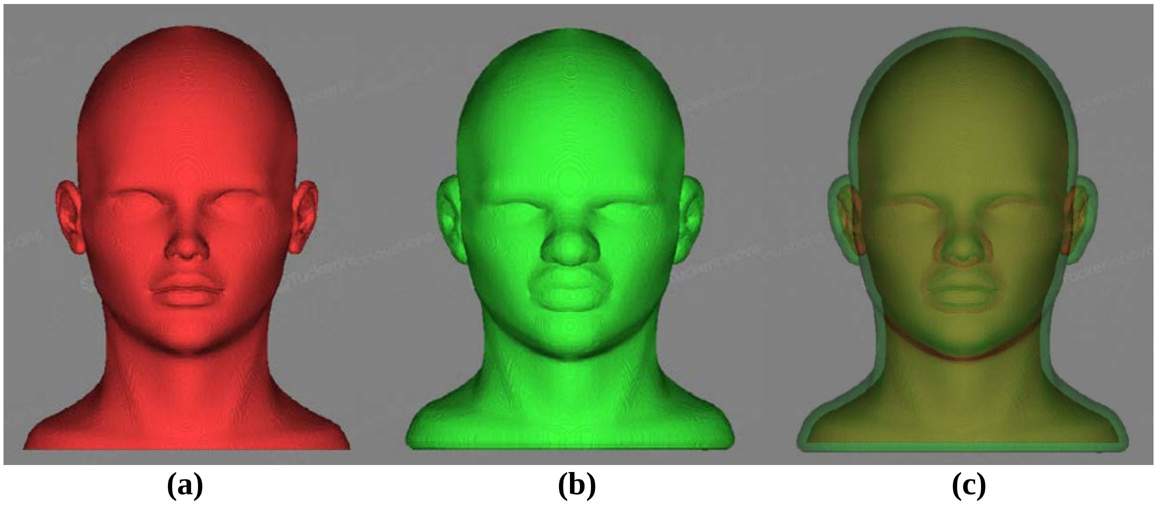
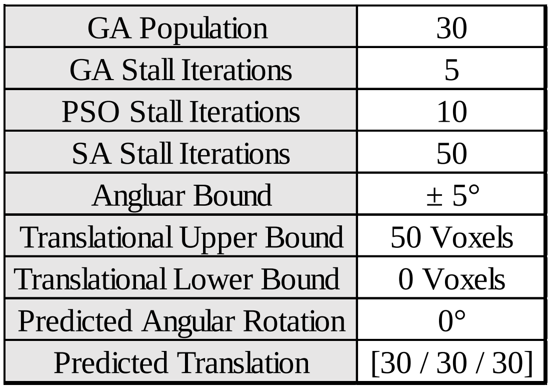


Figure 29. Head model with convolution offset
(a) voxelized floating model *F* (b) offset reference model *R* (c) centered *F* inside *R*

In practice, each metaheuristic optimization method requires trial-and-error tuning of certain performance parameters. The reported settings should not necessarily be taken as general rules of thumb since they are specific to the chosen problem suite. This research presents the results of different optimization techniques from an “out-of-the-box” implementation without a series of rigorous trial-and-error tests. Instead, the goal is to give an idea of how well each method responds with minimal tuning.

The conclusion that there doesn’t exist a “best” set of search parameters for each optimization technique comes from the *No Free Lunch Theorem* [78]. This states that whatever performance one algorithm gains in one class of problems, it is necessarily offset by loss in another. There is no optimal set of optimization parameters independent of a given problem context. Therefore, again, the chosen parameters used in all following tests should be recognized as being empirically chosen through minimal trial-and-error with values similar to those used in select published research [79].

Table 4. Optimization parameters for initial registration tests



GA Population	30
GA Stall Iterations	5
PSO Stall Iterations	10
SA Stall Iterations	50
Angluar Bound	± 5°
Translational Upper Bound	50 Voxels
Translational Lower Bound	0 Voxels
Predicted Angular Rotation	0°
Predicted Translation	[30 / 30 / 30]

The NNS reference volumes were generated by a 1.5mm selected convolution in SculptPrint. Each registration used the same starting offset for the floating volume of -30 voxels translated in each of the X, Y and Z world coordinate directions and with no rotation. The optimization algorithms was also constrained to a +50 voxel translation and ±5° rotation about each axis. Note that some rotation entries are struck through. This is because rotation about this axis is not meaningful due to the part's symmetry. For example, rotations about the Z-axis for the torus model do not contribute to any change in orientation because the part is completely symmetric about this axis. Similarly, rotation results are not geometrically meaningful for a sphere.

Three tests were executed for each part model using some moderate stopping criteria settings. Two subsequent tests were made with this criteria modified for a convergence within a lower tolerance. This was done to observe the increase in both iteration number and quality of result when stall iterations were increased and/or function tolerances were decreased.

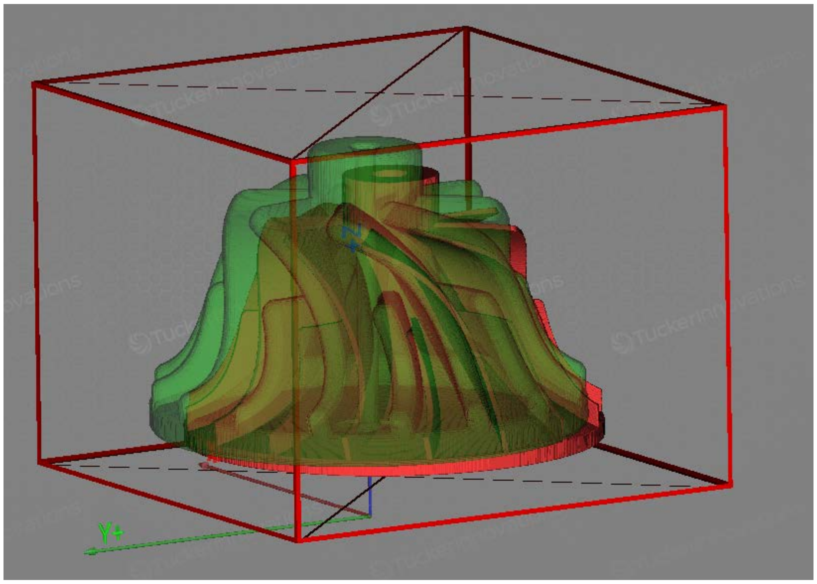


Figure 30. Starting orientation for initial optimization testing

Figure 31 summarize the error results in using a GA, PSO and SA algorithms to drive the registrations. Figure 32 illustrates the percentage of tests that yielded a registration with a minimum 1mm of material covering all to-be-machined surfaces. Additionally, the percentage of tests with a statistical mode of distances in a 0.2mm tolerance from the predicted offset are plotted (i.e., $1.5 \pm 0.2\ mm$). Tables listed in the appendix report the complete results as well as the errors based on the approximated, optimal placement. Again, the placement that was selected as the baseline is only approximate and derived from the creation of the offset R based on F. Errors attributed to using a low voxel number may cause an ideally centered position to be different from this baseline.

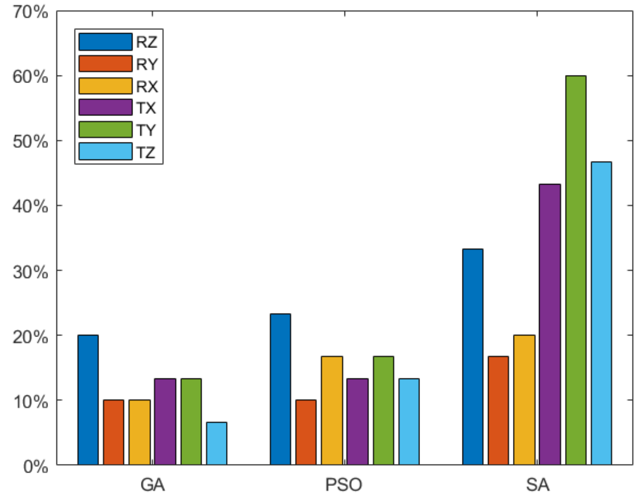


Figure 31. Error in rotational and translational motion across optimizing schemes used

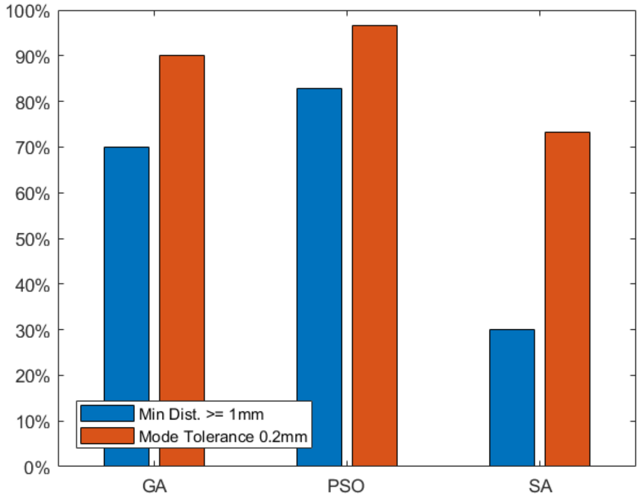


Figure 32. Minimum material distance and occurrence of mode within 0.2mm tolerance

All three optimization tests successfully placed the floating volume completely inside the corresponding NNS with a success rate of at least 97%. The tests that terminated with some portion of *F* outside of *R* are highlighted in the tabulated data. Although the tests using GA gave a 100% successful placement inside the *R* volume, it should be reiterated that this does not mean that implementing GAs will prevent registration failure. As a practical rule of thumb, all stochastic methods should be considered susceptible to failure under the right conditions.

Further examining the test data from the appendix, it can be seen that decreasing the GA function tolerance appears to give clear improvement to only the registration results for the head and impeller models. This is reasonable considering that their more complicated geometries would benefit from finer adjustments occurring in later iterations. By comparison, tightening the tolerances in the PSO tests did not greatly improve the results but did considerably increase the number of function evaluations. Although code performance is not a focus of this research, the increased number of evaluations should be noted as an indicator of possibly greater computation cost associated with the method. Implementing SA as the optimization scheme gave the poorest results across the board. This is not surprising since prior research and theory [77] state that although the algorithm is easy to implement, the actual parameter selection and tuning requires experience to return adequate results. Therefore, it is clear that although SA may be a very beneficial optimization technique, application of this optimization to a registration program should not be made without a commitment to further testing.

It should also be noted that several of the tests minimized the variance below the predicted or starting value through sub-voxel translation (see entries with negative variance penalty error). This emphasizes the fact that the accuracy of a discrete, intensity-based registration method will be directly influenced by the voxel resolution and the means by which intensity values are generated. Since this research uses a Euclidean distance transform, the measurement intensity values, which are not constrained to integer multiples of the voxel resolution, will drive the optimization to some sub-voxel global optimum position.

5.3 Further Testing Using GA

From the results in Section 5.2, a GA appeared to offer the best performance to function evaluation ratio when applied with minimal parameter tuning. Two additional test sets were conducted to incorporate rotational misalignment positions and the use of integer constraints. The results of these simulated registrations present a general description of the proposed technique under different conditions.

First, the similarity metric's performance was examined when the optimal alignment position should ideally terminate at a final rotation and translation. Figure 33 shows the new starting condition for the head model. Unlike the tests in section 5.2, each model was internally modified to have an initial 10° rotation about the X, Y and Z axes in addition to a 2mm translation. The convolution offset parameter used to generate the reference volume *R* was also increased to 2mm.

To misalign the volumes, the models could either be rotated by adding a rotational offset to the models used in section 5.2 or by importing already rotated data in image stack format. Both these methods were considered to gauge the possible effects of errors through volume interpolation. The former method of generating rotated model data was first examined.

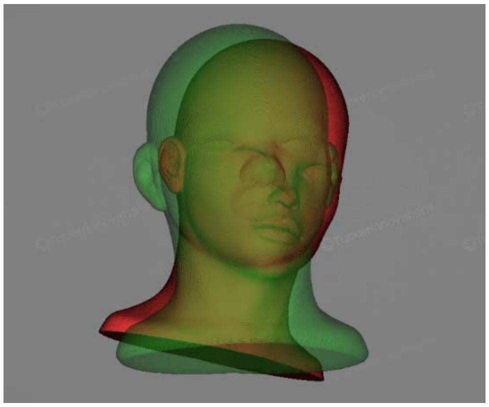


Figure 33. Rotational misalignment representation in SculptPrint

Table 5. Optimization parameters used in first rotational misalignment test

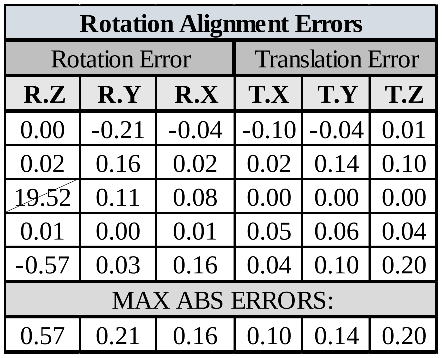
Population	50
Stall Iterations	10
Function Tolerance	1.00E-04
Angluar Bound	± 15°
Translational Upper Bound	40 Voxels
Translational Lower Bound	0 Voxels
Predicted Angular Rotation	[-10° / -10° / -10°]
Predicted Translation	[20 / 20 / 20]

Table 6 and Table 7 list the registration results and errors from registering each model from an incorporated rotational offset and translational misalignment. The sphere was excluded from these tests since rotation would have no effect on the results. As with the previous set of tests, the minimum and maximum material thickness around *F* for the positioning does not exactly reflect the offset parameter input. However, the mode and mean values indicate that the alignment is acceptable in generating a near uniform machining allowance. Overall, the final registration data indicate very favorable alignments across all models.

Table 6. Rotational misalignment registration results using GA

				Rotation			Translation								
	Iterations	**Variance Penalty**	**Variance Penalty Error**	**R.Z**	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**	**Min. Dist**	**Max. Dist**	**Mean**	**Mode**	**SSD**	**Function Evals**
Cube	59	34.302	-0.102	-10.00	-10.21	-10.04	1.90	1.96	2.01	1.60	2.30	2.13	21	0.74	3000
PP	97	33.751	-0.503	-9.98	-9.84	-9.98	2.02	2.14	2.10	1.57	2.33	2.10	21	0.85	4900
Torus	40	38.262	0.105	9.52	-9.89	-9.92	2.00	2.00	2.00	1.85	2.51	2.21	21	1.15	2050
Head	88	34.654	-0.585	-9.99	-10.00	-9.99	2.05	2.06	2.04	1.63	4.21	2.12	21	1.38	4450
Impeller	46	40.086	0.033	-10.57	-9.97	-9.84	2.04	2.10	2.20	1.47	3.35	2.21	23	2.37	2350

Table 7. Error results from rotational misalignment tests using GA



Rotation Alignment Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
0.00	-0.21	-0.04	-0.10	-0.04	0.01
0.02	0.16	0.02	0.02	0.14	0.10
~~19.52~~	0.11	0.08	0.00	0.00	0.00
0.01	0.00	0.01	0.05	0.06	0.04
-0.57	0.03	0.16	0.04	0.10	0.20
MAX ABS ERRORS:					
0.57	0.21	0.16	0.10	0.14	0.20

Since the rotation offset was programmed directly to the models from 5.2, the possibility of errors originating from a rotation that generated new voxels through trilinear interpolation was avoided. A more realistic rotation registration test should consider data of an already misaligned model. In practice, a part may be scanned in any orientation. Therefore, a second means of attaining an initial misalignment was used to confirm the technique's ability to register parts though rotational transformations. The HDT models were first rotated in SculptPrint and then sent as an image stack. Figure 34 compares images from the rotated (34a) and non-rotated (34b) stacks to illustrate the differences in how the volume was reconstructed in the registration program. Again, this was done to give a more realistic data input to the software. The prototype demonstration program used in this research may be prone to interpolation or aliasing errors from trilinear interpolation when attempting to match voxel sets that were generated from different orientations. Consequently, the results of registering data from these rotated image stacks gives an indication of the technique's robustness and flexibility. The parallelepiped and impeller models were chosen to represent a simplistic and complex geometry for the test.

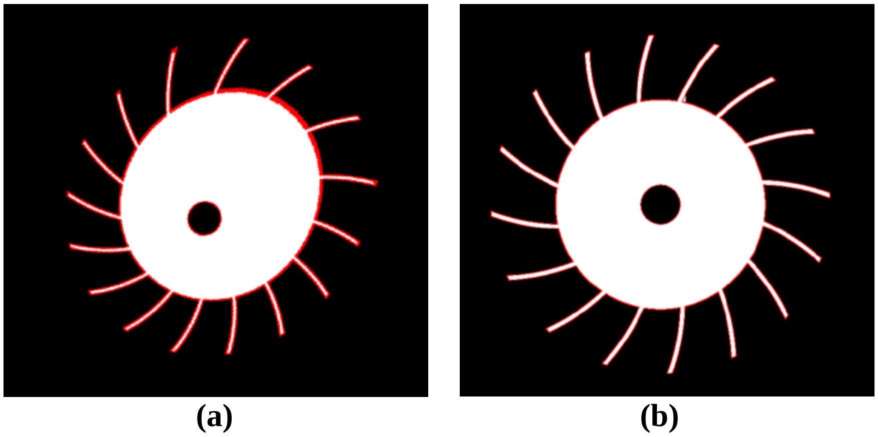


Figure 34. Difference between a rotated (a) and non-rotated (b) image stack file

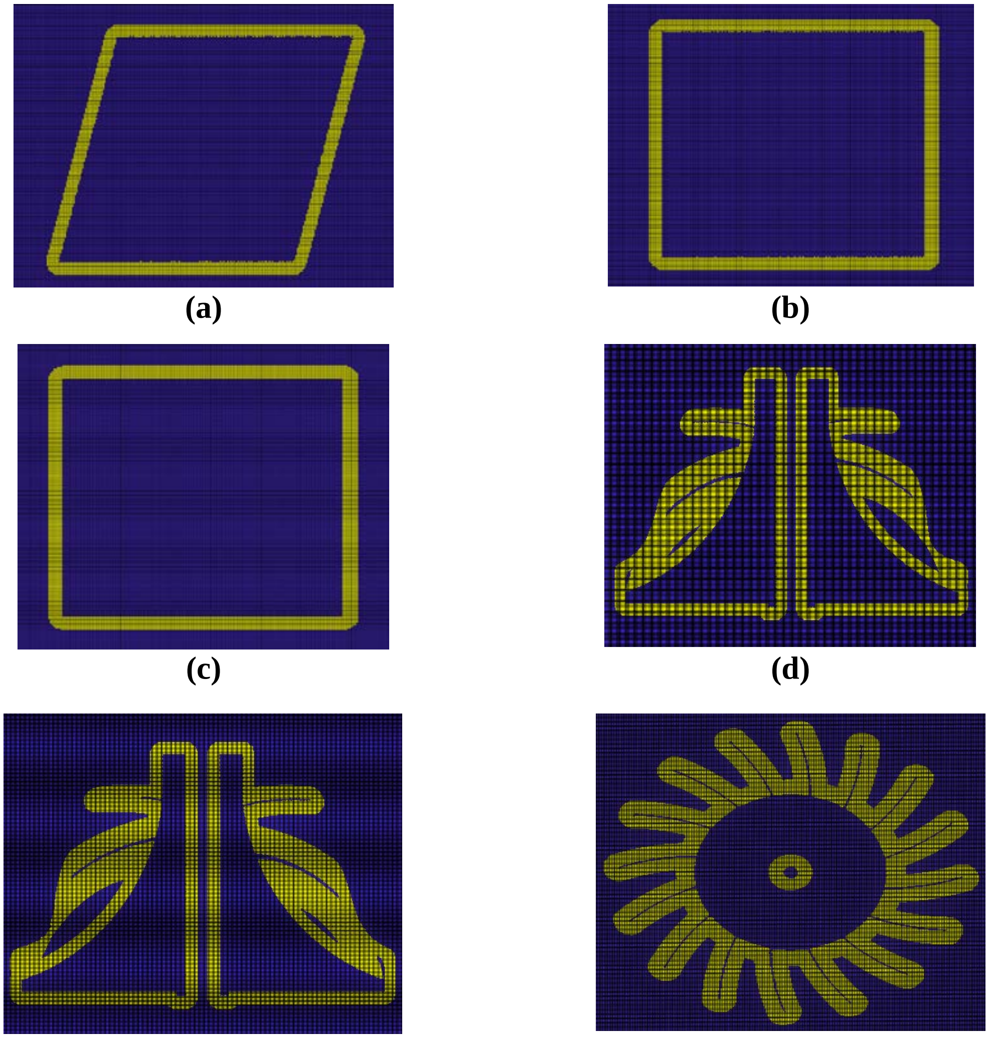
Since the image stacks were generated from different model orientations, it is difficult to approximate or predict the “correct” transformation results to judge the registration as with previous tests. Therefore, the error from a predicted solution will not be considered. Instead, the measurement data (i.e., minimum, mode and standard deviation) of the distance field will be used as the primary means of evaluating success. A visual inspection from figure plots is also used to determine whether the registration was successful. The simulation tests were run twice for a 2mm offset volume and the results are reported in Table 9.

Table 8. Optimization parameters used in second rotational misalignment test

Population	50
Stall Iterations	10
Function Tolerance	1.00E-04
Angluar Bound	± 15°
Translational Upper Bound	0 Voxels
Translational Lower Bound	(-50) Voxels

Table 9. Registration results for rotated image stack alignment

			Rotation [Degrees]			Translation [Voxels]			Distance [mm]					
	Iterations	**Variance Penalty**	**R.Z**	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**	**Min. Dist**	**Max. Dist**	**Mean**	**Mode**	**SSD**	**Function Evals**
Parallelepiped	42	38.893	-8.22	-11.47	-8.23	-43.55	-31.69	-22.73	1.30	2.50	2.22	2.30	1.61	2150
	25	39.238	-8.03	-10.80	-8.18	-42.03	-31.03	-21.38	1.27	2.80	2.22	2.20	1.74	1300
Impeller	79	43.938	-8.50	-11.41	-8.42	-20.07	-36.19	-32.74	1.55	3.49	2.30	2.40	2.13	4000
	43	43.963	-8.61	-11.40	-8.20	-19.46	-36.17	-33.30	1.63	3.55	2.30	2.30	2.16	2200



(a) (b) (c) (d) (e) (f)

Figure 35. Cross-sectional views of boolean subtracted volumes (rotation image stack) (a)(d) X-Z, (b)(e) Y-Z and (c)(f) X-Y planes bisecting the parallelepiped and impeller models

Figure 35 shows the boolean subtraction plots for the X-Z, Y-Z and X-Y cross-sectional planes dividing both volumes. The yellow volume represents the larger model *R* and the subtracted void represents the superimposed *F*. The hollowed appearance of the images in Figure 35 clearly indicate that *F* was centrally superimposed on *R* in both registrations.

A second set of additional tests conducted using GA was aimed at determining the efficiency of incorporating mixed-integer programming constraints to the registration process. MathWorks® Global Optimization Toolbox™ used in this research allows for integer constraints to be applied to specific variables in the GA function. Due to the fact that a minimal voxel resolution is known for both octree and gird structures, it was hypothesized that constraining the transformations of *F* would avoid unnecessary sub-voxel motions and improve overall efficiency by decreasing the number of total function evaluations. A monotonic vector set representing the constrained bounds of rotation/translation was incorporated into the voxel class. The integer constraint in the GA function was then used to index this vector for specific values. This is in contrast to the direct, classical bit-string manipulation of chromosomes where the range or increment is determined by the data type.

Each of the parts was first registered with X, Y and Z translations restricted to integer values for volume offsets of 1.5mm and 2mm. Table 10 summarizes the optimization parameters used in the test which are similar to section 5.2.

Table 10. Optimization parameters for integer constrained translation tests

Population	30
Stall Iterations	10
Angluar Bound	± 20°
Translational Upper Bound	50 Voxels
Translational Lower Bound	0 Voxels
Predicted Angular Rotation	0°
Predicted Translation	[30 / 30 / 30]

Table 18 and Table 19 included in the appendix list the results and approximated errors from implementing the integer constrained translation optimization. The most obvious difference in these test results and those from section 5.2 is that the rotational errors are slightly larger. This is reasonable considering that the predicted "correct" solution is an approximation-based on the relative positioning from offsetting F to generate R. Recall that several tests found a variance below the predicated value through sub-voxel translations. Since these sub-voxel translations were constrained, it is understandable that a non-integer constrained rotation resulted in a new global optimum slightly off-axis from the predicted result. Most importantly, the total number of function evaluations performed by the registration program before termination dropped as predicted.

Next, an experiment was run where the rotations were also constrained to $1°$ increments. Instead of testing all the voxel models again, the head and impeller were singled out as the two most complicated models to be used in the test. Table 11 shows that discretizing the rotations to $1°$ increments eliminated the slight angular error from the previous test. Further, the number of function evaluations reported is practically half the value from tests with similar optimization tolerances. It should be noted that although discretizing rotations has been shown here to decrease the number of function evaluations, and consequently shorten the time for the optimization to reach a terminating criteria, a smaller increment should be used in more serious applications. Unlike constraining translations to the length of the voxel resolution, choice of an appropriate discretized rotation amount is not clear. Since too large of an increment can cause the registration to completely miss the desired global minimum, it would be safest to maintain a discrete step size between $0.5° - 0.1°$.

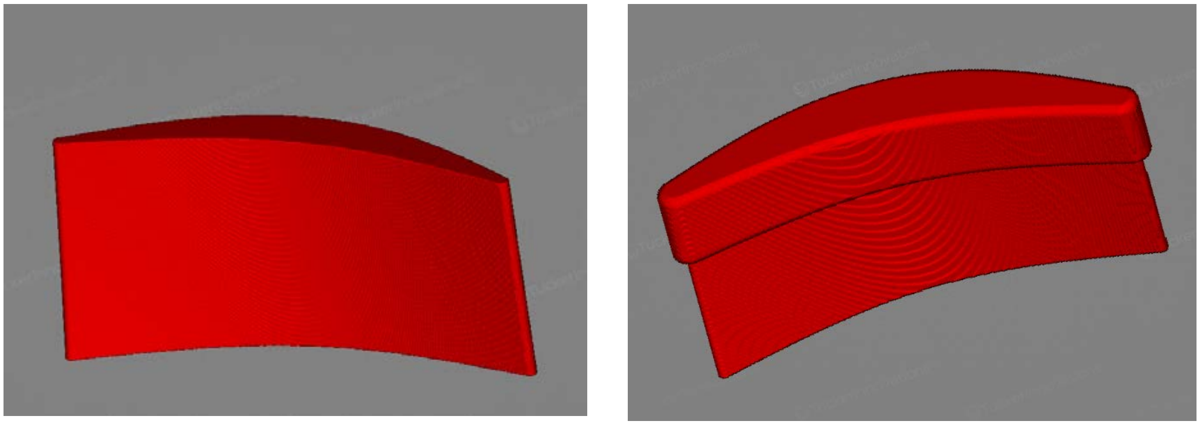
Table 11. Registration results using incremental integer sets for rotation and translation

				Rotation [Degrees]			Translation [Voxels]			Distance [mm]					
	Iterations	Variance Penalty	Variance Penalty Error	R.Z	R.Y	R.X	T.X	T.Y	T.Z	Min. Dist	Max. Dist	Mean	Mode	SSD	Function Evals
Head	39	20.148	0.137	0	0	0	30	31	30	1.14	3.53	1.62	1.60	1.25	1200
	40	20.012	0	0	0	0	30	30	30	1.17	3.56	1.62	1.60	1.12	1230
Impeller	30	21.939	-0.063	0	0	0	30	30	31	1.15	2.60	1.66	1.70	1.64	930
	40	21.939	-0.063	0	0	0	30	30	31	1.53	2.60	1.66	1.70	1.64	1230

5.4 Robustness and Limitations

The previous tests investigated the registration of voxel models with their respective NNS volumes to simulate rough castings. However, to present a more complete picture of the proposed technique, conditions where the registration fails or is applied to non-NNS geometries should be examined. This section presents several test results where these cases are addressed. Since a predicted “correct” answer to the registration is either not known or difficult to determine and failed alignments are readily identifiable, this section relies on a visual examination of the final transformation plots to judge the success of the program. As a consequence of this choice, the voxel resolution was increased from the previously used 0.1mm value to approximately 0.3mm. This was done to increase program efficiency and simply demonstrate the end behavior of the technique since the previously reported tests established the accuracy of the results.

First, a registration was attempted to align a turbine blade with a slightly modified volume of the same model. Specifically, the same model with additional material added to its perimeter was used as the reference volume *R* instead of a NNS offset volume. Figure 36 shows the turbine blade model (36a) and the same model with an additional weld lip added to the top rim (36b). This simulation test has direct applications to hybrid machining and remanufacturing processes. In one possible scenario, a damage blade could be repaired by fusing additional, arc welded material around the perimeter. In the context of CLMS, the registration algorithm can process the DT models of the blade to align the target part geometry so that the additional volume can be identified as excess material to be machined away. However, this is unlike registering parts to a NNS. Instead, no additional volume is present around certain areas of the part. . Figure 36a is be the floating, final model and Figure 36b is be the volume as it would be scanned and read as DT.



(a) **(b)**

Figure 36. Turbine blade models (a) nominal and (b) with weld

The results of a registration with the same optimization parameters used in 5.2 is portrayed in Figure 37 as a set of boolean subtracted volumetric image slices. The absence of the yellow volume in comparing the subplots (a) and (d) indicate that the blade model *F* was aligned to superimpose the regions in *R* where additional material was not added. In other words, the common regions between the volumes were correctly matched. This result appears to show that the technique can even be used in limited situations where NNS volumes are not being registered. However, we can analytically see that the reason for this particular successful registration was that there were very few orientations where *F* could be entirely inside *R*.

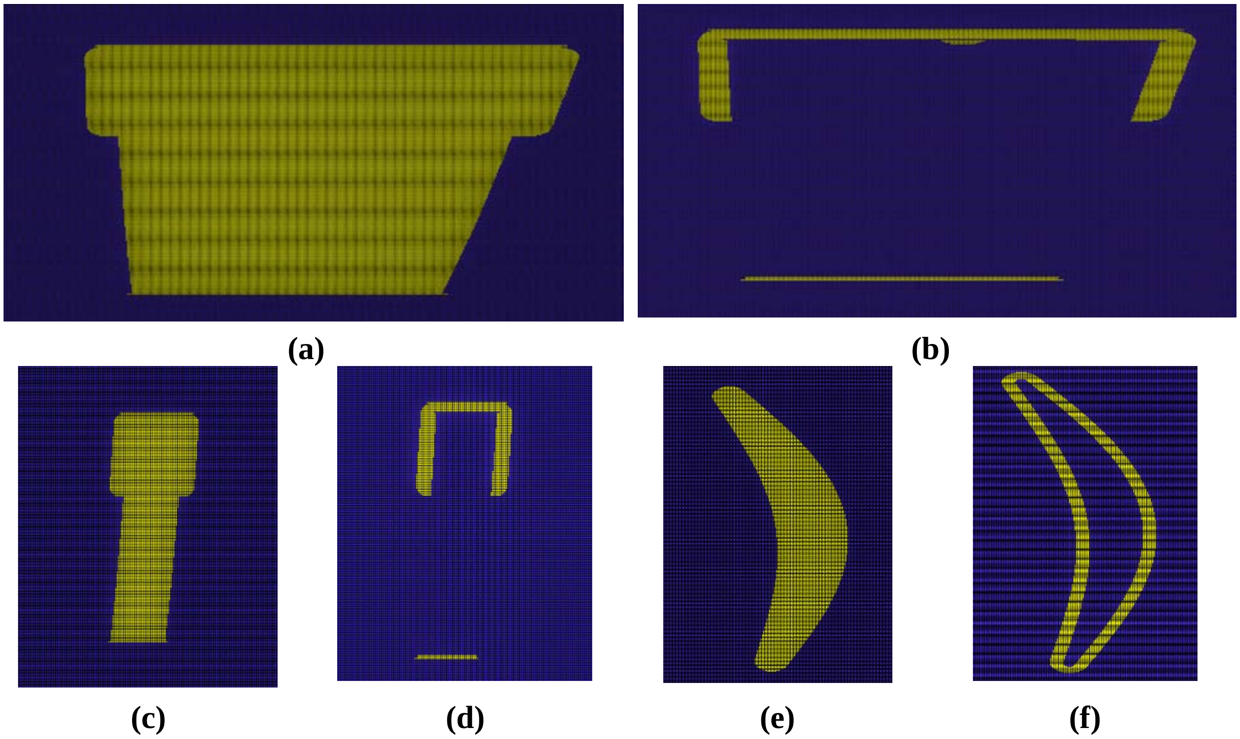


Figure 37. Turbine blade boolean subtracted volumetric image slices

Therefore, this example should be considered a special case given that so few orientations were available for the turbine blade to assume while still remaining inside its corresponding reference volume. Also note that in Figure 37b and Figure 37d the lower surface is still visible. This offset is not desirable for situations where only the additional material added to the target model *F* needs to be identified for machining.

Figure 38 shows two dissimilar parts that give a better example of a scanned volume and target model in the context of hybrid machining. We can consider Figure 38b to represent a scanned volume after material has been added to a part undergoing a hybrid machining operation. Again, Figure 38a would be the final target geometry in this scenario. Unlike the parts previously described, more material has been added to the upper portion of the rod and only a small section of the parts are nearly identical or unmodified.

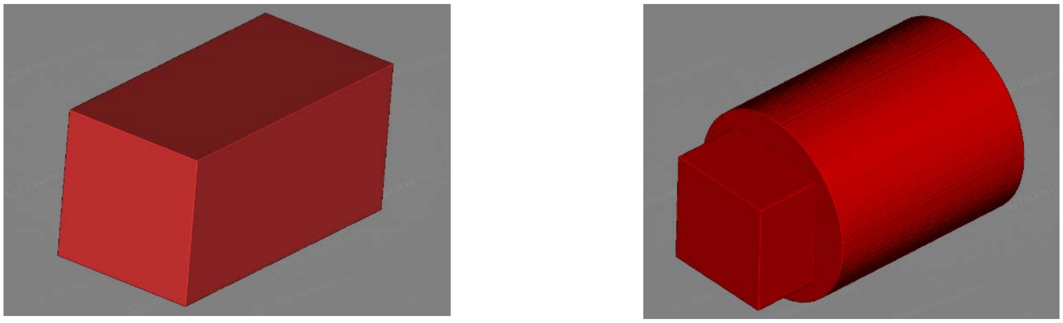

(a) (b)

Figure 38. Additional modified volume example for more realistic hybrid registration

The previous test indicated that the technique of minimizing voxel intensity variances from a material non-implication index was robust enough to still register slightly modified parts with additional volume present. However, Figure 39 shows the same registration results when the parts from Figure 38 are used.

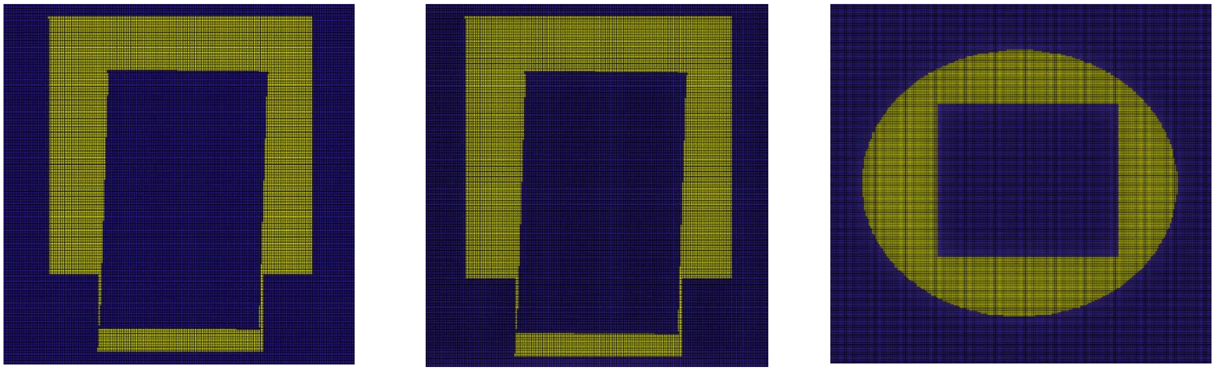

(a) (b) (c)

Figure 39. Boolean subtraction plots for alternative modified structure

This registration failed to match the near identical regions of the parts despite successfully terminating the optimization to maintain all portions of *F* inside of *R*. The resulting vertical offset and tilt in Figure 39a and Figure 39b can be attributed to similarity metric being designed to minimize the variance of non-overlapped voxel values. Considering this further, we can qualitatively reason that such an alignment taken in Figure 39 possesses a smaller variance than one with the bottom edges directly overlapping. All of the outer surface voxel intensities in *R* are valued 1 since they are one voxel length away from empty space as assigned by the Euclidean distance transform. Therefore, overlapping and thus negating a large amount of single-valued voxels located on the surface of *R* would result in a higher variance value than negating the higher valued, central ones. MMI would be a better metric to use in such cases where regions of identical structure are present between the target and real-time volumes of the DTs. This is discussed further in the concluding section of this thesis.

The second registration scenario examined was selected to demonstrate the limitations of the presented method when it is applied to larger dissimilar volumes. Figure 38 shows the two models selected for a test using dissimilar structures. Again, the conical starting stock is not a generated offset volume of the impeller model. Ideally, the impeller should be centrally aligned inside the larger material. However, the intensity measures across the green conical volume *R* from the distance transform will not necessarily cause the material non-implication index to guide *F* to an on-axis, central region. If the impeller was a scaled version of the conic volume, this might be the case. However, the outer blade profiles are not similar to the conical pitch. Moreover, the hole in the impeller model would necessarily not overlap the most high intensity weighted voxels in *R* if the model was positioned centrally along the axis of rotation of the starting stock.

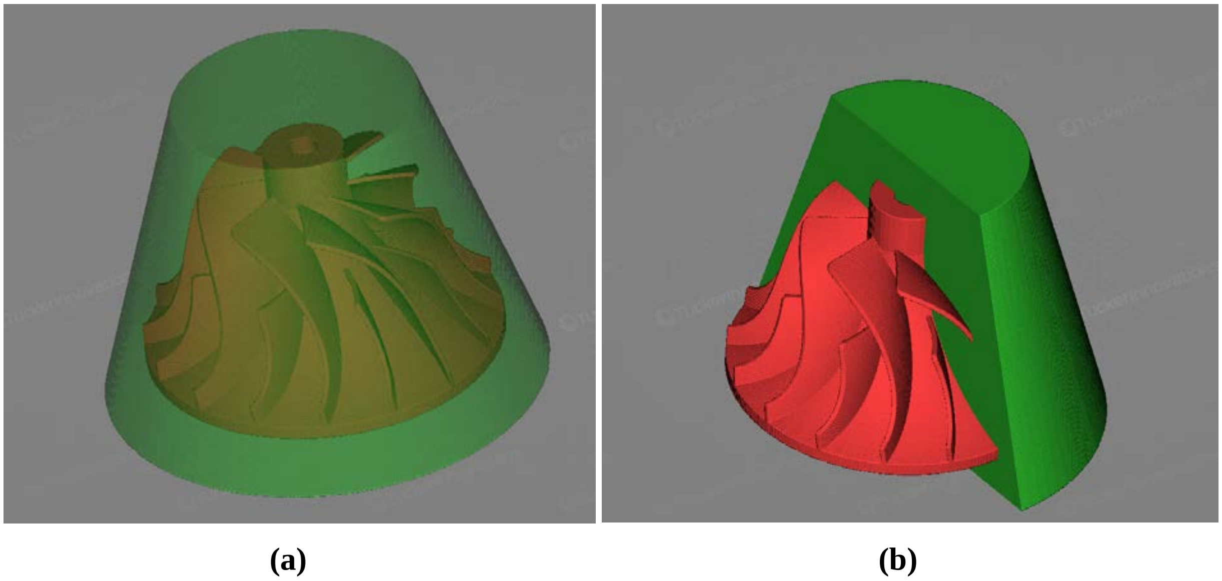


Figure 40. Impeller and conic stock part models used in dissimilar registration (a) transparent view (b) sectional view

Recall that a material non-implication operation was selected to calculate the distance field variance for uniform grid voxel structures. This means that those voxels in *R* that are not overlapped by *F* are considered through a variance calculation. Also, recall that the distance field generated across *R* results in voxels with the highest intensity at the most central locations in the volume. Since the impeller model incorporates a hole along its central axis, there is a conflict between the desired central alignment and the material non-implication operation. In other words, if *F* is placed centrally inside the conical *R*, the hole causes the volume of *F* to not overlap the highest weighted voxels of *R*. This has the potential to increase the subsequent variance calculation and drive the registration to an off-center alignment.

Figure 39 shows a set of volumetric boolean subtraction plots similar to Figure 35 and Figure 37 taken from the registration of the conical stock and impeller model. As predicted, the central, axial hole in the impeller drives the final alignment to the edges of the reference volume. A slight tilt about the X-axis is also observable.

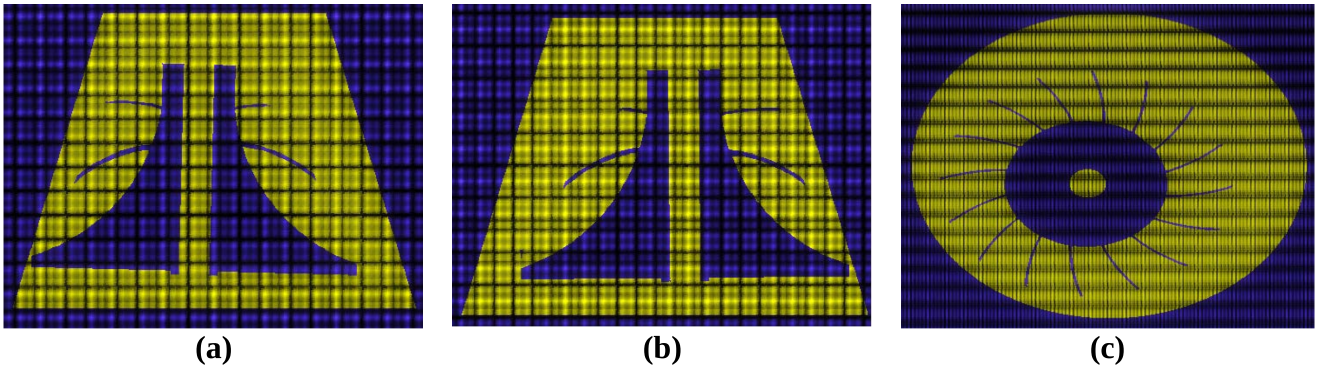


Figure 41. Boolean subtracted volumetric image slices for impeller-conic registration

The simplest solution is to eliminate (fill) the hole so that central alignments will cover, and thus negate, the central voxels in *R*. Although this sounds like a trivial operation, the task of altering geometries can be very challenging for NURBS and B-rep models. However, since voxel models incorporate independent, discrete volumetric elements, adding and subtracting volumes is simple. The hole was identified and filled by indexing 2D slices of the model along its Z-axis. It should be noted that although this modification changes the volume used in the registration, keeping track and removing these alterations after a registration to return the impeller to its original state is also quite simple. After the proper alignment has been identified, voxels previously added can be subtracted through a fast boolean operation. A second registration with the filled-hole impeller model was then performed. Figure 40 shows the new results after performing the volume modification. Note that filling in the axial hole did not drive the registration to converge to a more centrally aligned position. Instead, the optimization appears to have terminated at a slightly worse position. This fact signifies that the modified impeller is still too dissimilar to *R*. Because of this, a global minimum still doesn't exist near the central regions of the conical stock.

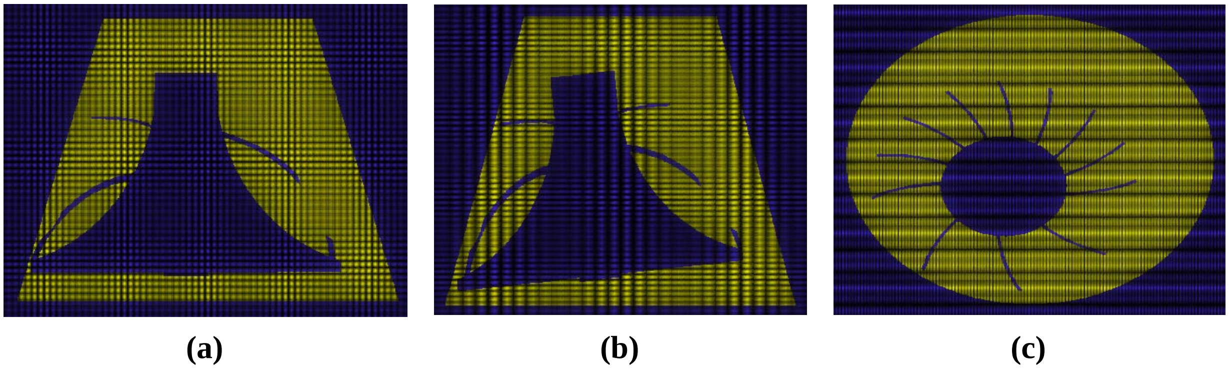


Figure 42. Boolean subtracted volumetric image slices for hole-filled registration

A final volume modification was then performed. This operation eliminated the blades by extending additional material from the central axis to the outer bounds of the part for each 2D slice along the Z-axis of symmetry. The resulting volume is shown in Figure 41. Figure 42 shows the results for the final registration using the further modified impeller model.

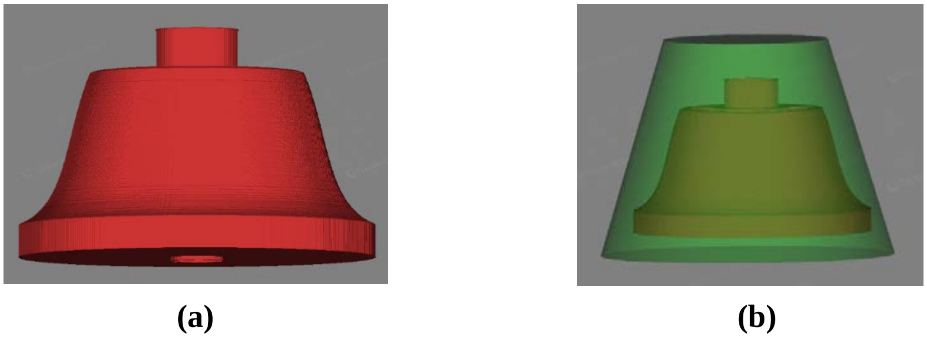


Figure 43. Further modified impeller volume

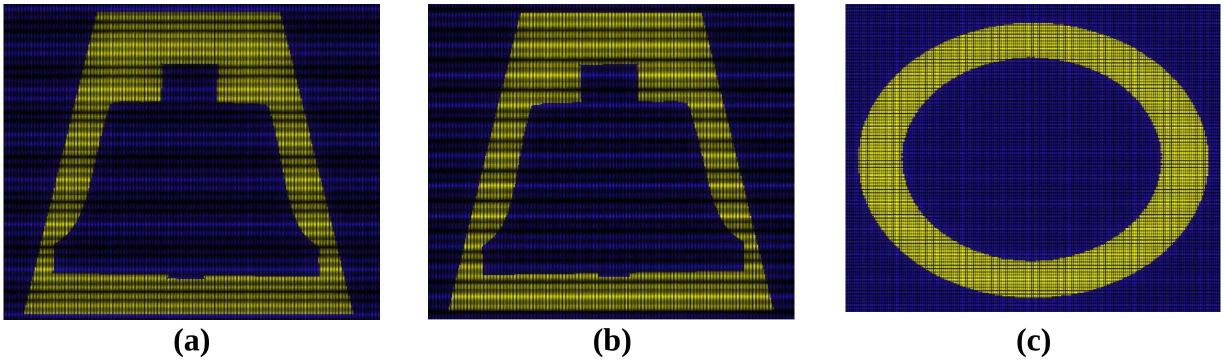


Figure 44. Boolean subtracted volumetric image slick for modified impeller registration

The additional modification presented a more favorable registration result than the previous program runs. However, the image segmentation and modification operations are by no means an ideal solution. Different part geometries would require different approaches for modification and there is no clear way to automate this process. In conclusion, registrations between two dissimilar volumes appear to suffer from similar problems as MMI as described in section 4.1. This being said, there is a notable advantage in the ability to easily modify volumes in the context of part/volume registration. For example, the inclusion of fiducial spheres to a CAD model to help drive the registration to a proper positioning is infinitely easier with these discrete models. Voxel-based models are therefore superior when attempting to debug alignment optimizations where the similarity metric is not behaving as expected.

CHAPTER 6: CONCLUSIONS AND FUTURE WORK

This thesis has presented a simple but novel technique for voxel volume registration in CLMS and has demonstrated its basic viability in a prototype uniform grid application. The presented ideas and data are not meant to describe an exhaustive testing or directly implementable program. Instead, a practical use of this method must necessarily be programmed in a more efficient voxel format (i.e., octree or HDT). The choice to program a uniform grid structure was made so that the basic concept could be validated in the simplest context. However, this decision affected other aspects of the application which may not directly translate to further research.

It was previously found that programs which indexed the intensities of *R* based on the overlap of the surface voxels in *F* performed slightly worse than a material non-implication index. As stated in section 4.2, this was due to the voxel sample size on the uniform grid. This research followed the convention of interpolating the floating volume into the coordinate space of *R* in order to determine which values should be indexed. As a result, the program suffered from interpolation artifacts. Trilinear interpolation tended to cause fluctuations in the total number of voxels present. However, an indexing based only on the outermost voxels in *F* is still the most intuitive means of measuring the distances about *F* and decreases the computation cost by calculating fewer floating point values for the variance. Since future uses of this technique will inevitably use more advanced voxel structures (e.g., octree, HDT), they should not be susceptible to the same errors. In short, future research that would build on this technique may be equally or better served by indexing *R* based on the surface positions of *F*.

The use of a uniform grid also lead to directly storing the voxel intensities of *R* in a lookup table for the subsequent indexing operations. With the voxel resolution set to 0.1mm in this research, this was acceptable. However, a smaller resolution is necessary to achieve the finer detail demanded by professional CAM applications. Since continually increasing the size of the lookup table is impractical, the natural solution is to define a separate grid over *R* prior to the start of the registration iterations. Instead of using the lookup table directly, as done in this research, values for points which lie between grid positions can simply be interpolated. Note that doing this will avoid increasing the voxel size through trilinear interpolation artifacts since the voxel center points of *F* are not being plotted onto the space of *R* in order to index the values.

The qualitative experiments in section 5.4 show that caution should be taken in applying the ideas presented in this research to registrations of dissimilar volumes, particularly larger offset volumes. Such cases may require a more advanced form of automatic segmentation in order to appropriately format the floating volume to conform to the larger stock. The overall results from the reported tests indicate that the proposed technique works best with narrow to moderately offset NNS volumes. It should be noted that the registrations were designed under the assumption that the NNS is offset in all directions. No mechanism exists to ignore a particular dimension although one transformation parameter may be simply turned off. However, if the floating volume must be completely translated in order to be aligned, the results could be unacceptable. For example, a worker might wish to machine a rough casting on all surfaces except the bottom of the workpiece. In this case the program would create an undesirable offset by producing a machining allowance about the bottom of the casting. A more advanced technique, such as MMI may be more applicable in such situations.

As a corollary, further investigation into the use of MMI and information theory registration metrics in computer-aided modelling and alignment is strongly encouraged. This thesis did not directly focus on implementing MMI in CAM registration of rough castings and weldments but it is evident that the technique is powerful and applicable to many described challenges encountered by industry. MMI can be observed to terminate at a desirable position that maintains a consistent amount of material encompassing the floating volume across all dimensions even without an explicit mechanism to minimize the intensity variance as designed in the proposed method. This is because the convergence of MMI is based solely on the information reflected by the intensity values between the images. As seen in section 4.1, MMI will align *F* inside *R* for Euclidean distance transform intensities as long as the general shapes conform to each other. The proposed method in this research performs slightly better under conditions where the volumes are more dissimilar, but only after some alterations to modify the floating volume.

As mentioned previously, the registration of NNS and irregular volume geometries has direct implications to hybrid technologies. Section 5.3 showed that the technique of minimizing voxel intensity variances from a material non-implication index was robust enough to still register slightly modified parts with additional volume present. However, the example should be considered a special case given that so few orientations were available for the turbine blade to assume while still remaining inside its corresponding reference volume. Figure 38 showed a more realistic example of two modified, dissimilar parts. The results using these parts in section 5.3 showed the use of minimization of voxel intensity variance failed to give a favorable alignment for hybrid machining. However, the application of MMI can yield the desired result for such a situation. Figure 45 shows the boolean subtraction plots after a MMI registration using the parts from Figure 38. Notice that the nearly identical sections are perfectly aligned, showing no excess material on the bottom face.

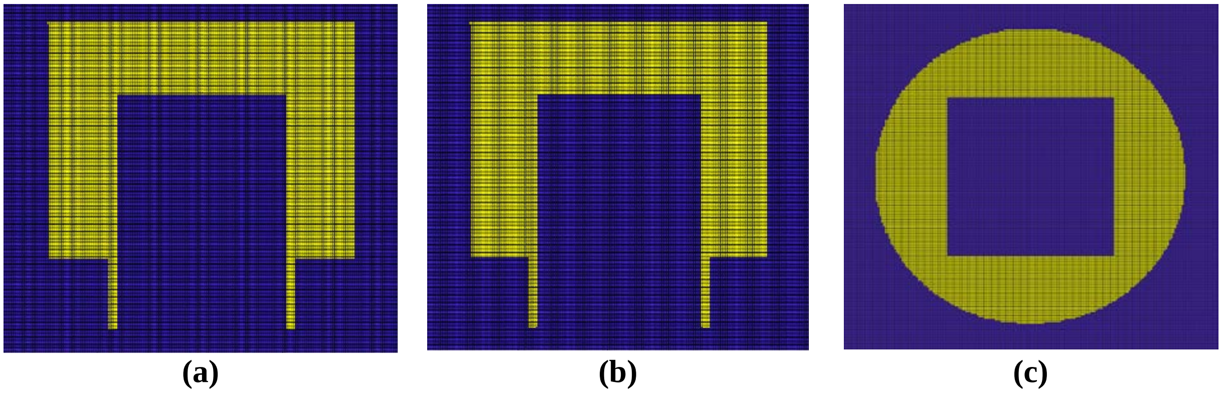


Figure 45. Modified structure registration using MMI

This example clearly demonstrates that MMI would be a better technique in those situations where *F* and *R* retain some nearly identical structures. Like many registration methods, the initial positions that the floating and reference images take relative to each other is important to the convergence of the algorithm [80]. In the case of this example, the registration could easily fail if the models were not aligned closely enough. However, this is not of particular concern since it is not as difficult to achieve a decent positioning to put the volumes in close proximity with each other.

As previously mentioned, emerging hybrid machining technologies will require volume registration techniques to confirm material placement on the workpiece for secondary subtractive operations. It is worth mentioning that although this research proposes the basic idea, the specifics of volume registration in hybrid machining systems has yet to be explored. For example, the uncertainties in proper material placement in additive operations could be addressed on a layer-by-layer basis by incrementally scanning the workpiece.

Overall, the proposed technique of minimizing voxel intensity variance offers flexibility in cases where a central alignment must be identified. Specifically, the technique directly incorporates a means to minimize the variance of voxel distances of *F* from the outer edges of *R.*

Also, it considers only one set of voxel intensities and thus offers a slight computational advantage over MMI when applied to simple registration scenarios. Although further refinements are needed, the similarity metric and registration technique can contribute to certain situations in the future of CLMS and associated volume registrations.

Finally, this research hopes to encourage future research into CLMS as well as the acquisition and processing of sensor data from machine tools. Further research into the actual acquisition of spatial data and integration of NC architectures with CAM software is a monumental task which will require years of further research. The integration of sensor feedback describing the real-time state of a workpiece is an inevitable outcome of the DT concept. How this sensor data is used to update the machining process will become a pivotal manufacturing innovation in the near future.

APPENDIX:

Table 12. Initial registration results using genetic algorithm

				Rotation [Degree]			Translation [Voxel]			Distance [mm]						
	Iterations	Variance Penalty	Variance Penalty Error	R.Z	R.Y	R.X	T.X	T.Y	T.Z	Minimum Distance	Maximum Distance	Mean	Mode	Standard Deviation	Function Evaluations	Function Tolerance
Cube	18	19.813	1.138	-0.18	0.01	0.72	29.18	29.91	31.35	0.67	2.00	1.59	1.50	1.56	570	1.00E-03
	26	18.954	0.279	0.11	0.18	0.07	28.91	29.76	30.00	0.94	1.80	1.60	1.60	0.82	810	1.00E-03
	37	18.572	-0.104	-0.09	-0.02	0.02	29.74	29.53	29.57	1.21	1.70	1.60	1.60	0.51	1140	1.00E-03
	70	18.236	-0.439	0.02	0.02	0.01	30.43	29.53	29.60	1.21	1.70	1.60	1.60	0.41	2130	1.00E-05
	108	18.726	0.050	0.02	0.02	0.01	30.43	29.53	29.60	1.21	1.70	1.60	1.60	0.41	3270	1.00E-05
Sphere	16	20.193	0.327	~~0.27~~	~~2.60~~	~~-3.44~~	29.84	30.02	29.99	1.39	1.81	1.63	1.63	0.520	510	1.00E-03
	15	20.217	0.351	~~2.12~~	~~2.04~~	~~-1.23~~	30.23	30.23	30.41	1.34	1.82	1.63	1.63	0.579	480	1.00E-03
	19	20.184	0.319	~~0.70~~	~~-1.18~~	~~0.73~~	29.93	30.24	29.95	1.40	1.81	1.63	1.63	0.507	600	1.00E-03
	33	20.167	0.301	~~4.47~~	~~-0.37~~	~~-1.03~~	30.01	30.03	30.06	1.40	1.80	1.63	1.63	0.485	1020	1.00E-05
	51	20.173	0.307	~~4.78~~	~~-0.15~~	~~-0.45~~	30.12	30.05	30.03	1.40	1.80	1.63	1.63	0.486	1560	1.00E-05
Parallelepiped	27	18.974	-0.431	0.06	0.01	0.03	29.61	29.50	30.50	1.10	1.80	1.62	1.60	0.78	840	1.00E-03
	50	21.177	1.773	0.36	0.03	0.06	29.58	33.43	31.51	0.75	2.14	1.62	1.70	2.12	1530	1.00E-03
	55	19.946	0.542	0.18	0.07	0.17	28.89	31.54	30.94	0.88	1.96	1.62	1.70	1.23	1680	1.00E-03
	134	20.312	0.908	0.02	0.00	0.18	32.37	30.07	30.98	0.91	1.90	1.62	1.40	1.66	4050	1.00E-05
	48	18.866	-0.538	0.02	0.00	0.18	32.37	30.07	30.98	0.91	1.90	1.62	1.40	1.66	1470	1.00E-05
Torus	37	22.904	0.533	~~-1.56~~	-0.08	0.40	30.40	30.01	29.88	1.31	2.03	1.71	1.50	1.41	1140	1.00E-03
	24	22.699	0.327	~~4.38~~	0.23	0.07	29.76	29.61	30.49	1.36	2.03	1.71	1.60	1.22	750	1.00E-03
	23	22.541	0.169	~~1.88~~	0.13	0.01	30.50	29.73	30.01	1.41	2.02	1.71	1.60	1.12	720	1.00E-03
	37	22.423	0.052	~~0.43~~	0.04	0.00	30.36	29.79	30.49	1.41	2.01	1.71	1.60	1.15	1140	1.00E-05
	101	22.282	-0.090	~~0.43~~	0.04	0.00	30.36	29.79	30.49	1.41	2.01	1.71	1.60	1.15	3060	1.00E-05
Head	69	20.077	0.065	0.27	0.04	-0.03	29.84	30.26	30.36	1.22	1.70	1.60	1.60	0.611	2100	1.00E-03
	29	20.900	0.888	-0.07	0.20	-0.38	31.61	32.18	32.41	0.81	2.00	1.59	1.80	2.223	900	1.00E-03
	27	20.366	0.354	-0.64	0.08	-0.11	30.07	29.89	30.56	1.05	3.56	1.62	1.60	1.275	840	1.00E-03
	58	20.264	0.252	-0.64	0.08	-0.11	30.07	29.89	30.56	1.05	3.56	1.62	1.60	1.275	1770	1.00E-05
	75	20.896	0.885	-0.64	0.08	-0.11	30.07	29.89	30.56	1.05	3.56	1.62	1.60	1.275	2280	1.00E-05
Impeller	29	22.632	0.630	-0.15	0.07	-0.01	28.76	32.03	31.52	1.00	2.67	1.66	1.80	2.03	900	1.00E-03
	35	26.679	4.677	0.03	-0.03	0.08	35.35	25.88	30.55	0.70	3.00	1.59	1.70	3.28	1080	1.00E-03
	27	25.043	3.041	-0.71	0.13	0.03	30.45	30.62	34.88	0.77	2.57	1.66	2.10	3.07	840	1.00E-03
	76	22.581	0.579	-0.02	0.04	0.18	32.26	30.34	30.82	1.04	2.69	1.66	1.70	1.93	2310	1.00E-05
	76	21.563	-0.438	0.00	0.03	0.02	30.42	29.56	31.38	1.12	2.60	1.66	1.70	1.70	2310	1.00E-05

Table 13. Error sets for genetic algorithm initial testing

Cube Genetic Algorithm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
-0.18	0.01	0.72	-0.82	-0.09	1.35
0.11	0.18	0.07	-1.09	-0.24	0.00
-0.09	-0.02	0.02	-0.26	-0.47	-0.43
0.02	0.02	0.01	0.43	-0.47	-0.40
0.02	0.02	0.01	0.43	-0.47	-0.40
MAX ABS ERRORS:					
0.18	0.18	0.72	1.09	0.47	1.35

Sphere Genetic Algorithm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
~~4.47~~	~~-0.37~~	~~-1.03~~	-0.16	0.02	-0.01
~~2.12~~	~~2.04~~	~~-1.23~~	0.23	0.23	0.41
~~0.70~~	~~-1.18~~	~~0.73~~	-0.07	0.24	-0.05
~~4.47~~	~~-0.37~~	~~-1.03~~	0.01	0.03	0.06
~~4.78~~	~~-0.15~~	~~-0.45~~	0.12	0.05	0.03
MAX ABS ERRORS:					
~~4.78~~	~~2.04~~	~~1.23~~	0.23	0.24	0.41

Parall. Genetic Algorithm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
0.06	0.01	0.03	-0.39	-0.50	0.50
0.36	0.03	0.06	-0.42	3.43	1.51
0.18	0.07	0.17	-1.11	1.54	0.94
0.02	0.00	0.18	2.37	0.07	0.98
0.02	0.00	0.18	2.37	0.07	0.98
MAX ABS ERRORS:					
0.36	0.07	0.18	2.37	3.43	1.51

Torus Genetic Algorithm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
~~-1.56~~	-0.08	0.40	0.40	0.01	-0.12
~~4.38~~	0.23	0.07	-0.24	-0.39	0.49
~~1.88~~	0.13	0.01	0.50	-0.27	0.01
~~0.43~~	0.04	0.00	0.36	-0.21	0.49
~~0.43~~	0.04	0.00	0.36	-0.21	0.49
MAX ABS ERRORS:					
~~4.38~~	0.23	0.40	0.50	0.39	0.49

Head Genetic Algorithm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
0.27	0.04	-0.03	-0.16	0.26	0.36
-0.07	0.20	-0.38	1.61	2.18	2.41
-0.64	0.08	-0.11	0.07	-0.11	0.56
-0.64	0.08	-0.11	0.07	-0.11	0.56
-0.64	0.08	-0.11	0.07	-0.11	0.56
MAX ABS ERRORS:					
0.64	0.20	0.38	1.61	2.18	2.41

Impeller Genetic Algorithm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
-0.15	0.07	-0.01	-1.24	2.03	1.52
0.03	-0.03	0.08	5.35	-4.12	0.55
-0.71	0.13	0.03	0.45	0.62	4.88
-0.02	0.04	0.18	2.26	0.34	0.82
0.00	0.03	0.02	0.42	-0.44	1.38
MAX ABS ERRORS:					
0.71	0.13	0.18	5.35	4.12	4.88

Table 14. Initial offset registration tests using particle swarm

				Rotation [Degree]			Translation [Voxel]			Distance [mm]							
	Iterations	Variance Penalty	Variance Penalty Error	R.Z	R.Y	R.X	T.X	T.Y	T.Z	Minimum Distance	Maximum Distance	Mean	Mode	Standard Deviation	Function Evaluations	Function Tolerance	Swarm Size
Cube	30	18.683	0.008	0.04	0.01	-0.12	30.50	30.32	29.94	1.21	1.70	1.60	1.60	0.55	1550	1.00E-04	100
	31	18.832	0.156	0.02	-0.18	-0.06	29.18	29.65	30.71	1.17	1.80	1.60	1.60	0.76	1600	1.00E-04	100
	8	27501342	27501324	-0.11	1.26	0.04	21.90	42.96	21.14	NA	NA	NA	NA	NA	450	1.00E-04	100
	56	18.113	-0.563	0.00	0.00	0.00	29.56	29.52	30.49	1.24	1.60	1.60	1.60	0.19	2850	1.00E-06	150
	70	18.137	-0.539	0.00	0.01	0.01	29.59	30.41	29.56	1.24	1.60	1.60	1.60	0.19	3550	1.00E-06	150
Sphere	25	20.167	0.302	5.00	-2.56	-0.19	29.90	29.99	30.02	1.40	1.81	1.63	1.63	0.498	1300	1.00E-04	100
	25	20.186	0.320	3.68	0.42	3.81	30.02	29.99	30.08	1.39	1.81	1.63	1.63	0.508	1300	1.00E-04	100
	9	21.535	1.669	-5.00	-5.00	1.91	31.96	30.42	32.22	1.14	2.01	1.63	1.63	1.811	500	1.00E-04	100
	34	20.168	0.302	4.43	0.16	0.19	29.96	29.84	30.15	1.40	1.80	1.63	1.63	0.495	1750	1.00E-06	150
	137	20.100	0.234	-0.18	-0.17	-0.18	30.01	30.01	30.08	1.40	1.74	1.63	1.63	0.461	6900	1.00E-06	150
Parallelepiped	58	18.772	-0.632	0.00	0.00	0.01	29.53	29.59	30.46	1.10	1.71	1.62	1.60	0.578	2950	1.00E-04	100
	50	18.803	-0.602	0.03	0.01	0.02	29.42	29.57	29.53	1.10	1.75	1.62	1.60	0.613	2550	1.00E-04	100
	13	25.831	6.427	-0.27	0.49	-1.20	32.68	25.77	30.59	0.20	2.45	1.61	1.90	3.898	700	1.00E-04	100
	88	18.756	-0.649	0.00	0.00	0.01	29.56	30.37	30.47	1.04	1.70	1.62	1.60	0.569	4450	1.00E-06	150
	70	18.751	-0.653	0.00	0.00	0.01	29.55	30.37	30.46	1.04	1.70	1.62	1.60	0.569	3550	1.00E-06	150
Torus	11	31.580	9.208	-0.53	-0.90	-0.73	24.34	31.18	33.70	0.57	2.81	1.69	1.50	4.71	600	1.00E-04	100
	8	29.641	7.269	2.45	0.86	1.27	27.54	26.19	28.30	0.70	2.57	1.69	1.30	4.29	450	1.00E-04	100
	8	26.512	4.140	-1.99	-0.46	-0.98	30.64	26.39	31.24	0.89	2.48	1.70	1.30	3.24	450	1.00E-04	100
	106	22.319	-0.053	-3.14	0.00	0.00	30.00	29.93	29.56	1.49	2.00	1.71	1.60	1.05	5350	1.00E-06	150
	91	22.296	-0.076	0.00	-0.08	-0.09	30.48	29.54	30.00	1.45	1.99	1.71	1.60	1.09	4600	1.00E-06	150
Head	28	20.565	0.553	0.87	-0.03	0.01	30.33	30.51	31.48	1.12	3.63	1.62	1.70	1.53	1450	1.00E-04	100
	51	19.690	-0.322	-0.01	0.00	-0.01	29.54	29.63	29.43	1.14	3.62	1.61	1.50	1.25	2600	1.00E-04	100
	73	19.594	-0.418	0.00	0.00	0.01	29.59	30.47	30.41	1.17	3.56	1.62	1.60	1.17	3700	1.00E-04	100
	37	19.902	-0.109	0.00	0.00	0.01	29.59	30.47	30.41	1.17	3.56	1.62	1.60	1.17	1900	1.00E-06	150
	115	19.590	-0.422	0.00	0.00	0.01	29.59	30.47	30.41	1.17	3.56	1.62	1.60	1.17	5800	1.00E-06	150
Impeller	34	21.479	-0.523	-0.35	-0.01	0.00	29.96	29.48	31.45	1.14	2.60	1.66	1.70	1.66	1750	1.00E-04	100
	30	21.582	-0.419	-0.34	0.00	0.01	30.37	29.50	31.35	1.10	2.60	1.66	1.70	1.67	1550	1.00E-04	100
	8	34.037	12.035	-0.16	-0.55	-1.23	31.20	34.30	24.22	0.17	3.47	1.54	1.30	5.70	450	1.00E-04	100
	33	22.050	0.048	-0.33	0.16	0.03	30.06	30.37	31.29	1.12	2.63	1.67	1.70	1.72	1700	1.00E-06	150
	144	21.396	-0.606	-0.32	0.00	0.00	29.57	29.60	31.50	1.17	2.60	1.66	1.70	1.66	7250	1.00E-06	150

Table 15. Error sets for particle swarm initial tests

Cube Particle Swarm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
0.04	0.01	-0.12	0.50	0.32	-0.06
0.02	-0.18	-0.06	-0.82	-0.35	0.71
-0.11	1.26	0.04	-8.10	12.96	-8.86
0.00	0.00	0.00	-0.44	-0.48	0.49
0.00	0.01	0.01	-0.41	0.41	-0.44
MAX ABS ERRORS:					
0.11	1.26	0.12	8.10	12.96	8.86

Sphere Particle Swarm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
5.00	-2.56	-0.19	-0.10	-0.01	0.02
3.68	0.42	3.81	0.02	-0.01	0.08
-5.00	-5.00	1.91	1.96	0.42	2.22
4.43	0.16	0.19	-0.04	-0.16	0.15
-0.18	-0.17	-0.18	0.01	0.01	0.08
MAX ABS ERRORS:					
5.00	5.00	3.81	1.96	0.42	2.22

Parall. Particle Swarm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
0.00	0.00	0.01	-0.47	-0.41	0.46
0.03	0.01	0.02	-0.58	-0.43	-0.47
-0.27	0.49	-1.20	2.68	-4.23	0.59
0.00	0.00	0.01	-0.44	0.37	0.47
0.00	0.00	0.01	-0.45	0.37	0.46
MAX ABS ERRORS:					
0.27	0.49	1.20	2.68	4.23	0.59

Torus Particle Swarm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
-0.53	-0.90	-0.73	-5.66	1.18	3.70
2.45	0.86	1.27	-2.46	-3.81	-1.70
-1.99	-0.46	-0.98	0.64	-3.61	1.24
-3.14	0.00	0.00	0.00	-0.07	-0.44
0.00	-0.08	-0.09	0.48	-0.46	0.00
MAX ABS ERRORS:					
3.14	0.90	1.27	5.66	3.81	3.70

Head Particle Swarm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
0.87	-0.03	0.01	0.33	0.51	1.48
-0.01	0.00	-0.01	-0.46	-0.37	-0.57
0.00	0.00	0.01	-0.41	0.47	0.41
0.00	0.00	0.01	-0.41	0.47	0.41
0.00	0.00	0.01	-0.41	0.47	0.41
MAX ABS ERRORS:					
0.87	0.00	0.01	0.46	0.51	1.48

Impeller Particle Swarm Errors					
Rotation Error			Translation Error		
R.Z	**R.Y**	**R.X**	**T.X**	**T.Y**	**T.Z**
-0.35	-0.01	0.00	-0.04	-0.52	1.45
-0.34	0.00	0.01	0.37	-0.50	1.35
-0.16	-0.55	-1.23	1.20	4.30	-5.78
-0.33	0.16	0.03	0.06	0.37	1.29
-0.32	0.00	0.00	-0.43	-0.40	1.50
MAX ABS ERRORS:					
0.35	0.55	1.23	1.20	4.30	5.78

Table 16. Initial offset registration tests using simulated annealing

				Rotation			Translation			Distance [mm]						
	Iterations	Variance Penalty	Variance Penalty Error	R.Z	R.Y	R.X	T.X	T.Y	T.Z	Minimum Distance	Maximum Distance	Mean	Mode	Standard Deviation	Function Evaluations	Stall Iterations
Cube	530	23.386	4.710	0.13	0.08	-0.34	30.86	34.79	32.27	0.66	2.20	1.59	1.50	3.17	543	50
	290	20.354	1.678	-0.33	-0.20	-0.21	27.37	31.05	29.13	0.60	2.00	1.59	1.70	1.90	297	50
	360	38.835	20.159	-0.15	0.19	-0.09	21.01	37.79	29.27	0.00	2.60	1.57	0.80	6.86	367	50
	150	39.752	21.076	0.10	0.21	-0.25	29.33	40.10	36.78	0.14	2.70	1.57	1.60	7.02	151	75
	280	43.054	24.378	0.10	0.21	-0.25	29.33	40.10	36.78	0.14	2.70	1.57	1.60	7.02	287	75
Sphere	100	22.036	2.170	~~4.92~~	~~-4.09~~	~~1.64~~	27.07	28.13	30.22	1.17	2.12	1.63	1.39	2.06	101	50
	80	22.172	2.306	~~1.11~~	~~4.13~~	~~-1.36~~	29.85	31.47	33.29	1.04	2.10	1.63	1.39	2.15	81	50
	110	20.394	0.528	~~2.54~~	~~-2.58~~	~~4.94~~	30.21	30.62	29.09	1.30	1.91	1.63	1.59	0.84	111	50
	130	20.885	1.019	~~4.51~~	~~4.79~~	~~2.47~~	30.06	28.67	31.67	1.20	1.96	1.63	1.55	1.34	131	75
	210	20.229	0.363	~~0.46~~	~~-0.43~~	~~-4.08~~	29.47	29.96	30.16	1.36	1.83	1.63	1.68	0.60	217	75
Parallelepiped	380	21.525	2.121	0.15	0.46	0.16	32.54	28.58	29.60	0.87	2.10	1.62	1.50	2.14	387	50
	140	29.907	10.502	0.20	-0.17	0.09	26.93	33.45	38.03	0.30	2.50	1.61	0.80	4.99	141	50
	240	41.164	21.760	-0.02	-0.01	-1.33	39.36	26.08	23.50	0.00	2.80	1.59	0.70	7.07	241	50
	370	20.143	0.739	0.10	-0.29	0.02	31.27	30.05	29.66	0.81	2.00	1.62	1.60	1.24	377	75
	240	21.667	2.262	0.16	0.06	-0.12	31.64	31.54	33.00	0.75	2.00	1.62	1.90	2.26	247	75
Torus	80	29.516	7.145	~~0.70~~	-1.52	-0.19	34.86	28.79	30.00	0.76	2.66	1.69	1.30	4.20	81	50
	160	25.344	2.972	~~-4.89~~	0.28	0.18	26.66	27.74	32.03	1.10	2.40	1.70	1.30	2.69	161	50
	170	22.736	0.364	~~-2.21~~	-0.13	-0.05	30.95	30.88	30.17	1.40	2.10	1.71	1.60	1.25	171	50
	210	25.098	2.727	~~-3.06~~	0.27	0.03	27.40	26.05	30.73	1.16	2.46	1.70	1.60	2.50	217	75
	380	24.293	1.921	~~1.07~~	0.03	-0.03	28.28	26.13	30.58	1.24	2.39	1.70	1.50	2.22	387	75
Head	240	24.269	4.257	1.82	-0.27	-0.36	27.75	33.62	27.84	0.54	3.87	1.61	1.40	3.19	247	50
	160	23.842	3.831	-1.86	0.02	0.32	30.91	29.89	25.56	0.58	3.77	1.60	1.20	3.01	161	50
	230	21.850	1.838	-0.46	0.09	0.46	29.33	27.11	28.20	0.99	3.67	1.62	1.50	2.14	237	50
	390	21.664	1.653	-0.46	0.09	0.46	29.33	27.11	28.20	0.99	3.67	1.62	1.50	2.14	397	75
	340	22.841	2.829	-0.46	0.09	0.46	29.33	27.11	28.20	0.99	3.67	1.62	1.50	2.14	347	75
Impeller	180	37.805	15.804	0.04	-0.14	0.00	32.80	41.37	24.96	0.14	3.44	1.45	1.10	6.17	181	50
	470	24.816	2.814	-0.36	0.17	-0.14	28.35	25.78	32.29	0.81	2.75	1.63	1.80	2.78	483	50
	190	27.137	5.135	-0.01	-0.12	-0.05	27.16	33.28	34.83	0.57	2.86	1.61	2.10	3.62	191	50
	140	28.699	6.697	-0.14	0.12	-0.43	28.80	37.83	31.62	0.36	3.19	1.55	1.70	3.92	141	75
	280	1897.957	1875.956	0.15	-0.55	-0.86	18.82	19.52	25.31	NA	NA	NA	NA	NA	281	75

Table 17. Error sets for simulated annealing initial testing

Cube Simulated Annealing Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
0.13	0.08	-0.34	0.86	4.79	2.27
-0.33	-0.20	-0.21	-2.63	1.05	-0.87
-0.15	0.19	-0.09	-8.99	7.79	-0.73
0.10	0.21	-0.25	-0.67	10.10	6.78
0.10	0.21	-0.25	-0.67	10.10	6.78
MAX ABS ERRORS:					
0.33	0.21	0.34	8.99	10.10	6.78

Sphere Simulated Annealing Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
~~4.92~~	~~-4.09~~	~~1.64~~	-2.93	-1.87	0.22
~~1.11~~	~~4.13~~	~~-1.36~~	-0.15	1.47	3.29
~~2.54~~	~~-2.58~~	~~4.94~~	0.21	0.62	-0.91
~~4.51~~	~~4.79~~	~~2.47~~	0.06	-1.33	1.67
~~0.46~~	~~-0.43~~	~~-4.08~~	-0.53	-0.04	0.16
MAX ABS ERRORS:					
~~4.92~~	~~4.79~~	~~4.94~~	2.93	1.87	3.29

Parall. Simulated Annealing Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
0.15	0.46	0.16	2.54	-1.42	-0.40
0.20	-0.17	0.09	-3.07	3.45	8.03
-0.02	-0.01	-1.33	9.36	-3.92	-6.50
0.10	-0.29	0.02	1.27	0.05	-0.34
0.16	0.06	-0.12	1.64	1.54	3.00
MAX ABS ERRORS:					
0.20	0.46	1.33	9.36	3.92	8.03

Torus Simulated Annealing Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
~~0.70~~	-1.52	-0.19	4.86	-1.21	0.00
~~-4.89~~	0.28	0.18	-3.34	-2.26	2.03
~~-2.21~~	-0.13	-0.05	0.95	0.88	0.17
~~-3.06~~	0.27	0.03	-2.60	-3.95	0.73
~~1.07~~	0.03	-0.03	-1.72	-3.87	0.58
MAX ABS ERRORS:					
~~4.89~~	1.52	0.19	4.86	3.95	2.03

Head Simulated Annealing Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
1.82	-0.27	-0.36	-2.25	3.62	-2.16
-1.86	0.02	0.32	0.91	-0.11	-4.44
-0.46	0.09	0.46	-0.67	-2.89	-1.80
-0.46	0.09	0.46	-0.67	-2.89	-1.80
-0.46	0.09	0.46	-0.67	-2.89	-1.80
MAX ABS ERRORS:					
1.86	0.27	0.46	2.25	3.62	2.16

Impeller Simulated Annealing Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
0.04	-0.14	0.00	2.80	11.37	-5.04
-0.36	0.17	-0.14	-1.65	-4.22	2.29
-0.01	-0.12	-0.05	-2.84	3.28	4.83
-0.14	0.12	-0.43	-1.20	7.83	1.62
0.15	-0.55	-0.86	-11.18	-10.48	-4.69
MAX ABS ERRORS:					
0.36	0.55	0.86	11.18	11.37	5.04

Table 18. Registration tests using integer constrained translation

				Rotation [Degree]			Translation [Voxel]			Distance [mm]							
	Iterations	Variance Penalty	Variance Penalty Error	R.Z	R.Y	R.X	T.X	T.Y	T.Z	Minimum Distance	Maximum Distance	Mean	Mode	Standard Deviation	Function Evaluations	Function Tolerance	Offset Input
Cube	32	18.789	0.113	0.13	-0.19	0.00	30	30	30	1.06	1.70	1.60	1.60	0.48	990	1.00E-03	1.5 mm
	28	19.361	0.685	-0.05	-0.53	-0.22	30	29	30	0.81	1.90	1.60	1.50	1.19	870	1.00E-03	1.5 mm
	62	18.813	0.138	0.23	0.05	-0.17	30	30	30	1.00	1.70	1.60	1.60	0.60	1890	1.00E-05	1.5 mm
	37	34.645	0.241	-0.15	-0.16	-0.03	29	30	31	1.39	2.30	2.13	22.00	0.90	1140	1.00E-03	2 mm
	20	36.021	1.616	-0.01	-0.23	1.09	29	30	31	1.17	2.60	2.12	21.00	2.02	630	1.00E-03	2 mm
Sphere	14	20.165	0.299	-1.36	1.10	-0.12	30	30	30	1.40	1.80	1.63	1.63	0.48	450	1.00E-03	1.5 mm
	13	20.170	0.305	-1.09	2.30	0.31	30	30	30	1.40	1.80	1.63	1.63	0.49	420	1.00E-03	1.5 mm
	40	20.103	0.237	0.18	-0.21	0.15	30	30	30	1.40	1.74	1.63	1.63	0.46	1230	1.00E-05	1.5 mm
	14	34.765	0.340	-0.15	-1.80	-1.47	30	30	30	1.90	2.30	2.12	21.12	0.35	450	1.00E-03	2 mm
	13	34.751	0.326	2.19	-0.32	-0.47	30	30	30	1.92	2.25	2.12	21.12	0.35	420	1.00E-03	2 mm
Parallelepiped	22	20.311	0.906	-0.77	0.19	-0.21	29	30	30	0.49	2.10	1.62	1.60	1.51	690	1.00E-03	1.5 mm
	24	19.784	0.380	-0.28	-0.17	-0.02	30	30	31	0.97	1.82	1.62	1.70	0.99	750	1.00E-03	1.5 mm
	91	19.491	0.087	-0.18	-0.05	-0.13	30	30	30	1.10	1.80	1.62	1.60	0.64	2760	1.00E-05	1.5 mm
	24	34.128	0.397	-0.11	0.07	0.00	31	30	30	1.57	2.30	2.11	21.00	0.67	750	1.00E-03	2 mm
	26	33.782	0.050	0.00	0.09	0.18	30	30	30	1.71	2.28	2.10	21.00	0.50	810	1.00E-03	2 mm
Torus	14	22.520	0.149	2.45	0.07	0.13	30	30	30	1.41	2.00	1.71	1.60	1.12	450	1.00E-03	1.5 mm
	22	22.591	0.220	-4.67	0.08	-0.03	30	30	30	1.51	2.00	1.71	1.60	1.04	690	1.00E-03	1.5 mm
	43	22.566	0.194	-1.49	-0.03	0.09	30	30	30	1.51	2.00	1.71	1.60	1.04	1320	1.00E-05	1.5 mm
	18	38.537	0.380	1.54	-0.07	0.34	30	30	30	1.84	2.51	2.21	22.00	1.32	570	1.00E-03	2 mm
	15	38.637	0.481	-2.36	0.23	-0.27	29	30	30	1.82	2.60	2.21	20.00	1.40	480	1.00E-03	2 mm
Head	24	20.262	0.250	0.12	-0.13	0.17	30	30	30	1.12	3.56	1.62	1.60	1.20	750	1.00E-03	1.5 mm
	22	20.259	0.247	0.16	-0.16	-0.14	30	30	30	1.16	3.54	1.62	1.60	1.21	690	1.00E-03	1.5 mm
	65	20.329	0.317	-0.11	0.13	-0.12	29	29	30	1.10	3.64	1.62	1.65	1.49	1980	1.00E-05	1.5 mm
	16	36.165	0.926	0.74	0.11	0.03	30	30	32	1.63	4.22	2.13	23.00	1.79	510	1.00E-03	2 mm
	31	35.667	0.428	-0.11	0.12	-0.32	30	30	31	1.64	4.21	2.13	22.00	1.51	960	1.00E-03	2 mm
Impeller	24	21.958	-0.044	-0.18	-0.15	-0.05	30	30	31	1.12	2.66	1.66	1.70	1.67	750	1.00E-03	1.5 mm
	38	21.905	-0.097	-0.15	0.13	0.06	30	30	31	1.12	2.63	1.66	1.70	1.65	1170	1.00E-03	1.5 mm
	65	21.948	-0.054	-0.34	-0.10	0.10	30	30	31	1.12	2.65	1.66	1.70	1.70	1980	1.00E-05	1.5 mm
	11	46.525	6.472	1.67	0.68	0.53	29	31	30	0.94	3.73	2.06	21.00	3.75	360	1.00E-03	2 mm
	19	40.444	0.390	-0.82	0.15	-0.17	30	30	31	1.45	3.38	2.20	22.00	2.52	600	1.00E-03	2 mm

Table 19. Error sets for registrations using integer constrained translation

Cube Constrained Translation Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
0.13	-0.19	0.00	0	0	0
-0.05	-0.53	-0.22	0	-1	0
0.23	0.05	-0.17	0	0	0
-0.15	-0.16	-0.03	-1	1	1
-0.01	-0.23	1.09	-1	0	1
MAX ABS ERRORS:					
0.23	0.53	1.09	1	1	1

Sphere Constrained Translation Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
~~-1.36~~	~~1.10~~	~~-0.12~~	0	0	0
~~-1.09~~	~~2.30~~	~~0.31~~	0	0	0
~~0.18~~	~~-0.21~~	~~0.15~~	0	0	0
~~-0.15~~	~~-1.80~~	~~-1.47~~	0	0	0
~~2.19~~	~~-0.32~~	~~-0.47~~	0	0	0
MAX ABS ERRORS:					
~~2.19~~	~~2.30~~	~~1.47~~	0	0	0

Parall. Constrained Translation Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
-0.77	0.19	-0.21	-1	0	0
-0.28	-0.17	-0.02	0	0	1
-0.18	-0.05	-0.13	0	0	0
-0.11	0.07	0.00	1	0	0
0.00	0.09	0.18	0	0	0
MAX ABS ERRORS:					
0.77	0.19	0.21	1	0	1

Torus Constrained Translation Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
~~2.45~~	0.07	0.13	0	0	0
~~-4.67~~	0.08	-0.03	0	0	0
~~-1.49~~	-0.03	0.09	0	0	0
~~1.54~~	-0.07	0.34	0	0	0
~~-2.36~~	0.23	-0.27	-1	0	0
MAX ABS ERRORS:					
~~4.67~~	0.23	0.34	1	0	0

Head Constrained Translation Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
0.12	-0.13	0.17	0	0	0
0.16	-0.16	-0.14	0	0	0
-0.11	0.13	-0.12	-1	-1	0
0.74	0.11	0.03	0	0	2
-0.11	0.12	-0.32	0	0	1
MAX ABS ERRORS:					
0.74	0.16	0.32	1	1	2

Impeller Constrained Translation Errors					
Rotation Error			Translation Error		
R.Z	R.Y	R.X	T.X	T.Y	T.Z
-0.18	-0.15	-0.05	0	0	1
-0.15	0.13	0.06	0	0	1
-0.34	-0.10	0.10	0	0	1
1.67	0.68	0.53	-1	1	0
-0.82	0.15	-0.17	0	0	1
MAX ABS ERRORS:					
1.67	0.68	0.53	1	1	1

REFERENCES

[1] A. Gessner, R. Staniek, and T. Bartkowiak, “Computer-aided alignment of castings and machining optimization,” *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.*, vol. 229, no. 3, pp. 485–492, Feb. 2015.

[2] F. E. Peters and E. J. Salisbury, “Effect of casting form variability on machining fixturing error,” *Trans. Am. Foundrymen’S Soc. Vol 106*, vol. 106, pp. 659–663, 1998.

[3] P. Maloney, N. Moroz, C. Stanfill, and N. Zalenski, “Turbine Blade Fixture for Inspection and Grinding,” no. April, 2008.

[4] E. Kurokaiva, “Flexible Conformable Clamps for a Machiniilg Cell with Applications to Turbine Blade Machining,” no. May, 1983.

[5] E. H. Glaessgen and D. S. Stargel, “The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles,” in *53rd Structures, Structural Dynamics and Materials Conference - Special Session: Digital Twin*, 2012, no. April, pp. 1–14.

[6] S. Boschert and R. Rosen, “Digital twin-the simulation aspect,” in *Mechatronic Futures: Challenges and Solutions for Mechatronic Systems and Their Designers*, Springer International Publishing, 2016, pp. 59–74.

[7] B. Schleich, N. Anwer, L. Mathieu, and S. Wartzack, “Shaping the digital twin for design and production engineering,” *CIRP Ann. - Manuf. Technol.*, vol. 66, no. 1, pp. 141–144, 2017.

[8] M. Grieves, “How A ‘Digital Twin’ Can Warrant Products Are Built As Designed,” *Manuf. Bus. Technol.*, Jan. 2015.

[9] F. Tao *et al.*, “Digital twin-driven product design framework,” *Int. J. Prod. Res.*, pp. 1–19, Feb. 2018.

[10] J. D. Hochhalter, *Coupling damage-sensing particles to the digital twin concept*. 2014.

[11] A. M. Miller, R. Alvarez, and N. Hartman, “Towards an extended model-based definition for the digital twin,” *Comput. Aided. Des. Appl.*, pp. 1–12, Apr. 2018.

[12] H. Zhang, Q. Liu, X. Chen, D. Zhang, and J. Leng, “A Digital Twin-Based Approach for Designing and Multi-Objective Optimization of Hollow Glass Production Line,” *IEEE Access*, vol. 5, pp. 26901–26911, Oct. 2017.

[13] G. L. Knapp *et al.*, “Building blocks for a digital twin of additive manufacturing,” *Acta Mater.*, vol. 135, no. C, pp. 390–399, Aug. 2017.

[14] M. J. Pratt, “Introduction to ISO 10303—the STEP Standard for Product Data Exchange,” *J. Comput. Inf. Sci. Eng.*, vol. 1, no. 1, 2001.

[15] Hamilton, Hascoet, and J.-Y. M. Rauch Kelvin, “Implementing STEP-NC: Exploring Possibilities for the Future of Advanced Manufacturing.” pp. 199–239, 2014.

[16] M. Rauch, R. Laguionie, J.-Y. Hascoet, and S.-H. Suh, “An advanced STEP-NC controller for intelligent machining processes,” *Robot. Comput. Integr. Manuf.*, vol. 28, no. 3, 2011.

[17] P. Hu, Z. Han, H. Fu, and D. Han, “Architecture and implementation of closed-loop machining system based on open STEP-NC controller,” *Int. J. Adv. Manuf. Technol.*, vol. 83, no. 5, pp. 1361–1375, Mar. 2016.

[18] R. Lynn, M. Sati, T. Tucker, J. Rossignac, C. Saldana, and T. Kurfess, “Realization of the 5-Axis Machine Tool Digital Twin Using Direct Servo Control from CAM,” *NIST Model-Based Enterprise Summit*. National Institute of Science and Technology, Gaitherburg, MD, pp. 1–22, 2018.

[19] A. Masood, R. Siddiqui, M. Pinto, H. Rehman, and M. A. Khan, “Tool Path Generation, for Complex Surface Machining, Using Point Cloud Data,” *Procedia CIRP*, vol. 26, no. C,

pp. 397–402, 2015.

[20] J. D. Barnfather and T. Abram, “Efficient compensation of dimensional errors in robotic machining using imperfect point cloud part inspection data,” *Measurement*, vol. 117, pp. 176–185, Mar. 2018.

[21] H.-Y. Feng and Z. Teng, “Iso-planar piecewise linear NC tool path generation from discrete measured data points,” *Comput. Des.*, vol. 37, no. 1, pp. 55–64, Jan. 2005.

[22] Q. Zou and J. Zhao, “Iso-parametric tool-path planning for point clouds,” *Comput. Des.*, vol. 45, no. 11, pp. 1459–1468, Nov. 2013.

[23] D. Zhang, P. Yang, and X. Qian, “Adaptive NC Path Generation From Massive Point Data With Bounded Error,” *J. Manuf. Sci. Eng.*, vol. 131, no. 1, 2009.

[24] Y. Liu, S. Xia, and X. Qian, “Direct numerical control (NC) path generation: From discrete points to continuous spline paths,” *J. Comput. Inf. Sci. Eng.*, vol. 12, no. 3, 2012.

[25] J. Xu, L. Zhang, and Y. Wang, “Tool Path Generation for Machining Point Cloud Data by Least Squared Projection,” *Appl. Mech. Mater.*, vol. 635–637, pp. 555–559, Sep. 2014.

[26] W. Liu, L. Zhou, and L. An, “Constant scallop-height tool path generation for three-axis discrete data points machining,” *Int. J. Adv. Manuf. Technol.*, vol. 63, no. 1, pp. 137–146, Nov. 2012.

[27] J. F. Chatelain and C. Fortin, “A balancing technique for optimal blank part machining,” *Precis. Eng.*, vol. 25, no. 1, pp. 13–23, 2001.

[28] P. Haghighi, S. Ramnath, N. Kalish, J. V Shah, J. J. Shah, and J. K. Davidson, “Method for automating digital fixture-setups that are optimal for machining castings to minimize scrap,” *J. Manuf. Syst.*, vol. 40, pp. 15–24, Jul. 2016.

[29] N. J. Kalish, S. Ramnath, P. Haghighi, J. K. Davidson, J. J. Shah, and J. V. Shah, “The

set-up-map for automating the positioning of castings and weldments in fixtures to ensure completely machined surfaces," *Proc. ASME Des. Eng. Tech. Conf.*, vol. 1A–2016, pp. 1–17, 2016.

[30] S. Ramnath, "Automating Fixture Setups Based on Point Cloud Data & CAD Model." ProQuest Dissertations Publishing, Jan-2016.

[31] P. Lei and L. Zheng, "An automated in-situ alignment approach for finish machining assembly interfaces of large-scale components," *Robot. Comput. Integr. Manuf.*, vol. 46, pp. 130–143, Aug. 2017.

[32] L. Zheng, X. Zhu, R. Liu, Y. Wang, and P. G. Maropoulos, "A Novel Algorithm of Posture Best Fit based on Key Characteristics for Large Components Assembly," *Procedia CIRP*, vol. 10, pp. 162–168, 2013.

[33] M. M. Hossain, "Voxel-Based Offsetting at High Resolution with Tunable Speed and Precision Using Hybrid Dynamic Trees," no. December, 2016.

[34] J. Kalojanov, M. Billeter, and P. Slusallek, "Two-level grids for ray tracing on GPUs," *Comput. Graph. Forum*, vol. 30, no. 2, pp. 307–314, 2011.

[35] E. Langetepe, *Geometric data structures for computer graphics*. Wellesley, Mass.: A K Peters, 2006.

[36] M. M. Hossain, T. M. Tucker, T. R. Kurfess, and R. W. Vuduc, "Hybrid Dynamic Trees for Extreme-Resolution 3D Sparse Data Modeling," *Proc. - 2016 IEEE 30th Int. Parallel Distrib. Process. Symp. IPDPS 2016*, pp. 132–141, 2016.

[37] D. Ibaroudene and R. Acharya, "Coordinate relationships between vertices of linear octree nodes and corners of the universe," *Comput. Graph.*, vol. 15, no. 3, pp. 375–381, 1991.

[38] M. M. Hossain, T. M. Tucker, T. R. Kurfess, and R. W. Vuduc, "A GPU-parallel

construction of volumetric tree," *Proc. 5th Work. Irregul. Appl. Archit. Algorithms - IA3 '15*, pp. 1–4, 2015.

[39] D. Konobrytskyi, M. M. Hossain, T. M. Tucker, J. A. Tarbutton, and T. R. Kurfess, "5-Axis tool path planning based on highly parallel discrete volumetric geometry representation: Part I contact point generation," *Comput. Aided. Des. Appl.*, vol. 15, no. 1, pp. 76–89, 2018.

[40] M. M. Hossain, R. W. Vuduc, C. Nath, T. R. Kurfess, and T. M. Tucker, "A Graphical Approach for Freeform Surface Offsetting With GPU Acceleration for Subtractive 3D Printing," no. 49903. p. V002T04A031, 2016.

[41] I. N. Bankman, "Segmentation: Handbook of Medical Imaging," in *Handbook of Medical Imaging*, 2000, pp. 67–68.

[42] R. P. Woods, "Introduction to Registration," *Handb. Med. Imaging*, pp. 421–423, 2000.

[43] R. P. Woods, *36.Within-Modality Registration Using Intensity-Based Cost Functions*. Academic Press, 2008.

[44] D. L. G. Hill and D. J. Hawkes, "37.Across-Modality Registration Using Intensity-Based Cost Functions," *Handb. Med. Image Process. Anal.*, pp. 613–628, 2008.

[45] X. Sun, F. D. McKenzie, S. Bawab, J. Li, Y. Yoon, and J.-K. Huang, "Automated dental implantation using image-guided robotics: registration results," *Int. J. Comput. Assist. Radiol. Surg.*, vol. 6, no. 5, pp. 627–634, Sep. 2011.

[46] Z. Wang, H. Kieu, H. Nguyen, and M. Le, "Digital image correlation in experimental mechanics and image registration in computer vision: Similarities, differences and complements," *Opt. Lasers Eng.*, vol. 65, pp. 18–27, 2015.

[47] A. A. Farag *et al.*, "Chapter 1 Medical Image Registration : Theory , Algorithm , and Case

Studies in Surgical Simulation , Chest Cancer , and Multiple Sclerosis," pp. 1–46.

[48] E. Grimson and R. Kikinis, *Registration for Image-Guided Surgery*. Academic Press, 2000.

[49] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens, "Multimodality image registration by maximization of mutual information," *Med. Imaging, IEEE Trans.*, vol. 16, no. 2, pp. 187–198, Apr. 1997.

[50] J. A. Bærentzen, *Guide to Computational Geometry Processing Foundations, Algorithms, and Methods*. 2012.

[51] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, 2001, pp. 145–152.

[52] A. W. Fitzgibbon, "Robust registration of 2D and 3D point sets," *Image Vis. Comput.*, vol. 21, no. 13, pp. 1145–1153, 2003.

[53] M. W. Jones, J. A. Baerentzen, and M. Sramek, "3D distance fields: a survey of techniques and applications," *Vis. Comput. Graph. IEEE Trans.*, vol. 12, no. 4, pp. 581–599, Jul. 2006.

[54] R. Fabbri, L. Da Costa, J. Torelli, and O. Bruno, "2D Euclidean distance transform algorithms: A comparative survey," *ACM Comput. Surv.*, vol. 40, no. 1, pp. 1–44, Feb. 2008.

[55] C. R. Maurer, R. Qi, and V. Raghavan, "A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions," *Pattern Anal. Mach. Intell. IEEE Trans.*, vol. 25, no. 2, pp. 265–270, Feb. 2003.

[56] M. Mitchell, *An introduction to genetic algorithms*. Cambridge, Mass.: MIT Press, 1996.

[57] A. Fraser, “Simulation of Genetic Systems by Automatic Digital Computers I. Introduction,” *Aust. J. Biol. Sci.*, vol. 10, no. 4, p. 484, 1957.

[58] N. A. Barricelli, “Numerical testing of evolution theories - Part I Theoretical introduction and basic tests,” *Acta Biotheor.*, vol. 16, no. 1–2, pp. 69–98, 1962.

[59] K. P. Misegades, “Optimization of Multi-Element Airfoils,” 1994.

[60] S. N. Sivanandam and S. N. Deepa, *Introduction to genetic algorithms*. 2008.

[61] S. Koziel, *Computational optimization, methods and algorithms*. New York: Springer, 2011.

[62] X. Yang, “Genetic Algorithms,” in *Engineering Optimization*, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2010, pp. 171–180.

[63] S. Kiranyaz, T. Ince, and M. Gabbouj, *Swarm Optimization for Machine Learning and*. 2014.

[64] S. Kiranyaz, *Multidimensional particle swarm optimization for machine learning and pattern recognition*. Berlin ; New York: Springer, 2014.

[65] X. Yang, “Particle Swarm Optimization,” in *Engineering Optimization*, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2010, pp. 203–211.

[66] S. M. Mikki, *Particle swarm optimization a physics-based approach*. San Rafael, Calif.]: Morgan & Claypool Publishers, 2008.

[67] C.-Y. Tsai and I.-W. Kao, “Particle swarm optimization with selective particle regeneration for data clustering,” *Expert Syst. Appl.*, vol. 38, no. 6, pp. 6565–6576, Jun. 2011.

[68] J. C. Bansal, P. K. Singh, M. Saraswat, A. Verma, S. S. Jadon, and A. Abraham, “Inertia Weight strategies in Particle Swarm Optimization,” in *Nature and Biologically Inspired*

Computing (NaBIC), 2011 Third World Congress on, 2011, pp. 633–640.

[69] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, 1995, vol. 4, pp. 1942–1948.

[70] E. Mezura-Montes and C. A. Coello Coello, “Constraint-handling in nature-inspired numerical optimization: Past, present and future,” *Swarm Evol. Comput.*, vol. 1, no. 4, pp. 173–194, 2011.

[71] M. E. Pedersen, “Good Parameters fo Particle Swarm Optimization,” *Luxemb. Hvass Lab.*, 2010.

[72] M. Gendreau, *Handbook of Metaheuristics*, 2nd ed.. 2010.

[73] W. Wang, J. Machta, and H. G. Katzgraber, “Comparing Monte Carlo methods for finding ground states of Ising spin glasses: Population annealing, simulated annealing, and parallel tempering,” *Phys. Rev. E. Stat. Nonlin. Soft Matter Phys.*, vol. 92, no. 1, Jul. 2015.

[74] H.-J. Cho, S.-Y. Oh, and D.-H. Choi, “Population-oriented simulated annealing technique based on local temperature concept,” *Electron. Lett.*, vol. 34, no. 3, pp. 312–313, Feb. 1998.

[75] P. Rossmanith, “Simulated annealing,” in *Algorithms Unplugged*, Springer Berlin Heidelberg, 2011, pp. 393–400.

[76] X. Yang, “Simulated Annealing,” in *Engineering Optimization*, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2010, pp. 181–188.

[77] J. Dréo, *Metaheuristics for Hard Optimization Simulated Annealing, Tabu Search, Evolutionary and Genetic Algorithms, Ant Colonies,… Methods and Case Studies*. 2006.

[78] D. H. Wolpert and W. G. Macready, “No free lunch theorems for optimization,” *Evol. Comput. IEEE Trans.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[79] F. G. Lobo, C. F. Lima, and Z. Michalewicz, *Parameter Setting in Evolutionary Algorithms*, no. 54. 2007.

[80] V. S. Roshni and K. Revathy, “Using mutual information and cross correlation as metrics for registration of images,” *J. Theor. Appl. Inf. Technol.*, vol. 4, pp. 474–481, 2008.