**Automated Weld Path Generation Using Random Sample Consensus and Iterative Closest Point Workpiece Localization with Low-Cost 3D LiDAR**

**Abstract**

Jobs performed by small to medium enterprises (SMEs) are infrequently automated due to high setup costs and lack of technical expertise needed for robot training, however productivity and worker safety can be improved in SMEs with the use automated tooling. In a traditional automated manufacturing environment, tasks such a welding or painting are accomplished through execution of pre-programmed tool motions which rely on the location and orientation of the workpiece to be fixed and known. The lack of this spatial information is typically treated through positioning of the workpiece with respect to the robot arm using jigs or fixtures which are costly in initial setup and not easily modified. Further, the resulting toolpath associated with a desired task is typically defined through manual teaching resulting in a path appropriate for an individual job. For this reason, SMEs requiring variation in part geometry or arrangement are not commonly automated. This work presents a method for automated weld path generation for a 6DOF co-bot arm using random sample consensus (RANSAC) and iterative closest point (ICP) workpiece localization from LiDAR pointclouds. Scans from a low cost 2D LiDAR mounted to the co-bot arm are used to generate 3D pointclouds of the workspace scene with the Robot Operating System (ROS). The Point Cloud Library (PCL) is used to compare the generated pointcloud with a CAD model to produce a rigid transformation to localize the workpiece. The estimated pose of the workpiece with respect to a fixed frame is used offline to generate a weld path as series of tool poses. Two example welding processes in which a cylinder or rectangular tube is joined to a flat plate and two square tubes are joined through weldment are investigated and a physical implementation of the method is demonstrated using a 2D LiDAR mounted to a 6DOF co-bot carrying a MIG welding torch.

**Introduction**

Small to medium enterprises perform manufacturing tasks associated with relatively low part volume and increased variation in assembly geometry as compared to jobs performed in large scale manufacturing environments. This type of manufacturing operation is infrequently automated due to high setup costs; however, productivity and worker safety can be improved in small to medium enterprises with the use of flexible automation such as automated tooling (Co-bot and Lasers too!?!?).

In a traditional automated manufacturing environment, a task such a welding or painting is accomplished through execution of pre-programmed tool motions which rely on the location and orientation of the workpiece to be fixed and known with respect to a global coordinate system. The need for spatial information is typically treated through positioning of the workpiece with respect to the robot arm using jigs or fixtures which are costly in initial setup and are not easily modified. In large scale production environments, this can be accomplished with dedicated infrastructure built into the environment such as moving jigs on assembly lines and other features available in a highly structured environment. Further, the resulting toolpath associated with a desired task is typically defined through manual teaching resulting in a path appropriate for an individual job. For this reason, SMEs requiring lower volume manufacturing with variation in part geometry or arrangement are not commonly automated.

As technology advances, humans and robots must adapt to remain relevant in our respective environments and this can currently be seen in the emergence of the co-bot workcell paradigm. This new model allows for more flexible manufacturing techniques in which robots are able to work with humans on both parallel and sequential tasks [9].

Presented in this paper is a method for automated weld path generation for a 6DOF co-bot arm using random sample consensus (RANSAC) and iterative closest point (ICP) workpiece localization implemented using the Robot Operating System (ROS) and the Point Cloud Library (PCL). Point Cloud Library provides an open-source C++ implementation of several 3D point cloud and image processing algorithms including: object recognition, filtering, feature estimation, surface reconstruction, registration, model fitting, and segmentation [5]. This library is an attractive research tool due to its stability, ability to integrate with ROS, and it includes example code and standard data sets that can be used for comparison repeatability in research.

This work employs a collection of these algorithms to locate, or register, a point cloud representing the workpiece in a point cloud of the working environment collected by a LiDAR scanner located on the robot. Once the known part is located with respect to a fixed frame, an automated weld path generation routine is used to plan a weld tool-path offline. Two example welding applications are presented. In the first, a square tube is joined to a flat plate through weldment. In the second, two square tubes are joined orthogonally to each other to form a tee. Simulations of both applications are investigated and a physical implementation of the method is demonstrated using a 2D LIDAR mounted to a 6DOF co-bot carrying a MIG welding torch which can be seen below in Figure 1.

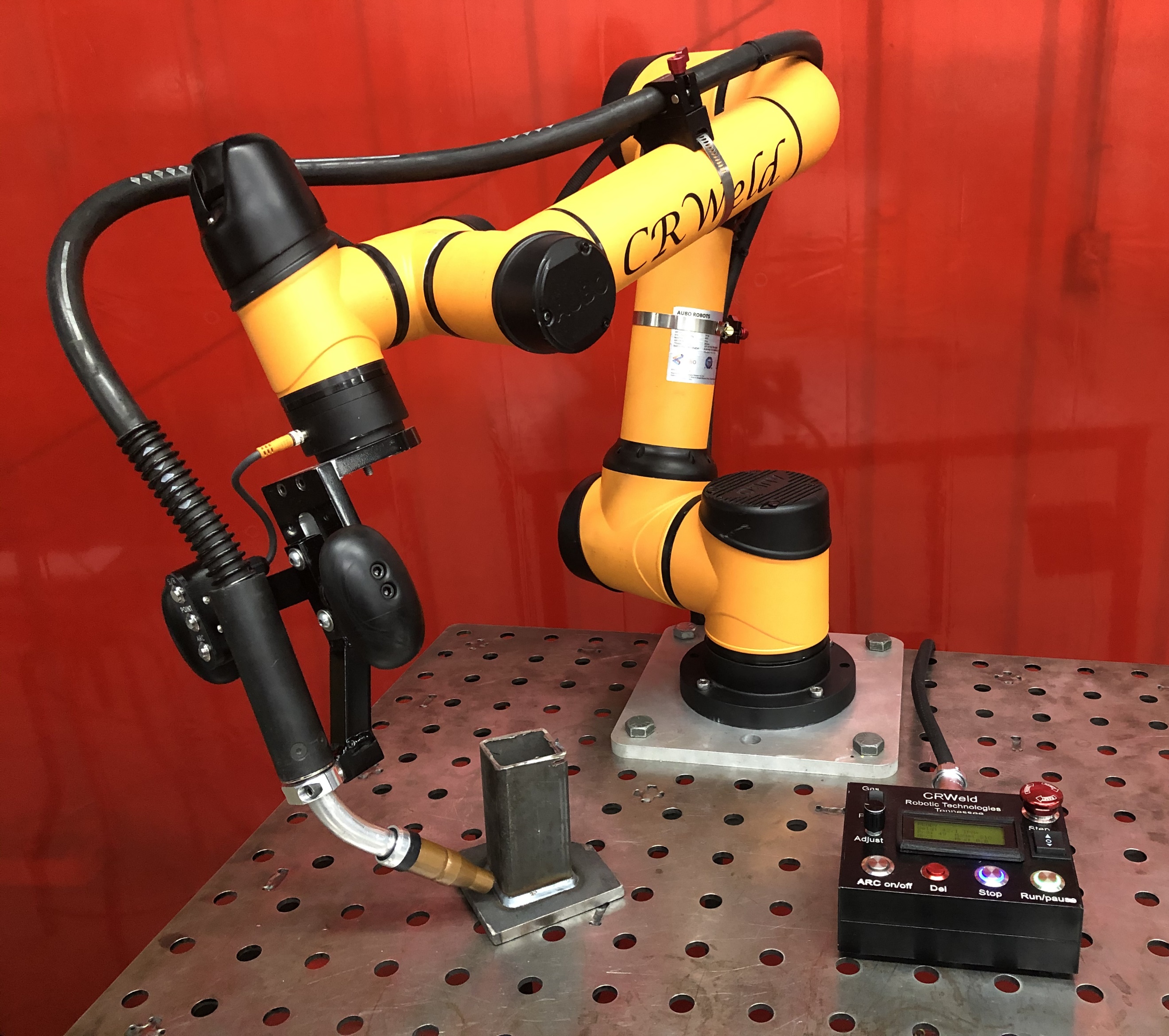


Figure 1

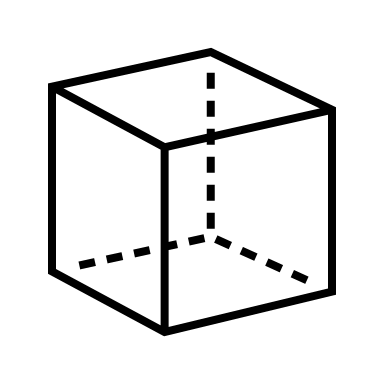
A pointcloud is a list of points in 3D space representing a physical object or collection of objects [6][8], and this data is generated through measurements from a sensing device such as a LiDAR or 3D camera. These geometrical data, or features of a pointcloud contain the locations of the boundaries of a solid object. Features may also include point normals which can be measured or inferred from the feature locations. Non-geometrical data such as color or other surface properties that are independent of the transformations between features are known as descriptors [review of point reg algs]. Descriptors are known to also be used in other registration methods such as feature based registration, which depend on a high level of unique, descriptive features in order to obtain a match between pointclouds [6]. The two types of data contained in a pointcloud are stored separate because they are treated differently in processing algorithms such as segmentation or registration.

Environment sensing devices are frequently used in the mobile robotics industry, and improved sensors and being developed with the increased demand [5] for automation in manufacturing and transportation. Widespread applications and research involving spatial data has led to the development of standard file types and storage containers for efficiently processing and transferring pointclouds [5]. The .pcd file is used for permanent storage and several pointcloud data types exist in common programming languages (C++, Python, MATLAB) for integration with various libraries (PCL, OpenCV) and software frameworks (ROS,). Although, there is an effort to standardize these data types many variants are used. Conversion between different types is common feasible because they are all equivalent representations of a collection of features and descriptors.

Model Preparation

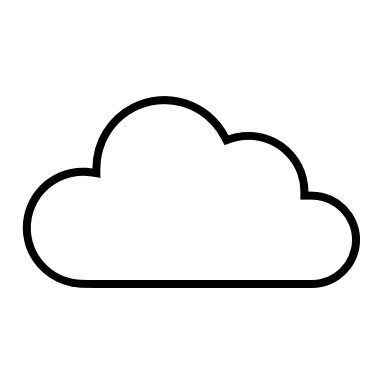
Workpiece CAD

Model Generation



Conversion to

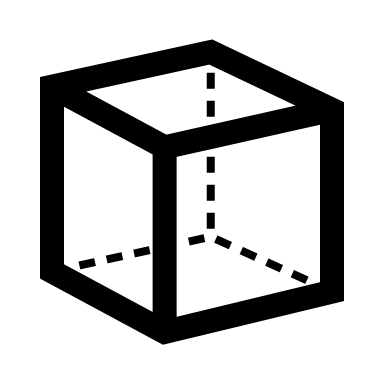
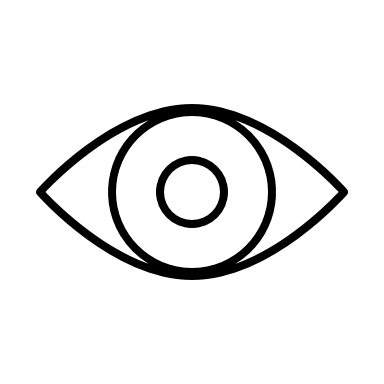
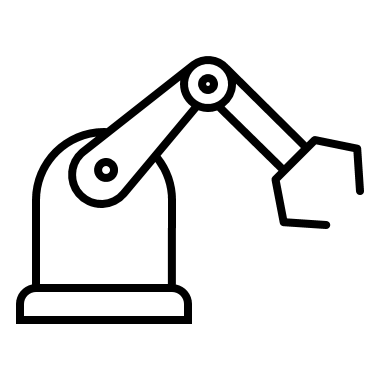
Pointcloud



Workspace Sensing Stage

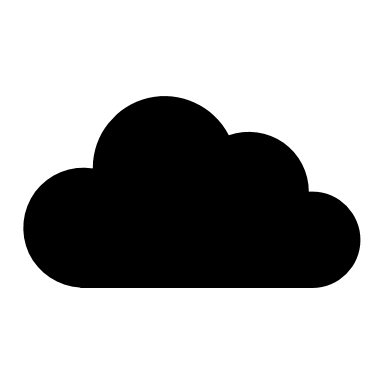
Collection of

2D LiDAR Scans



Conversion to

Pointcloud



Path Generation

Weld Seam

Transformation

Joint Velocity

Profile Generation

Workpiece Localization

Voxel

Filtering

RANSAC

Segmentation

ICP

Registration

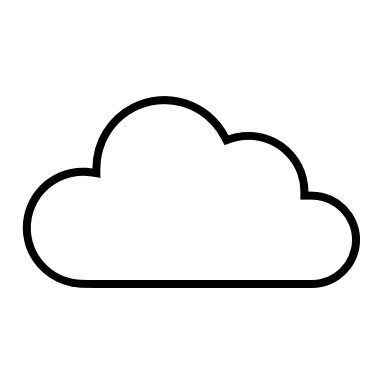
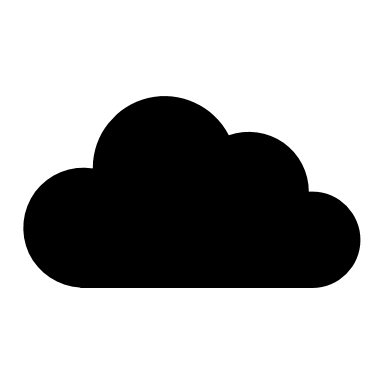
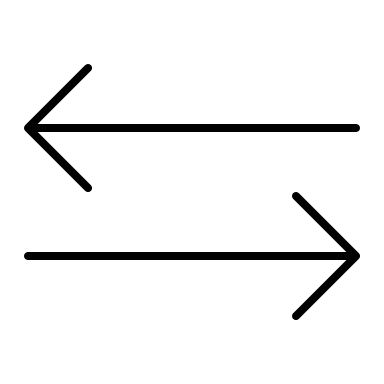
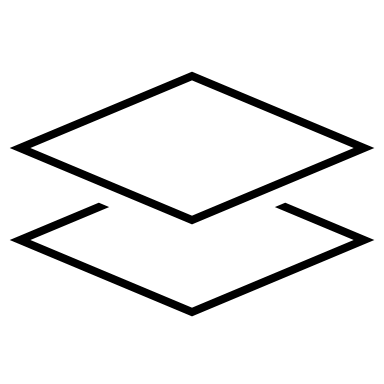


Figure 2 Method for Automated Weld Path Generation

**Approach**

The proposed approach to automated weld path generation shown in **Figure 1** consists of a model data preparation stage, a workspace sensing stage, a workpiece localization stage, followed by an offline robot path generation stage. The resulting path can be used to automate a welding process on the component in the workspace with a 6-DOF co-bot carrying a welding torch.

**Model Preparation Stage**

In the model data preparation stage, the geometry of the workspace and the workpiece is defined based on the prescribed application. An ideal model of the workpiece including the weldment is generated using CAD. Part models are first generated of the individual workpiece components which are then assembled to represent the workpiece. The CAD assembly representing the workpiece is converted into a pointcloud through a uniform sampling technique to be used for workpiece registration. The pointcloud associated with the CAD model is known as the source pointcloud.

A simplified model of the workspace and environment including the welding table and the robot base is also created for simulation purposes, and the environment model is also converted into a pointcloud file. The 3D models are generated using standard CAD software from which they can be exported as .ply files or other standard file formats.

**Workspace Sensing Stage**

Prior to the sensing stage, the workpiece is placed in the robot workspace by the operator in the proper relative orientation to be joined by a weldment. The relative orientation of the parts must match that of the model to an extent and the global location of the workpieces is restricted to the usable workspace of the robot.

In the sensing stage a sweeping motion of the arm is performed, and the workpiece and environment are scanned with the 2D LiDAR mounted on linkS of the robot. Multiple 2D lidar scans are measured along with corresponding sensor poses at (linkS). As the scanning stage continues the data are transformed from the sensor frame linkS to the base frame link0 through the robot forward kinematics and accumulated into a 3D pointcloud with respect to the base frame. This process produces sparse data sets with redundant points. Therefore, the scans are filtered and downsampled [verify – ref. 6 discusses this] to improve results and reduce the resource requirements of storage and processing. The resulting pointcloud contains an image of the workpiece and fixtures as well as the top of the welding table and the background. The pointcloud associated with the LiDAR scan is known as the reference or target cloud.

**Workpiece Localization Stage**

In the workpiece localization stage, the **source pointcloud** derived from the CAD model is compared to the reduced **reference cloud** acquired from lidar in the sensing stage. The relative transformation between clouds is found using the iterative closest point algorithm (ICP). The pose of the two parts can be used to determine the required location of the weld seam in a global sense.

The reference cloud, collected from LiDAR, contains a larger volume of points, but not necessarily more points, than the source cloud. Also, the percentage of the workpiece represented in the LiDAR cloud depends on the sweeping motion used in the scanning stage and the amount of interference caused by the clamps or other obstructions. In the best-case scenario, approximately half of the points associated with the external faces of the workpiece are available in the LiDAR cloud.

The LiDAR cloud is first reduced to the usable workspace of the robot using a 3D bounding box removing points from the surrounding walls and extents of the table. Next, the point cloud is downsampled with a voxel filter [] to ensure uniform density of points in the reference. The remaining image contains points from the workpiece, the clamps holding the workpiece, and the table. The robot arm may also be included in the remaining pointcloud. At this point, RANSAC based segmentation is used to compare geometrical information such as the planar nature of the table or the orthogonality of the workpiece to the LiDAR cloud to separate, or segment, the points associated with the workpiece. The results of a cascaded RANSAC segmentation are stored as the reference pointcloud cloud. Finally, the rigid transformation between the reference and source pointclouds is found with the Iterative Closest Point cloud registration algorithm. This transformation matrix represents the location and orientation of the workpiece with respect to a fixed origin.

**Path Generation Stage**

The geometry and location of the desired weld seam in the workspace is required for offline generation of an appropriate toolpath. This information is determined by measuring the pose of the two individual workpieces with respect to a frame fixed to the robot base. Once this is information is known the appropriate toolpath can be generated. Determination of the poses of the individual workpieces represents to majority of this work and the method is described in detail. (should the detail above go below?)

Different methods have been shown for reducing or down sampling images while still retaining the useful features [6]. (this might go to lit rev.)

**Filtering with and bounding box and voxel**

Can or should we add the math here!?!?

**Segmentation with RANSAC**

The main drawback of ICP is that it may not reach the global minimum of convergence. This can be due to things such as false correspondences, which can cause poor initial alignment and therefore increase the chance of getting stuck in a local minimum. Outlier rejection based on Random Sample Consensus is one of the several methods including distance-based rejection, or duplicate target point rejection, that reduces the number of outliers in pointclouds. Furthermore, RANSAC may also be utilized to provide a good initial guess for the transformation estimated in ICP [6]. This method is a resampling technique that uses the minimum number of data points required to develop the correct pointcloud. The steps involved for using RANSAC are as follows:

1: Randomly select the minimum number of points required to determine the model parameters.

2: Solve for the parameters of the model

3: Determine how many points form the set of all points fit with a predefined tolerance.

4: If the fraction of the number of inliers over the total number of points in the set exceeds a predefined threshold, re-estimate the model parameters using all the identified inliers and terminate.

5: Otherwise, repeat steps 1 through 4 (max N times)

The method of RANSAC used in this paper involves plane-fitting to detect the planes of the pointcloud using the least amount of data as possible while providing an initial solution and then attempting to eliminate the invalid data points [2]. Fischler and Bolles [2] formally state the RANSAC model as follows:

Given a model that requires a minimum of n data points to instantiate its free parameters, and a set of data points P such that the number of points in P is greater than n [#(P) ≥ n], randomly select a subset SI of n data points from P and instantiate the model. Use the instantiated model M1 to determine the subset SI\* of points in P that are within some error tolerance of Ml. The set SI\* is called the consensus set of S1.

If g (SI\*) is greater than some threshold t, which is a function of the estimate of the number of gross errors in P, use SI\* to compute (possibly using least squares) a new model MI \*. If g (SI\*) is less than t, randomly select a new subset S2 and repeat the above process. If, after some predetermined number of trials, no consensus set with t or more members has been found, either solve the model with the largest consensus set found, or terminate in failure.

Discuss cascaded perpendicular plane RANSAC (with dot product ?)

Can or should we add the math here!?!?

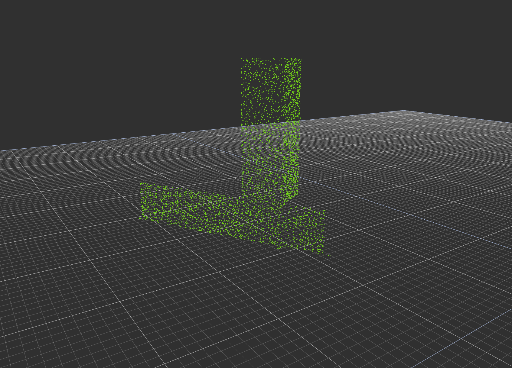
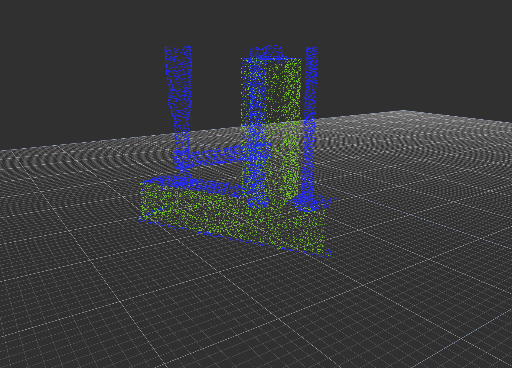
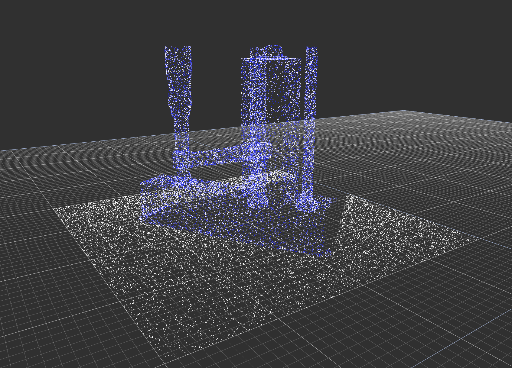
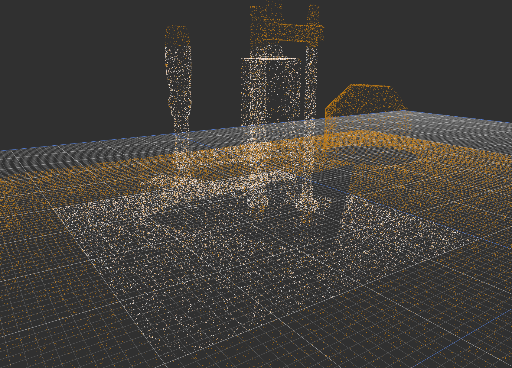


Figure 3a – Before Filtering

Figure 3a – After Filtering

Figure 3a – Before Segmentation

Figure 3a – Before Segmentation

Figure 3: Filtering and RANSAC Segmentation

**Iterative Closest Point**

The objective of the original ICP algorithm is to find a rigid transformation, with which the reference cloud is in the best alignment with the source pointcloud set. This method considers the closest corresponding points between two pointclouds and estimates a transformation to minimize the distance between them using a method of least squares [6]. By iteratively registering the reference cloud with the source cloud and applying rotation matrix R and a translation vector *t,* the source point set is expected to converge as the correspondences achieve alignment*.* This method however, has also been proven to be locally convergent, which means that the algorithm easily fails when the rotation angle between two point-sets is large. For this reason, a good initial transformation is important such that it guarantees that the algorithm converges to the global minimum [7]. For this algorithm, false correspondences are known to inhibit convergence. An implementation to reduce these false correspondences, such as RANSAC, included.

The primary challenge in the localization stage is the selection of point clouds to use as inputs to the ICP algorithm. It has been shown [] and verified in this work that the success of the alignment process is highly dependent on the correspondence between input data sets. The existence of points in one cloud which are not represented in the other cloud can only add cost [] to the alignment process. Further, significant amounts of outliers will cause the alignment to fail or perform poorly. Modifications to ICP and alternative algorithms have shown improved performance [] in the presence of outliers, and methods are available (used in this approach) for automatic rejection of non-corresponding outliers (in let rev?). However, the approach in this work addresses the problem by reducing the reference point cloud to a subset of the LiDAR cloud which contains a portion of the workpiece without the surrounding table or clamps. (this paragraph was in the approach section above)

**Correspondence Matching and Alignment (ICP)**

Let be the CAD pointcloud set for to be aligned with the source cloud set for where , and where each point corresponds to the point with the same index. The mean square objective function (Point-to-Point Error Metric) to be minimized is

If the correct correspondences are known, the correct relative rotation/translation can be calculated in closed form. The close form implementation from PCL consists of using singular value decomposition (SVD). The process of using SVD to begin minimizing distances between corresponding point requires the center of mass be calculate for each point set where, and

For the reference point set and the target point set, respectively. The idea is to subtract the corresponding centers of mass from every point in the two point sets before calculating the transformation. The resulting point sets are given as:

Let the cross-covariance matrix, denote the singular value decomposition (SVD) of W by:

Where D is a diagonal matrix, ,

U, V ,

and are the singular values of W.

If rank(W)=3, the parameters minimizing E(R,t) are unique and given by:

Threshold for E(R,t) can be set for the determination of convergence.

Termination Criteria

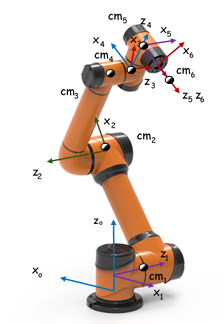
1. Max iterations
2. Absolute transformation threshold
3. Relative transformation threshold
4. Maximum number of similar iterations
5. Relative mean square
6. Absolute mean square error

**Implementation using ROS and PCL**

The robot and sensor are operated simultaneously using ROS, and the resulting pointcloud is saved as a .pcd file which can be processed using the Point Cloud Library or converted to a .ply polygon file.

**ICP – Iterative Closest Point –**

* “Reference” cloud ( Target ) is fixed
* “Source” is transformed to match reference
* The data association (step1) is the most expensive part. It can be done in different ways.
  + Closest Point (kd-Trees)
  + Normal Shooting
  + Closest Compatible Point
  + Projection Based
* The error metric can also be calculated in different ways.
  + Point-Point
  + Point-Plane
* There are also other variants.
  + Point subsets
  + Weighting correspondences
  + Data association
  + Outlier pair rejection
* Here is the basic ICP algorithm:
  1. For each point in the **source**, find the closest point in the **reference**.
  2. Estimate a transformation **T** using root mean squared minimization to best align each **source** point to its **corresponding** reference point found in step 1.
  3. Transform the **source** points using **T** from step 2.
  4. Back to step 1 (re-associate).



Just in case we want to include this (most likely wont)

**Manufacturing Application**

A manufacturing task is considered, in which a weldment is performed on a workpiece resting on a welding table. The workpiece in this task consists of multiple components to be joined through weldment. The relative alignment of the multiple components of the workpiece is assumed to be correct within the physical constraints of the designed part prior to the automated process. In practice, this alignment is set by the operator and secured using clamps or other fixtures.

Variation in surface quality and workpiece dimension and shape are likely present however these are not the focus of this process. The workpiece geometries are generally assumed to match those in the model within a working tolerance. These local model inaccuracies certainly affect the global information produced regarding the geometry and location of the weld, but these affects are minor.

Two example applications are considered. In the first of which, two square tubes are joined perpendicular to one another with a fillet weld along two of the shared edges. In the second application, a square tube is joined to a flat plate with a fillet weld along the shared edge between the two components. In each of these examples, the assembly is temporarily joined together by clamps which will be included in the lidar scan. Prior to the alignment process, these clamps will be removed from the pointcloud data via segmentation with RANSAC.

In example application 1 the workpiece consists of two square tubes to be joined by weldment so that the tubes are perpendicular and form a tee.

In example application 2 the workpiece consists of a square tube to be joined by weldment to a flat plate so that the tube is perpendicular to the plate.

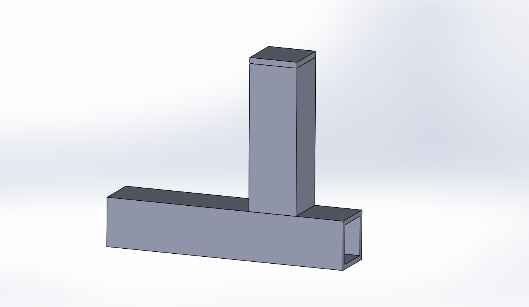
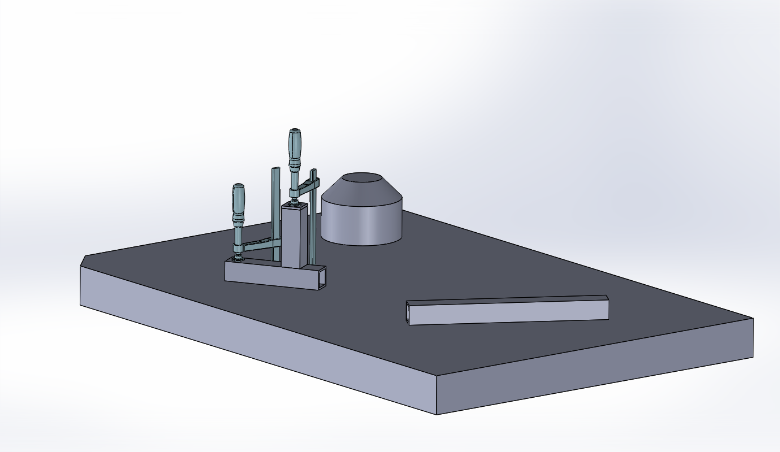


Figure showing workpiece alone

Figure showing workpiece on table

Figure showing segmentation process (above – do we need one for each?)

Table showing expected and measured transformation

**Results**

“PCL incorporates methods for the initial alignment of point clouds using a variety of local shape feature descriptors as well as for refining initial alignments using different variants of the well-known Iterative Closest Point (ICP) algorithm.” [6]

Registraction (ICP or TEASER): The registration stage is required for the alignment of two or more pointcloud sets by finding the relative position and orientation between views in a global coordinate frame, iteratively aligning the point clouds such that processing stages such as segmentation can be applied. In this paper, registration of two pointclouds is carried out by the classical variant of Iterative Closest Point (ICP).

“Due to the non-convexity of the optimization, ICPbased approaches require initialization with a rough initial transformation in order to increase the chance of ending up with a successful alignment. Good initialization also speeds up their convergence.” [6]

General Notes

* This approach provides a benefit over classical online seam tracking in that it does not require measuring the joint itself.
* It is important to distinguish between ‘assembly variations’ and ‘form variations’
* This process could be extended and used to check for unacceptable workpiece geometries and or relative placement.
* The ability to locate objects in the working environment could also be used to assist the operator in workpiece placement as in [] where a laser projector is used to provide a visual guide for manual placement of the workpiece prior to weldment.
* The group at ‘Fraunhofer’ then uses the transformations to update the reference models…
* Rajaraman Dawson-Haggerty Shimada Bourne use a ‘laser projector’ to help the operator in ‘placement’ of the workpeice. ‘robot guided placement procedure’ + ‘3d sensing for preplanned tool paths’

**Bibliography**

Numbered references are tied to notes, please do not change.

[1] Segal, A. V., Haehnel, D., and Thrun, S., 2010, “Generalized-ICP,” Robotics: Science and Systems.

[2] Fischler, M. A., and Bolles, R. C., 1981, “Random Sample Consensus: A Paradigm for Model Fitting With,” Commun. ACM.

[4] Zhang, J., Yao, Y., & Deng, B. (2021). Fast and Robust Iterative Closest Point. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[5] Rusu, R. B., and Cousins, S., 2011, “3D Is Here: Point Cloud Library (PCL),” Proceedings - IEEE International Conference on Robotics and Automation.

[6] Zhao, H., Anwer, N., et. Al., 2016, “Registration with the Point Cloud Library PCL,” IEEE Int. Conf. Intell. Robot. Syst.

[7] Du, S., Xu, Y., Wan, T., Hu, H., Zhang, S., Xu, G., and Zhang, X., 2017, “Robust Iterative Closest Point Algorithm Based on Global Reference Point for Rotation Invariant Registration,” PLoS One.

[8] Schwarz, S., Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P. A., Cohen, R. A., Krivokuca, M., Lasserre, S., Li, Z., Llach, J., Mammou, K., Mekuria, R., Nakagami, O., Siahaan, E., Tabatabai, A., Tourapis, A. M., and Zakharchenko, V., 2019, “Emerging MPEG Standards for Point Cloud Compression,” IEEE J. Emerg. Sel. Top. Circuits Syst.

[9] Galin, R., Meshcheryakov, R., Kamesheva, S., and Samoshina, A., 2020, “Cobots and the Benefits of Their Implementation in Intelligent Manufacturing,” *IOP Conference Series: Materials Science and Engineering*.

Detection of Assembly Variations for Automatic Program Adaptation in Robotic Welding Systems

Alexander Kuss, Ulrich Schneider, Thomas Dietz Fraunhofer

Automated Planning of Robotic MAG Welding Based on Adaptive Gap Model

Alexander Kuss, Ulrich Schneider, Thomas Dietz Fraunhofer

Generalized-ICP

Aleksandr V. Segal, Dirk Haehnel, Sebastian Thrun

Automated workpiece localization for robotic welding

Rajaraman Dawson-Haggerty Shimada Bourne

A method for registration of 3D shapes

Paul Besl, Neil McKay