**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.

In a traditional automated manufacturing environment, a task such a welding or painting is accomplished through the 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.

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.

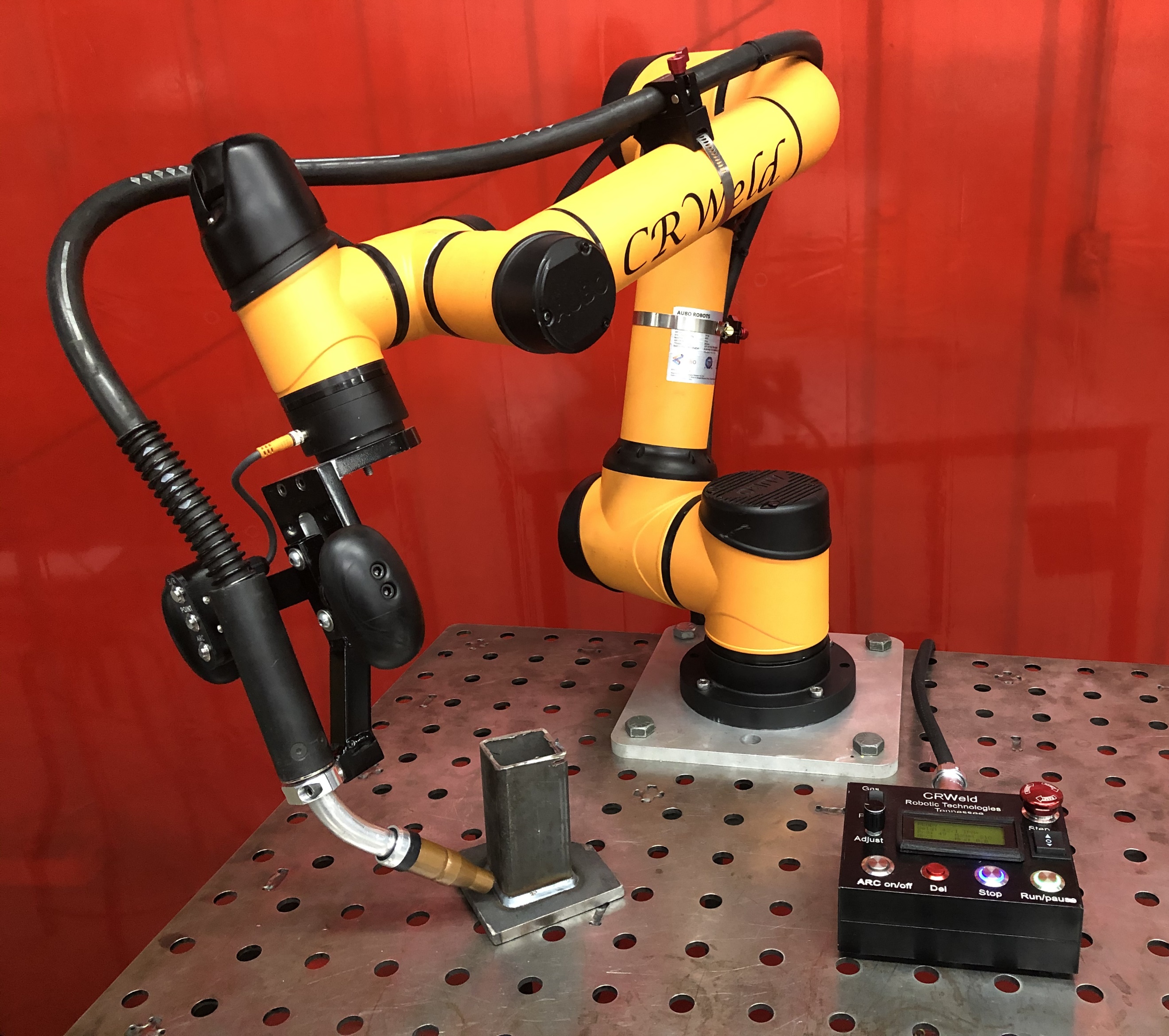


Figure 1

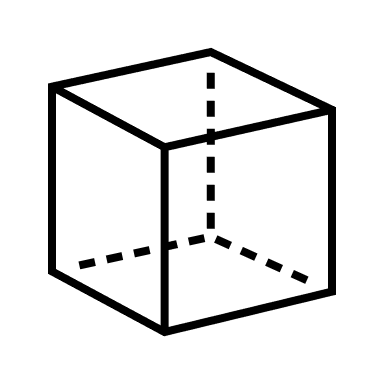
Environment sensing devices which generate 3D points are frequently used in the mobile robotics industry, and improved sensors are being developed with the increased demand [5] for automation in manufacturing and transportation. A pointcloud is a list of points in 3D space representing a physical object or collection of objects [6][8], and this data is generated through measurements from a sensing device such as a LiDAR or 3D camera. Widespread applications and research involving spatial data has led to the development of standard file types, storage containers, and libraries for efficiently processing of pointclouds [5]. Common programming languages (C++, Python, MATLAB) support integration of pointcloud data with various libraries (PCL, OpenCV) and software frameworks (ROS).

The geometrical data, or features, stored in 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 [13]. Descriptors are also used in feature based registration methods, which primarily depend on unique, descriptive features in order to obtain a match between pointclouds [6]. These two types of data contained in a pointcloud are stored separate because they are different in nature and are processed differently in algorithms such as segmentation or registration.

Model Preparation

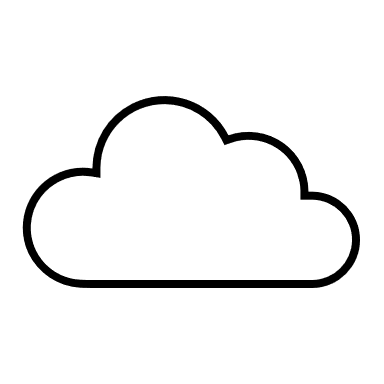
Workpiece CAD

Model Generation



Conversion to

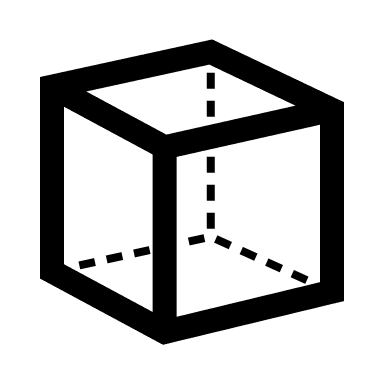
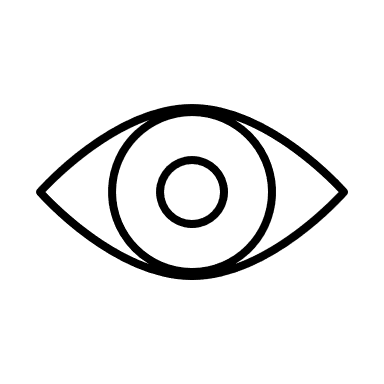
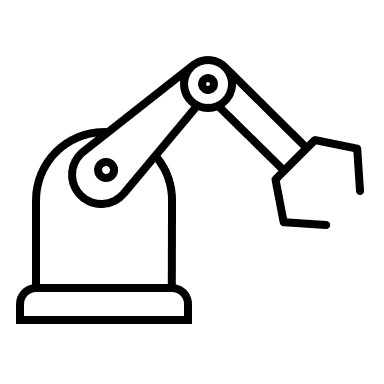
Pointcloud



Workspace Sensing Stage

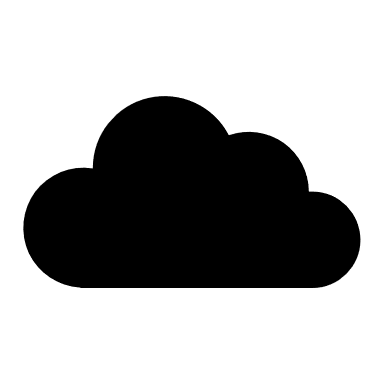
Collection of

2D LiDAR Scans



Conversion to

Pointcloud



Path Generation

Weld Seam

Transformation

Joint Velocity

Profile Generation

Workpiece Localization

Voxel

Filtering

RANSAC

Segmentation

ICP

Registration

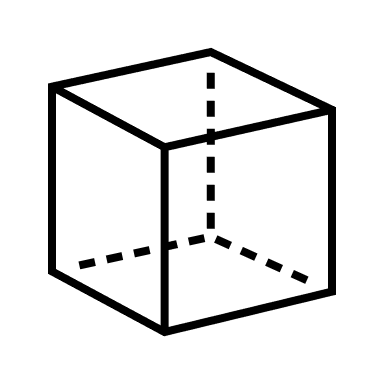
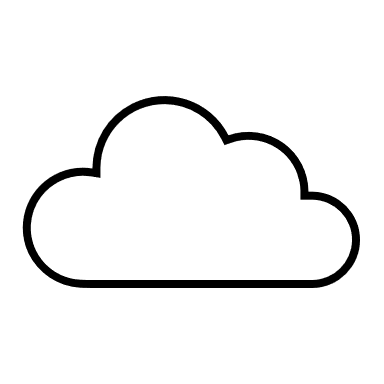
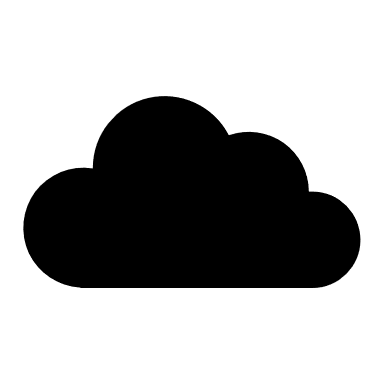
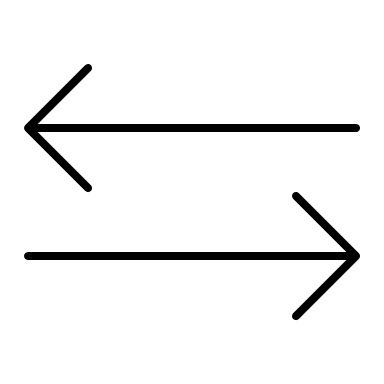
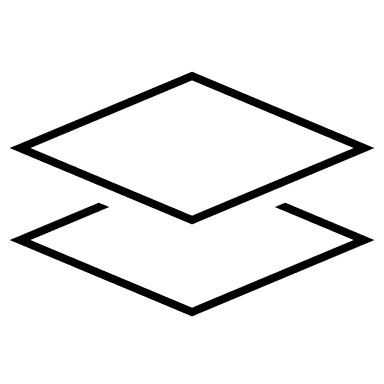


Figure 2 Method for Automated Weld Path Generation (NEEDS ARROWS!)

**Overview of 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.

**Description of Algorithms**

Filtering w/ Bounding Box and Voxel

Segmentation w/ RANSAC

Correspondence Matching with ICP

**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 (with methods within PCL) 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. The sensing stage along with the methods of filtering and downsmpling can be seen below in figure 3.



Fig 3b - After Voxel Filter

Fig 3c - After Bounding Box

Fig 3a - Before Filtering

Fig 3 - Filtering

**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 [15] 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 (ICP) cloud registration algorithm. This transformation matrix represents the location and orientation of the workpiece with respect to a fixed origin.

**Segmentation with RANSAC**

One of the key issues with standard ICP is that it may not reach the global minimum of convergence which can be due to things such as false correspondences. These false correspondences cause poor initial alignment and therefore increase the chance of getting stuck in a local minimum [7]. Outlier rejection based on Random Sample Consensus (RANSAC) 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 estimation in ICP [6]. The RANSAC algorithm, in general, is a resampling technique that uses the minimum number of data points required to correctly detect and segment pointcloud shapes [10][11].

Although variations have been developed, the RANSAC algorithm can generally be thought of as an iterative two-part process. The first part involves a hypothesis in which the first minimal sample set (MSSs) is selected at random from the input dataset (source pointcloud) which is then used as the basis for computing the model parameters. Following the selection of the minimal sample set, RANSAC checks which elements of the instantiated model are consistent with the entire dataset – these elements are referred to as the consensus set (CS) [10]. During each iteration, the instantiated model’s elements are comparted to the original dataset. If the new MSS increases the number of correct correspondences compared to the best CS, it will then overwrite the previous CS. The algorithm continuously iterates and is terminated only when the CS reaches a certain threshold. When this threshold has been reached, the instantiated model based on model parameters of the newest MSS, the MSS is said to be consistent with the entire dataset.

Figure 4a – Input Cloud: Table, Workpiece, and Clamps

Figure 4 – Segmentation to Remove Table



Figure 4b – Plane Inliers: Table

Figure 4c – Plane Outliers: Workpiece and Clamps

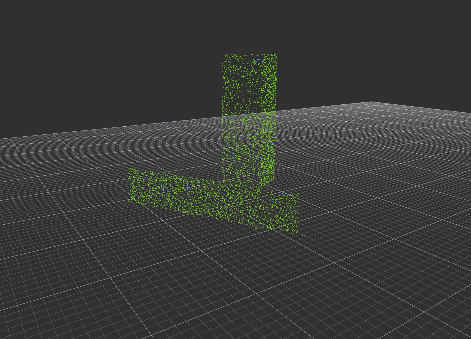
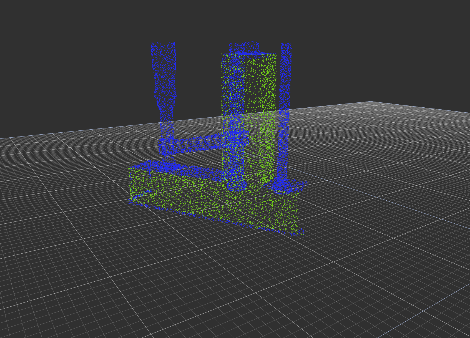


Figure 5 – Segmentation to Remove Clamps

Figure 5a – Input Cloud: Workpiece, and Clamps

Figure 4b – Plane Inliers: Table

Figure 4c – Plane Outliers: Clamps

**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 [7]. For this reason, a good initial transformation is important such that it guarantees that the algorithm converges to the global minimum [7]. In this work, false correspondences from the environment scans are known to inhibit convergence. An implementation to reduce these false correspondences, such as RANSAC, is included.

The primary challenge in the localization stage is the selection of point clouds to use as inputs to the ICP algorithm [4],[7]. 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 [6] 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 [4] [6] 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.

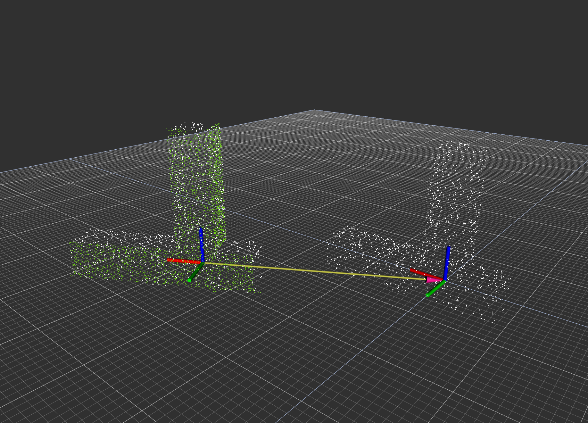
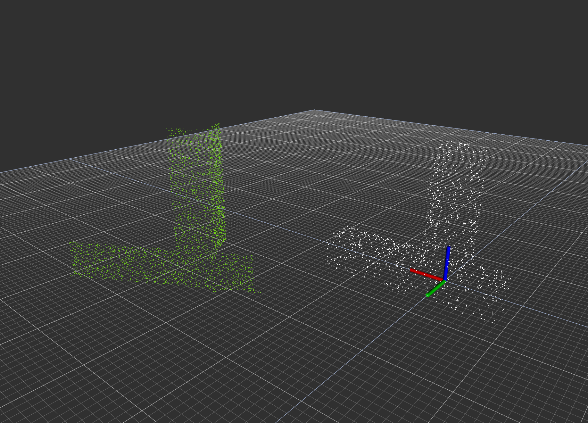


Figure 6 – Workpiece Localization with ICP

**Correspondence Matching and Alignment (ICP)**

**(the notation with R here is confusing, careful R is not Rp)**

The ICP method as described in [12] is summarized here. Let be the reference point cloud set for representing the workpiece to be aligned with the source point cloud set for where , and where each point corresponds to the point with the same index (*i=k*). The reference set will come from a CAD model while the source cloud set will be collected from a data scan of the real environment. The mean square objective function (Point-to-Point Error Metric) to be minimized is,

. (1)

where is an array that projects onto and that translates onto . If the correct correspondences are known, the correct relative rotation/translation can be calculated in closed form. A closed form implementation that can be found in the PCL library (cite) is briefly described. The center of mass for the reference point cloud set and source point cloud set respectively is calculated for each set as,

and . (2)

The reference point cloud set and source point cloud set are shifted by their center mass such that they are distributed around zero as,

(3)

A cross-covariance matrix, is defined as

. (4)

A singular value decomposition (SVD) of is given as

(5)

where D is a diagonal matrix containing the singular values, , ordered such that and are the left and right singular vectors of . When the , the rotation and translation minimizing are unique and given by:

(6)

and

(7)

In practice, the correspondences assumed in Eq. (1) are not truly known. Therefore, this process is performed iteratively, by assuming a correspondence between the reference and source point cloud set based on a minimum distance between points. The source is corrected and the process repeats until convergence of the source and reference point cloud set occurs as given in the error .

**Path Generation Stage**

**Transformation of Seam Points**

Consider a pointcloud set for located along the weld seam as described by the example application and defined in the CAD model. The weld seam exists at the shared location of the connecting faces of the two parts which make up the workpiece. The pointcloud set is a subset of the pointcloud set which represents the workpiece.

The pointcloud set and are defined with respect to a frame {{part}} located on the workpiece.

To complete the desired operation the robot must carry the torch along the weld seam. The path generation stage requires the robot tool path points to be described with respect to the fixed frame of the robot. The pointcloud sets and can be projected into the fixed frame of the robot with the rigid transformation for resulting from the ICP routine described previously as it is the best available approximation of the workpiece pose.

The set of points for is found by applying the rotation and translation separately as shown. = fn(R, t)

for

**Joint Velocity Profile Generation**

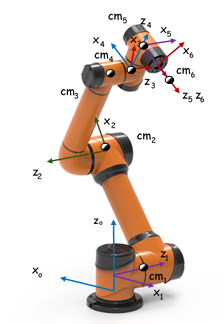
**Implementation using ROS and PCL**

This research has been implemented in ROS on Ubuntu Linux which provides a multi-threaded and distributed software framework for robotics applications.

The combination of 2D LiDAR scans into 3D pointclouds was done using a custom ROS package *scan2cloud* that is based on *ROS* *laser\_geometry*. The rigid transformation from the sensor frame to the base of the robot is programmed with *ROS tf* so that individual 2D scans collected using *ROS rplidar* can be processed and saved as a .pcd file with respect to a global origin.

The .pcd file is used for permanent storage and several pointcloud data types

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.



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

I think we should take the picture of the CR weld robot a make a similar picture.

**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.

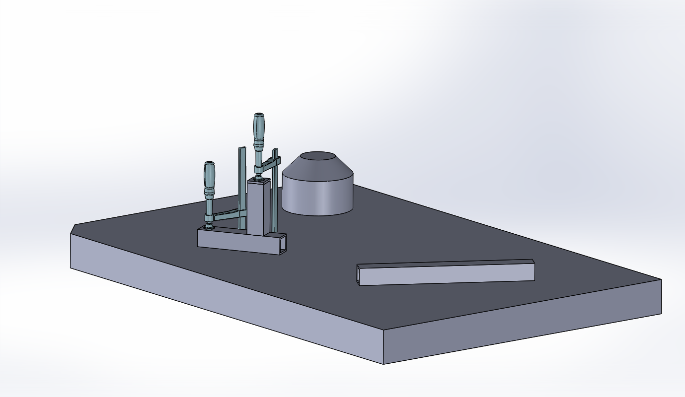


Figure 7 – Example Application A



Figure 8 – Example Application B

Figure showing workpiece alone

Figure showing workpiece on table

**Simulation Results**

Example application A and B were performed for testing and validation of the proposed approach using synthetic data generated from CAD models of the workspace including the table and clamps as shown in the figure above. The models of the scene were converted to pointclouds using the uniform sampling process described for the conversion of the source cloud.

**Experimental Results**

Example Application A was performed with an Aubo i5 on a welding table with a RP-LiDAR A2 mounted to the end effector for generating 3D pointsclouds. In the scanning stage the arm performed a sweeping motion while collecting a point cloud containing approximately half of the workpiece, a large portion of the table, and a small portion of the arm itself. The recorded points are restricted to those that fall in a selected region of the usable workspace of the robot. This collection process produces redundant data points representing the objects and the cloud data can become large. The approach presented is applied to the raw projected LiDAR points as described with respect to the base frame of the robot.

Experimental Calibration

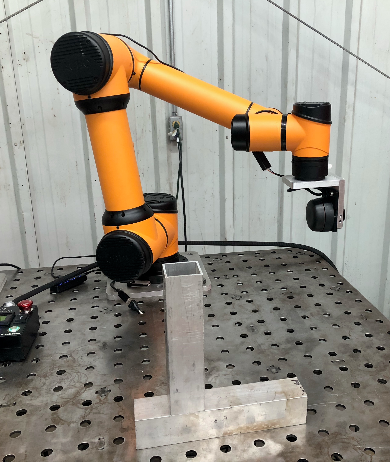


Figure - Experimental Setup for Application A

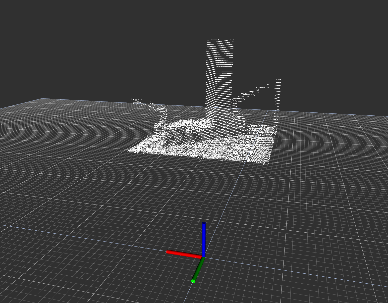
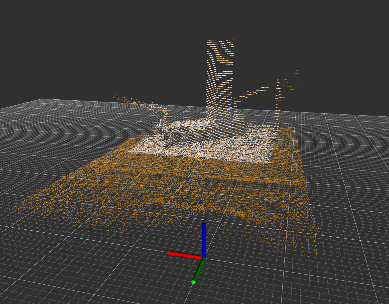


Figure – Filtering

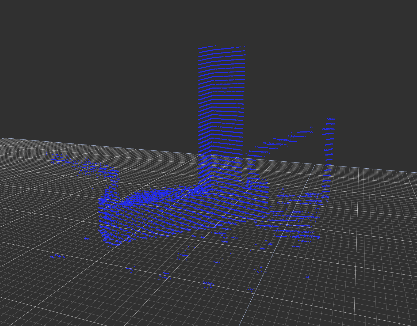
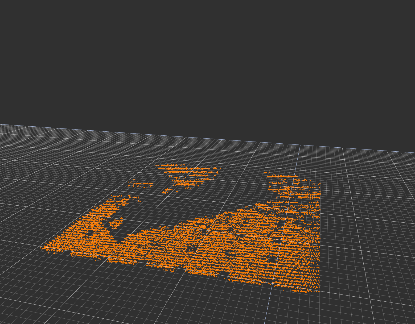


Figure – Segmentation of Table

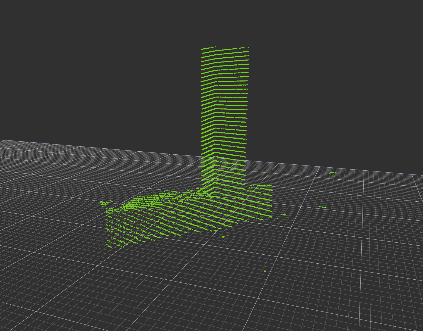
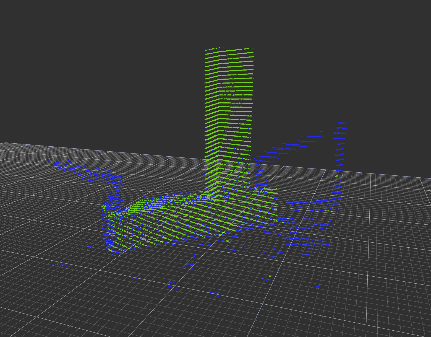


Figure – Segmentation of Table

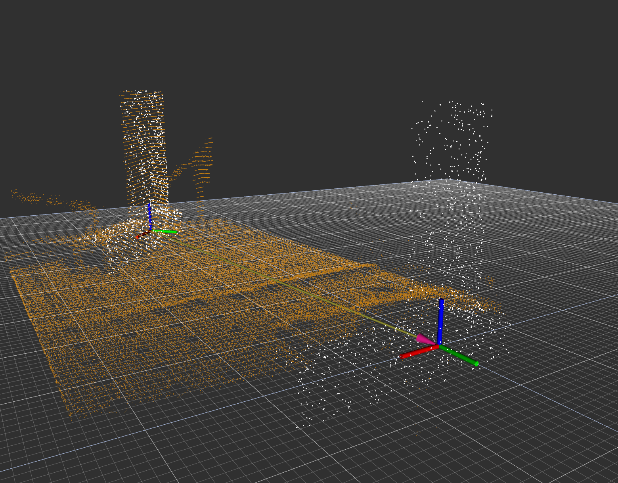
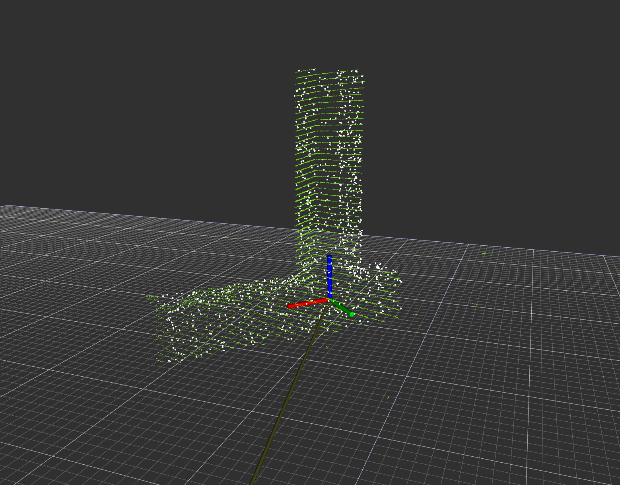


Figure – Workpiece Localization with ICP for Example A

Experiment Results

3D LiDAR Calibration Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | X (m) | Y (m) | Z (m) |  |  |
| Expected Translation | 0.0 | -0.60960 | 0.02540 |  |  |
| Measured Translation | 0.00893 | -0.60874 | 0.02003 |  |  |
| Difference | -0.00893 | -0.00086 | 0.00537 |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Roll (rad) | Pitch (rad) | Yaw (rad) |  |  |
| Expected Rotation | 0.0 | 0.0 | 0.0 |  |  |
| Measured Rotation | -0.00493 | -0.00071 | 0.019938 |  |  |
| Difference | 0.00493 | 0.00071 | -0.00071 |  |  |

Example Application A

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | X (m) | Y (m) | Z (m) |  |  |
| Expected Translation | 0.10160 | -0.60960 | 0.02540 |  |  |
| Measured Translation | 0.10840 | -0.61096 | 0.02034 |  |  |
| Difference | -0.00680 | 0.00136 | 0.00506 |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Roll (rad) | Pitch (rad) | Yaw (rad) |  |  |
| Expected Rotation | 0.0 | 0.0 | 0.785 |  |  |
| Measured Rotation | -0.00457 | 0.02186 | 0.79461 |  |  |
| Difference | 0.00457 | -0.02186 | -0.00921293 |  |  |

Simulation Results – Example Application A

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | X | Y | Z |  |  |
| Expected Translation |  |  |  |  |  |
| Measured Translation |  |  |  |  |  |
| Difference |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Roll | Pitch | Yaw |  |  |
| Expected Rotation |  |  |  |  |  |
| Measured Rotation |  |  |  |  |  |
| Difference |  |  |  |  |  |

Simulation Results – Example Application B

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | X | Y | Z |  |  |
| Expected Translation |  |  |  |  |  |
| Measured Translation |  |  |  |  |  |
| Difference |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Roll | Pitch | Yaw |  |  |
| Expected Rotation |  |  |  |  |  |
| Measured Rotation |  |  |  |  |  |
| Difference |  |  |  |  |  |

“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*.

[10] Zuliani, M., 2008, “RANSAC for Dummies,” Citeseer.

[11] Li, L., Yang, F., Zhu, H., Li, D., Li, Y., and Tang, L., 2017, “An Improved RANSAC for 3D Point Cloud Plane Segmentation Based on Normal Distribution Transformation Cells,” Remote Sens.

[12] Besl, P. J., and McKay, N. D., 1992, “A Method for Registration of 3-D Shapes,” IEEE Trans. Pattern Anal. Mach. Intell.

[13] Wang, F., Liang, C., Ru, C., and Cheng, H., 2019, “An Improved Point Cloud Descriptor for Vision Based Robotic Grasping System,” Sensors (Switzerland).

[14] Moreno, C., and Li, M., 2016, “A Comparative Study of Filtering Methods for Point Clouds in Real-Time Video Streaming,” *Lecture Notes in Engineering and Computer Science*.

[15] Zygmunt, M., 2013, “The Testing of PCL: An Open-Source Library for Point Cloud Processing,” Geomatics, Landmanagement Landsc.

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