Towards Robotic Metal Scrap Cutting: A Novel Workflow and Pipeline for Cutting Path Generation

James Akl[1], Fadi Alladkani[1], Berk Calli[1]

***Abstract*— We propose a novel framework for robotic metal scrap cutting in unstructured scrap yards. In this framework the robots and workers collaborate: the worker marks the cutting locations on the scrap metal with spray paint and the robot then generates the cutting trajectories. This leverages worker expertise, while deferring the dull, dirty, dangerous aspects to the robot. For the robot, this requires a 3-D exploration and curve reconstruction stage for path generation. We use a non-uniform rational basis spline (NURBS) model and a topological skeletonization method for path generation, and implement and compare these methods via simulations. These simulations employ a realistic sensor noise model and highly-detailed 3-D scans of complex, real-life scrap pieces. Real-robot experiments with three different shapes are also provided.**

I. Introduction

Recycling decommissioned large metal structures, e.g. oil rigs and ships, or equipment, e.g. large engines requires them to be dismantled, moved to a metal scrap yard and cut into small workable chunks. Currently, the cutting operation is conducted manually by skilled workers using a gas torch (s. Fig. 1). This manual operation is slow and laborious. Due to the variety of the scrap pieces and the difficult and unstructured nature of this process, automating this task has many challenges: For each piece, the cutting locations and trajectories need to be determined, the cutting parameters need to be identified, and the cut needs to be executed at certain torch speed and poses. All these operation variables are successfully determined and applied by skilled workers, but very challenging to translate into robot task parameters.

We present a human-robot collaboration workflow that combines the strengths of skilled workers and robots. Workers draw the desired cutting paths on scrap pieces using spray paint, and the robot inspects the drawing with its camera to generate cutting trajectories based on object properties. This workflow has many advantages:

- Cutting locations are determined by the worker, significantly reducing cognitive effort on the robot.
- The robot does not scan the whole object; it works with local information on the cutting locations.
- The labor intensive work is minimized for the worker, since the slow, dangerous and tedious cutting operation is done by the robot.

Fig. 1. Manual metal cutting in an European Metal Recycling Group scrap yard. The picture is from the company website [1].

In this workflow, the robot's role becomes:

1) Scan an arbitrary drawn curve on the scrap surface,
2) Generate a 3-D cutting path from these scans,
3) Generate a cutting trajectory within the constraints,
4) Execute the cutting operation while monitoring it.

This paper focuses on steps 1 and 2. We use point cloud information to track and acquire the drawn curve via two distinct curve fitting methods (namely a B-spline method [2] and a skeletonization algorithm [3]) within an active vision framework. To the best of our knowledge, our framework and proposed pipeline present a novel solution to the metal scrap cutting problem.

II. Related Work

We first present the existing methods in robotic cutting and welding applications. Next, we review spatial curve reconstruction techniques relevant to our extraction step.

A. Robotic Cutting and Welding Applications

The problem of robotic cutting varies greatly across application domains and depends on the specific tooling

[1]Authors are with Robotics Engineering Program, Worcester Polytechnic Institute, 85 Prescott Street, Worcester, MA-01605, USA. {jgakl; fmalladkani; bcalli}@wpi.edu

This work was supported in part by funding from European Metal Recycling Ltd. via the NSF I/UCRC ROSE-HUB under grant 1939061.

used [4], [5], which in turn defines the cutting properties (quality, speed, compatible materials). There is abundant work on automated laser cutting; e.g., analytical methods that assume object knowledge [6] and [7], as well as path planning in structured settings [8]. These methods rely on prior knowledge, i.e. a full object model, and do not directly translate to gas torch cutting. Our framework does not rely on prior knowledge of object geometry.

To the best of our knowledge the only robotic gas cutting work is presented by [9], which develops a vision-less reactive control architecture for identifying poor strips in sensitive yet constrained surroundings. This method is designed for a specific object shape and application. We are not aware of any work in the literature on robotic methods for gas cutting that is general enough to be applied to metal scrap recycling.

A close application domain to metal cutting is welding. In this domain, the robots rely on weld seam tracking; and seam identification such as the developed in [10]. However, this method requires full view of thin-enough line. Other methods borrow ideas from active vision [11]. These algorithms enable precise following of a weld seam, but are unsuitable for scrap metal cutting. This is because the drawings encountered are noisier and thicker, and the objects explored are much larger. Other robust methods [12] are noise-resistant, but they require prior knowledge of the welding seam.

B. Spatial Curve Reconstruction

Spatial line reconstruction is researched extensively [13], [14], especially on unorganized data [15]. Such methods have special interest to our problem, which operates on 3-D point clouds. Common reconstruction methods rely on the optimization formulation of B-Splines, Non-uniform rational basis spline (NURBS), or Bézier curves. For example, there are iterative methods for surface fitting [16] in the presence of obstacles, as well as reconstruction [17] of self-intersecting lines. More complicated shapes have been reconstructed by partitioning them [18] for further fitting using multiple curves. An alternative approach is using principal curves [19] that are based on principal component analysis. These resemble [20] the typical skeletonization algorithms but the latter are instead used to represent the connectedness of N-dimensional binary shapes and easily represent branching paths [21]. Skeletonization is traditionally implemented in thinning algorithms [22] for 2-D images, but extend to 3-D [23].

Each fitting approach has special advantages and limitations for spatial curve reconstruction. We adopt the NURBS and skeletonization approaches, and evaluate their strengths and weaknesses in our application.

III. Overview of the Workflow

We received domain knowledge from our corporate sponsor European Metal Recycling (EMR) Group [1]. We assume that the target metal scrap pieces are located at a scrap yard (similar to Fig. 1). These pieces vary greatly in shape, and due to practical limitations it is infeasible to acquire full scans of their shapes.

The workers can easily identify the metal types and the cutting locations via quick inspection. Then, they cut the parts using a gas torch. Although the cutting locations on a scrap piece can be determined in a few minutes, it requires domain-specific expertise of the skilled worker and a global shape knowledge of the target object. On the other hand, the cutting operation itself is repetitive, but quite laborious and time-consuming.

As a solution to this problem, we propose a human-robot collaboration framework that takes advantage of human expertise and minimizes the dull, dirty and dangerous aspects of the manual work. Determining the cutting locations requires worker's intuition, and is difficult to automate. Therefore, the worker's role is to mark the desired cutting locations with a distinctive color spray paint. After this step, the robot autonomously detects the 3-D curve on the object surface, reconstructs it, generates a cutting path and executes the cut.

While there are many exciting research questions within this pipeline, this paper focuses on the curve acquisition and path generation steps. We do not assume that the robot has a full view of the curve, nor that the extremities (start, end) of the curve are in sight. The drawn curve is reconstructed from partial observations in an automated process, akin to a simplified, surface-based active vision problem. We propose a 3-D curve reconstruction pipeline, while using spatial curve fitting techniques to obtain a next-view for iterative scanning. The acquired curve segments are then registered, and the full cutting path is obtained by generating collision free set-points at a desired cutting distance, where the cutting torch is always perpendicular to the object surface. This path can then be converted to a cutting trajectory by imposing tool speed constraints based on the scrap piece's properties. Here, we assume that the scrap properties are entered by the worker, and the robot can use a look-up table for determining the cutting speed.

IV. Implementation

We present our implementation as in Fig. 2. The worker provides the desired cutting path in the form of a drawing, whose specific color is known to the robot and is distinguishable from its surroundings. Next, the robot starts scanning this drawing segment by segment, and stitches them together with the following steps.

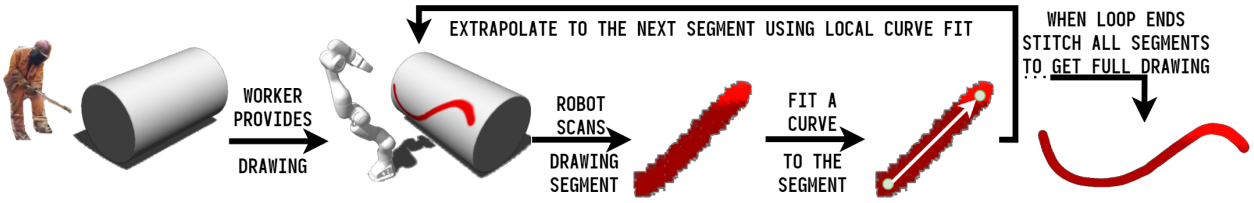

Fig. 2. Visual overview of Algorithm 1 resulting in a fully-stitched point cloud of the drawing.

A. Point Cloud Preprocessing and Curve Fitting

This section assumes an eye-in-hand RGB-D setup. With the initial segment of the drawing in view, we apply color filtering using HSV color space to obtain the points that belong to the drawn curve. We, then, apply statistical outlier removal to eliminate noisy points that do not belong to the dominant curve. To reduce excess growth of cloud data, the point clouds are downsampled.

The next step is to fit a curve to the acquired data. We use two different methods for curve fitting (and later compare their performance in Section V), namely the NURBS method and the a skeletonization algorithm. The NURBS curves have desirable properties for this pipeline. They are memory-efficient and have a configurable smoothness parameter. Fitting a Bézier curve to a point cloud (obtained from the RGB-D camera) is a non-trivial optimization problem, however available implementations such as the PCL Library [2] provide NURBS fitting support. The implementation was modified to allow for open curves. The NURBS curve is especially robust to noise and gaps in the point clouds. Its functionality extends to discontinuous mesh shapes.

The skeletonization method's primary goal is identifying the set of points equidistant to at least two boundary points, called *medial axis* of a 2D image, or of a 3-D set of voxels. The skeleton obtained is a voxel-wide representation of a mesh's connectedness; useful for working with unstructured point clouds from the RGB-D camera. The 3-D skeletonization implementation as used in the pipeline is described in [3]. The skeletonization component gradually thins an image (removing boundary voxels) until it a voxel-wide line is left. One limitation of skeletonization comes from converting raw point clouds into binary voxel occupancy grids whose resolution directly correlates with the medial axis accuracy. Finer leaf size leads to better accuracy, but with a robustness tradeoff, as sparsely-sampled point clouds can lead to fragmented occupancy (falsely disconnected voxels), thus skeletonization fails. This trade off is demonstrated in Fig. 3, where increase in leaf size causes a larger average error in curve fitting.

The fitted curve is used to estimate the curve direction and determine the next view for curve acquisition.

B. Extrapolation & Exploration using Curve Data

This section describes the core component of the exploration loop as outlined in Algorithm 1.

Algorithm 1: Exploration Procedure

```
Result: Fully-stitched point cloud of drawing
end_points_found ← 0;
k ← 0;
while end_points_found < 2 do
    cloud ← obtain_point_cloud();
    full_cloud ← concat(full_cloud, cloud);
    curve ← fit_curve(cloud);
    if size_diff(k+1, k) <= thresh
     then
        ++end_points_found;
        return_to_initial_point();
    else
        next_pos ← extrapol(curve, cloud);
    end
    ++k;
end
```

The robot must determine its next end-effector position to reveal the rest of the drawn line. The next viewpoint is generated by extrapolating the fitted curve. A running log of previously visited coordinates avoids revisiting an explored direction. Due to a lack of prior knowledge of the target shape, active collision avoidance is needed using feedback from the RGB-D sensor. For efficient collision avoidance, `octomap` [24] is used.

Exploration is done by sampling two points near the end of the line's representation. For skeletonization, those points are the last two voxels on the edge. The NURBS curve is instead extrapolated by sampling two points near the parametrized curve's edge. The robot moves along the extrapolated chunk by a constrained distance close enough to the edge to avoid overshoot and missing unscanned chunks. A conservative estimate is to move towards the fitted curve's edge. Although this slows down scanning, it outputs a more robust line.

There remains three (orientation) plus one (distance-to-surface) DOFs to constrain. For robot orientation, we first constrain the end-effector's direction normally to

the surface, which maximizes scan quality and motion safety. The end-effector's rotation about the normal axis, is kept free to search for collision-free configurations.

The last DOF, the distance-to-surface, can be determined based on the camera noise model and required performance. To reduce noise, most cameras should be placed close to the object surface. An iterative solution is to start from the camera's minimum distance and increment until a collision-free pose is found with a viable trajectory. However, moving the robot closest to the surface forces it to move slower along the drawing, as vision is now constrained to a smaller view of the drawing, and thus more steps are required for the same distance. This tradeoff is a user-defined parameter for the pipeline regarding scanning speed and accuracy.

C. Stopping Criterion and Global Curve Fit

As in Algorithm 1, the agent must detect both end points to terminate properly. A single endpoint is determined by examining the amount of new information per step. The agent keeps track of the previous fully-stitched cloud's size. After frame k is processed, the agent registers the new cloud and obtains a new fully-registered cloud. The stopping criteria compares the incremented size of the fully-stitched clouds within a certain threshold: $\text{size}(k) - \text{size}(k-1) \leq \delta_{\text{size}}$.

Once this condition is met, the robot backtracks to unexplored parts of the drawing, and re-runs the loop. Once the condition is met again, the loop terminates and the drawing is considered fully-explored. The robot now has a fully-registered cloud of the entire filtered drawing, on which it performs global path generation and normal estimation. The latter improves accuracy by providing more information to compute the normal planes.

After the fully-stitched cloud is available, we curve-fit the filtered data using either aforementioned technique. With skeletonization, we use the smallest possible leaf size when discretizing the grid, pruning smaller branches and ensuring all line points are fully-connected throughout the skeleton. Alternatively, the NURBS method generates a global fit while minimizing its error by tuning the control points, degree, or smoothness constraints.

V. Simulations & Experiments

We assess our pipeline with realistic simulations and provide proof of concept real-robot implementations.

A. Simulations

We developed a simulation environment using the Robot Operating System (ROS) [25] and the Gazebo simulator [26] to test our path generation pipeline. Four real scrap pieces from our industrial partner's scrap

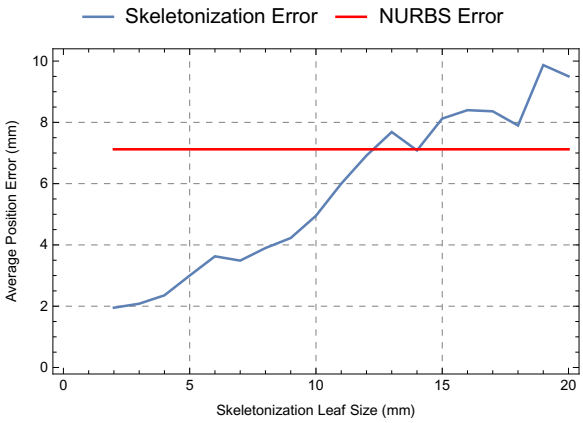

Fig. 3. Effect of leaf size on average skeletonization error versus average NURBS error.

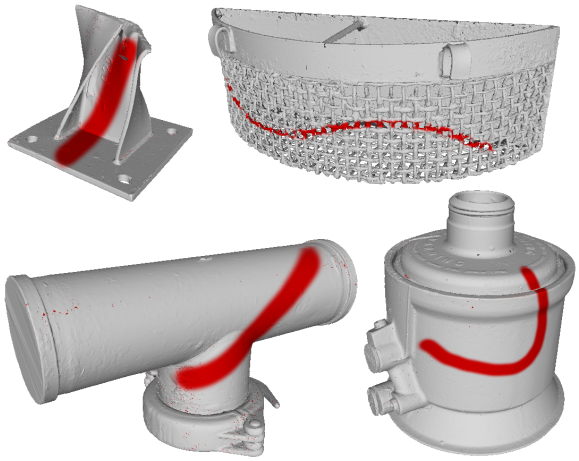

Fig. 4. High-precision scans of real-life metal scrap pieces: (1,1) I-Beam, (1,2) Basket, (2,1) T-Piece, (2,2) Air Chamber.

yard were scanned to produce high-quality 3-D models (Fig. 4). These models are used as test objects to examine the robustness and accuracy of the methods in various challenging geometries, such as mesh-like gaps, sharp turns and intricate surface changes. We also added the cylindrical object shown in Fig. 2 as a simple shape example. We simulated spray-painted drawings on these models by loading them in Blender, drawing a parametric reference curve on the object, and then generating the drawing from that curve. These curves are used as baselines for our error calculations. We emphasize that the object models are not used in the execution of the path generation pipeline; it does not use any *a priori* shape information. A virtual Intel RealSense D435i RGB-D camera is attached at the end effector of the robot, and a probabilistic noise model was developed based on information from the camera datasheet [27].

Two runs are performed on each scrap piece; one

with the NURBS method and one with the skeletonization algorithm (10 simulations total). The smoothness parameter (weight) of the NURBS algorithm is set to 1.0. The skeletonization leaf size is set to 5 mm for all simulations, except for the basket object (the reasons are discussed later in this section). The path errors are presented in Table I. The mean path errors are calculated along the drawn curve by sampling and averaging the difference between the ground truth of the drawing and the resulting path. In addition, the sampled error values along the basket object is given in Fig. 5.

Overall, the accuracy of both curve fitting methods is similar and for the simple cylinder shape. More complex shapes present various challenges, and significant differences are obtained for each curve fitting method. For the air chamber, while the average errors of both algorithms are similar, the maximum error of the NURBS algorithm is significantly higher due to the sharp curvature change towards the top groove, which violates NURBS smoothness assumptions. Skeletonization, on the other hand, relies on thinning and shows robustness to such sharp corners, having no assumptions on the input's smoothness. Similarly, the joint location of the T-piece causes the NURBS method to have a much higher maximum error than the skeletonization algorithm. This difference is very pronounced for the I-beam, where the curvature change is close to 90 degrees at the joint location. The basket object, however, showcases the limitations of skeletonization algorithm. This object features a mesh surface, where the drawing is not necessarily connected in all the locations, thus requiring the methods to "bridge" the gaps. Due to this feature, the skeletonization algorithm with a 5 mm leaf size fails due to unconnected voxels, and stops the execution before completing the whole curve. When we increase the leaf size to 2 cm, the algorithm successfully tracks the whole curve, but results in significantly larger errors compared to the NURBS algorithm (please refer to the leaf size discussion in Section IV-A and Fig. 3).

B. Experiments

Proof of concept experiments are conducted with a Franka Emika Panda robot equipped with an Intel RealSense D435i RGB-D camera attached to its end-effector. Three artificial shapes and their "cutting locations" were prepared as in Fig. 6.

Applying the same pipeline with two different curve fitting methods, the robot was able to successfully reconstruct the curves for the planar box and the cylindrical object. Its execution ended prematurely for both algorithms at the sharp corner of the slanted object, having only one side reconstructed (Fig. 7). Both algorithms failed to cope with self occlusions of the object. The curve exploration algorithm thinks that the line is fully-scanned and finishes the execution. Such limitations may be addressed via more elaborate active vision methods that take into account not just the curve fit, but also information entropy when determining the next view.

Our multimedia attachment presents a realtime execution of the pipeline with the cylindrical object.

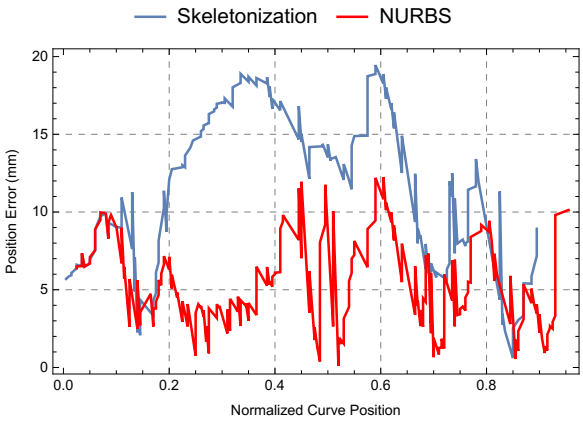

Fig. 5. Comparison of the position error across the normalized curve per method for the basket.

TABLE I

POSITION ERROR FOR ALL OBJECTS IN SIMULATIONS. ALL VALUES ARE IN MILLIMETERS.

Object	Method	Mean	SD	Min	Max
Air Chamber	NURBS	3.61	1.73	2.2	13.37
	Skelet.	2.96	1.66	0.42	7.48
Basket	NURBS	5.43	2.62	0.11	12.24
	Skelet.	11.58	4.71	0.61	19.45
Cylinder	NURBS	3.01	1.89	0.30	7.03
	Skelet.	3.10	1.35	0.45	6.46
I-Beam	NURBS	12.9	10.66	3.40	37.23
	Skelet.	6.44	3.11	0.57	13.09
T-Piece	NURBS	7.43	4.27	1.08	17.59
	Skelet.	4.04	1.69	0.81	7.71

VI. CONCLUSION

In this paper, we proposed a novel workflow for the solution of the robotic metal scrap cutting problem. This workflow leverages human expertise and transfers the

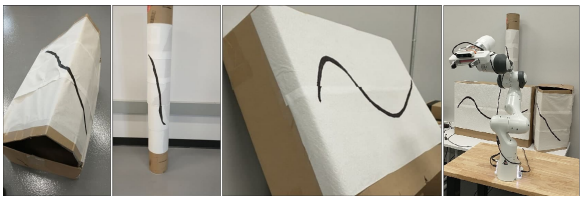

Fig. 6. Experimental setup and the three model scrap pieces. From left to right: slanted object, cylinderical object, planar box.

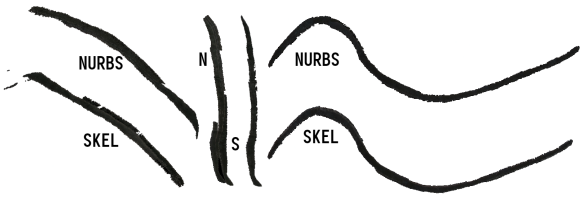

Fig. 7. Drawings extracted for each method per object in the experiment. From left to right: Sharp Surface, Cylinder, Planar Box.

operation's laborious aspects to the robot. We proposed and implemented a pipeline to acquire the cutting locations and generate a cutting path autonomously, without relying on prior object models. We compared two curve fitting methods in our active exploration scheme, and demonstrated their advantages and drawbacks.

VII. Acknowledgements

We thank Roger Morton from EMR Group for his domain knowledge and guidelines in scrap cutting.

References

[1] "EMR Global," https://uk.emrgroup.com/.

[2] Thomas Mörwald, "Fitting a 3D B-Spline curve to point-clouds using point-distance-minimization and optionally asymmetric-distance-minimization," https://github.com/PointCloudLibrary/pcl/blob/master/surface/include/pcl/surface/on_nurbs/fitting_curve_pdm.h, PCL Library.

[3] T. Lee, R. Kashyap, and C. Chu, "Building skeleton models via 3-d medial surface axis thinning algorithms," *CVGIP: Graphical Models and Image Processing*, vol. 56, no. 6, pp. 462–478, 1994. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S104996528471042X

[4] R. Bogue, "Cutting robots: A review of technologies and applications," *Industrial Robot*, vol. 35, no. 5, pp. 390–396, 2008.

[5] J. Bickendorf, "New applications of cutting and welding robots with automatic offline-programming," in *Proceedings of ISR 2016: 47st International Symposium on Robotics*, 2016, pp. 1–6.

[6] J. Cai, Z. Ding, Y. Zhang, and M. Liu, "Trajectory planning and simulation for intersecting line cutting of the industry robot," *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, vol. 2015-March, no. March, pp. 63–68, 2015.

[7] Z. Xia, K. Zhou, X. Li, and X. Cui, "A method of robot laser cutting for small holes," in *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 2016, pp. 1887–1891.

[8] J. Hatwig, P. Minnerup, M. F. Zaeh, and G. Reinhart, "An automated path planning system for a robot with a laser scanner for remote laser cutting and welding," in *2012 IEEE International Conference on Mechatronics and Automation*, 2012, pp. 1323–1328.

[9] K. S. Yoo, H. R. Ryu, and C. Choi, "Control architecture design for an gas cutting robot," *Proceedings of the 2008 2nd International Conference on Future Generation Communication and Networking, FGCN 2008*, vol. 4, pp. 66–71, 2008.

[10] M. Dinham and G. Fang, "Autonomous weld seam identification and localisation using eye-in-hand stereo vision for robotic arc welding," *Robotics and Computer-Integrated Manufacturing*, vol. 29, no. 5, pp. 288–301, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584513000057

[11] J. Muhammad, H. Altun, and E. Abo-Serie, "Welding seam profiling techniques based on active vision sensing for intelligent robotic welding," *The International Journal of Advanced Manufacturing Technology*, vol. 88, no. 1-4, pp. 127–145, 2017.

[12] Z. Ye, G. Fang, S. Chen, and M. Dinham, "A robust algorithm for weld seam extraction based on prior knowledge of weld seam," *Sensor Review*, 2013.

[13] W. Zheng, P. Bo, Y. Liu, and W. Wang, "Fast b-spline curve fitting by l-bfgs," *Computer Aided Geometric Design*, vol. 29, no. 7, pp. 448–462, 2012, geometric Modeling and Processing 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167839612000301

[14] W. Wang, H. Pottmann, and Y. Liu, "Fitting b-spline curves to point clouds by curvature-based squared distance minimization," *ACM Trans. Graph.*, vol. 25, no. 2, p. 214–238, Apr. 2006. [Online]. Available: https://doi.org/10.1145/1138450.1138453

[15] L. Fang and D. C. Gossard, "Multidimensional curve fitting to unorganized data points by nonlinear minimization," *Computer-Aided Design*, vol. 27, no. 1, pp. 48–58, 1995. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0010448595907522

[16] S. Flöry, "Fitting curves and surfaces to point clouds in the presence of obstacles," *Computer Aided Geometric Design*, vol. 26, no. 2, pp. 192–202, 2009.

[17] K. Khanna and N. Rajpal, "Reconstruction of curves from point clouds using fuzzy logic and ant colony optimization," *Neurocomputing*, vol. 161, pp. 72–80, 2015.

[18] D. Zhu, P. Bo, Y. Zhou, C. Zhang, and K. Wang, "Fitting multiple curves to point clouds with complicated topological structures," in *2013 International Conference on Computer-Aided Design and Computer Graphics*, 2013, pp. 60–67.

[19] T. Hastie and W. Stuetzle, "Principal curves," *Journal of the American Statistical Association*, vol. 84, no. 406, pp. 502–516, 1989.

[20] B. Kegl and A. Krzyzak, "Piecewise linear skeletonization using principal curves," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 59–74, 2002.

[21] D. Jin, K. S. Iyer, C. Chen, E. A. Hoffman, and P. K. Saha, "A robust and efficient curve skeletonization algorithm for tree-like objects using minimum cost paths," *Pattern recognition letters*, vol. 76, pp. 32–40, 2016.

[22] K. Saeed, M. Tabedzki, M. Rybnik, and M. Adamski, "K3m: A universal algorithm for image skeletonization and a review of thinning techniques," *International Journal of Applied Mathematics and Computer Science*, vol. 20, no. 2, pp. 317–335, 2010.

[23] S. Tran and L. Shih, "Efficient 3d binary image skeletonization," in *2005 IEEE Computational Systems Bioinformatics Conference-Workshops (CSBW'05)*. IEEE, 2005, pp. 364–372.

[24] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, no. 3, pp. 189–206, 2013.

[25] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[26] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, 2004, pp. 2149–2154 vol.3.

[27] "Intel RealSense D400 Series Product Family," https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf, Accessed: 2021-03-15.