International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Integrated Robotic Path Planning for Complex Geometry Industrial Spraying Operations

Pranjal Garg[1], Piyush Kumar Jain[2], Harimohan Soni[3]

[1]*Mechanical Engineering Department, Bansal Institute of Science and Technology, Bhopal (M.P.), India.*
[2]*Mechanical Engineering Department, Bansal Institute of Science and Technology, Bhopal (M.P.), India.*
[3]*Mechanical Engineering Department, Bansal Institute of Science and Technology, Bhopal (M.P.), India.*

Abstract - The implementation of automated robotic solutions for complex tasks currently faces a few major hurdles. For instance, lack of effective sensing and task variability – especially in high-mix/low-volume processes – creates too much uncertainty to reliably hard-code a robotic work cell. Current collaborative frameworks generally focus on integrating the sensing required for physical collaborative implementation. While this paradigm has proven effective for mitigating uncertainty by mixing human cognitive function and fine motor skills with robotic strength and repeatability, there are many instances where physical interaction is impractical but human reasoning and task knowledge is still needed. The proposed framework consists of key modules such as a path planner, path simulator, and result simulator. An integrated user interface facilitates the operator to interact with these modules and edit the path plan before ultimately approving the task for automatic execution by a manipulator that need not be collaborative. Application of the collaborative framework is illustrated for a pressure washing task in a remanufacturing environment that requires one-off path planning for each part. The framework can also be applied to various other tasks, such as spray-painting, sandblasting, deburring, grinding, and shot peening. Specifically, automated path planning for industrial spraying operations offers the potential to automate surface preparation and coating in such environments. Autonomous spray path planners in literature have been limited to generally continuous and convex surfaces, which is not true of most real parts. There is a need for planners that consistently handle concavities and discontinuities, such as sharp corners, holes, protrusions or other surface abnormalities when building a path. The path planner uses a slicing-based method to generate path trajectories. It identifies and quantifies the importance of concavities and surface abnormalities and whether they should be considered in the path plan by comparing the true part geometry to the convex hull path. If necessary, the path is then adapted by adjusting the movement speed or offset distance at individual points along the path. Which adaptive method is more effective, and the trade-offs associated with adapting the path are also considered in the development of the path planner.

***Key Words*:** spraying operation,reliability,robotics.

1.INTRODUCTION

When the idea for this research was first conceived, the task was to design an automated pressure washing work cell capable of handling a large majority of the parts present at one of the Army's rework and rebuild depots. This presented a particular challenge due to the vast differences in part size and geometry that needed to be cleaned on a daily basis. The facility is responsible for cleaning pallets of smaller parts, as well as full tank bodies. Further compounding the problem was the realization that there was almost no way of consistently identify the exact geometry of a part. Whether that be from lack of existing data, easy to miss differences between parts or the fact that the process is still manual and most parts are still custom made, especially for rework and rebuild facilities like this one.

In the past, these challenges have deterred most facilities from attempting to automate the process and choosing to do it manually instead. While this is certainly the most common method, the physical toll these jobs take on the people doing them is undeniable and until recently the technology needed to automate these tasks has been relatively inaccessible, whether that be due to cost or the sheer difficulty of the task being automated. Specifically, full coverage path planning is one of the most difficult tasks to automate reliably and economically. Not to say it isn't doable, but most cases where these tasks are automated don't need to build a new path plan for each part. They are typically used to repetitively do the same set of preprogrammed parts over and over again.

Given the knowledge that human operators are very good at making the judgement calls of what needs to really be cleaned and that a generally good path plan can be built on the fly by an automated system, the task became how to blend a human's cognitive function with the precision and endurance of an automated robotic system, an idea pioneered by the collaborative robotics community. Taking this idea a step further, adaptive path planning was embraced to create a better path than the generally good path created by the naïve path planner. Due to the growing scope of this project, it was broken down into two separate problems. The first being what does the collaborative system look like from the initial input to user verification and ultimately process execution, and the second being what does an adaptive path planner for pressure washing look like.

1.1 GENERAL FRAMEWORK DESIGN

This section discusses the modules required within the framework as well as reveals the key attributes for the success of each module. Figure 1 illustrates how the individual modules interact with each other, the external components of the system, and the human operator. From a high level, the system takes the provided 3D data and initial user input as parameters into the path planner to generate a path. The path is then sent to the path analyzer before the simulation displays the original 3D input, the path, and the analysis. From here, the operator can decide to accept the proposed task as is or make adjustments. If necessary, the adjustments are made by the

path modifier module and then sent back through analysis before the operator has the opportunity to make another decision. Upon approval, the path is passed to the robot and the task is completed. However, if the operator notices that there are still unsatisfactory spots, the process can be started again with either the full part or a smaller section being passed to the path planning module.

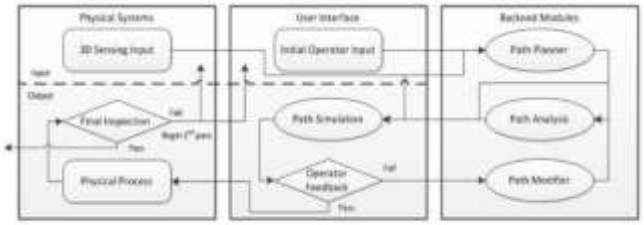

Figure 1: System Framework Design

2. Methodology

This approach involves taking some initial 3D data, given as an STL file in this case because the algorithm requires normal vector, and building a convex hull around it to eliminate the collision and accessibility issues created by non-continuous and concave surfaces. However, given a method for determining normal vectors and building a tessellated mesh from a point cloud, any number of 3D data gathering methods could be used. At this point, the mesh is converted into a point cloud where the centroid of each facet is linked to the normal vector of that facet. The path is then built based on the convex hull using a slicing based method that relies on the following input parameters: a rotation axis and the degrees of rotation, which serve to modify the slicing direction, as it is unlikely that the part will actually be moved if the scanner is calibrated and registered correctly; and a slice thickness, an offset distance, and an overlap percentage, which serve to quantify how much work will be applied to the part. Once the path has been built, the points from the original mesh, now represented as a point cloud, are mapped to specific segments of the path. In some cases, a point can be mapped to multiple segments.

The individual segments of the path are then adapted based on the points mapped to each segment. The adaptive phase considers the true geometry and will either modify the path to be closer to the part or it will slow the end effector down to facilitate more cleaning power applied to a particular area. From an experimental point of view, the system was used to test two factors; the type of adaptive algorithm used and the statistical aggregation method used within the adaptive algorithm. The user can choose between a time adapted path and a distance adapted path with the option to also use both or neither adaptive algorithm and a choice of mean, mode, min and max is provided for the aggregation method. The process is illustrated in Figure 3.

Outside factors aside and without any adaptive methods, slice thickness, wA, and offset width, wO, are the two parameters that most significantly affect the algorithm's performance, but they can be derived from a variety of other measurements based on the specific process. For the proposed spraying process, the slice thickness can be calculated using simple right-angle trigonometry from the angle of the sprayer, θ, and the end effectors distance from the part surface, referred to as the offset distance, dO. The offset width is calculated as a percentage of the slice thickness based on the provided overlap percentage, σ. Figure 4 illustrates these calculations as well as how the slices can overlap on the surface of the part itself. Base velocity and sprayer intensity can also be provided, but they only serve to augment the results equally and have no effect on the distribution of work done.

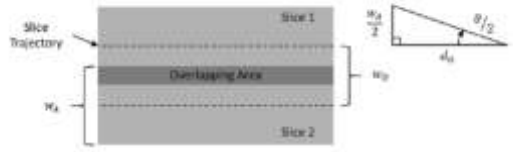

Figure 4: Overlapping Slice Paths and Input Parameter Derivation

2.1 Tool Path Trajectory

In order to better understand the path generation process, the form of the finished trajectory is presented here first. After the algorithm has run its course, the initial 3D data has been taken and converted into a final toolpath with seven data points for each position in the trajectory. The first three points represent the X, Y and Z positions of the end effector relative to the center of the part, in this case the origin. These values would need to be translated for any real implementation, but that translation is entirely dependent on the specific implementation. The second three points represent the orientation of the end effector as a directional vector from the end effector to the part, which is found by taking the inverse of the corresponding facets normal vector. This vector can be converted to any other representation as needed. While this vector does not fully define the pose of the end effector, as the rotation about the tool is not addressed, it does allow motion planners a free parameter to find a kinematic solution. This can change based on the process, but assuming a conical spray pattern, this free parameter does not have any effect on the process. The seventh point represents the timestamp of that particular point, which can be used by a robot to generate a trajectory. Each path point is defined as,

$$p = [x, y, z] \tag{1}$$

and each orientation is defined as,

$$r = [x_r, y_r, z_r] \tag{2}$$

and the finished trajectory is defined as,

$$G = \{ [p_k, r_k, t_k] \mid \forall k = (0, \ldots, \beta) \} \tag{3}$$

where β is the number of observations in G, P, R and T, P is the ordered set of all path points p, R is the ordered set of all orientation vectors r, and T is the ordered set of all time values t.

2.2. Slicing the Part

The algorithm builds the path in a similar fashion to additive manufacturing processes. The main difference is that when an additive manufacturing process slices a part, it is slicing to build a solid piece, which needs many thin slices with an interior raster pattern, whereas this process is slicing for

exterior surface coverage, which uses a few thick slices without the interior raster pattern, just the exterior perimeter. Throughout this process, all of the slicing and path building action are taken with regards to the convex hull of the part. The original part data is used to inform the adaptive algorithms and all analysis is done using the original part as well. The appropriate number of slices for the convex hull is defined as,

$$m=\lceil(Z\max-Z\min)wO\rceil \quad (4)$$

where $Zmax$ and $Zmin$ are the extreme values in the Z axis of the part and wO is the offset width. The offset width, which represents the distance between each slice, is redefined so that the slices are equally spaced along the entire part, which can be defined as,

$$wO=(Z\max-Z\min)\,m \quad (5)$$

This ensures that there is total coverage of the part and can be modified to create overlap on the edges of the part if necessary. The height hs of each slice s is defined as,

$$hs=Z\max-wO/2-swO \quad \forall\, s=\{0,\cdots,m\} \quad (6)$$

where s is the slice index and $wO/2$ represents the offset needed to shift the slice from the edge to the center of that slice. For each slice, s, the process described in the following sections is repeated until all slices have been planned for and the slices are combined into one complete path. Additionally, the path is also checked for consistent segment sizes as illustrated in Figure 6. Given the unknown nature of the 3D data and especially with the way convex hulls are built, where flat surfaces are represented by the smallest number of facets required, some facets can be quite large. This results in path segments, which will later be redefined as bins, that may capture too many data points to be effectively adapted. However, before dealing with too large path segments, too small segments are combined into one segment that can be broken up later if necessary. Let $\Phi s=\{[\boldsymbol{p}k,\hat{\boldsymbol{n}}k]\}$ denote the ordered set of all points and their corresponding normal vectors in the path for slice s. Let $\Xi k=\{\varphi j\}$ denote the set of consecutive adjacent points that are colinear to point $\boldsymbol{p}k$, which includes φk. These colinear points can be defined in the same manner as the duplicate points in Equation 13.

2.3. Full Path Concatenation

When adding each slice's path to the master path, there are a few extra pieces needed to ensure a quality path. In this instance, a raster pattern is created by alternating the order with which the points from the individual slices are added to the path, as illustrated in Figure 6. This is useful for robots with limited reach, so that the path does not require the robot to continuously circle the part. To achieve this, the algorithm has ordered the path from the lowest angular polar coordinate to the highest and the points are added in this order for the first slice, and then from highest to lowest in the next and so on.

Another method not covered above is the transition from slice to slice. To ensure that there are no collisions with the convex hull in between slices, the part must be sliced using virtually the same method described above, but instead of slicing on the Z axis, it slices on the Y axis. Since this is used only for the transition between slices, and the slices are ordered by the angular component of their polar coordinate, the part should be sliced on the plane where Y = 0 and all facets with negative X values should be thrown out. This assumes that the reference vector for the polar coordinate system is (1,0).

Using the points generated from this method, the algorithm takes only the points in between the two slices, orders them based on their Z axis values, and then adds them to the path in between the two slices. This could also be modified to generate a path based on a specific angular polar coordinate by redefining the plane as an equation instead of an absolute value in one axis. After all of the slices have been added and the path is complete, it still needs to be converted into a timestamped path as opposed to its current format with only the time required for each move being reported and the normal vectors need to be inverted to represent tool orientation.

2.4. Partial Path Creation

In some instances, a full path plan may not be necessary. A user could have pre-existing knowledge of the parts condition and only need to plan for a specific area, or the user could observe some flaws in the original path plans simulation and opt to add an additional partial path to address the issue. Partial path planning can also be utilized to prevent robotic accessibility issues, such as singularities, collisions, and reach issues. Previous research lays out a framework and infrastructure for facilitating these decisions by allowing the user to select individual facets on the part for partial path planning [38]. In these scenarios, this path planner uses the extreme values of the selected facets Z values and angular components of the centroids polar coordinate to select the necessary parts of the path. Aside from the selection process, the algorithm proceeds as normal. The slices selected as part of the partial path are defined as,

$$S_{part}=\{s \mid Z_{r_{min}} \le s_z \le Z_{r_{max}} \wedge s \in S\} \quad (25)$$

where $Z_{r_{min}}$ and $Z_{r_{max}}$ are the minimum and maximum Z values of the selected facets and s_z is the Z value of slice s. Within each selected slice, the planner defines the path points to be selected as,

$$P_s=\{p \mid \varphi_{min} \le \varphi_p \le \varphi_{max} \wedge p \in P_s\} \quad (26)$$

While this method is relatively easy to implement, it is computationally inefficient. An alternative method would be to define a partial convex hull of only the selected facets and then run the algorithm as usual with the partial hull. If a partial convex hull is implemented, there needs to be some methods developed to ensure that the resulting path does not violate the convex hull of the entire part, otherwise a collision is very likely for non-continuous selections and even a possibility for continuous selections. Specifically, since a convex hull is built without regards to the orientation of the original facets inside the convex hull, so the resulting toolpath would still wrap completely around the new convex hull even if the selected facets were only on one side. The only way to then avoid the method utilized in this planner, would be to successfully link facets on the convex hull to facets on the

original point and build a partial convex hull from only the corresponding facets.

2.5. Adaptive Methods

To this point, all path planning has been done on the convex hull. This will henceforth be referred to as the "naïve" tool trajectory since it does not consider the underlying surface topology. The following sections describe methods for adapting the naïve trajectory based on the actual part surface. This is accomplished by associating features of the underlying surface with segments of the toolpath and adjusting the tool offset distance and/or velocity based on aggregate descriptions of the underlying surface's position and orientation with respect to the convex hull.

The true surface is represented by a set of ordered pairs of points and normal vectors, $C=\{\boldsymbol{c},\boldsymbol{n}\}$. The points are sampled from the workpiece's tessellated mesh, and each point is paired with the unit normal vector of the facet from which it was sampled. To avoid ambiguity regarding the unit To this point, all path planning has been done on the convex hull. This will henceforth be referred to as the "naïve" tool trajectory since it does not consider the underlying surface topology. The following sections describe methods for adapting the naïve trajectory based on the actual part surface. This is accomplished by associating features of the underlying surface with segments of the toolpath and adjusting the tool offset distance and/or velocity based on aggregate descriptions of the underlying surface's position and orientation with respect to the convex hull.

The true surface is represented by a set of ordered pairs of points and normal vectors, $C=\{\boldsymbol{c},\boldsymbol{n}\}$. The points are sampled from the workpiece's tessellated mesh, and each point is paired with the unit normal vector of the facet from which it was sampled. To avoid ambiguity regarding the unit normal, points should not be sampled from facet edges. Sampling strategy is a tradeoff between resolution and computational load, and it has implications for how each surface facet influences the adjusted tool trajectory. A simple method is to take the centroid of each facet. This insures that each facet is represented in the ensuing calculations but has disadvantages when facet size varies significantly or facet aspect ratios are high: Areas of high curvature (many small facets) can dominate areas of low curvature (fewer, larger facets), and the centroids of high-aspect-ratio facets can be far from their associated vertices. The disadvantages may be mitigated by enforcing a uniform sampling resolution within facets; however, this can significantly increase the number of points and thus the computational load. A third option is to densely sample the mesh and then down sample via a voxel grid filter (VGF) [39].

The VGF overlays a three-dimensional grid onto the point cloud; all points within each voxel are represented their centroid and normal vectors are represented by their average. The resolution versus computation tradeoff is managed by setting the grid size, though the VGF itself consumes computational resources. For simplicity, and without loss of generality, this research uses the first method – representing each facet via its centroid.

2.6. Distance Based Adapting

Given the bins of the path being adapted and the point cloud points that have been matched to each bin, the algorithm defines the distance from each point to the line created by the two end points of the matched bin [40]. Using this distance, the adaptive process can begin by defining the adjustment value for each bin as,

$$\psi b = \min(dAGb - d\mathrm{O}, xd\mathrm{O}) \; \forall \; b \in Bs$$

where Bs is the set of all bins on slice s, x represents the maximum percentage that the path can adapt by with regards to dO, and $dAGb$ is defined as the aggregate distance value of all points in the bin, which can vary based on the aggregation method. Ideally, this returns a value of zero, meaning no adjustment is needed and the aggregate distance is equal to the desired offset distance. The adjustment value is limited in range so that the path will not be moved within the convex hull by over adjusting. In this case, x = 0.95. This is necessary because a value greater than or equal to 1 would result in an adjustment greater than the offset distance itself which would cause the new position to be inside or on the convex hull. While this may not create any collision issues, there is always the possibility and thus is must be accounted for. Alternatively, a minimum distance could be defined and the adjustment would occur on the remaining distance beyond the minimum distance. It should also be noted that the aggregate method used depends on the user's initial input. If necessary, the new path points are determined by,

$$\boldsymbol{p} = \boldsymbol{p} - (\boldsymbol{np}\hat{}\,\psi p) \; \forall \; \boldsymbol{p} \in Ps$$

where $\boldsymbol{np}\hat{}$ is the unit vector of the corresponding surface normal, Ps is the path for slice s, and ψp is the adjustment value of the path point which is defined as,

$$\psi p = \max(\psi bp, \psi bp - 1)$$

where ψbp is the adjustment value of point p in bin b. The max of the two bins that share the point is used to ensure that the bin needing the most adjustment gets it. For path smoothness, a moving average of the adjacent bins can be used to determine how much adjustment is needed for each bin by taking the mean of a few adjacent values on both sides of the bin. An example of distance-based adaptation is shown below in Figure 8 as an individual slice of Part C, where both the naïve path, in blue, and the distance adapted path, in orange, have been plotted along with the facets that are captured within the slice, the exact slicing height is shown to the right of the path analysis.

3.Results and discussion

These results come from a set of five test parts that are shown in Appendix A. Two of the parts were specifically designed to demonstrate how the planner makes adjustments to the path, while the other three were taken from online 3D CAD databases to represent common geometries encountered in the real world. Preliminary analysis was performed via ANOVA analysis to test for interactions between the two factors. A two-way analysis considered the statistical method to be multi-colinear and threw it out. Upon further analysis, a one-way ANOVA analysis showed that there was some

significance to the type of statistical method used, which is expected as the aggregation method directly effects all values in the same and generally equal manner. In other words, common sense would indicate that adaptation based off of the minimum values would be greater than for the mean, which would be greater than if the max was used. Going forward this analysis will focus on the difference between adaptive methods by averaging the values from each treatment grouped by adaptive method. The aggregate results of all parts are shown in Table 1.

Table 1: Average Adaptive Method Results

Adaptive Algorithm	Mean Impingement	Bin Metric	Total Path Time	Total Path Length
Distance	0.00270	1.264	318.856	3215.599
Distance + Time	0.00390	1.474	513.151	3215.599
Naïve	0.00177	1.131	309.094	3117.976

Along with the data above, both a histogram and a kernel density estimate (KDE) of the probability density function were created for each treatment of each part. For clarity, the KDE's have been trimmed to show only positive values up to the 95th percentile. This is done because these estimates do not take into account that the values cannot be negative and the extreme values can cause the data to be very hard to read. The histograms have also been trimmed to zoom in on the densest areas of the distribution to better observe the change between each treatment. In this case, the histogram zooms in on the range 0 - 0.01, and all values greater than 0.01 are contained in their own bin. Along with these comparisons between each treatment as a function of impingement, the tradeoffs with making the adaptations were also considered. The resulting values were plotted against the total time taken to complete the path in both actual values and percent difference from the naïve path. Figure 11 contains the KDE's for adaptive algorithm and aggregation method as well as a plot of each treatments bin metric vs time in columns 1-3 respectively, and each row A-E references the corresponding part. Individual part results and heat maps are also included in the text below. In the heatmaps, red indicates more cleaning and blue indicates less or none at all. All other figures and results are contained in the appendices. Appendix A contains the original parts, Appendix B contains the KDE's, Appendix C contains the histograms, Appendix D contains the trade-off plots, Appendix E contains the individual treatment results for each part, and Appendix F contains a nomenclature table for all equations.

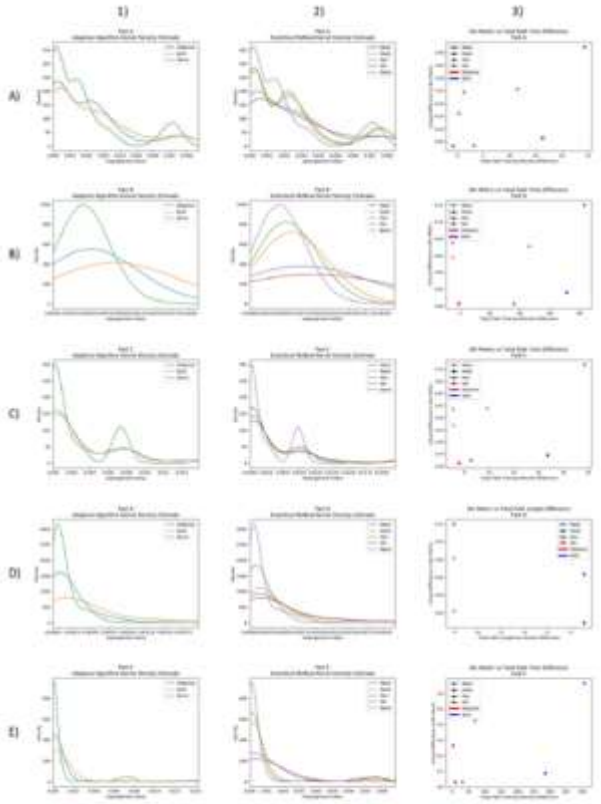

Figure 13: Summary Plots for All Parts and Treatments

Part A is a specifically designed test cube intended to show how the algorithm reacts to concavities and other changes in geometry. Each side and corner are designed to test a different geometric form. Shown here in Figure 12 is a concave half sphere at each level of cleaning. Here the blue center indicates areas beyond the effective reach of the sprayer. This area lessens when distance adaptation is used, but remains roughly the same when time is added, although the colors around it shift closer to red. This can also be seen in Figure 11(A1) where there is a spike in higher values as time is added on to the distance adapted path. This results in an improvement of the bin metric by ~0.1 at the cost of an ~10% increase in time to execute shown in Figure 11(A3).

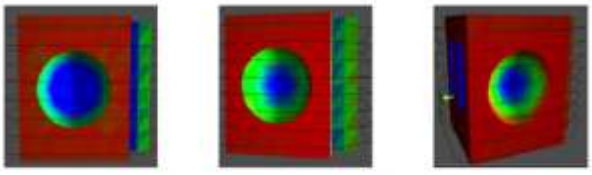

Figure 14: Part A Heatmap at Each Adaptive Algorithm Level (Naïve, Distance, Distance+Time)

Part B a specifically designed test cube intended to show how the algorithm reacts to changes in incidence angle and thus how it adapts velocity and time. While the results are a bit difficult to see as the mean surface is mostly convex and requires little path variation, when viewed in real time, a real change in velocity can be observed. This part also demonstrates one of the key issues with flat surfaces on the convex hull. There is a distinct difference of coloration

between facets on the same plane due to the large facet size which causes the centroids to be captured in different slices and thus adapted differently. Figure 17(a) demonstrates this issue for a single slice. This also causes the path to be unevenly adapted despite the mean surface being consistent throughout. Here, the bin metric only improves by ~0.02 at the cost of nearly a ~40% increase in total path time shown in Figure 11(B3).

Figure 15: Part B Heatmap at Each Adaptive Algorithm Level (Naïve, Distance, Distance+Time)

From these results, the most consistent, effective and cheapest, in terms of computation time, method for adapting the naïve path is as follows: Begin by adapting the naïve path using the distance-based method, evaluate the quality of the path and if a standard is not met, adapt the path based on time if required. Otherwise, the time-based adaptive method is not needed. This can be seen across all of the adaptive algorithm charts in column 1 of Figure 11 which shows a significant decrease in low impingement values from the naïve to the distance adapted path and again when adapting by both distance and time and is summarized in Figure 18. While the values in the figure for parts B and D are not exactly convincing, they can be explained by geometric abnormalities as discussed above. Overall, these findings are consistent with the fact that work done by spraying is highly nonlinear when adjusting distance, which makes it a critical part of the process when calculating actual work done. This method was also chosen because the benefits of adapting based on time are nowhere close to the benefits of adapting by distance and choosing only to use it if the distance method cannot meet the requirements saves significant computational time. Aside from computational time, adjusting the time for each bin generally creates a longer path completion time, which is not ideal. Further compounding the issue of time, some distance adjustments can actually lessen the overall completion time. It should also be noted that the mean aggregation method is preferred because it gives a better representation of all points in a bin. While taking the minimum definitely returns better values, it can be unnecessarily affected by outliers. The same holds true for the maximum value. And while the mode might make sense, these are continuous values and thus it is not very likely to consistently find multiple occurrences of the same value. The difference between using max and min values can also be made up by increasing the sprayer pressure.

4. Conclusion

As robots continue to ingrain themselves within industrial settings, it is only a matter of time until fully autonomous systems are the norm. This two-pronged approach to generalized path planning, planning for the convex hull and then adjusting for the original part, removes a lot of the difficult geometrical problems from the equation. When comparing adaptive methods and considering tradeoffs for the simplified paths, it is clear that adjusting based on distance is the most effective way of achieving better results, although using both methods does produce better toolpaths and can be justified depending on the trade-off with time on that part. If nothing else, these results indicate that it is possible to achieve good toolpaths without intricate and time-consuming path planning algorithms by sacrificing some precision. Moving forward, there are plenty of improvements that can be made, in any number of areas, but this algorithm does provide a good stepping off point for agile path planning for industrial spraying operations of novel complex parts. A groundwork was laid for the implementation of a collaborative robotic pressure washing work cell. The framework described in chapter 2 sets a standard for system design and infrastructure and the path planner described in chapter 3 can be utilized in both the initial planning and rework modules from the framework. In the process of developing both the framework and path planner, a prototype system was built to model the pressure washing work cell that inspired this project. Currently, the prototype system encompasses everything included in both papers except for the physical implementation of the work cell. The user is prompted to input the initial 3D data and input parameters before the adaptive path planner takes over and generates an initial path plan. After seeing the visualization, the user can select specific areas of the part for replanning or approve the path. If replanning is required, the user is again prompted for some input parameters and then the new replanned path is added on to the initial path or can replace it entirely.

Moving forward, work still needs to be done on developing better analysis techniques that more accurately model what facets are actually being covered by the path plan, especially with regards to complex geometries that may hide pieces of the part from the sprayer. There is also a need for even more advanced path planners that can dynamically change the structure of the path when the adaptive measures cause once covered pieces to go uncovered. Aside from the research-based advances, the obvious next step is to turn the virtual prototype into a physical prototype. To do this, two things need to happen. First, the resulting path from the path planner needs to be checked and reconfigured to reflect the physical limitations of the robot performing the task, and second, there needs to be some system in place to gather the 3D data from the part relative to the robot.

REFERENCES

[1] A. Woods and H. Pierson, "Developing an Ergonomic Model and Automation Justification for Spraying Operations," in Institute of Industrial and Systems Engineers Annual Conference, Orlando, FL, 2018.
[2] H. Chen, T. Fuhlbrigge and X. Li, "Automated Industrial Robot Path Planning for Spray Painting," in 4th IEEE Conference on Automation Science and Engineering, Washington, D.C., 2008.
[3] J. S. Oh, Y. H. Choi, J. B. Park and Y. F. Zheng, "Complete Coverage Navigation of Cleaning Robots Using Triangular-Cell-Based Map," IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, vol. 51, pp. 718-726, 2004.
[4] Z. He, B. Lu, J. Hong, Y. Wang and Y. Tang, "A novel arc-spraying robot for rapid tooling," International Journal of Advanced Manufacturing Technologies, 2007.
[5] W. Chen and D. Zhao, "Path Planning for Spray Painting Robot of Workpiece Surfaces," Mathematical Problems in Engineering, 2013.

[6] W. Sheng, N. Xi, M. Song, Y. Chen and P. Macneille, "Automated CAD-guided robot path planning for spray painting of compound surfaces," in IEEE/RSJ International Conference on Intelligent Robots and Systems, Takamatsu, Japan, 2000.
[7] A. M. Kabir, J. D. Langsfeld, S. Shriyam, V. S. Rachakonda, C. Zhuang, K. N. Kaipa, J. Marvel and S. K. Gupta, "Planning Algorithms for Multi-Setup Multi-Pass Robotic Cleaning with Oscillatory Moving Tools," in IEEE International Conference on Automation Science and Engineering (CASE), Fort Worth, TX, 2016.
[8] Y.-H. Choi, T.-K. Lee, S.-H. Baek and S.-Y. Oh, "Online Complete Coverage Path Planning for Mobile Robots Based on Linked Spiral Paths Using Constrained Inverse Distance Transform," in The IEEE/RSJ International Conference on Intelligent Robots and Systems , St. Louis, MO, 2009.
[9] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," Rovotics and Autonomous Systems, vol. 61, pp. 1258-1276, 2013.
[10] C. Luo, S. X. Yang, D. A. Stacey and J. C. Jofriet, "A Solution to Vicinity Problem of Obstacles in Complete Coverage Path Planning," in IEEE International Conference on Robotics 8 Automation, Washington DC, 2002.
[11] J. H. Lee, J. S. Choi, B. H. Lee and K. W. Lee, "Complete Coverage Path Planning for Cleaning Task using Multiple Robots," in IEEE International Conference on Systems, Man, and Cybernetics, San Antonio, TX, 2009.
[12] S. X. Yang and C. Luo, "A Neural Network Approach to Complete Path Planning," IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, vol. 34, pp. 718-724, 2004.
[13] A. UGUR, "Path planning on a cuboid using genetic algorithms," Information Sciences, vol. 178, pp. 3275-3287, 2007.
[14] J. C. Rubio, J. Vagners and R. Rysdyk, "Adaptive Path Planning for Autonomous UAV Oceanic Search Missions," in AIAA 1st Intelligent Systems Technical Conference, Chicago, IL, 2004.
[15] F. Schwarzer, M. Saha and J.-C. Latombe, "Adaptive dynamic collision checking for single and multiple articulated robots in complex environments," IEEE Transactions on Robotics, vol. 21, no. 3, pp. 338-353, 2005.
[16] R. Bohlin and L. Kavraki, "Path planning using lazy PRM," in IEEE International Conference on Robotics and Automation, San Francisco, CA, 2000.
[17] A. Pichler, M. Vincze, K. Hausler, H. Andersen and O. Madsen, "A Method for Automatic Spray Paiiitiiig of Uiiknowii Parts," in IEEE International Conference on Robotics & Automation, Washington DC, 2002.
[18] E. U. Acar, H. Choset, Y. Zhang and M. Schervish, "Path Planning for Robotic Demining: Robust Sensor-based Coverage of Unstructured Environments and Probabilistic Methods," The International Journal of Robotics Research , vol. 22, pp. 441-466, 2003.
[19] M. Hebert and E. Krotkov, "3-D Measurements From Imaging Laser Radars: How Good Are They?," in International Workshop on Intelligent Robots and Systems , Osaka Japan, 1991.
[20] F. Remondino and S. El-Hakim, "IMAGE-BASED 3D MODELLING: A REVIEW," The Photogrammetric Record, vol. 21, pp. 269-291, 2006.
[21] F. Blais, "Review of 20 years of range sensor development," Journal of Electronic Imaging, vol. 13, pp. 231-240, 2004.
[22] D. K. Pai, K. v. d. Doel, D. L. James, J. Lang, J. E. Lloyd, J. L. Richmond and S. H. Yau, "Scanning Physical Interaction Behavior of 3D Objects," in SIGGRAPH, Los Angeles, CA, 2001.
[23] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade and D. Fulk, "The Digital Michelangelo Project: 3D Scanning of Large Statues," in SIGGRAPH, New Orleans, LA, 2000.
[24] C.-F. Huang, Y.-C. Tseng and L.-C. Lo, "The Coverage Problem in Three-Dimensional Wireless Sensor Networks," in IEEE Globecom, Dallas, 2004.
[25] J. D. Schutter, T. D. Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aertbeliën, K. Claes and H. Bruyninckx, "Constraint-based Task Specification and Estimation for Sensor-Based Robot Systems in the Presence of Geometric Uncertainty," The International Journal of Robotics Research, vol. 26, pp. 433-455, 2007.
[26] J. Tegin and J. Wikander, "Tactile sensing in intelligent robotic manipulation – a review," Industrial Robot: An International Journal , pp. 64-70, 2005.
[27] P. Meng, E. S. Geskin, M. C. Leu, F. Li and L. Tismeneskiy, "An Analytical and Experimental Study of Cleaning With Moving Waterjets," J. Manuf. Sci. Eng., vol. 120, no. 3, pp. 580-589, 1998.
[28] S.-H. Suh, I.-K. Woo and S.-K. Noh, "Development of An Automatic Trajectory Planning System (ATPs) for Spray Painting Robots," in Intemational Conference on Robotics and Automation, Sacremento, CA, 1991.
[29] H. Chen, W. Sheng, N. Xi, M. Song and Y. Chen, "Automated Robot Trajectory Planning for Spray Painting of Free-Form Surfaces in Automotive Manufacturing," in IEEE International Conference on Robotics 8 Automation, Washington DC, 2002.
[30] A. Djuric, R. J. Urbanic and J. L. Rickli, "A Framework for Collaborative Robot (CoBot) Integration in Advanced Manufacturing Systems".
[31] ISO/TS 15066:2016 Robots and robotic devices -- Collaborative robots, International Organization for Standardization, 2016.
[32] P. Waurzyniak, "Putting Safety First in Robotic Automation," Manufacturing Engineering, pp. 61-66, September 2016.
[33] T. Anandan, "Robotic Industry Insights: Collaborative Robots and Safety," Robotic Inustries Association, 26 Jan 2016. [Online]. Available: https://www.robotics.org/content-detail.cfm?content_id=5908.
[34] Universal Robots, "Afraid to Commit? No Problem, with Flexible, Redeployable Cobots," 3 June 2016. [Online]. Available: https://blog.universal-robots.com/flexible-redeployable-cobots. [Accessed 30 April 2017].
[35] Robotiq, "Teaching Robots Welding," [Online]. Available: http://robotiq.com/solutions/robot-teaching/. [Accessed 6 May 2017].
[36] P. Stone and M. Veloso, "Towards collaborative and adversarial learning: a case study in robotic soccer," International Journal of Human-Computer Studies, vol. 48, pp. 83-104, 1998.
[37] R. Mitnik, M. Recabarren, M. Nussbaum and A. Soto, "Collaborative robotic instruction: A graph teaching experience," Computers and Education, vol. 53, pp. 330-342, 2009.
[38] S. Brown and H. Pierson, "A Collaborative Framework for Robotic Task Specification," Procedia Manufacturing, vol. 17, pp. 270-277, 2018.
[39] C. Connolly, "Cumulative generation of octree models from range data," in IEEE International Conference, 1984.
[40] Roman, "Find the shortest distance between a point and line segments (not line)," August 2017. [Online]. Available: https://stackoverflow.com/questions/27161533/find-the-shortest-distance-between-a-point-and-line-segments-not-line.
[41] D. Dakdouk, "Tool Accessibility with Path and Motion Planning for Robotic Drilling and Riveting," Ryerson University, 2016.
[42] D. Vinayagamoorthy, "Rotor Blade," 29 Ocotober 2017. [Online]. Available: https://grabcad.com/library/rotor-blade-8.
[43] C. Domingos, "Wing_Section_NACA23015C200," 2 January 2018. [Online]. Available: https://grabcad.com/library/wing_section_naca23015c200-1.
[44] F. H. Macias, "rueda motriz e inducida grua viajera," 21 September 2017. [Online]. Available: https://grabcad.com/library/rueda-motriz-e-inducida-grua-viajera-1.