International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 06 | June - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Spraying Operations in the Industrial Sector that Need Integrated Robotic Path Planning for Complex Geometry

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 - A number of significant challenges are presently standing in the way of the widespread use of automated robotic solutions to complicated activities. In high-mix/low-volume operations, for example, there is often too much uncertainty to successfully hard-code a robotic work cell due to ineffective sensing and job unpredictability. The majority of existing collaborative frameworks are devoted on including the necessary senses for physical collaboration. Mixing human cognitive function and fine motor abilities with robotic strength and repeatability has been a successful model for reducing uncertainty. However, there are numerous situations when physical engagement is not feasible but human reasoning and task understanding are still required. Important components of the suggested framework are a route planner, a route simulator, and a result simulator. The operator can communicate with these modules using an integrated user interface, making edits to the path plan before automatically authorising the task for execution by a manipulator that doesn't need to be collaborative. In a remanufacturing setting where each component needs one-off route planning, the collaborative framework is demonstrated for a pressure washing job. Other processes that may be included into the framework include shot peening, deburring, grinding, sandblasting, and spray painting. In such settings, surface preparation and coating might be automated using automated route planning for industrial spraying operations. Most real-world components do not conform to the assumptions made by autonomous spray route planners in the literature, which have focused on continuous and convex surfaces. Concave and discontinuous surfaces, such as sharp corners, holes, protrusions, and other surface anomalies, must be systematically handled by planners while paths are being constructed. The route planner creates paths by means of a slicing-based algorithm. By comparing the actual part geometry with the convex hull path, it finds if surface irregularities and concavities should be taken into account in the path plan and, if so, how significant they are. At certain locations along the route, the movement speed or offset distance can be adjusted to suit the needs of the way. The construction of the route planner also takes into account the trade-offs involved with path adaptation and the relative effectiveness of different adaptive methods.

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

1.INTRODUCTION

The original goal of this project was to construct an automated pressure washing work cell that could handle most of the Army's rework and rebuild depot's parts. This was difficult owing to the wide range of part sizes and geometry that needed daily cleaning. The facility cleans pallets of smaller pieces and tank bodies. The fact that part geometry was practically impossible to determine compounded the issue. Lack of data, easy-to-miss part variances, and a manual procedure with unique parts make it difficult, especially for repair and rebuild facilities like this one. Due to these issues, most institutions have avoided automating the procedure and done it manually. This is the most prevalent way, but these occupations are physically demanding, and until recently, the technology needed to automate them was either too expensive or too complicated. Full coverage path planning is one of the hardest activities to automate economically and consistently. It's possible, however most automated jobs don't require a fresh path design for each step. They are usually used to repeat preprogrammed segments. Given that human operators are good at deciding what needs to be cleaned and that an automated system can build a good path plan on the fly, the collaborative robotics community pioneered the idea of blending a human's cognitive function with an automated robotic system's precision and endurance. Adaptive path planning was implemented to improve the path quality compared to the naïve approach. Due to its expanding breadth, this project was split into two issues. What does the collaborative system look like from input to user verification to process execution? What does an adaptive path planner for pressure washing look like?

1.1 OVERALL FRAMEWORK DESIGN

This section outlines the necessary modules inside the framework and identifies the essential criteria for the success of each module. Figure 1 depicts the interactions among the separate modules, the system's external components, and the human operator. The system utilises the supplied 3D data and initial user input as parameters for the path planner to produce a path. The path is thereafter transmitted to the path analyser prior to the simulation exhibiting the original 3D input, the path, and the analysis. The operator can either accept the suggested task as presented or modify it. Adjustments are done by the path modification module if required and thereafter returned for analysis before the operator may make another decision. Upon authorisation, the trajectory is transmitted to the robot, and the assignment is executed. Nevertheless, if the operator observes persistent deficiencies, the procedure may be

reinitiated with either the entire component or a reduced segment being forwarded to the path planning module.

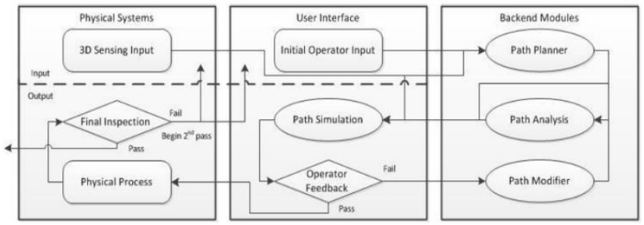

Figure 1: System Framework Design

2. Methodology

To circumvent the collision and accessibility problems caused by non-continuous and concave surfaces, this method entails constructing a convex hull around some initial 3D data—here provided as an STL file—because the procedure needs a normal vector. Any variety of 3D data collection methods could be employed, though, if there was a way to determine normal vectors and generate a tessellated mesh from a point cloud. The mesh is now transformed into a point cloud with each facet's centroid connected to its normal vector. Using the convex hull as a foundation, the path is constructed using a slicing-based approach that takes into account the following parameters: a rotation axis and degrees of rotation to change the slicing direction (since moving the part is highly improbable with properly calibrated and registered scanners); and a slice thickness, offset distance, and overlap percentage to measure the amount of work that will be applied to the part. The points from the initial mesh, which are now in the form of a point cloud, are assigned to certain sections of the route when its construction is complete. It is possible to map a location to more than one segment in certain situations.

The points assigned to each segment of the path are then used to adjust the various parts of the path. Depending on the actual geometry, the adaptive phase will either change the path's velocity to bring the end effector closer to the component or slow it down so more cleaning force may be delivered to a specific location. Using the system as an experimental tool, we were able to examine two aspects: the kind of adaptive algorithm and its statistical aggregation approach. Users have the option to select between a time-adapted path and a distance-adapted path, or to employ both or neither adaptive algorithm. The aggregation technique also gives them the choice between mean, mode, min, and max. Figure 3 depicts the procedure.

The algorithm's effectiveness is largely affected by two parameters—slice thickness (wA) and offset width (wO)—in the absence of adaptive techniques and other external influences. These values can be determined from a number of additional data depending on the operation at hand. Using the sprayer's angle, θ, and the end effector's distance from the component surface, di, the slice thickness may be determined for the suggested spraying procedure using basic right-angle trigonometry. The given overlap percentage, σ, is used to determine the offset width as a percentage of the slice thickness. These computations and the possibility of slice overlap on the part's surface are shown in Figure 4. While inputs like base velocity and sprayer intensity are welcome, they do nothing to alter the workload allocation and rather enhance the final product.

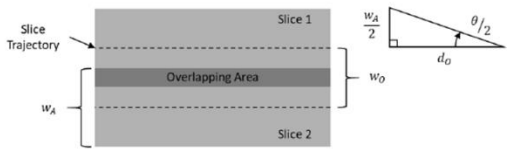

Figure 4: Overlapping Slice Paths and Input Parameter Derivation

2.1 Tool Path Trajectory

To enhance comprehension of the path generating process, the completed trajectory is displayed first. Upon completion of the procedure, the original 3D data has been processed and transformed into a definitive toolpath including seven data points for each place along the trajectory. The initial three coordinates denote the X, Y, and Z locations of the end effector in relation to the part's centre, namely the origin. These values must be converted for any practical use, but the conversion is wholly contingent upon the particular implementation. The last three points denote the orientation of the end effector as a directional vector from the end effector to the component, determined by inverting the associated facet's normal vector. This vector can be transformed into any other format as required. Although this vector does not completely specify the end effector's pose, as it neglects the rotational aspect of the tool, it provides motion planners with a variable to determine a kinematic solution. The impact of this free parameter may vary depending on the process; nevertheless, under the assumption of a conical spray pattern, it does not influence the process. The seventh point denotes the date of that specific point, which a robot might utilise to create a trajectory. Each journey point is delineated as follows:

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

and each orientation is defined as,

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

and the finished trajectory is defined as,

$$G = \{ [\boldsymbol{p}_k , \boldsymbol{r}_k , t_k] \mid \forall k = \{0, \dots, \beta\} \} \tag{3}$$

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,

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

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 $\Phi s=\{[\boldsymbol{p}k,\boldsymbol{n}k]\}$ 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 connecting each slice's path to the master path, a few more parts are required to ensure a decent path. In this case, a raster pattern is generated by altering the sequence in which the points from each individual slice are added to the route, as seen in Figure 6. This is advantageous for robots with limited reach since the path does not need the robot to constantly circle the part. To do this, the algorithm arranges the path from the lowest angular polar coordinate to the highest, and the points are added in this order for the first slice, then from highest to lowest in the next, and so on.

Another way not previously discussed is the transition from slice to slice. To avoid collisions with the convex hull between slices, the portion must be sliced in a manner similar to that described above, but on the Y axis rather than the Z axis. Because this is only utilised for the transition between slices, and the slices are sorted by the angular component of their polar coordinates, the portion should be cut on the plane with Y = 0, and all facets with negative X values should be removed. This assumes the polar coordinate system's reference vector to be (1,0).

Using the points created by this technique, the algorithm selects just the points in between the two slices, arranges them according to their Z axis values, and adds them to the path between the two slices. This might also be changed to construct a route based on a chosen angular polar coordinate by specifying the plane as an equation rather than an absolute value in one axis. After all of the slices have been added and the route is complete, it must be changed into a timestamped path rather of its current format, which just reports the time necessary for each step and inverts the normal vectors to show tool orientation.

2.4. Partial Path Creation

There are times when you might not need a full road plan. The user might already know how the parts are doing and only need to plan for a certain area, or they might see some problems with the modelling of the original path plans and choose to add an extra partial path to fix the problem. You can also use partial path planning to stop artificial mobility problems like singularities, collisions, and reach problems. Research from the past has set up a framework and infrastructure that makes these choices easier by letting the user pick out specific parts of the part for partial path planning [38]. In these cases, this path planner picks out the necessary parts of the path based on the extreme values of the chosen facets Z values and the angle components of the centroids polar coordinate. The algorithm works normally except for the choosing step. These are the slices that were chosen as part of the incomplete path:

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

where $Z_{c_{min}}$ and $Z_{c_{max}}$ 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,

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

Although this approach is relatively straightforward to execute, it is computationally inefficient. An alternative approach would be to establish a partial convex hull that encompasses only the selected facets and subsequently execute the algorithm as usual with the partial hull. The implementation of a partial convex hull necessitates the development of methods to guarantee that the resulting path does not violate the convex hull of the entire part. Failure to do so may result in a collision for non-continuous selections and even a possibility for continuous selections. In particular, the resulting toolpath would still encircle completely around the new convex hull, even if the selected facets were only on one side, as a convex hull is constructed without regard for the orientation of the original facets within the convex hull. The sole method to circumvent the approach employed in this planner would be to effectively connect the facets on the convex hull to the facets on the original point and construct a partial convex hull using only the corresponding facets.

2.5. Adaptive Methods

Thus far, all path planning has been conducted on the convex hull. This will thereafter be designated as the "naïve" tool trajectory, as it disregards the underlying surface topology. The subsequent sections delineate techniques for modifying the naïve trajectory in accordance with the real surface of the component. This is achieved by correlating characteristics of the underlying surface with segments of the toolpath and modifying the tool offset distance and/or velocity based on comprehensive descriptions of the underlying surface's location and orientation relative to the convex hull.

The actual surface is shown as a collection of ordered pairs of points and normal vectors, $C=\{\boldsymbol{c},\boldsymbol{n}\}$. The points are extracted from the tessellated mesh of the workpiece, with each point associated with the unit normal vector of the corresponding facet from which it was derived. To eliminate ambiguities concerning the unit Thus far, all path planning has been conducted on the convex hull. This will thereafter be designated as the "naïve" tool trajectory, as it does not account for the underlying surface topology. The subsequent sections delineate techniques for modifying the naïve trajectory in accordance with the real surface of the component. This is achieved by correlating characteristics of the underlying surface with segments of the toolpath and modifying the tool offset distance and/or velocity based on comprehensive descriptions of the underlying surface's location and orientation relative to the convex hull.

The actual surface is denoted by a collection of ordered pairs including points and normal vectors, $C=\{\boldsymbol{c},\boldsymbol{\hat{n}}\}$. The points are extracted from the tessellated mesh of the workpiece, with each point associated with the unit normal vector of the corresponding facet from which it was derived. To eliminate uncertainty concerning the unit normal, points must not be sampled from the margins of facets. The sampling approach represents a compromise between resolution and computing burden, affecting the manner in which each surface facet impacts the modified tool trajectory. A straightforward approach involves determining the centroid of each feature. This ensures that each facet is included in the subsequent calculations, but it has drawbacks when facet sizes vary considerably or when facet aspect ratios are elevated. Regions of significant curvature (many tiny facets) may prevail over regions of minimal curvature (fewer, bigger facets), and the centroids of high-aspect-ratio facets can be far from their corresponding vertices. The drawbacks may be alleviated by implementing a consistent sampling resolution within aspects; however, this can considerably elevate the number of points and hence the computing burden. An alternative approach involves densely sampling the mesh and subsequently downsampling using a voxel grid filter (VGF) [39].

The VGF superimposes a three-dimensional grid over the point cloud; all points within each voxel are represented by their centroid, and normal vectors are denoted by their average. The balance between resolution and processing is governed by adjusting the grid size, however the VGF itself requires computer resources. This research use the first way, representing each facet by its centroid, for the sake of simplicity and without loss of generality.

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,

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

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,

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

where $\boldsymbol{\hat{np}}$ 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

The results are derived from a collection of five test elements, which are illustrated in Appendix A. The remaining three components were sourced from online 3D CAD databases to depict common geometries encountered in the real world, while two were specifically designed to illustrate how the planner makes adjustments to the path. In order to investigate potential interactions between the two variables, an ANOVA analysis was implemented. The statistical procedure was rejected as multi-colinear in a two-way analysis. Upon further

examination, a one-way ANOVA analysis revealed that the statistical method employed had some significance. This is to be anticipated, as the aggregation method directly affects all values in a uniform and generally equal manner. In other words, it would be logical to assume that adaptation based on the minimum values would be more effective than adaptation based on the mean, which would be more effective than adaptation based on the maximum value. In the future, this analysis will concentrate on the distinctions between adaptive methods by aggregating the values of each treatment, which will be grouped by adaptive method. Table 1 displays the combined outcomes of all components.

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

h a histogram and a kernel density estimate (KDE) of the probability density function. To make things clearer, the KDEs have been cut down so that they only show positive numbers up to the 95%ile. This is done because these figures don't think about the fact that the values can't be negative and that values that are too high or too low can make the data hard to read. Also, the histograms have been cut down to focus on the most thick parts of the distribution so that you can see the difference between each treatment more clearly. This time, the histogram focusses on the range 0 to 0.01, and each number bigger than 0.01 is in its own bin. Along with these similarities of each treatment based on impingement, the costs and benefits of making the changes were also thought about. The values that came out were plotted against the total time it took to go along the path, showing both the real values and the percent change from the naïve path. Figure 11 shows the KDEs for the adaptive algorithm and the aggregation method. It also shows a plot of the treatments' bin measure vs. time in columns 1-3, and each row (A–E) refers to a different part. There are also heat maps and statistics for each part in the text below. The heatmaps show that red means more cleaning and blue means less or no cleaning at all. The appendices have all the other numbers and findings. There are five sets of files: Appendix A has the original parts, Appendix B has the KDEs, Appendix C has the histograms, Appendix D has the trade-off plots, Appendix E has the treatment results for each part, and Appendix F has a table with all the equations' names.

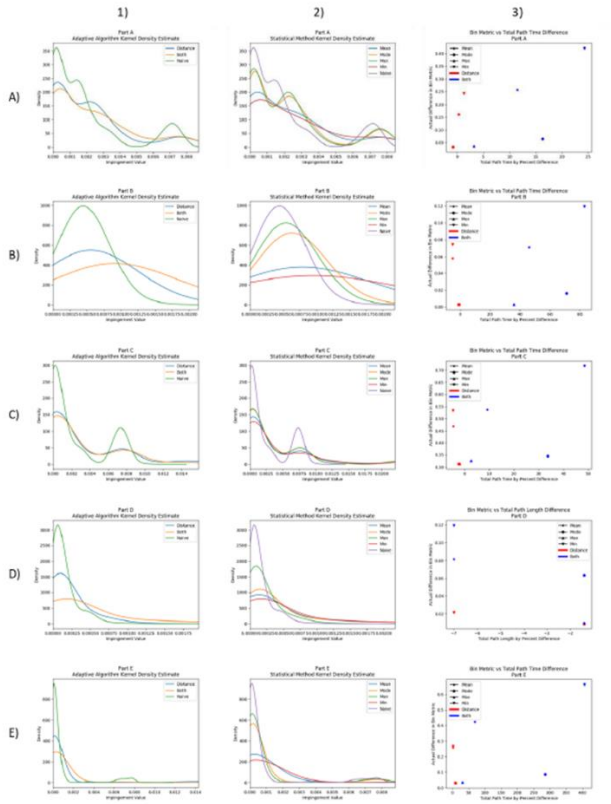

Figure 13: Summary Plots for All Parts and Treatments

Part A is a test cube that has been particularly designed to demonstrate how the algorithm responds to changes in geometry, such as concavities. Each side and corner are intended to evaluate a distinct geometric form. A concave half sphere is depicted in Figure 12 at each cleansing level. The cyan centre in this instance denotes regions that are beyond the sprayer's effective range. This region experiences a decrease in size when distance adaptation is implemented; however, it maintains a similar size when time is incorporated, despite the fact that the colours surrounding it become increasingly crimson. This is also evident in Figure 11(A1), where a rise in higher values is observed as the distance-adapted path is extended over time. This results in a ~0.1 enhancement in the bin metric, but at the expense of a ~10% increase in the time to implement, as illustrated in Figure 11(A3).

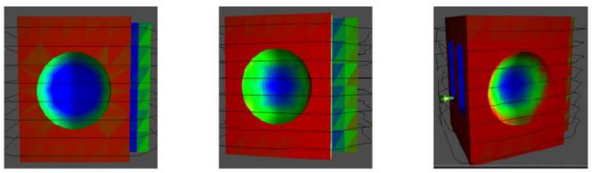

Figure 14: Part A Heatmap at Each Adaptive Algorithm Level

(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).

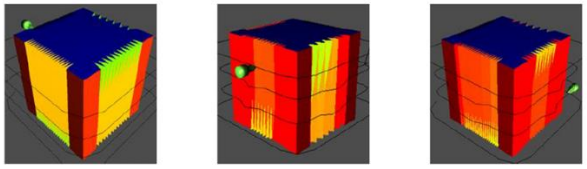

Figure 15: Part B Heatmap at Each Adaptive Algorithm Level (Naïve, Distance, Distance+Time)

The most consistent, successful, and cost-efficient strategy for modifying the naïve route, based on these results, is as follows: Commence by modifying the naïve way using the distance-based approach, assess the path's quality, and if it fails to meet the requirement, adjust the path based on time if necessary. Consequently, the time-based adaptive technique is unnecessary. Column 1 of Figure 11 illustrates a notable reduction of low impingement values across all adaptive algorithm charts, transitioning from the naïve to the distance-adapted route, and further when adapting by both distance and time, as summarised in Figure 18. The values in the figure for sections B and D may lack persuasiveness, however they can be elucidated by the aforementioned geometric anomalies. These findings align with the observation that the work performed by spraying exhibits significant nonlinearity with respect to distance adjustments, rendering it a crucial factor in the calculation of real work accomplished. This technique was selected because the advantages of temporal adaptation are much inferior than those of spatial adaptation, and utilising the latter only when the distance method fails to satisfy the criteria conserves considerable processing time. In addition to computing duration, modifying the time for each bin typically results in an extended path completion time, which is suboptimal. Moreover, certain distance modifications can indeed reduce the total finishing time. The mean aggregation approach is used as it provides a more accurate picture of all points inside a bin. Although the minimum consistently yields superior values, it may be unduly influenced by outliers. The same applies to the maximum value. Although the mode may seem logical, these are continuous values, making it improbable to constantly encounter several instances of the same value. The disparity between utilising maximum and minimum values can also be compensated by augmenting the sprayer pressure.

4. Conclusion

As robots become more prevalent in industrial settings, it will only be a matter of time until completely autonomous systems become the norm. This two-pronged method to generalised route planning, which involves planning for the convex hull first and then correcting for the original section, eliminates many of the more challenging geometrical difficulties. When comparing adaptive methods and considering tradeoffs for simplified paths, it is clear that adjusting based on distance is the most effective way to achieve better results, though using both methods produces better toolpaths and can be justified depending on the time trade-off. If nothing else, these findings show that it is feasible to produce effective toolpaths without using complex and time-consuming path planning algorithms by sacrificing some precision. Moving ahead, there are several enhancements that can be made in a variety of areas, but this method serves as an excellent starting point for agile route planning for industrial spraying operations using unique complicated elements. The foundation was built for the establishment of a collaborative robotic pressure washing work cell. The framework presented in chapter 2 establishes a standard for system design and infrastructure, and the route planner described in chapter 3 may be used for both initial planning and rework modules in the framework. During the development of the framework and path planner, a prototype system was constructed to simulate the pressure washing work cell that inspired this project. Currently, the prototype system includes everything mentioned in both articles except for the physical implementation of the work cells. The user is requested to provide basic 3D data and settings before the adaptive path planner develops an initial path plan. After seeing the visualisation, the user can choose particular portions of the component for replanning or approve the path. If replanning is necessary, the user is requested again for certain input parameters, after which the new replanned path is added to or replaces the previous path altogether.

Moving ahead, further effort has to be done to build better analytical algorithms that more precisely reflect what aspects are really covered by the route plan, particularly for complicated geometries that may hide parts from the sprayer. There is also a need for more complex path planners that can dynamically adjust the structure of the path when adaptive methods force previously covered sections to become exposed. Aside from the research-based gains, the apparent next step is to convert the virtual prototype into a real one. To do this, two things must occur. First, the route planner's output must be verified and adjusted to reflect the physical restrictions of the robot doing the work. Second, some mechanism must be in place to collect 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.