2017 IEEE International Conference on Robotics and Automation (ICRA)
Singapore, May 29 - June 3, 2017

Planning Cuts for Mobile Robots with Bladed Tools

Jeffrey I Lipton[1], Zachary Manchester[2], and Daniela Rus[1]

Abstract— **Linear bladed cutting tools, such as jigsaws and reciprocating saws are vital manufacturing tools for humans. They enable people to cut structures that are much larger than themselves. Robots currently lack a generic path planner for linear bladed cutting tools. We developed a model for bladed tools based on Reeds-Shepp cars, and used the model to make a generic path planning algorithm for closed curves. We built an autonomous mobile robot which can implement the algorithm to cut arbitrarily large shapes in a 2D plane. We tested the robots performance and demonstrated the algorithm on several test cases.**

I. INTRODUCTION

We believe flexible manufacturing infrastructure requires mobile robot systems for both assembly and fabrication tasks. Mobile robots based on arms and drones have been used in a wide variety of assembly tasks [1][2][3][4][5]. But, mobile robotic fabrications systems are still needed to enable scalable fabrication. Algorithms for planning cuts with linear bladed cutting (LBC) tools are critical to enabling the widespread deployment of mobile robotic fabrication systems. LBC tools, such as the jig saw, band saw, coping saw, saws-all, and scroll saw, are the most common type of cutting tool used by human workers. Despite their ubiquity for the human craftsman, robotic applications of these tools is limited by a lack of planning algorithms. We provide an algorithm for planning cuts with LBC tools in cases where the blade and material are neither inserted or removed. We modeled a blade as a modified Reeds-Shepp car with the constraint that it can reverse if and only if it is reversing along its previously traveled trajectories. Using this model we developed a curve processing algorithm that can process a closed shape into sections based on curvature and closing operations. To demonstrate this algorithm we developed a new robotic fabrication system. The mobile robot with an integrated jigsaw is able to produce 120mm diameter circles cut with 8mm maximum radial error. This mobile fabrication system allows for a scalable and parallelizable fabrication platform since it is limited in dimension only by its work surface, localization system, and battery life.

In this paper we contribute:

- Develop an algorithm for solving the path planning problem given a blades unique constraints
- Provide simulation results for the algorithm
- Develop a robotic system which can use bladed manufacturing to make large scale objects

This work was supported by the Boeing Corporation. [1]Computer Science and Artificial Intelligence Lab, 32 Vassar street, Cambridge, MA, 02139, USA. `jlipton at mit.edu` [2]School of Engineering and Applied Sciences, Harvard University, 60 Oxford St, Cambridge, MA USA 02138. `zmanchester at g.harvard.edu`

II. BACKGROUND

Most subtractive manufacturing robots,including CNC machines and robot arms, are equipped with rotary tools. Unlike humans, robotic arms and gantries can easily apply counter torques to control a rotary tool. Rotary tools are also holonomic, making cutting with them easier to plan.This makes them ideal for carving and milling operations. Mobile humanoid robots like CHIP have used rotary tools for this reason at the DARPA Robotics Challenge [6].

Rotary tools, however, have several drawbacks. While they are ideal for low aspect ratio ($depth/width \approx 1$) cuts, they are wasteful for high aspect ratio cuts. To prevent buckling, the cutting bit needs to become thicker as length increases, leading to wasted material and power. In mobile robot applications the need for a counter torque can increase system demands. Rather than simply applying the force for locomotion and cutting, a mobile robot must also apply a counter torque to prevent the robot from spinning.

A major limitation of current robotic manufacturing systems is the build space. CNC machines require a work piece to fit inside of them. Making a larger CNC machine can increase cost and complexity in non-linear ways since the maximum deflection of a beam is proportional to the length of the span to the fourth power [7]. Previous attempts at mobile manufacturing robotic systems have relied on building large systems which can move robotic arms [8][9] [10]. Other attempts have relied on making a robotic system plant itself in an as series of locations.[11]. By building a mobile robot around the mobile tools humans use to fabricate,we can decouple the robot size from the final part size.

Bladed tools are used in a wide range of industries, from meat processing to furniture making and construction [12]. These tools use a single-sided blade that either reciprocates or is part of a continuously moving band to cut curves [13][14]. They can be used to cut extremely thick parts, are far faster at cutting than rotary tools, and can be used to cut extremely fine features.

Currently robotic arms in manufacturing can integrate with LBC tools such as band saws. These systems rely on programmed movement patterns designed by humans[15]. This limits their ability to adapt to new designs and makes setup for these tools time intensive. Previous attempts to automate LBC planning have used connective circles to cut shapes but have not developed general solutions [16].

III. BLADE MODELING

A blade on the end of a robot, constrained to a plane has a unique set of movement requirements. As seen in Figure 2A,

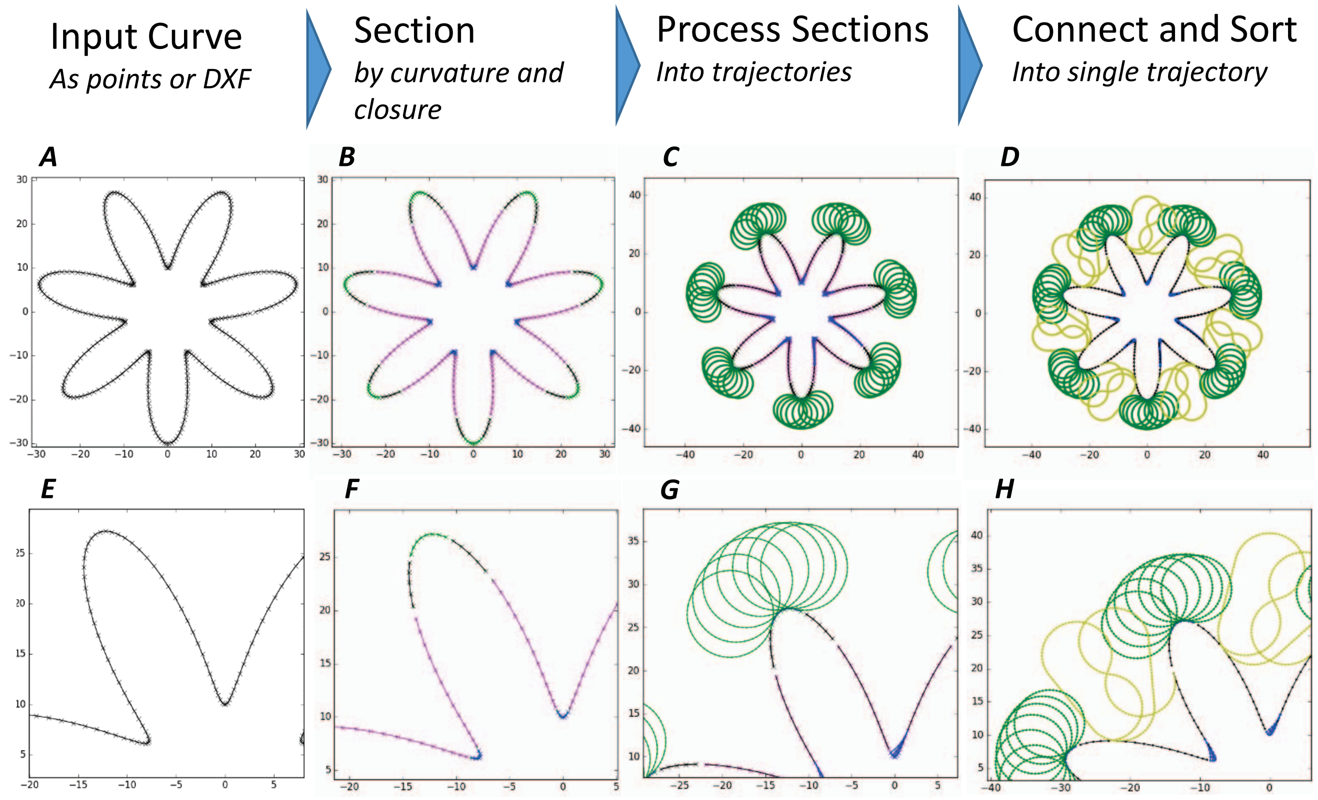


Fig. 1. Processing a target shape into a trajectory. The top row shows the overall shape being processed. The bottom row shows a representative sub-sample being processed. The target curve A,E is segmented into sections seen in B,F. The pink sections are interior type sections (I). The black sections are outside type sections (O). The green sections are convex type sections (C), and the blue type are Saw-tooth type sections (S). Each section type is then processed into a trajectory for part of the cut. This is seen in C and G. Finally the cut trajectories are ordered together and connected. They are connected directly, with single-Dubin or with double-Dubin paths. These are seen in in figures D and H as yellow trajectories.

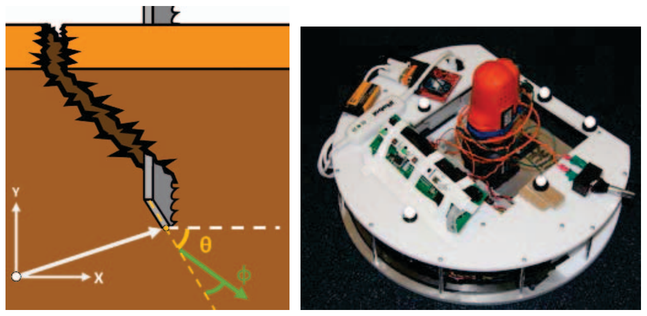


Fig. 2. Model of Blade contact (left) and Cutting Robot (right). We model the blade as having a position (x,y) in the $\mathbf{R}^2$ plane and angle θ. The blade has a depth l, and a turning angle ϕ. The blade moves at a speed u_s and changes angle at a rate u_ϕ such that $u_s \in \{0,1\}$ and $|u_\phi| \leq \phi_{max} < \pi/2$. The robot (Right) can be modeled as a tricycle and therefore the constraints from the blade dominate.

we model the blade as having a position (x,y) and angle (θ) in the $\mathbf{R}^2$ plane. The blade has a depth l and a turning angle ϕ. The blade moves at a speed u_s and changes angle at a rate u_ϕ such that $u_s \in \{0,1\}$ and $|u_\phi| \leq \phi_{max} \leq \pi/2$. The speed configuration space $U = [-1,1] \times (-\phi_{max}, \phi_{max})$ This leads to a minimum radius of $\rho_{min} = l/tan(\phi_{max})$. In free space a blade can be modeled as a tricycle with speed configuration space $U_{tric} = u_s \times u_\phi = [-1,1] \times (-\pi/2.0, \pi/2.0)$.

When moving forward into material, it must cut through the material, limiting the speed and angle change. The speeds u_s and rate u_ϕ are determined by the blade, motor and material interaction, and must be pre-characterized. The forward constraints are Identical to those of a Dubin car, building a speed configuration space of $U_{dubin} = \{0,1\} \times (-\phi_{max}, \phi_{max})$ [17] [18]. Reeds-Shepp cars, however, can move forward and backward with a minimum turning radius leading to a speed configuration space of $U_{rs} = \{-1,0,1\} \times (-\phi_{max}, \phi_{max})$ [18].

Unlike a Reeds-Shepp car, blades cannot move backward into the material, since there is no cutting surface on the back side of the blade. A blade can travel backwards if and only if it is traveling into space where there is no material. If we assume that no material falls out when being cut, a blade can only reverse through locations it has previously visited since material is removed along the cutting path. This unique set of constrains on the Reed-Shepp car differentiates it from a Dubin car and allows for unique pathing solutions.

IV. Shape Processing Algorithm

A. Planning Requirements and Constraints

The requirements on trajectory-following for a blade are also different from typical path planning problems. In order for a shape to be cut, the blade's cutting surface must reach every point along the curve that defines the shape. This can

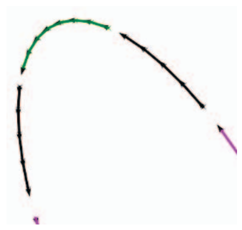
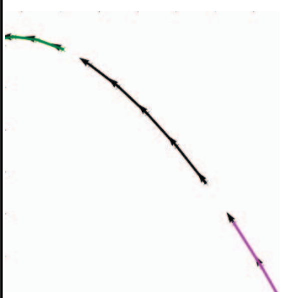
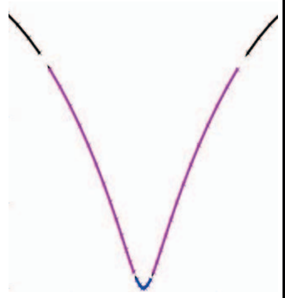
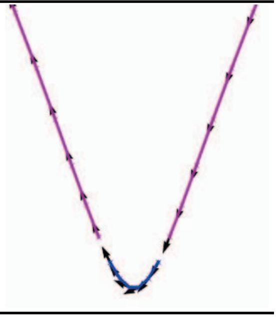
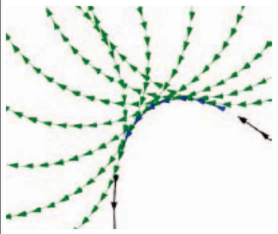
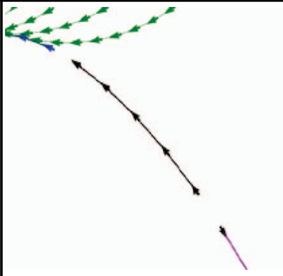
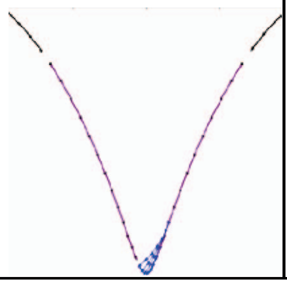
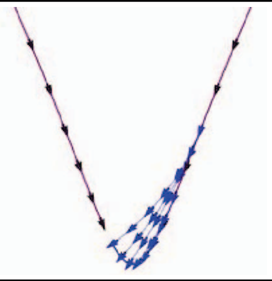
	C – Convex	O – Outside	I – Interior	S – Saw-tooth
Curvature	> limit	≤ limit	≤ limit	> limit
Closure	Outside	Outside	Inside	Inside
Curve Section				
Trajectory solutions				

Fig. 3. The various section types. the points on the target curve are segmented into the four types. C-sections are approximated with line segments connected with Dubin paths. The O-sections can be directly converted into trajectories. I-sections must be oriented to go from the closure curve towards the interior and then reversed along. S-sections cannot be moved along. They must be approached from other sections to have the blade terminate at the section. If the blade could not be reversed along trajectories it had cut, I and S types could not be made.

be accomplished in one of two ways: Either the blade follows the curve by moving tangential to the curve at each point, or the blade can be move off of the curve and stop non-tangent to the curve at each point. This second technique is often used by wood workers to cut features which require a turning radius smaller than the blade's turning radius. We exploit this ability to allow bladed tools to cut features smaller than their turning radius by leaving and returning to the desired curves.

To plan the cuts, it will be necessary to know the section of the curve where the blade can change its orientation along the curve form clockwise to counter-clockwise. As seen in Figure 4, a circle of radius $2R_{min}$ must be free of obstruction and tangent to a point on the curve for it to be reorientable. If the blade can move along and inside the circle of radius $R_{circle} = 2R_{min}$, the blade can travel along the circle and change directions along the circle without leaving the interior of the circle and without violating the $r \geq R_{min}$ constraint. In Figure 4A, we see that for a given tangent direction $\hat{T}$, the blade can move along a circle of radius R_{min} in a single direction. If two circles or R_{min} are tangent to each other, and one of them is tangent to the contact point of the blade, and there are lines tangent to both circles that terminate on them, it is possible for the blade to change from $\hat{T}$ to $-\hat{T}$ at the point. The blade can take an S shaped path from the current location to a point on the far circle, and traverse along said circle to the tangent line, and then along the tangent line to return to the original circle in the opposite orientation. Since these two circles are inscribed in the circle of $R_{circle} = 2R_{min}$, It will be possible to move along the large circle in either direction. Therefore the circle of $R_{circle} = 2R_{min}$ defines our minimum free area for reorienting.

B. Algorithmic Flow

We model a shape we wish to cut as a closed curve inside an infinite bulk of material. We call this the target curve $\mathscr{L}$. The target curve $\mathscr{L}$ need not be smooth, but must be continuous. As seen in Figure 1A and E, we approximate the target curve as a piece-wise linear function of lines between points of arbitrary distance. In Algorithm 1 we see that the the curve is first processed into sections based on the turning radius R_{min} in the MAKESECTIONS function. Once each section is identified, trajectories for each section are generated by the PROCESSSECTION function. Once the sections are turned into trajectories, they can be connected together using the CONNECT function.

C. Making Sections

We process the target curve into sections using a closing operation and a curvature test. First we perform a binary closing operation on the target curve using a circle $R_{circle} = 2R_{min}$ to generate the closure curve. The closure curve is used to determine the parts of the curve where the robot can arbitrarily reorient. At each point, we calculate the radius of curvature of the curve $R = |\frac{ds}{dT}|$, where T is the unit tangent

Algorithm 1 Process Shape Given R_{min}, and Curve $\mathscr{L}$

```
1: $sections \leftarrow$ MAKESECTIONS($\mathscr{L}, R_{min}$)
2: for all $section \in sections$ do
3:     $Trajs \leftarrow Trajs \cup$ PROCESSSECTION($section$)
4: end for
5: $JoinedTraj \leftarrow \{\}$
6: for all $Traj \in Trajs$ do
7:     $JoinedTraj \leftarrow JoinedTraj \cup$ CONNECT($Traj$)
8: end for
9: return $JoinedTraj$
```

Algorithm 2 CONNECT Given R_{min}, Curve $\mathscr{L}$

```
1: $P_s, V_s \leftarrow Start$
2: $P_e, V_e \leftarrow End$
3: if $distance(P_s, P_e) < threshold \wedge V_e \cdot V_s < threshold$ then
4:     return {}
5: else
6:     $Traj \leftarrow$ DUBIN($Start, End$)
7:     if Traj does not intersect $\mathscr{L}$ then
8:         return Traj
9:     else
10:        $mid \leftarrow$ FINDMID($Start, R_{min}$)
11:        $Traj \leftarrow$ DUBIN($Start, mid$)
12:        $Traj \leftarrow Traj \cup$ DUBIN($mid, End$)
13:        if Traj does not intersect $\mathscr{L}$ then
14:            return Traj
15:        end if
16:    end if
17: end if
18: return $Error$
```

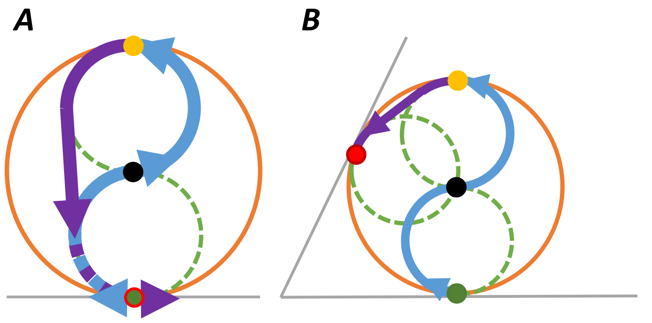


Fig. 4. Reorienting Along a circle of radius $2R_{min}$. A blade with a minimum turning radius of R_{min} (green) can reorient at a point if there is a circle of radius $2R_{min}$ (Orange) in which it can move, tangent to the point. In A, we see the blade start out along the blue path, from the green dot, it makes and S shaped curve to the Yellow point. At the yellow point it has changed its direction on the orange circle. It can then return to the green dot along the purple trajectory, maintaining its new orientation on the circle. In B we can see that this process can be used to move to other points on the orange circle.

vector and s is the distance along the curve. If a point has a radius of curvature smaller than the turning radius, we know that the robot cannot move tangentially along the curve at that point. Each point on the target curve is then classified by its radius of curvature and by being inside the closure curve or not.

Neighboring points with the same classification are then grouped into sections. In Figure 1B and F we can see a target curve broken into sections based on these criteria. Each section belongs to one of four categories: Outside, Convex, Interior, or Saw-toothed. In Figure 3 we see the sections of the various types generated from the target curve in Figure 1A. Outside sections have points lying on the closure curve and with radii greater than or equal to the minimum turning radius of the blade. Convex sections are sections where the points are on the closure curve, but have radii smaller than the turning radius. Interior-directional sections are points along the target curve with radii greater than or equal to the turning radius, but are inside of the closure curve. Saw-toothed sections are those points which are inside of the closure curve and have radii smaller than the minimum turning radius.

D. Process Sections into Trajectories

Figure 3 shows the solutions for each section type overlaid on the target curve. Each section type has a different solver which either keeps the blade tangential to the curve, normal to the curve, or approximates the curve. On-curve (O) sections can be oriented in either direction, and can be moved along by the blade, therefore they are trivial to process into cutting trajectories. Convex (C) sections must be approximated. For each point along a C-section, the incoming direction V_i cannot be the same as the outgoing direction V_o because of the curvature constraint. Therefore we approach from the previous point in a straight line. We must leave the current point in a straight line to the next point. Since the point is outside of the closure, it is safe to leave the curve if we remain inside a circle of radius $2R_{min}$. Therefore a Dubin curve can be used to transition from V_i to V_o at the point. In Figures 1C and G we see the Dubin paths in green connecting the blue linear approximations of the curve. Interior-directional (I) sections can be moved along with the blade tangent to the curve at all points, but only one end point of the section is on the closure curve. Therefore the section must be traveled along from the end on the closure curve to the other and reversed backwards along. Saw-tooth (S) sections can only be approximated by moving the blade up to the point on the curve and then stopped. Therefore for each point in the S-sections, a curve must be found such that $R \geq R_{min}$ at all points and it is tangent to a point on another section, and does not intersect the target curve. We use circles tangent to points in the preceding or following sections which terminate on the curve. We search each of the previous and following sections points. If the radius is too small or it intersects the tangent curve, it is rejected and another point is searched. If all points are exhausted and no solution is found, that point is deemed unreachable. If a solution is found, a trajectory forwards and backwards along the curve is added to the section. In Figure 3 the S-section trajectories are in blue and move from an I-section to the S-section. S-section solutions are merged into the preceding or following sections trajectories. This causes the other sections to branch to and from a point on the S section off of their original trajectory. Additionally each connection point is then along the closure curve.

E. Connection of Trajectories

A point which is the start of a section and on the target and closure curve is chosen as the starting point for the curve since it its reachable from the outside of the curve. Each section is then connected to the preceding curve using the CONNECT algorithm described in Algorithm 2. If a movement from the end of the previous section can be made to the start of the next without violating the curvature conditions, no extra trajectories are needed. Otherwise, the sections are connected using Dubin paths. Each connection path is tested for intersection with the target curve. If the single Dubin path intersects the curve, a double-Dubin path is used. These are paths seen in Figure 4B. The midpoint is the mirror of the start point through the line, parallel to the tangent line of the point, that bisects the circle of radius $2R_{min}$ which is tangent to the point. If there are points which cannot be connected without intersecting the target object, a cut from the outside can be added to separate the part into multiple cuts.

V. ROBOT DESIGN AND CONTROL

A. Robot Design

In order to test the algorithm we built a wheeled robot with a cutting blade mounted inside of it, as seen in Figure 2(b). The robot consists of a modified iRobot Create whose housing has been removed and replaced with a stronger and more open shell. The bottom is an aluminum plate and the top is an acrylic plate. We attached the robot to the top plate with standoffs and the bottom metal plate to the upper plate with standoffs. The standoffs are adjusted to sandwich the robot between the plates to secure the create components. We secured a Black and Decker LPS7000 Lithium-Ion CompactSaw to the metal plate with the cutting surface of the blade directly inline with the axes of the two wheels. We placed rechargeable lead-acid batteries next to the saw on the metal plate. These provided additional power to the saw and added weight to the robot. Teflon skids were placed beneath the saw to provide a secure and low friction contact to the surface being cut. Without a blade the robot can have a turning radius of zero. This, in theory allows the system to only be constrained by the turning radius of the blade. In practice, errors in motor speed control limit the turning radius at low speed. The robot has two DC motors attached to wheels via a gear train. Since the robot drives on top of the material it is cutting, it is necessary to have a sacrificial support layer underneath the cuts. This prevents the blade from hitting the floor, and prevents the cut material from falling out. Jigsaw blades pull the robot towards the material and generate down-force during the upstroke of the blade. This intern generates a vibration which added mass can dampen out. Other methods such as vibration dampening materials, could be employed to lighten the robot while maintaining cutting force [**?**].

B. Real-Time Control

Due to significant dead-band and saturation effects associated with the robot's motors, simple linear tracking control along trajectories proved ineffective. Instead, a nonlinear model-predictive control (MPC) scheme was used. Model-predictive control, sometimes referred to as receding-horizon control, involves solving an optimization problem over a finite horizon N of future time steps using a model of the robot's dynamics[19]. Unlike linear feedback controllers, MPC can explicitly account for non-linearity and actuator torque limits. However, these advantages come at a significantly higher computational cost.

Our MPC scheme solves the following optimization problem at each time step,

$$\begin{aligned} \text{minimize} \quad & J(\boldsymbol{x}, \boldsymbol{u}) \\ \text{subject to} \quad & x_{k+1} = f(x_k, u_k) \\ & u_{\min} \leq u \leq u_{\max} \end{aligned} \tag{1}$$

where J is a cost function, $x_k \in \mathbb{R}^3$ is the robot's state at time k, $u_k \in \mathbb{R}^2$ consists of the velocity commands sent to the robot's wheels at time k, $f(x_k, u_k)$ is a discrete-time model of the robot's dynamics, and $u_{\min}$ and $u_{\max}$ are limits on the motor speeds. We chose J to be the following quadratic cost function,

$$J = \sum_{k=1}^{N} \delta x_k^T Q \delta x_k + \delta u_k^T R \delta u_k \tag{2}$$

where δx and δu are predicted deviations from the desired state and control trajectories, respectively, and Q and R are positive-definite weighting matrices. This problem can be readily solved by a wide range of commercial and open-source nonlinear program (NLP) solvers such as SNOPT[20] and IPOPT[21]

At each time step, a measurement of the current state is made and a prediction of the future state trajectory over the next N time steps is performed using Runge-Kutta integration of the following continuous model of the robot's dynamics,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos(\theta) & \frac{1}{2}\cos(\theta) \\ \frac{1}{2}\sin(\theta) & \frac{1}{2}\sin(\theta) \\ \frac{-1}{2d} & \frac{1}{2d} \end{bmatrix} \begin{bmatrix} \ell \\ r \end{bmatrix} \tag{3}$$

where d is the distance between the robot's wheels and ℓ and r are the velocity commands sent to the robot's left and right wheels, respectively. An NLP solver is used to find the N future control inputs $\boldsymbol{u}^*$ that minimize the cost function over the prediction horizon. The first of these predicted commands, u_1^*, is then run on the actual robot. At the next time step, a new measurement of the state is made and the process is repeated.

VI. EXPERIMENTS AND DISCUSSION

Control of the robot was done using a XBee wireless serial port with interfaced with the iRobot Create. Commands were sent from an off-board computer which ran the controller and logged the state information. Measurements of the state of the robot were done with a Vicon tracking system which had an accuracy of 1 mm. The controller loop had a $5Hz$ update rate.

As a test of the robot control system, we had the robot cut circles out of pink foam. We chose pink foam as our cut material because it is homogeneous and can be easily locked together to form larger sheets of material. In Figure 5D we can see the robot cutting across two sheets of pink foam which have been locked together. The system is also capable of cutting parts smaller than itself, seen in 5D. We cut circles with a radius of 120mm 10 times. We compared the target position with the actual position using the Vicon system. As seen in Figure 5A, the system had a peak error of 8mm, or 6.67% while cutting the circle seen in Figure 5B.

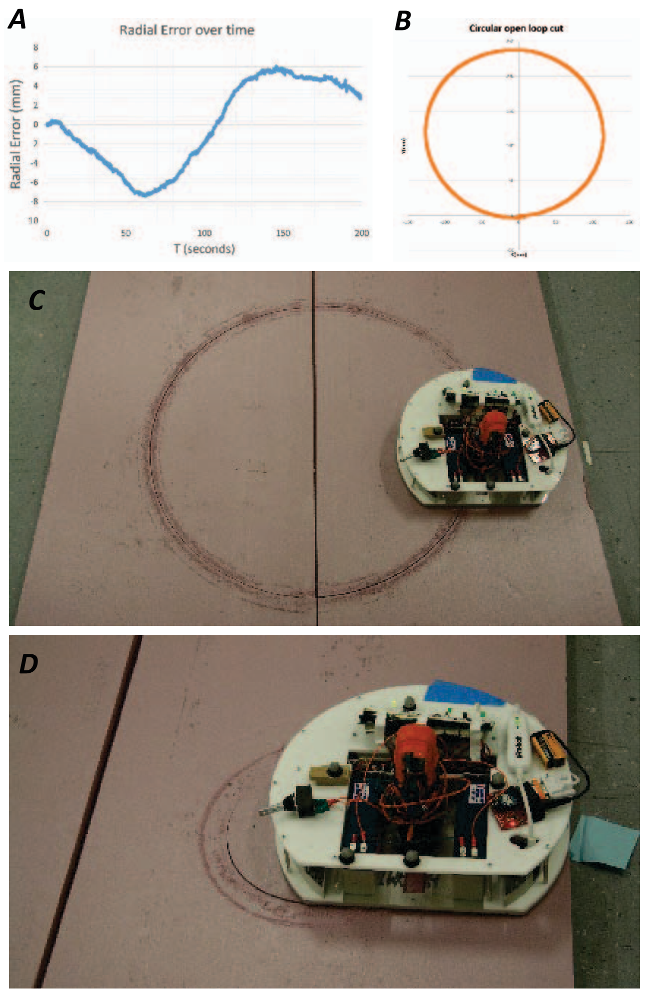


Fig. 5. Test of Robot and Controller. As a test of the robot and controller we had the system cut circles larger than itself (C) and smaller than itself (D). A cut of a 120mm radius circle (B) has a maximum radial error of 8mm seen in part A.

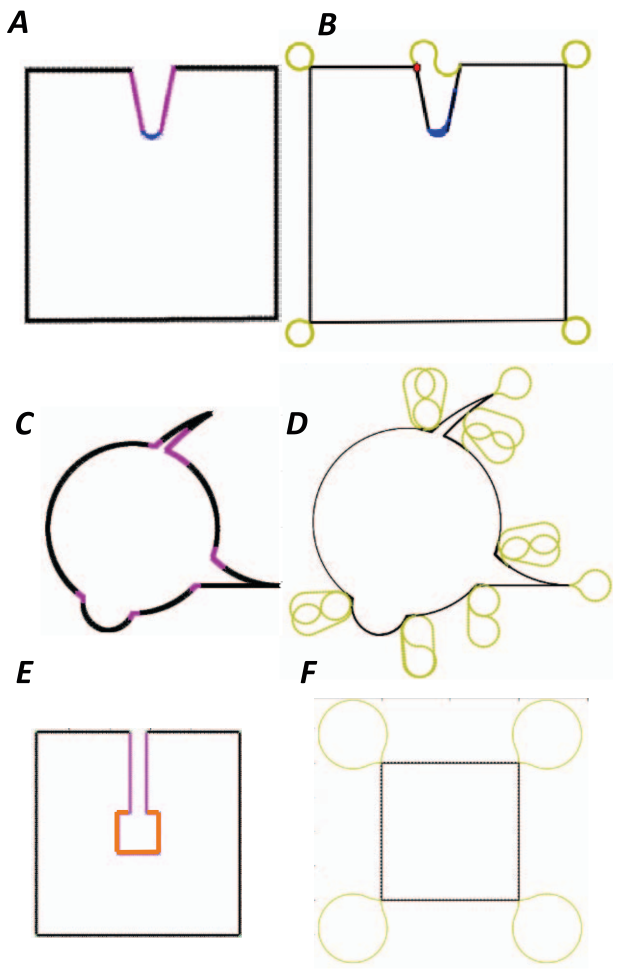


Fig. 6. Test of algorithm. As a test of the algorithm we processed several shapes. A Square (F) is easily decomposed into sides which are connected with Dubin paths in yellow. A square with a cut toward the interior (A) is processed into O type (black) I type(pink) and S type (blue). In B we see that it was processed into cuts along the curve (black), cuts off of the curve to S type(blue) and Dubin paths (yellow). A complex handmade shape (C) is processed into O and I type sections. Those sections were then processed into a single path along the target curve as seen in D. Some shapes cannot be solved (E). The sections highlighted in orange cannot be reached from the outside and would require a pilot hole to cut.

As a test of the algorithm performance, we tested the system on a variety of shapes. In Figure 6 you can see we processed a square, a square with a rounded cut to the interior, a square connected to a smaller square in the interior, and an arbitrary hand made shape. In each case the system was able to process the shape into sections, but only for the square within a square was it unable to a realizable cut. The algorithm fails to process the final shape. Shapes will fail to be processed if certain conditions are not met. For every point on the curve, there must exist a line from that point to a point at infinity that does not cross the curve. If such a line cannot be found, the shape cannot be guaranteed to be cut-able. For O and C sections we know that each point can be reached and the curve approximated. For I types, we know that if an endpoint is on the closure curve the end point is reachable so we can start there and move along the entire section. For S types, no such guarantee exists unless the points can reached from the closure curve by a curve with radius greater than the minimum radius. If a line to infinity exists, we are assured that from a sufficient distance, a path to that point, which terminates on that point can be found.

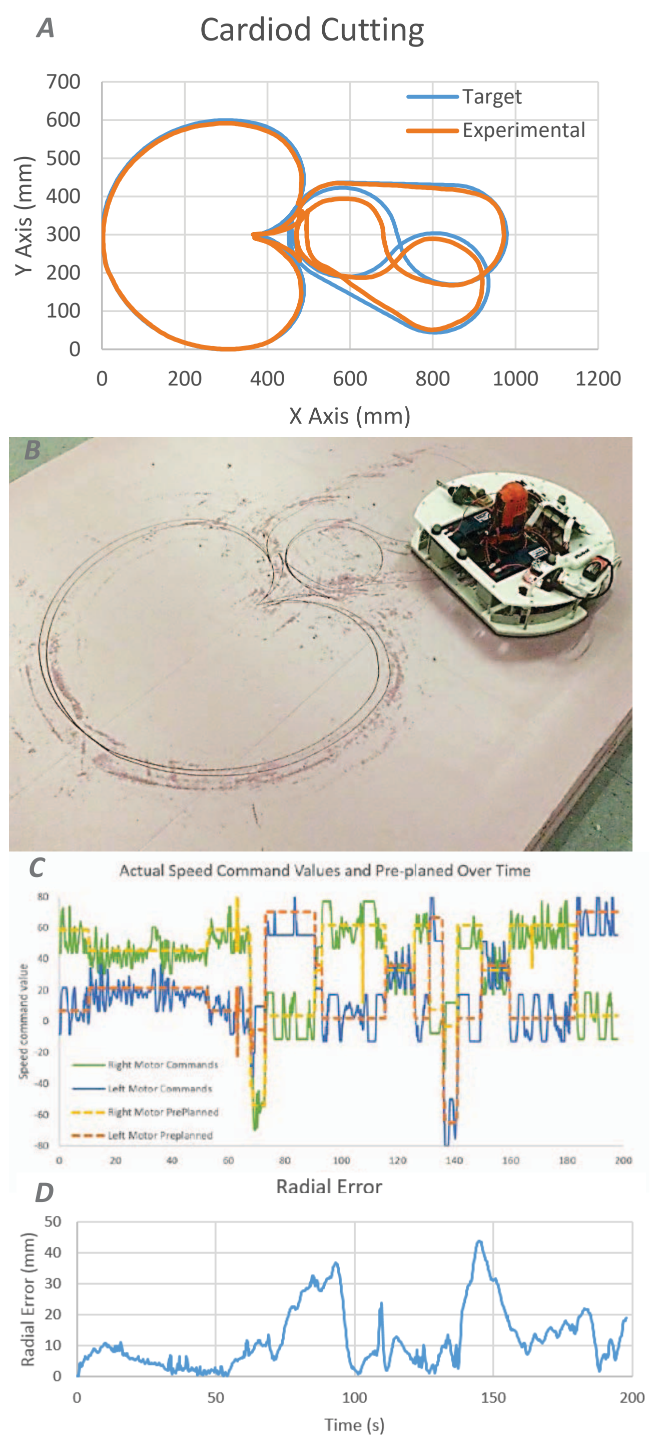


Fig. 7. Cutting cardiods using robot and planning algorithm. The Robot cut a cardiod several times (B) out of pink foam board. The actual vs target trajectories can be seen in A. The radial error between the two seen in D peaks above 40mm for part of the cut. The errors seem to occur after tight turns associated with the double-Dubin paths needed to connect the different sections. The controller effort can be inferred from the difference between pre-planned and actual controller speeds in C.

A test of the integrated robot, controller and planner system consisted of cutting the cardiod shape seen in Figure 7A and B 10 times. The shape decomposed into a main O-section, two I-sections and an S-section. The sections were connected with two double-Dubin paths between the I-sections and the O-sections. We can see in Figure 7A and D that there was significant deviations from the target trajectory. The peak error was over 40mm from target. The ability of the MPC controller to correct for errors in system positioning allowed the system to function properly. The effort of the controller can be observed in Figure 7C. The left and right motor pre-planned nominal values are compared with their actual commanded trajectories. The deviations are caused by the controller adjusting the system to account for positioning errors. The errors seem to be most severe after a tight turn of the robot.

Future versions of the system should be build with a more accurate and powerful drive system to allow for more forward force and accurate movements of the system. replacing the wheels with treads would allow the robot to distribute its load over a larger area, and cut more delicate structures such as cardboard as well as rougher structures such as plywood. The Vicon tracking system can be replaced by several other technologies. Since we desire a scalable localization system, we need a system that uses information contained in the internal area of the material to be cut or the robot itself. Any information on or beyond the boundary of the material could be lost since the boundary could be arbitrarily far away. April tags could be placed on the material or on the ceiling above the material for localization [22]. RFID tags could be placed below the sacrificial material layer to provide localization information [23]. Optical flow sensors could be used to derive movements based on variations of the local material surface [24]. IMUs can help provide telemetry. A Lighthouse system similar to the HTC Vibe's positioning system could be used to determine the robots location relative to a set of fixed way-points[25].

Other cutting methods such as rotary tools and laser cutters could be reconfigured into mobile robots. Such cutting devices are inherently holonomic and therefor would not require a complex pathing system for determining their movements. These however are often much larger and heavier than their linear bladed tool counterparts.

VII. Conclusions

We have demonstrated a new algorithm for planning cuts using linear bladed cutting tools on a robot. This algorithm is designed to consider the unique constraints LBC tools place on movement. By emulating the techniques used by woodworkers for centuries we were able to make an algorithm that could cut shapes tighter than its turning radius. With the compatible mobile robot this system creates a scalable CNC manufacturing system capable of cutting shapes out of large areas of material. Having mobile robots perform manufacturing tasks enables greater scalability in fabrication. A robot can be stored in a space much smaller than a CNC machine but could be deployed over much larger areas to cut

2D shapes. Multiple robots could be use to speed a single cut along by parallelizing it.

VIII. Acknowledgment

We would like to thank the Boeing Company for their support for the project.

References

[1] R. A. Knepper, T. Layton, J. Romanishin, and D. Rus, "Ikeabot: An autonomous multi-robot coordinated furniture assembly system," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 855–862.

[2] M. Dogar, A. Spielberg, S. Baker, and D. Rus, "Multi-robot grasp planning for sequential assembly operations," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 193–200.

[3] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction with quadrotor teams," *Autonomous Robots*, vol. 33, no. 3, pp. 323–336, 2012.

[4] M. Dogar, R. A. Knepper, A. Spielberg, C. Choi, H. I. Christensen, and D. Rus, "Towards coordinated precision assembly with robot teams," in *Experimental Robotics*. Springer, 2016, pp. 655–669.

[5] G. Reinhart and S. Zaidan, "A generic framework for workpiece-based programming of cooperating industrial robots," in *2009 International Conference on Mechatronics and Automation*. IEEE, 2009, pp. 37–42.

[6] C. M. Dellin, K. Strabala, G. C. Haynes, D. Stager, and S. S. Srinivasa, "Guided manipulation planning at the darpa robotics challenge trials," in *Experimental Robotics*. Springer, 2016, pp. 149–163.

[7] Y. Altintas, *Manufacturing automation: metal cutting mechanics, machine tool vibrations, and CNC design*. Cambridge university press, 2012.

[8] R. Holmberg and O. Khatib, "Development and control of a holonomic mobile robot for mobile manipulation tasks," *The International Journal of Robotics Research*, vol. 19, no. 11, pp. 1066–1074, 2000.

[9] B. Hamner, S. Koterba, J. Shi, R. Simmons, and S. Singh, "An autonomous mobile manipulator for assembly tasks," *Autonomous Robots*, vol. 28, no. 1, pp. 131–149, 2010.

[10] ——, "Mobile robotic dynamic tracking for assembly tasks," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 2489–2495.

[11] M. Denton, "Hexapod robot cnc router," May 2008. [Online]. Available: http://www.hexapodrobot.com/forum/viewtopic.php?f=14&t=12

[12] Z. Li, P. Ring, K. MacRae, and A. Hinsch, "Control of industrial robots for meat processing applications," in *Proceedings of the Australasian Conference on Robotics and Automation*. Citeseer, 2003.

[13] T. J. Ko and H. S. Kim, "Mechanistic cutting force model in band sawing," *International Journal of Machine Tools and Manufacture*, vol. 39, no. 8, pp. 1185–1197, 1999.

[14] B. Lehmann, "The cutting behavior of bandsaws," Ph.D. dissertation, University of British Columbia, 1993.

[15] I. Warsaw Machinery, "Mbd modell 4400 robotiic bandsawiing system," 2008. [Online]. Available: http://www.warsawmachinery.com/MBD_robotic_bandsawing_system.html

[16] A. J. Lubbe and H. Raath, "The development of a computer driven jigsaw," *The South African Journal of Industrial Engineering*, vol. 14, no. 1, 2011.

[17] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[18] J.-P. Laumond, S. Sekhavat, and F. Lamiraux, *Guidelines in nonholonomic motion planning for mobile robots*. Springer, 1998.

[19] E. Camacho and C. Bordons, *Model Predictive Control*, ser. Advanced Textbooks in Control and Signal Processing. Springer London, 2004.

[20] P. E. Gill, W. Murray, and M. A. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM Journal On Optimization*, vol. 12, pp. 979–1006, 1997.

[21] A. Wächter and T. L. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[22] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 3400–3407.

[23] S. S. Saab and Z. S. Nakad, "A standalone rfid indoor positioning system using passive tags," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 5, pp. 1961–1970, 2011.

[24] S. Lee and J. Song, "Mobile robot localization using optical flow sensors," *International Journal of Control, Automation, and Systems*, vol. 2, no. 4, pp. 485–493, 2004.

[25] J. Gibson, C. Haseler, H. Lassiter, R. Liu, G. Morrow, B. Oslund, M. Zrimm, and G. Lewin, "Lighthouse localization for unmanned applications," in *Systems and Information Engineering Design Symposium (SIEDS), 2016 IEEE*. IEEE, 2016, pp. 187–192.