UV Grid Generation on 3D Freeform Surfaces for Constrained Robotic Coverage Path Planning

Sean McGovern[1] and Jing Xiao[1]

***Abstract*— There are many industrial robotic applications which require a manipulator's end-effector to fully cover a 3D surface region in a constrained motion, such as painting, spray coating, abrasive blasting, polishing, shotcreting, etc. The manipulator must satisfy surface task constraints imposed on the end-effector while maintaining manipulator joint constraints. Coverage path planning (CPP) in this context generally involves placing commonly used coverage patterns (such as raster, spiral, or dual-spiral) onto the surface. There is substantial research for CPP on 2D surfaces, however, the problem of generating surface task constraints and evenly spaced coverage paths becomes particularly difficult when considering 3D freeform surfaces. Previous research concerning CPP on 3D surfaces consider parametric surfaces with limited surface curvature or produce unevenly spaced coverage paths. In this paper, we introduce a novel method to generate a uv grid on a 3D freeform surface (represented as a 3D polygon mesh) to facilitate feasibility checking for constrained coverage motion under task and manipulator constraints and to significantly ease the creation of more evenly spaced coverage paths for optimal application of task requirements. We applied our method to example 3D freeform surfaces to demonstrate its effectiveness.**

I. INTRODUCTION

Certain industrial manipulator applications (such as spray coating, abrasive blasting, polishing, shotcrete, laser ablation, etc.) require a manipulator's end-effector to traverse the surface once while satisfying task criteria in terms of spray thickness, cycle time, and material waste [1]–[9]. Surface coverage is often treated as an offline coverage path planning (CPP) problem on predefined surfaces with constraints. Methods for CPP on 2D surfaces are commonly online and intended for mobile robots [10]–[14]. Many previous methods for CPP on 3D surfaces apply to sensor coverage for aerial or submersible robots (i.e. view planning) [15]–[17], others relate to agriculture [18], [19], while some consider CPP for manipulator end-effector coverage [20]–[22]. Additionally, there is substantial work on constrained manipulator motion planning [23]–[27], which focuses on finding a feasible path connecting two configurations while the end-effector remains constrained, with the assumption that such a path exists prior to planning.

A feasible coverage path over a surface may not exist given certain manipulator and task constraints. Previous methods for constrained coverage path planning do not consider such feasibility issues. In [28], we introduced a general method to inform on coverage path feasibility, which uses a uv grid determined through parametric surface parameters to represent task constraints on 3D parametric surfaces. However, it is often difficult and not always possible to represent freeform surfaces with parametric equations. Moreover, there is little to no other work on uv grid generation on freeform surfaces for constrained CPP.

For coverage path planning on a constrained surface, some coverage path patterns (such as horizontal or vertical raster scan patterns) are used to ensure complete and even coverage. However, previous methods that place a coverage path pattern on a spatial surface are either limited to certain types of 3D surfaces (such as parametric, certain spline surfaces) or produce unevenly spaced coverage path patterns [2]–[5]. Whereas, evenly spaced coverage path patterns are necessary for optimal application of task requirements (e.g., even layer thickness) [13].

In this paper, we introduce a novel method to generate an evenly spaced 2D uv grid directly on 3D freeform surfaces represented as 3D polygon meshes. The uv grid contains cells to facilitate derivation of task constraint equations for coverage feasibility checking and makes it possible to place evenly spaced coverage path patterns on a 3D freeform surface. We derive geometric task constraint equations directly from 3D polygon meshes of freeform surfaces and convert generated constrained coverage paths from the uv grid onto 3D freeform surfaces using polygon mesh topology. Surfaces that are considered in this paper are found abundantly in industrial applications.

The rest of the paper is as follows. In Section II, we introduce notations and assumptions. We describe the method in Section III. We present implementation, testing results and discussions in Section IV and conclusions in Section V.

II. ASSUMPTIONS AND NOTATIONS

The physical surface of an object can be scanned using modern 3D scanning technology and approximated with a 3D polygon mesh, which is the most common way to represent a general surface when there is no other more precise model. We consider surfaces represented as 3D polygon meshes.

Let C be a planar, non self-intersecting, freeform curve. In this paper, we consider three types of 3D freeform surfaces created from C (see Fig. 1, which cover many surfaces in real world):

- **Surface of translation (SoT)**: translation of C, *along* one axis.
- **Surface of rotation (SoR)**: rotation of C, *about* one axis.
- **Surface of translation(s) and rotation(s) (SoTR)**: combinations of SoTs and SoRs from a single C, such

[1]The authors are affiliated with the Robotics Engineering Department, Worcester Polytechnic Institute. smmcgovern@wpi.edu, jxiao2@wpi.edu. This work is funded by US Army Research Lab Contract W911NF1920108.

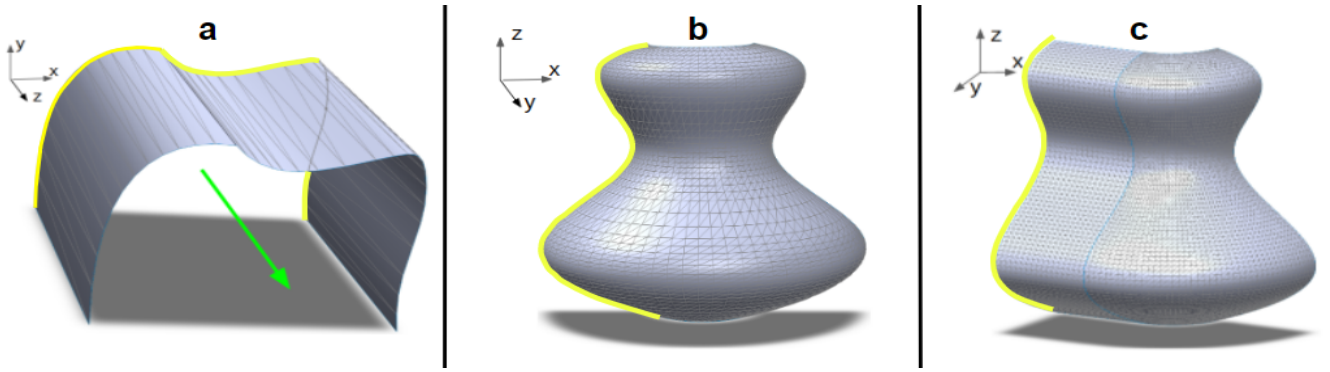


Fig. 1: Illustrations of 3D freeform surfaces (embedded polygon meshes). Curve C_0 for each definition is highlighted in yellow. (a) Freeform surface of translation (green arrow is direction of translation). (b) Freeform surface of rotation. (c) Freeform surface of rotation and translation.

that all rotation axes of C are parallel and all translation axes of C are on a plane normal to the rotation axes. The connecting curve between an SoT and an SoR sub-surfaces remains C.

We assume that the axes for the above surfaces and curve C can be known or approximated from the input of a human operator. Note that the above surface types share an important property that curve C remains the same along each axis of translation or about each axis of rotation. For an SoTR surface, the surface can be a continuous combination of translations and rotations of curve C which may occur simultaneously, as long as the rotation axis remains in the same direction, and the translation axis is always orthogonal to the rotation axis.

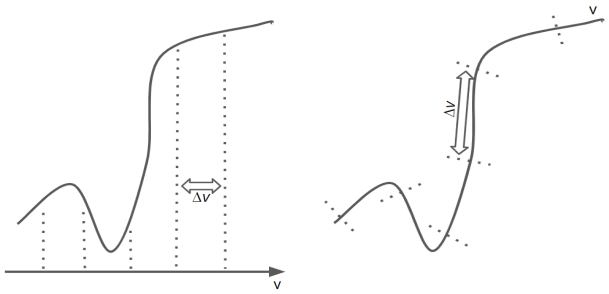
III. Methodology

We first describe our method to generate a 2D surface representation (a uv grid) for each freeform surface type defined above. We then discuss our method for deriving surface task constraints using the polygon mesh in the Cartesian space, which are used to inform on coverage feasibility checking. If a coverage path is feasible for the surface, we generate a commonly used coverage path pattern on the uv grid and convert it to Cartesian space onto the 3D freeform surface.

A. Generating uv Grid for 3D Freeform Surfaces

A uv grid that is evenly spaced on the 2D uv space may not result in an evenly spaced grid when it is projected to a 3D freeform surface, as illustrated in Fig. 2(a), and it is not trivial to find a uv grid that remains evenly spaced when projected to the 3D freeform surface. For a 3D freeform surface of one of the three types defined earlier, our idea is to let one of the parameters, say v, be a curve length parameter along the freeform curve C to achieve even spacing of the freeform surface along C, as illustrated in Fig. 2(b). In the following, we define and generate the uv space for each type of the freeform surfaces in detail, utilizing this idea.

*1) **Surfaces of Translation (SoT)**:* A coordinate frame is attached to the SoT such that the z-axis is along the direction of translation, see Fig. 1(a). Let C_0 be the curve C on the xy plane at the beginning of translation. Let (x_i, y_i, z_i) be



(a) Discretization along an axis results in uneven spacing on curve C. (b) Even discretization of curve C with v as a length parameter.

Fig. 2: Illustration of uneven discretization along C (a) and even discretization along C (b).

.

the local coordinates of the ith triangle vertex on the curve C_0 or intersection point of a triangle edge with C_0, $i = 0, ..., m$. For the ith point, we can compute the corresponding u_i coordinate in uv:

$$u_i = z_0 = 0 \tag{1}$$

Let $\Delta x_i = x_i - x_{i-1}$ and $\Delta y_i = y_i - y_{i-1}$. We can compute v_i by initializing $v_0 = 0$ and:

$$v_i = v_{i-1} + \sqrt{\Delta x_i^2 + \Delta y_i^2}, \tag{2}$$

for $i = 1, ..., m$, thus:

$$v_i = \sum_{j=1}^{i} \sqrt{\Delta x_j^2 + \Delta y_j^2} \tag{3}$$

Now, by incrementing z from z_0 to $z_1, z_2, ..., z_n$, we can obtain corresponding C_0, C_1, ..., C_n, which are the same curve as C but at different z coordinates.

*2) **Surfaces of Rotation (SoR)**:* A coordinate frame is attached to the SoR such that the z-axis is along the axis of rotation, see Fig. 1(b). We denote the angle of rotation as ϕ and radius (i.e. vertex distance from axis of rotation) as $r(z)$, which is an unknown function of z due to the freeform curve C. Let C_0 indicate the curve C on the plane of $\phi = \phi_0$.

Let (x_i, y_i, z_i) be the local coordinates of the ith triangle vertex on the curve C_0 or intersection point of the triangle edge with C_0, for $i = 0, ..., m$. For the ith point, we compute ϕ_i and $r(z_i)$. Let $u = r(z) \cdot \phi$, where u is in $[\phi_{min} r_{min}, \phi_{max} r_{max}]$, with the range determined by the size of the surface. We can compute:

$$u_i = r(z_i) \cdot \phi_i. \tag{4}$$

Let $\Delta z_i = z_i - z_{i-1}$ and $\Delta r_i = r_i - r_{i-1}$. We can compute v_i by initializing $v_0 = 0$ and:

$$v_i = v_{i-1} + \sqrt{\Delta z_i^2 + \Delta r_i^2}, \tag{5}$$

for $i = 1, \ldots, m$, thus:

$$v_i = \sum_{j=1}^{i} \sqrt{\Delta z_j^2 + \Delta r_j^2} \tag{6}$$

Now, by incrementing ϕ from ϕ_0 to $\phi_1, \phi_2, ..., \phi_n$, we can obtain corresponding C_0, C_1, ..., C_n, which are the same curve as C but correspond to different ϕ values.

*3) **Surfaces of Translation and Rotation (SoTR)**:* Denote the curve C at one edge of the SoTR as C_0. A coordinate frame is attached to the SoTR such that curve C_0 lies on the plane created by the yz-axes, all rotation axes are parallel to the z-axis, and all translation axes are orthogonal to the z-axis, see Fig. 1(c). We can create uv coordinates across the surface using the following method.

The SoTR surface and the xy plane intersects at curve U, which intersects C_0 at point p_0. Let $(x_i, y_i, 0)$ be the coordinates of the ith triangle vertex or edge intersection point on U, $i = 0, ..., n$, starting from p_0. Unlike in the case of SoT, where the u axis is a straight-line axis, here the u parameter is along the curve U, defined in the following way. Let $\Delta x_i = x_i - x_{i-1}$ and $\Delta y_i = y_i - y_{i-1}$. We can compute u_i by initializing $u_0 = 0$ (at p_0) and:

$$u_i = u_{i-1} + \sqrt{\Delta x_i^2 + \Delta y_i^2}, \tag{7}$$

for $i = 1, ..., n$, thus:

$$u_i = \sum_{k=1}^{i} \sqrt{\Delta x_k^2 + \Delta y_k^2} \tag{8}$$

Recall that curve C remains the same shape along the curve U. Denote C corresponding to u_1, u_2, ... u_n and perpendicular to U at those u_i point as C_0, C_1, C_2, ..., C_n, such that: for the curve C_i, $i = 0, ..., n$. Let $\Delta x_{i,j} = x_{i,j} - x_{i,j-1}$, $\Delta y_{i,j} = y_{i,j} - y_{i,j-1}$, $\Delta z_j = z_j - z_{j-1}$, $j = 0, ..., m$. $v_{i,j}$ can be computed by initializing $v_{i,0} = 0$ and:

$$v_{i,j} = v_{i,j-1} + \sqrt{\Delta x_{i,j}^2 + \Delta y_{i,j}^2 + \Delta z_j^2}, \tag{9}$$

for $i = 1, ..., n$, $j = 1, ...m$, thus:

$$v_{i,j} = \sum_{k=1}^{j} \sqrt{\Delta x_{i,k}^2 + \Delta y_{i,k}^2 + \Delta z_k^2} \tag{10}$$

4) **Discretization of uv space:** We now have (u_1, v_1), (u_2, v_2), $\ldots$, (u_m, v_m), which are based on the order of the triangle vertices or edge intersections with C, could be sparsely distributed, and are NOT evenly distributed. Next, let Δu and Δv be the desired interval between two adjacent u values and v values respectively, such that $\Delta u \leq min(u_i - u_{i-1})$, and $\Delta v \leq min(v_i - v_{i-1})$, $i = 1, ..., m$. We can interpolate between u_{i-1} and u_i and between v_{i-1} and v_i to create new uv points for an even discretization of the uv space. Note that our method used to obtain (u_{i-1}, v_{i-1}) and (u_i, v_i) makes sure that the two points are on the same triangle in the 3D surface mesh. Hence, linear interpolation between the two points creates new uv points on the same triangle too, and the 3D coordinates of the new uv points can be determined in the Cartesian space easily.

We now achieve an evenly distributed uv grid, and each (u_j, v_k) point ($j = 1, ..., J$ and $k = 1, ..., K$) corresponds to a 3D point in the surface Cartesian coordinate system.

B. Deriving Constraint Equations for 3D Freeform Surfaces

With the uv grid established, we can relate each uv cell in the grid to a robot end-effector position and orientation cell, called an E cell [28], taking into account the task constraints on the end-effector position and orientation. Unlike in [28], which maps a uv cell to 3D Cartesian coordinates using parametric surface equations, here we need to map a uv cell directly to the triangle mesh of the surface to obtain its 3D Cartesian coordinates, which in turn can be used to constrain the robot end-effector position in the corresponding E cell. Next, the end-effector orientation in the E cell can be determined and constrained by applying task constraints.

1) **Position Constraint on the End-effector:**
Let T denote the triangle a uv-cell is located on the 3D polygon mesh, which is described by vertices with 3D position vectors $\boldsymbol{\tau_1}$, $\boldsymbol{\tau_2}$, and $\boldsymbol{\tau_3}$ in Cartesian space and with 2D position vectors $\boldsymbol{\mu_1}$, $\boldsymbol{\mu_2}$, and $\boldsymbol{\mu_3}$ in uv space. A point inside the triangle can be checked through those vertex positions [29].

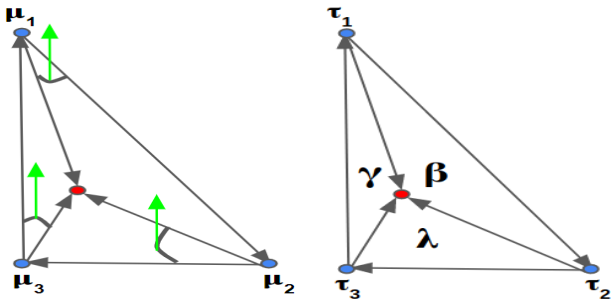
We denote $\boldsymbol{\mu_c}$ as the center position of the E-cell corresponding to the uv-cell in uv space. Let $\boldsymbol{\mu_{12}} = (\boldsymbol{\mu_2} - \boldsymbol{\mu_1})$, $\boldsymbol{\mu_{23}} = (\boldsymbol{\mu_3} - \boldsymbol{\mu_2})$, $\boldsymbol{\mu_{31}} = (\boldsymbol{\mu_1} - \boldsymbol{\mu_3})$, $\boldsymbol{\mu_{c1}} = (\boldsymbol{\mu_1} - \boldsymbol{\mu_c})$, $\boldsymbol{\mu_{c2}} = (\boldsymbol{\mu_2} - \boldsymbol{\mu_c})$, and $\boldsymbol{\mu_{c3}} = (\boldsymbol{\mu_3} - \boldsymbol{\mu_c})$. The E-cell is within the triangle face T in uv space if (illustrated in Fig. 3(a)):

$$0 < \begin{cases} \boldsymbol{\mu_{12}} \times \boldsymbol{\mu_{c1}} \\ \boldsymbol{\mu_{23}} \times \boldsymbol{\mu_{c2}} \\ \boldsymbol{\mu_{31}} \times \boldsymbol{\mu_{c3}} \end{cases} \quad \text{or} \quad 0 > \begin{cases} \boldsymbol{\mu_{12}} \times \boldsymbol{\mu_{c1}} \\ \boldsymbol{\mu_{23}} \times \boldsymbol{\mu_{c2}} \\ \boldsymbol{\mu_{31}} \times \boldsymbol{\mu_{c3}} \end{cases} \tag{11}$$

Next, with the triangle face T and its corresponding vertex positions $\boldsymbol{\tau_1}$, $\boldsymbol{\tau_2}$, and $\boldsymbol{\tau_3}$ and A as the surface area of T, we can compute:

$$\mathbf{A} = \|(\boldsymbol{\tau_2} - \boldsymbol{\tau_1}) \times (\boldsymbol{\tau_3} - \boldsymbol{\tau_1})\|/\mathbf{2} \tag{12}$$

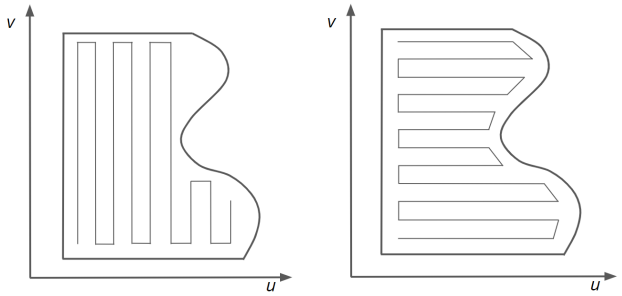
We denote the end-effector position in Cartesian space with respect to the surface coordinate frame as $\boldsymbol{\varepsilon}$. Let $\delta\varepsilon << 1$



(a) Cross products (green), center (red), and vertices $\boldsymbol{\mu_1}$, $\boldsymbol{\mu_2}$, and $\boldsymbol{\mu_3}$ in uv space.

(b) Subtriangles β, γ, and λ created by vertex position vectors $\boldsymbol{\tau_1}$, $\boldsymbol{\tau_2}$, and $\boldsymbol{\tau_3}$ and end-effector position $\boldsymbol{\varepsilon}$ (red) in Cartesian space.

Fig. 3: Illustration for eq. 11, 12, and 13 on triangle T in uv space and Cartesian space.



(a) Vertical raster scan pattern cannot cover the entire uv space.

(b) Horizontal raster scan pattern can cover the entire uv space.

Fig. 4: Different coverage patterns on concave surface in uv space.

be the allowable error for $\boldsymbol{\varepsilon}$ by the positional task constraint, c_p. We now compute the area of the three subtriangles, λ, β, and γ, created by the $\boldsymbol{\varepsilon}$ and the triangle face vertices $\boldsymbol{\tau_1}$, $\boldsymbol{\tau_2}$, and $\boldsymbol{\tau_3}$ (illustrated in Fig. 3(b)):

$$\boldsymbol{\lambda} = \frac{\|(\boldsymbol{\tau_2} - \boldsymbol{\varepsilon}) \times (\boldsymbol{\tau_3} - \boldsymbol{\varepsilon})\|}{2 \cdot \mathbf{A}}, \quad \boldsymbol{\beta} = \frac{\|(\boldsymbol{\tau_3} - \boldsymbol{\varepsilon}) \times (\boldsymbol{\tau_1} - \boldsymbol{\varepsilon})\|}{2 \cdot \mathbf{A}}, \quad \boldsymbol{\gamma} = \frac{\|(\boldsymbol{\tau_1} - \boldsymbol{\varepsilon}) \times (\boldsymbol{\tau_2} - \boldsymbol{\varepsilon})\|}{2 \cdot \mathbf{A}} \tag{13}$$

The position task constraint, c_p, on the end-effector is:

$$1 - \delta\varepsilon \leq \boldsymbol{\lambda} + \boldsymbol{\beta} + \boldsymbol{\gamma} \leq 1 + \delta\varepsilon. \tag{14}$$

In some manipulator applications, the end-effector tip is not placed directly on the surface but is offset from the surface at a distance d, with unit normal $\mathbf{b}$ at each vertex position defining the offset direction. Let $\mathbf{b_i}$, the unit normal of the vertex with position $\boldsymbol{\tau_i}$, be the average of the normals of the triangle faces that contain the vertex $\boldsymbol{\tau_i}$. Since the surface application area of the end-effector is still constrained by the plane created by T, we adjust eq. 13:

$$\boldsymbol{\lambda} = \frac{\|((\boldsymbol{\tau_2} + \mathbf{b_2} \cdot d) - \boldsymbol{\varepsilon}) \times ((\boldsymbol{\tau_3} + \mathbf{b_3} \cdot d) - \boldsymbol{\varepsilon})\|}{2 \cdot \mathbf{A}}, \quad \boldsymbol{\beta} = \frac{\|((\boldsymbol{\tau_3} + \mathbf{b_3} \cdot d) - \boldsymbol{\varepsilon}) \times ((\boldsymbol{\tau_1} + \mathbf{b_1} \cdot d) - \boldsymbol{\varepsilon})\|}{2 \cdot \mathbf{A}}, \quad \boldsymbol{\gamma} = \frac{\|((\boldsymbol{\tau_1} + \mathbf{b_1} \cdot d) - \boldsymbol{\varepsilon}) \times ((\boldsymbol{\tau_2} + \mathbf{b_2} \cdot d) - \boldsymbol{\varepsilon})\|}{2 \cdot \mathbf{A}} \tag{15}$$

2) **Orientation Constraint on the End-Effector:** The end-effector orientation is constrained by the triangle face normal of T in Cartesian space. We denote the end-effector approach vector as the unit approach vector $\mathbf{a}$ (along the z axis of the end-effector) and the desired approach direction to the surface as $\mathbf{b}$, such that:

$$\mathbf{a} = [r_{13}, r_{23}, r_{33}]^T \tag{16}$$

where r_{**} is from the rotation matrix of the manipulator transformation matrix, and $\mathbf{b}$ is the normal of triangle face T created by $\boldsymbol{\tau_1}$, $\boldsymbol{\tau_2}$, and $\boldsymbol{\tau_3}$. The orientation task constraint allows $\mathbf{a}$ to deviate from $\mathbf{b}$ within a small angle α at each position. Thus, we have the following orientation task constraint, c_o:

$$-\mathbf{b} \cdot \mathbf{a} \leq \cos\alpha \tag{17}$$

Now, with the position and orientation task constraints c_p and c_o on the robot end-effector expressed in terms of 3D coordinates of mesh vertices, they can be further related to the robot manipulator constraints on link parameters and joint variables via forward kinematics [28], which can then be used by the method in [28] to check coverage motion feasibility on the freeform surface.

C. Converting Coverage Pattern from uv space to Cartesian space

Once a 3D freeform surface is checked feasible for constrained coverage motion given the task constraints and a robot manipulator, a suitable coverage path pattern can be put on the surface through the uv grid on the surface.

The 2D uv grid makes it easy to detect if a coverage pattern is suitable. The convexity of a flattened (planar) uv space can be used to decide whether a coverage pattern can be applied, since for some concave planar region, certain coverage patterns cannot provide entire surface coverage (Fig. 4). Such a concave region can be decomposed into convex regions [30] so that each convex region can be covered by a desired coverage pattern.

Now, we denote $\mathbf{H}$ as a coverage path on the uv grid with $\mathbf{h_i}$ being the ith waypoint on the path such that $\mathbf{H} = [\mathbf{h_1}, \mathbf{h_2}, ..., \mathbf{h_m}]$. Every point in the uv grid that is within a triangle face (use eq. 11) is a viable location for $\mathbf{h_i}$.

Algorithm 1 outlines how to create a horizontal raster pattern $\mathbf{H}$ in the uv space. Let ω be the interval between two discrete (u, v) points and $\frac{\omega}{2}$ be the distance between viable (u, v) points and polygon mesh edges. ω can be determined based on the task constraint in an application, such as the spray width in spray painting. We can rediscretize the uv grid using ω. The algorithm begins setting variables u^* and v^* to those of the top-left viable point coordinates in the uv grid. The algorithm then iterates through each v to put the coordinates of all viable points between the minimum and maximum u values for each v into the $\mathbf{H}$ in the order of

Algorithm 1: Create horizontal raster H in uv space

Input uv grid; $i = 0$; $direction = Right$;
$(u^*, v^*) = (u_{min}(v_{max}), v_{max})\ \forall\ u, v \in uv$ grid;
while $v^* \geq v_{min}$ **do**
  $H(i) = (u^*, v^*)$; $i++$;
  if $direction == Right$ **then**
    $direction = Left$;
    While $u^* < u_{max}(v^*)$ **do:**
      $u^* = u^* + \omega$ **;** $H(i) = (u^*, v^*)$**;** $i++$**;**
  else
    $direction = Right$;
    While $u^* < u_{min}(v^*)$ **do:**
      $u^* = u^* - \omega$ **;** $H(i) = (u^*, v^*)$**;** $i++$**;**
  end
  $H(i) = (u^*, v^*)$; $i++$;
  $v^* = v^* - \omega$;
end

alternating right and left scan directions. To generate vertical raster scans, Algorithm 1 is altered by swapping u and v coordinates values, and after the algorithm finishes, swap the u and v coordinates for each waypoint in $\mathbf{H}$.

We now have a pattern $\mathbf{H}$ covering a 3D freeform surface in the uv space and need to convert the waypoints to Cartesian space. For each waypoint $\mathbf{h_i}$, we find the corresponding triangle face T whose vertices enclose the waypoint (eq. 11). Once a triangle face T is assigned to every waypoint $\mathbf{h_i}$, the corresponding vertices $\boldsymbol{\mu_1}$, $\boldsymbol{\mu_2}$, and $\boldsymbol{\mu_3}$ are used to compute baycentric coordinates $\mathbf{bu}$ and $\mathbf{bv}$:

$$\begin{bmatrix} \mathbf{bu} \\ \mathbf{bv} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu_{2u}} - \boldsymbol{\mu_{1u}} & \boldsymbol{\mu_{3u}} - \boldsymbol{\mu_{1u}} & \boldsymbol{\mu_{1u}} \\ \boldsymbol{\mu_{2v}} - \boldsymbol{\mu_{1v}} & \boldsymbol{\mu_{3v}} - \boldsymbol{\mu_{1v}} & \boldsymbol{\mu_{1v}} \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{h_u} \\ \mathbf{h_v} \\ 1 \end{bmatrix} \quad (18)$$

We denote $\mathbf{P}$ as the coverage path in Cartesian space with $\mathbf{p_i}$ being the ith waypoint on the path such that $\mathbf{P} = [\mathbf{p_1}, \mathbf{p_2}, ..., \mathbf{p_m}]$. Finally, the baycentric coordinates with the corresponding triangle face vertices in Cartesian space, $\boldsymbol{\tau_1}$, $\boldsymbol{\tau_2}$, and $\boldsymbol{\tau_3}$, are used to transform the waypoints $\mathbf{h_i}$ into 3D cartesian space, $\mathbf{p_i}$:.

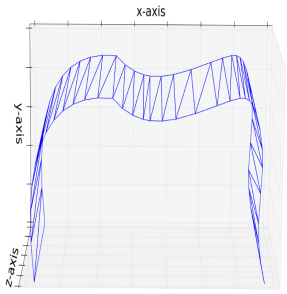
$$\begin{aligned} \mathbf{p_x} &= \boldsymbol{\tau_{1x}} + (\boldsymbol{\tau_{2x}} - \boldsymbol{\tau_{1x}}) \cdot \mathbf{bu} + (\boldsymbol{\tau_{3x}} - \boldsymbol{\tau_{1x}}) \cdot \mathbf{bv} \\ \mathbf{p_y} &= \boldsymbol{\tau_{1y}} + (\boldsymbol{\tau_{2y}} - \boldsymbol{\tau_{1y}}) \cdot \mathbf{bu} + (\boldsymbol{\tau_{3y}} - \boldsymbol{\tau_{1y}}) \cdot \mathbf{bv} \\ \mathbf{p_z} &= \boldsymbol{\tau_{1z}} + (\boldsymbol{\tau_{2z}} - \boldsymbol{\tau_{1z}}) \cdot \mathbf{bu} + (\boldsymbol{\tau_{3z}} - \boldsymbol{\tau_{1z}}) \cdot \mathbf{bv} \end{aligned} \quad (19)$$

In applications where the robot end-effector position is not located directly on the surface, an offset d needs to be applied to each waypoint $\mathbf{p_i}$. The corresponding triangle face normal $\mathbf{b}$ (unit vector) for each waypoint may be used to compute new offset path $\mathbf{P}$:

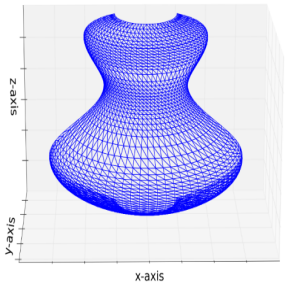
$$\mathbf{p_i} = \mathbf{p_i} + \mathbf{b} \cdot d \quad (20)$$

IV. Implementation and Results

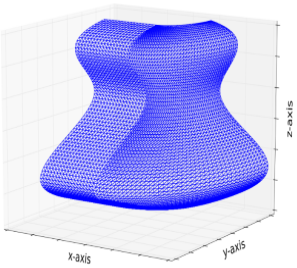
To test our method, we used Solidworks to generate polygon meshes of a SoT, SoR, and SoTR, see Fig. 5, containing 68, 3743, and 9200 triangle faces respectively. The meshes were transformed into evenly spaced uv grids, which were used for feasibility checking and generating coverage patterns. The coverage patterns were then converted from the uv space to Cartesian space.



(a) Polygon mesh of SoT.



(b) Polygon mesh of SoR.



(c) Polygon mesh of SoTR.

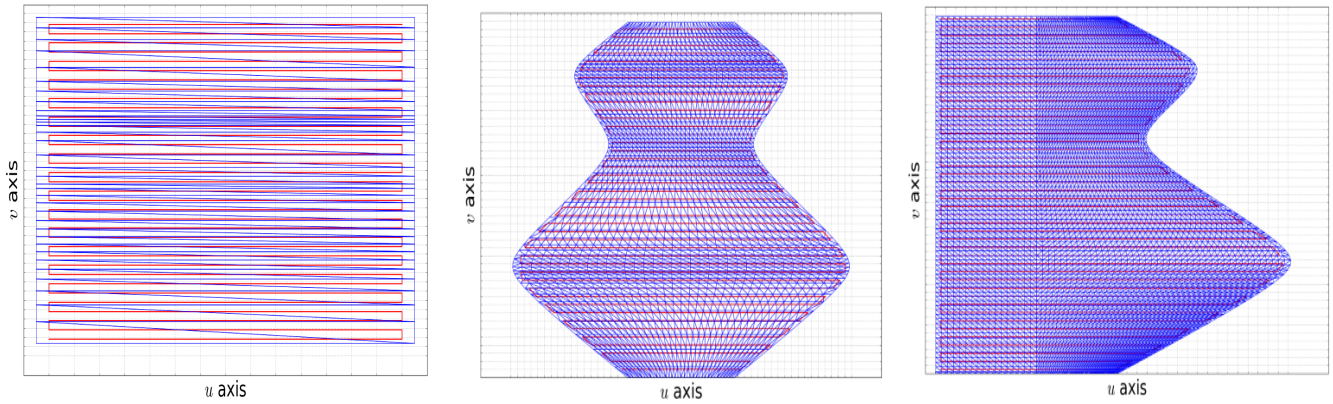
Fig. 5: Polygon meshes of (a) SoT, (b) SoR, and (c) SoTR, as shown in Fig. 1.

A. Results of uv Grid Generation and Constrained Motion Feasibility Checking

The mesh vertices of each surface were transformed into the uv space and corresponding triangle faces were reconstructed in the uv space to give a 2D representation of the polygon mesh. See Fig. 6 for resulting polygon mesh in uv space. To check coverage feasibility, we used a Franka Emika Panda manipulator which contains seven revolute joints, with the joint vector $\mathbf{q} = [\theta_1, \theta_2, ..., \theta_7]^T$ and link parameters $\mathbf{l} = [d_1, d_3, a_4, a_5, d_5, a_7] = [0.3, 0.3, 0.08, -0.08, 0.3, 0.08]$(m). Values for the link parameters and value ranges for the joint variables are expressed in the end-effector positions and orientations via forward kinematics, which are then related to the task constraints in the joint-space task constraint equations. The end-effector orientation constraint parameter α is set to $20°$. Equations 14 and 17 were used as constraint equations in Alg. 2 in [28].

B. Coverage Pattern Results

We create coverage path patterns in the uv space using an application parameter ω. Fig. 6 shows the uv grid re-descretized using ω, the generated raster coverage pattern,



(a) uv gird on SoT with coverage pattern. (b) uv grid on SoR with coverage pattern. (c) uv grid on SoTR with coverage pattern.

Fig. 6: Generated uv grid (blue) with applied horizontal raster coverage patterns $\mathbf{H}$ (red) of the three freeform surfaces.

$\mathbf{H}$, and surface uv polygon mesh. Fig. 7 shows the resulting coverage patterns on the 3D freeform surfaces.

Fig. 8 shows an example result of a coverage pattern that was generated on a cone surface using our method. This pattern is not trivial to generate using other methods but is easily generated and evenly spaced using our uv grid. The vertical raster pattern is straight near the center yet spiral on either side. This is a simple example to demonstrate that using the uv grid generated by our method, non-trivial coverage patterns may be produced on 3D freeform surfaces.

Recall that these patterns provide end-effector coverage paths that are feasible to produce continuous manipulator joint paths for successful execution of the coverage (see Section III B and C).

C. Discussion

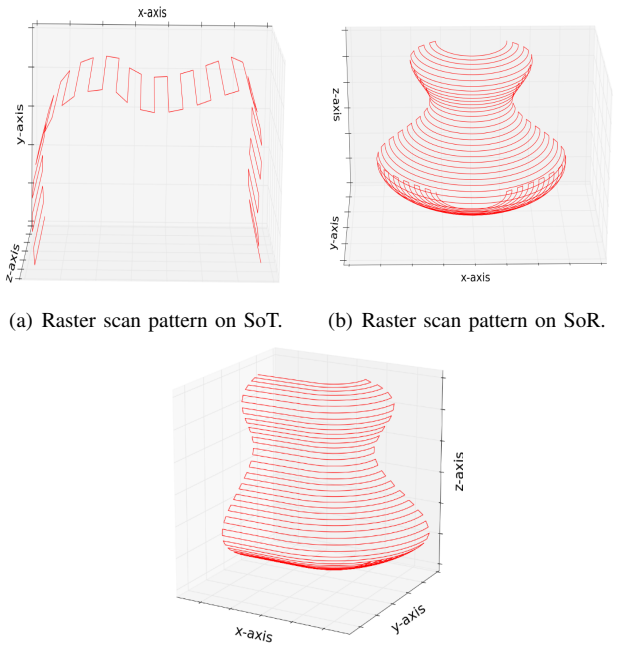
In the figures shown, we flattened the uv space and used straight-line axes to represent u and v. This results in some distortion of the polygon meshes, however, surface area is approximately maintained using the topology of the 3D polygon mesh.

Although we only showed the raster scan pattern in the examples, other suitable coverage patterns can be applied depending on the surface. For example, for the SoT example surface, a raster scan pattern perpendicular to the one shown can also be applied. The key point is that our generation of the uv grid allows evenly spaced application of coverage patterns.

Pattern waypoints can be further interpolated between uv cells and at the turns of a raster pattern in $\mathbf{H}$ to produce a coverage path $\mathbf{P}$ that follows the surface more smoothly.

V. Conclusions

This paper introduced a novel method for generating a uv grid on a 3D freeform surface to facilitate coverage motion feasibility checking and application of more evenly spaced coverage path patterns. Our method converted polygon mesh vertices from 3D Cartesian space to the uv space and discretize the uv space evenly. The points in uv space



(a) Raster scan pattern on SoT. (b) Raster scan pattern on SoR.

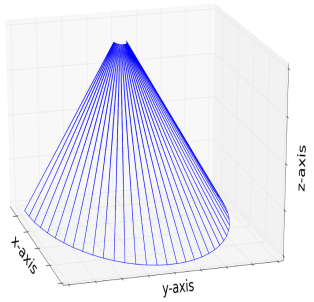
(c) Raster scan pattern on SoTR.

Fig. 7: Raster scan coverage patterns on different surfaces.

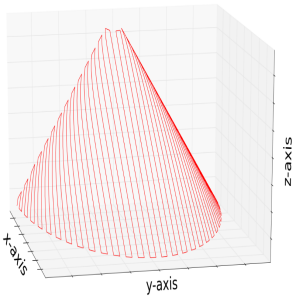
were also mapped back to the 3D Cartesian space. Our method further used the surface's polygon mesh to derive task constraint equations on a robot end-effector position and orientation to be used during coverage feasibility checking. Finally, coverage patterns were applied in the uv space and converted to the Cartesian space onto the freeform surface.

The introduced methods have been implemented and tested on example surfaces of three defined types: surface of translation, surface of rotation, and surface of combined translation and rotation.

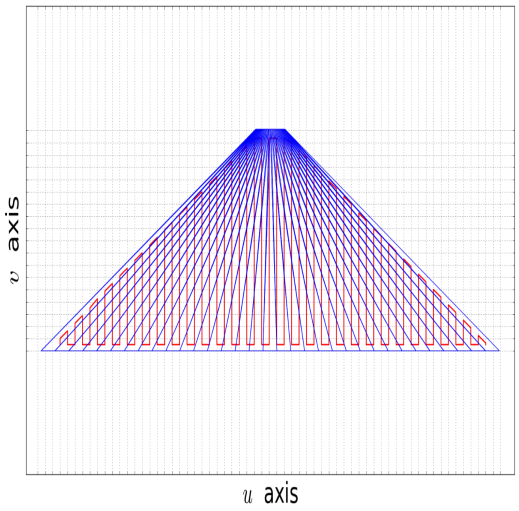
Future research includes expanding the approach to more types of complex freeform surfaces, those that do not fall within the three categories described, and to include methods



(a) Polygon mesh of cone surface (SoR) in Cartesian space.



(b) Spiral 3D raster scan coverage pattern on cone surface (SoR).



(c) Polygon mesh of cone surface (SoR) with vertical raster scan coverage pattern.

Fig. 8: Spiral coverage pattern results on cone surface (SoR).

of human interaction to divide an arbitrary freeform surface into well-defined surface types.

REFERENCES

[1] C. Chen, S. Gojon, Y. Xie, S. Yin, C. Verdy, Z. Ren, H. Liao, S. Deng, "A novel spiral trajectory for damage component recovery with cold spray," *Surface and Coatings Technology 309*, vol. 309, pp. 719-728, 2017.

[2] W. Chen, J. Liu, Y. Tang, H. Ge, "Automatic Spray Trajectory Optimization on Bezier Surface," *Electronics*, vol.8, no. 2, pp. 168-184, 2019.

[3] M. Andulkar, S. Chiddarwar,"Incremental approach for trajectory generation of spray painting robot," *Industrial Robot: An International Journal*, vol. 42, no. 3, pp.228-241, 2015.

[4] H.Chen, N.Xi, W. Sheng, M. Song, Y. Chen, "CAD-based automated robot trajectory planning for spray painting of free-form surfaces," *Industrial Robot: An International Journal*, vol. 29, no.5, pp. 426-433, 2002.

[5] G. Teodora, G. Florin, M. Gheorghe, "Virtual Planning of Robot Trajectories for Spray Painting Applications," *Applied Mechanics and Materials*, vol. 658, pp. 632-637, 2014.

[6] G.Trigatti, P.Boscariol, L. Scalera, D. Pillan, A. Gasparetto, "A new path-constrained trajectory planning strategy for spray painting robots," *The International Journal of Advanced Manufacturing Technology*, vol. 98, pp. 2287-2296, 2018.

[7] H.Chen, T. Fuhlbrigge, X. Li, "A review of CAD-based robot path planning for spray painting," *Industrial Robot: An International Journal*, vol. 36, no. 1, pp. 45-50, 2009.

[8] G. Liu, X. Sun, Y. Liu, C. Li, X. Zhang, "Automatic spraying motion planning of a shotcrete manipulator," *Intelligent Service Robotics (2021)*, https://doi.org/10.1007/s11370-021-00348-9.

[9] X. Ye, L. Lui, L. Hou, Y. Duan, Y. Wu, "Laser Ablation Manipulator Coverage Path Planning Method Based on an Improved Ant Colony Algorithm," *Applied Sciences*, vol.10, no. 23, pp.8641, 2020.

[10] Chen, C.H.; Song, K.T, "Complete coverage motion control of a cleaning robot using infrared sensors," in *Proc. IEEE International Conference on Mechatronics (ICM)*, Taipei, Taiwan, pp. 543–548, July 2005.

[11] T. Lee, S. Baek, Y. Choi, S. Oh, "Smooth coverage path planning and control of mobile robots based on high-resolution grid map representation," *Robotics and Autonomous Systems*, vol. 59, no. 10, pp. 801-812, 2011.

[12] J.Song, S. Gupta,"ε*: An Online Coverage Path Planning Algorithm," *IEEE Transactions on Robotics*, vol. 34, no.2, pp. 526-533, 2018.

[13] E. Galceran, M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258-1276, 2013.

[14] C. Tan, R. Mohd-mokhtar, M. Arshad, "A Comprehensive Review of Coverage Path Planning in Robotics Using Classical and Heuristic Algorithms," *IEEE Access*, vol. 9, pp. 119310-119342, 2021.

[15] K. Schid, H. Hirshmuller, A. Domel, "View Planning for multi-stereo 3D Reconstruction using an autonomous multicopter," *Journal of Intelligent and Robotic Systems*, vol. 65, no. 1-4, pp. 309-323, 2012.

[16] J. Mooney, E. Johnson, "A Comparison of Automatic Nap-of-the-Earth Guidance Strategies for Helicopters," *Journal of Field Robotics*, Vol. 33, no. 1, pp. 1-17, 2014.

[17] M. Na, J. Hyun, J. Song, "CAD-based View Planning with Globally Consistent Registration for Robotic Inspection,"*International Journal of Precision Engineering and Manufacturing*, vol. 22, no. 8, pp. 1391-1399, 2021.

[18] J. Jin, L. Tang, "Coverage Path Planning on Three-Dimensional Terrain for Arable Farming," *Journal of Field Robotics*, vol. 22, pp.424-440, 2011.

[19] L. Santos, F. Santos,S. Pires, E.J. Pires, A. Valente, P. Costa, "Planning for ground robots in agriculture: A short review," in *Proc. of the 2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, Ponta Delgada, Portugal, pp. 61–66, April 2020.

[20] P. Atkar, H. Choset, A. Rizzi, E. Acar, "Exact Cellular Decomposition of Closed Orientable Surfaces Embedded in R," *International Conference on Robotics and Automation*, vol. 1, pp. 699 - 704, 2001.

[21] T. Yang, J. Miro, Q. Lai, "Cellular Decomposition for Nonrepetitive Coverage Task with Minimum Discontinuities," in *IEEE/ASME Transactions on Mechatronics*, vol. 25, No. 4, pp. 1698-1708, August 2020.

[22] P. Atkar, H. Choset, A. Rizzi, "Towards Optimal Coverage of 2-Dimensional Surfaces Embedded in R: Choice of start Curve," in *Proc. International Conference on Intelligent Robotics and Systems*, vol. 4, pp. 3581-3587, Oct. 2003.

[23] Z. Kingston, M. Moll, and L. E. Kavraki, "Decoupling constraints from sampling-based planners," in *Proc. Int. Symp. of Robot.* Res., vol. 38, no. 10-11, pp. 1151-1178, 2017.

[24] M. Stilman, "Task constrained motion planning in robot joint space," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3074–3081, 2007.

[25] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation planning on constraint manifolds," *Proc. IEEE International Conference on Robotics and Automation*, IEEE, pp. 625–632, 2009.

[26] L. Jaillet and J. M. Porta, "Path planning under kinematic constraints by rapidly exploring manifolds," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 105–117, 2013.

[27] T. McMahon, S. Thomas, and N. M. Amato, "Sampling-based motion planning with reachable volumes: Theoretical foundations," in *Proc. IEEE International Conference on Robotics and Automation*, pp. 6514–6521, 2014.

[28] S. McGovern, J. Xiao, "Efficient Feasibility Checking on Continuous Coverage Motion for Constrained Manipulation," *IEEE 17th International Conference on Automation Science and Engineering (2021)*

[29] E. Angel, D. Shreiner, *Interactive Computer Graphics: A Top-down Approach with Shader-Based OPENGL* 6th edition, Addison-Wesley 2012.

[30] L Nielson, I. Sung, P. Nielson, "Convex Decomposition for a Coverage Path Planning for Autonomous Vehicles: Interior Extension of Edges," *Sensors (Basel, Switzerland)*, vol. 19, iss.19, 2019.