AIAA SciTech Forum
7-11 January 2019, San Diego, California
AIAA Scitech 2019 Forum

10.2514/6.2019-1717

Flends: Generalized Fillets via B-splines

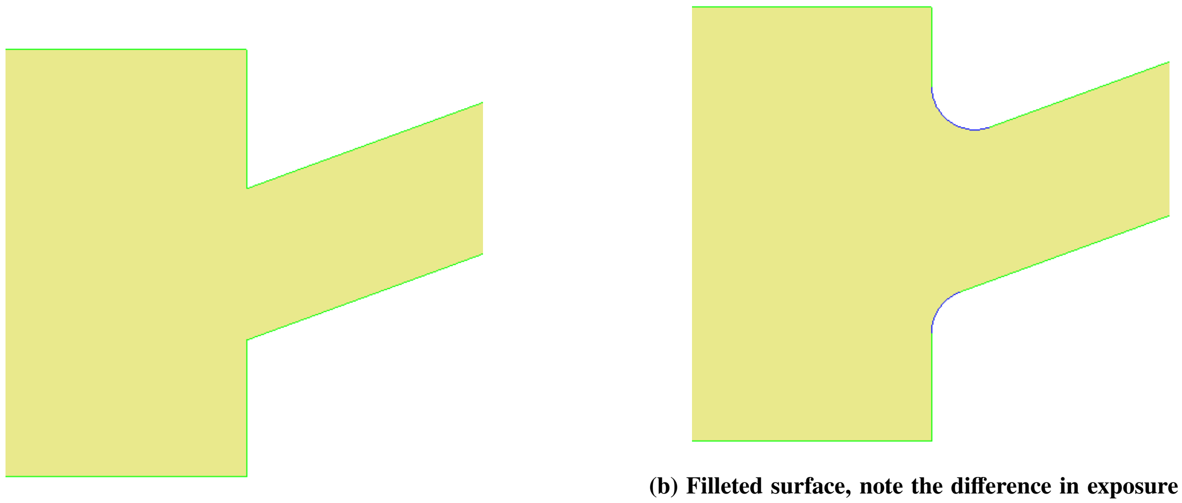
Zachary D. Eager* and John F. Dannenhoffer III†
Aerospace Computational Methods Laboratory (ACML)
Syracuse University, Syracuse, New York, 13244, USA

The methods for creating transition surfaces (e.g. fillets, chamfers, etc.), as implemented in many solid modelers, are restrictive and fragile. For example, the extent of a fillet on the two surfaces are linked and cannot be specified separately. Also, both fillets and chamfers yield curvature discontinuities where they meet the original surfaces, which is known to be sub-optimal aerodynamically. In this paper, we present more robust and flexible methods for creating transition surfaces using B-spline surfaces. These methods are solid modeler agnostic, however we've implemented them using the Engineering Sketch Pad (ESP).

I. Introduction

A common desire in solid modeling is the ability to create a transition surface between two adjacent faces of a solid, as well as creating a transition surface between two faces of two disjoint solids. Currently existing approaches available in most solid modeling software include fillets, chamfers, sweeps, lofts, etc. These methods typically take minimal input from the user and are thus restrictive and can additionally be highly fragile depending on the complexity of the geometry. Our goal is to create methods that allow the creation of transition surfaces that are more flexible and robust than the traditional methods. The solid modeling software we will focus on in this paper is the Engineering Sketch Pad (ESP) [1]. We note that the methods presented here are not inherently specific to this software and could be implemented for any system that provides analogous supporting functions.

II. Generalized Fillets



(a) Original surface.

(b) Filleted surface, note the difference in exposure on the upper edge and lower edges.

Fig. 1 Demonstration of unequal exposure on the original surfaces by a fillet.

The issue with fillets and chamfers is that they are, by definition, restrictive. The user input for these methods is typically a fixed radius, resulting in minimal control over where fillet/chamfer meets the original surfaces (see Fig. 1). Furthermore, since fillets are created by circular arcs of the input radius and chamfers are ruled, there is inherently a

*Ph. D. Student, Mechanical and Aerospace Engineering
†Associate Professor, Mechanical and Aerospace Engineering, AIAA Associate Fellow

discontinuity in curvature where the fillet/chamfer meets the original surfaces. It can also be common for the solid modeler to fail when creating a fillet/chamfer involving complex geometries.

In this section, we develop a method for producing a generalized fillet surface where the user has control over the curvature of surface as well as more control over where the generalized fillet surface meets the original surfaces. We also demonstrate how this method can be more robust than ESP's fillet and chamfer methods. For the remainder of this manuscript, this generalized fillet surface will be referred to as a "flend."

A. Blending Surface Generation

Consider the simple case where we wish to create a flend between two intersected boxes (see Fig. 2(a)). We assume at this stage that the locations where the flend meets the original surfaces has been determined, denoted by the offset edges on the two boxes (see Fig. 2(b)). For this example, the offset edges are 0.25 units from the intersection edge such that a fillet or chamfer of radius 0.25 will join the two surfaces at these edges.

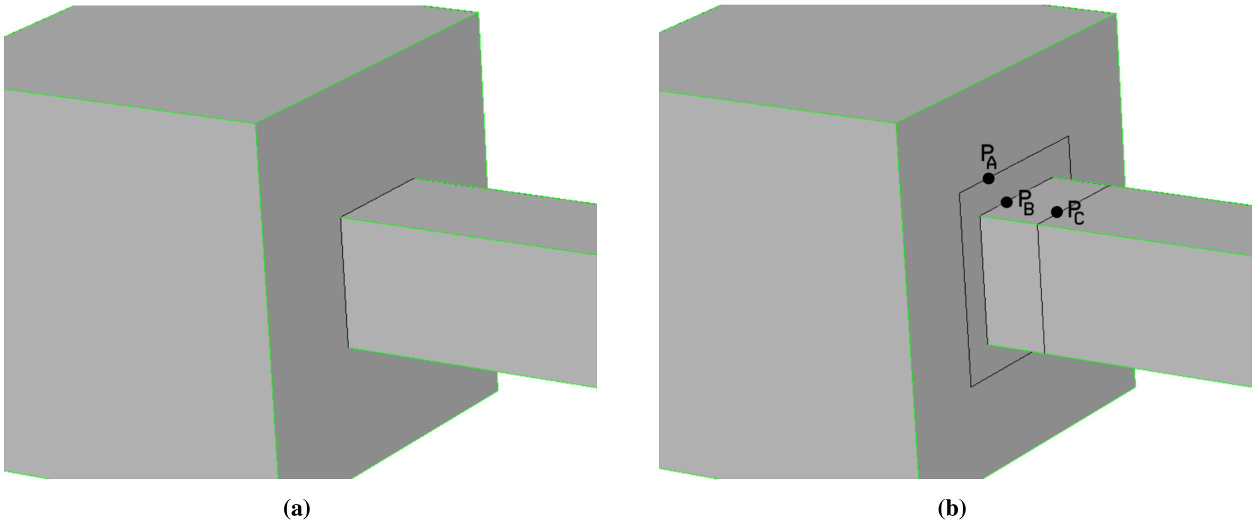


Fig. 2 The original intersected boxes (a) and the intersected boxes with the offset edges and corresponding points (b).

We begin by determining a set of corresponding points for an edge group. An edge group consists of the first offset edge, the intersection edge, and second offset edge (the order of offset edges is arbitrary). Therefore, the number of edge groups is equal to the number of intersection edges (e.g. in Fig. 2(b), there are four edge groups). Assuming that each of the three edges in a group is parameterized on $t \in [a, b]$, the corresponding points would be the points on each edge for a given value of t. We will call these points P_A, P_B, and P_C, respectively (see Fig. 2(b)).

Next, we compute P_I, the three-plane intersection [2] of: (1) the plane that is parallel to the surface at P_A, (2) the plane perpendicular to the intersection edge at P_B, and (3) the plane that is parallel to the surface at P_C.

Consider first the 2D plane formed by the points P_A, P_I, and P_C (see Fig. 3). In this plane, we define a clamped, cubic B-spline (of equal knot spacing) with 5 control points (P_0, P_1, ..., P_4), and introduce a parameter called "dip"$\in [0, 1]$ for controlling the curvature. We take the beginning and end points (P_0 and P_4) and place them at the points where the B-spline will meet the original surfaces, P_A and P_C, respectively. For the second point (P_1), we take the distance between P_0 and P_I (the intersection point) and multiply it by the dip level. We choose point P_1 to be this distance away from P_0 in the direction of P_I (tangent to original surface at P_0), $P_1 = P_0 + |P_0 - P_I| \cdot \text{dip}$. The fourth point ($P_3$) is determined analogously using P_4 and P_I. We then take the distance between P_1 (computed previously) and P_I and multiply it by the dip level. We choose point P^* to be this distance away from P_1 in the direction of P_I, $P^* = P_1 + |P_1 - P_I| \cdot \text{dip}$. P^{**} is determined analogously using P_3 and P_I. Finally, we choose P_2 to be the midpoint of P^* and P^{**}, $P_2 = \frac{P^* + P^{**}}{2}$.

In order to create the 3D flend, we sample as many corresponding point sets as desired (as many points required to accurately represent the offset edges) for an edge group and compute the cubic B-spline control points for each corresponding point set. We then construct a B-spline surface which is degree 1 in one direction (along the intersection edge) and degree 3 in the other. The control points of the B-spline surface are the sets of the cubic B-spline control points computed previously; the control points (P_0, P_1, ..., P_4) are the points in the degree 3 direction. Note that choosing P_I as we did, and moving the inner control points (P_1 and P_3) towards this point, ensures C^1 continuity where

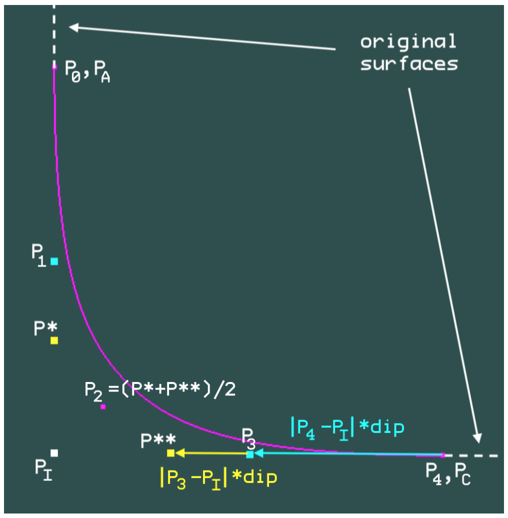


Fig. 3 5 control-point B-spline curve generation schematic.

the B-spline surface meets the original surfaces [3]. This process is repeated for each edge group, and the collection of surfaces created from each edge group makes up the flend. For the two intersecting boxes example, the flend consists of four B-spline surfaces stitched together. Fig. 4(a)-(f) show flends for various values of dip in juxtaposition to the fillet and chamfer. We observe that zero dip recreates the chamfer and a dip value of about 0.3 strongly resembles a traditional fillet for the given configuration. Note that, for higher values of dip (see Fig. 4(f)), flends do not have the strong discontinuity in curvature that is inherent when using fillets.

B. Offset Curve Generation & Intersection

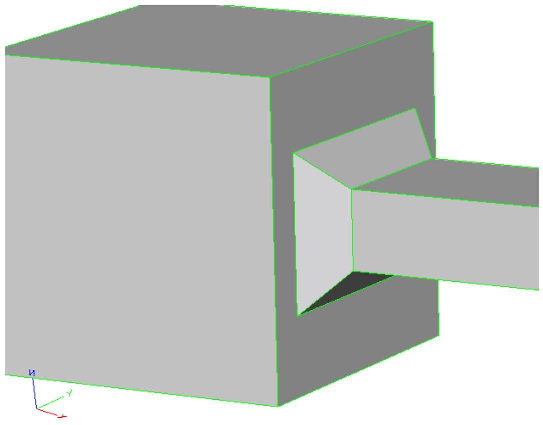
Suppose now that we are not given the offset edges on the surfaces to be joined. Furthermore, we would like to choose each offset edge to be an arbitrary distance (d_1 and d_2, respectively) from the intersection edge, independently of the dip.

For a given point on an intersection edge, we compute the corresponding offset point on the offset surface (the surface that will contain the offset) as follows. First, determine the outward normal on the offset surface at the given point as well as the direction vector on the intersection edge at this point. Typically, 3D surfaces are parameterized by two values, u and v. At any point on a surface we can query the derivative vectors w.r.t u and v, i.e. $\langle dx_u, dy_u, dz_u \rangle$ and $\langle dx_v, dy_v, dz_v \rangle$. The cross product of these yields the outward normal $\langle dx, dy, dz \rangle$ at that point on the surface. Further, the cross product of the outward normal on the offset surface and the direction vector $\langle rx, ry, rz \rangle$ on the intersection edge results in a new vector $\langle x, y, z \rangle$ which represents the desired traversal direction on the offset surface. See Fig. 5 for a pictorial description.

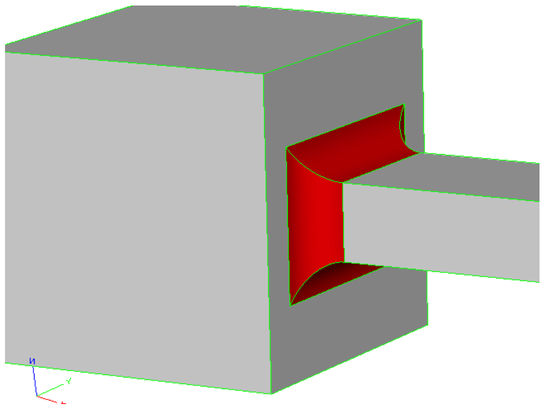
Beginning at the given point on the offset surface, we solve the following set of equations for $\langle du, dv \rangle$:

$$\begin{bmatrix} dx_u & dx_v \\ dy_u & dy_v \\ dz_u & dz_v \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}.$$

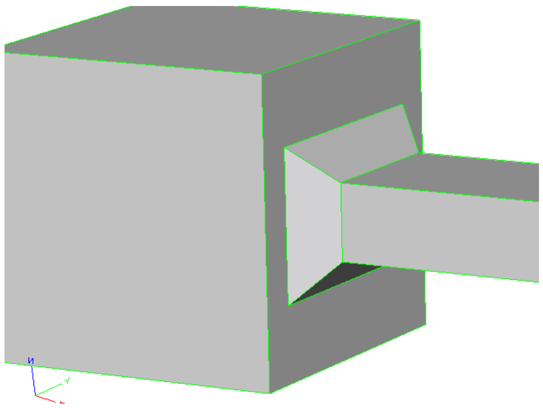
Since this is an over-determined system we compute the least-squares solution [4]. The resulting $\langle du, dv \rangle$ is the direction vector in uv-space on the offset surface that most closely matches the desired direction vector $\langle x, y, z \rangle$. We then take a "small" step in this $\langle du, dv \rangle$ direction on the offset surface. Since the directions of the unit vectors in uv-space are not constant w.r.t to xyz-space, we can not simply move the desired distance d in this direction. Instead, after taking a small step we re-compute $\langle dx_u, dy_u, dz_u \rangle$ and $\langle dx_v, dy_v, dz_v \rangle$ at this new (u, v) location and re-solve the equations for $\langle du, dv \rangle$ and take another small step in that direction. We repeat this until we've travelled the desired distance d along the offset surface from the given point on the intersection edge. We compute the offset points for an arbitrary number of points along the intersection edge (as many points as necessary to accurately represent the offset edge) and fit a cubic B-spline to this set of points. This B-spline curve is the offset edge on the offset surface.



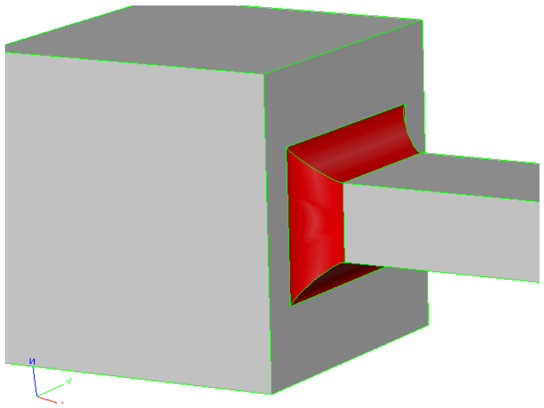
(a) chamfer.



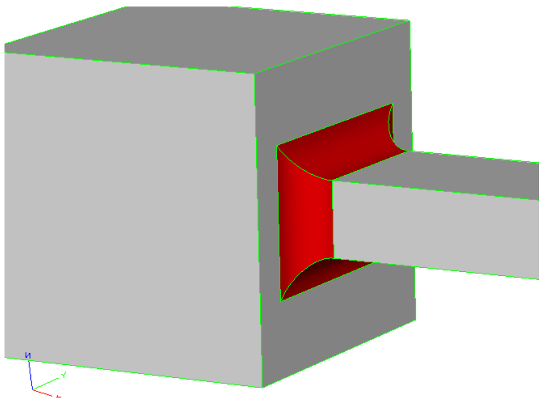
(b) fillet.



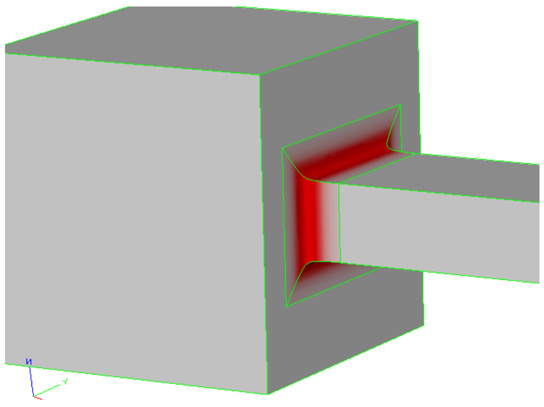
(c) Flend, dip = 0.0**.**



(d) Flend, dip = 0.15**.**



(e) Flend, dip = 0.3**.**



(f) Flend, dip = 0.7**.**

Fig. 4 Flends for various values of dip as well as the fillet and chamfer for two intersecting boxes. Colors showing maximum curvature.

Performing the aforementioned steps using the other surface (adjacent to the intersection edge) as the offset surface yields the offset edge on that surface. We emphasize that the distances for the offset edges on each face can be chosen independently. Together, the intersection edge and two newly created offset edges form an edge group (described in the previous subsection). Once we have determined all of the edge groups from the intersection edges, we can apply the logic outlined in the previous subsection to create a flend.

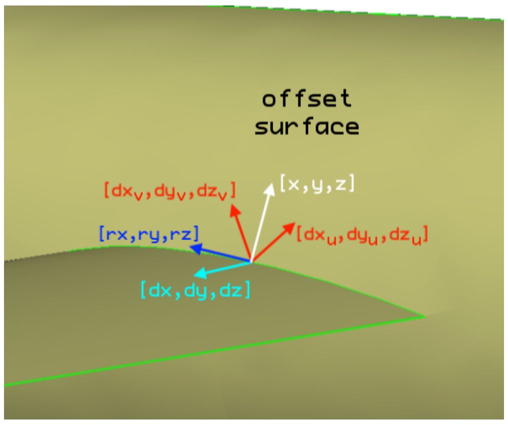


Fig. 5 Offset direction schematic.

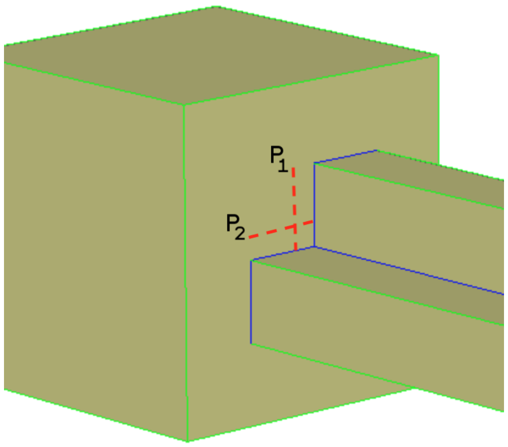


Fig. 6 Example where offset edges (denoted as red dashed lines) intersect.

It is often the case that offset edges will intersect. It is necessary to detect this when creating the edge groups so that the resulting flend does not intersect itself. Consider the example in Fig. 6 where we wish to create a fillet about the two intersected bodies. Clearly, the offset edges (denoted as red dashed lines) will intersect in the region shown. In order to detect this we currently apply a two-step process.

First, we take the two intersecting offset edges ($\mathbf{P}_1(t) = (u, v)$ and $\mathbf{P}_2(t) = (u, v)$) and divide each into several segments using the running parameter t for each. We determine the (u, v) on the offset surface at the beginning and end of each segment for each offset edge and linearize each curve in uv-space using these points for each segment. We then check all pairs of the linearized segments for intersection. If the points describing the first line segment are $(u_1, v_1), (u_2, v_2)$ and the second are $(u_3, v_3), (u_4, v_4)$, we can then apply the following formula [5]:

$$
\begin{aligned}
t_1 &= \frac{(v_3 - v_4)(u_1 - u_3) + (u_4 - u_3)(v_1 - v_3)}{(u_4 - u_3)(v_1 - v_2) - (u_1 - u_2)(v_4 - v_3)} \\
t_2 &= \frac{(v_1 - v_2)(u_1 - u_3) + (u_2 - u_1)(v_1 - v_3)}{(u_4 - u_3)(v_1 - v_2) - (u_1 - u_2)(v_4 - v_3)}
\end{aligned}
.
$$

If $0 \leq t_1 \leq 1$ and $0 \leq t_2 \leq 1$, then the linearized segments intersect at t_1 on the first segment and t_2 on the second.

Now, we perform a multivariate Newton's search [4] to find the intersection of the offset edges using the t_1, t_2 found previously as the initial guess. We define two functions which are both functions of two variables:

$$f_1(t_1, t_2) \equiv \mathbf{P}_1(t_1).u - \mathbf{P}_2(t_2).u = 0$$
$$f_2(t_1, t_2) \equiv \mathbf{P}_1(t_1).v - \mathbf{P}_2(t_2).v = 0,$$

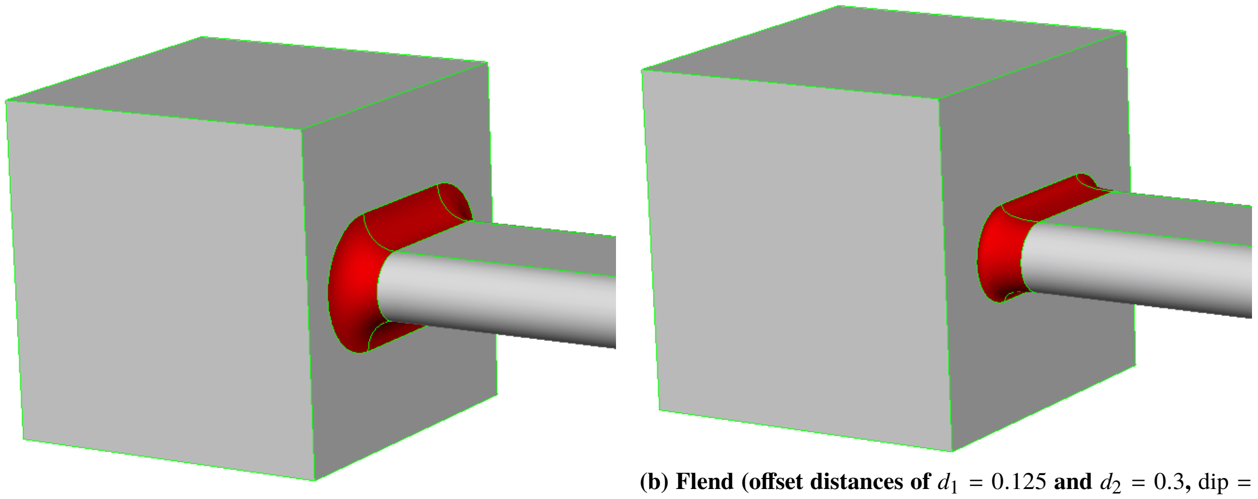
Essentially, these functions measure the distance between the u and v components of $\mathbf{P}_1$ and $\mathbf{P}_2$ which we require to be zero. The iterative scheme for Newton's method is

$$\begin{bmatrix} t_1 \\ t_2 \end{bmatrix}_{n+1} = \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}_{n} - \mathbf{J}(t_1, t_2)_n^{-1} \begin{bmatrix} f_1(t_1, t_2) \\ f_2(t_1, t_2) \end{bmatrix}_n ,$$

where the Jacobian is

$$\mathbf{J}(t_1, t_2) = \begin{bmatrix} \frac{\partial f_1}{\partial t_1} & \frac{\partial f_1}{\partial t_2} \\ \frac{\partial f_2}{\partial t_1} & \frac{\partial f_2}{\partial t_2} \end{bmatrix} = \begin{bmatrix} (\mathbf{P}_1(t_1))_u & -(\mathbf{P}_2(t_2))_u \\ (\mathbf{P}_1(t_1))_v & -(\mathbf{P}_2(t_2))_v \end{bmatrix}.$$

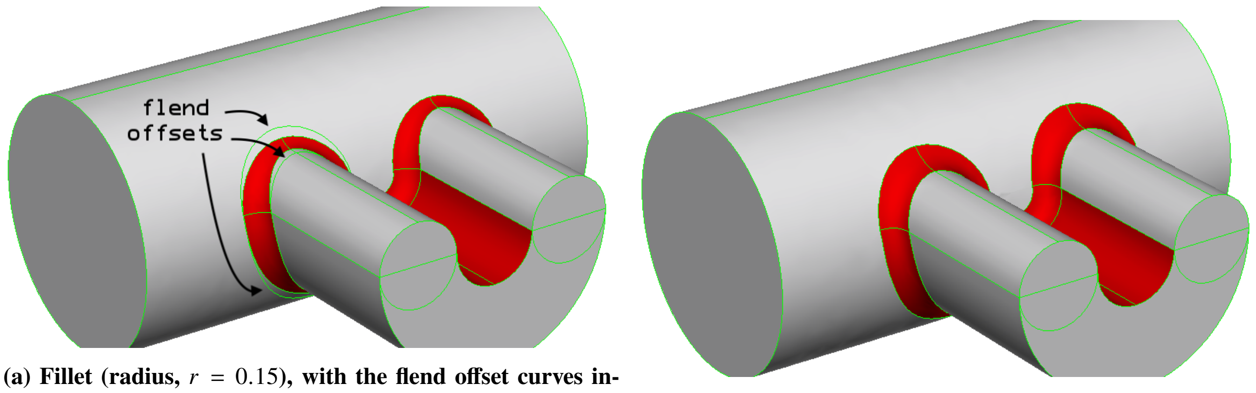
Once we have found t_1, t_2 to the desired accuracy, we trim the offset curves $\mathbf{P}_1$ and $\mathbf{P}_2$ to these parameterized values.



(a) Fillet (radius, $r = 0.15$).

(b) Flend (offset distances of $d_1 = 0.125$ and $d_2 = 0.3$, dip = 0.3).

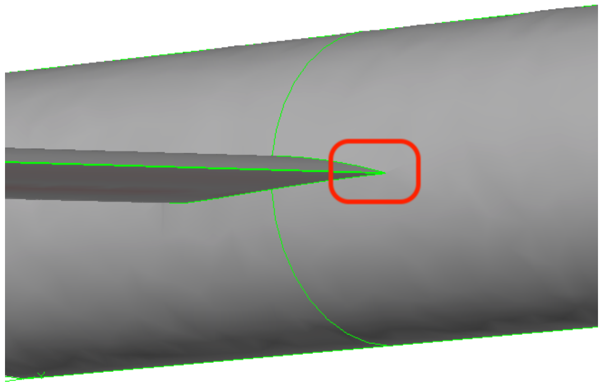
Fig. 7 Comparison of fillet and flend for box-"oval" intersection. Colors show maximum curvature.



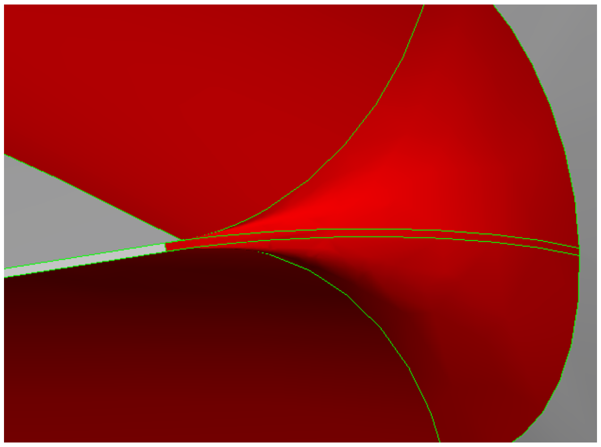
(a) Fillet (radius, $r = 0.15$), with the flend offset curves inscribed.

(b) Flend (offset distances of $d_1 = d_2 = 0.15$, dip = 0.3).

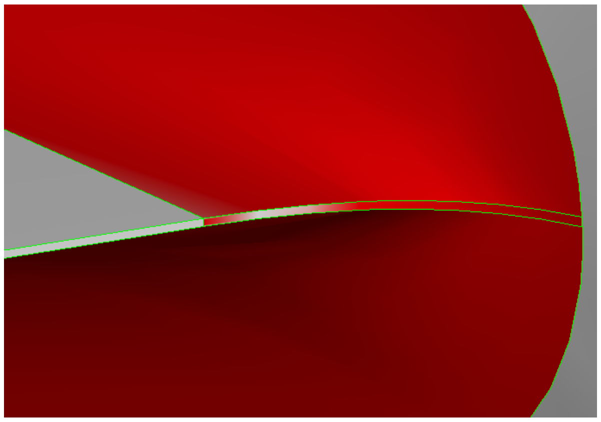
Fig. 8 Comparison of fillet and flend for cylinder-"horseshoe" intersection. Colors show maximum curvature.



(a) Cylinder-NACA4412 intersection, the region of interest shown in (b) and (c) is denoted in red.



(b) Failed fillet (radius, $r = 0.25$) at trailing edge.



(c) Flend (offset distances of $d_1 = d_2 = 0.25$, dip $= 0.3$) at trailing edge.

Fig. 9 Comparison of failed fillet and flend for cylinder-NACA4412 intersection. Colors in (b) and (c) show maximum curvature.

C. Demonstrations & Comparisons

This section contains several examples of flends, demonstrating their flexibility and robustness in comparison to fillets.

Fig. 7 demonstrates the ability to choose the offset distances (i.e. the location where the transition surface meets the original surfaces) independently. Fig. 8 further shows how the offset edges computed using our method remain a fixed distance from the intersection edge, whereas the width of the fillet fluctuates (note the inscribed curves in Fig. 8(a) that are the offsets for the flend in Fig. 8(b)). The example in Fig. 9 of the cylinder intersected with a NACA 4412 airfoil is a subtle but important demonstration of robustness. In Fig. 9(b), we observe that the fillet on the trailing edge of the airfoil does not join the fillet on the wing at the same location, whereas the offset distances for the flend (see Fig. 9(c)) are identical for all surfaces on the airfoil.

III. Flends about disjoint bodies

A user may wish to create a smooth transition surface between two faces on disjoint bodies (see Fig. 10), as opposed to transition surfaces between adjacent faces discussed in the previous sections. This is, however, essentially the same problem as discussed in the previous section. The only catch is that we do not have the intersection edges as a guide for generating the transition surface (i.e. the edge on which P_B lies). Therefore, we we need to develop an additional method for creating this faux intersection.

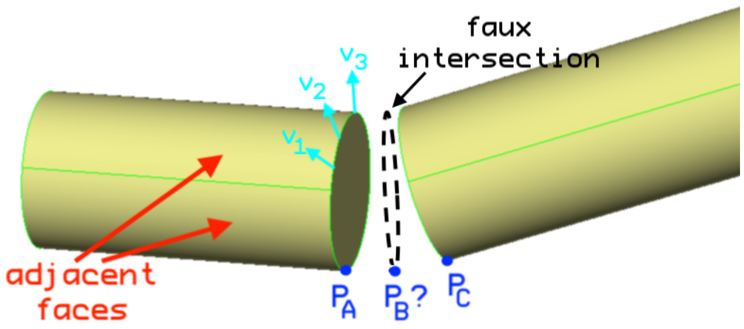


Fig. 10 Adjacent and normal face definitions.

A. Blending Surface Generation

First, the directions at the ends blending surface (where the blending surface meets the original surfaces) must to be determined. The current approach for determining the desired direction is to find the most perpendicular vector to the normal vectors on the adjacent faces. We begin by computing m normal vectors equally distributed along the edges on the adjacent faces (see Fig. 10), which can be expressed as $\mathbf{A} = \langle \vec{v}_1, \vec{v}_2, \ldots, \vec{v}_m \rangle$. The most perpendicular vector $\vec{x}$ is then any non-trivial solution to $\mathbf{A}^T\vec{x} = \vec{0}$, which can be computed via SVD [4].

Starting from a point on the adjacent face (i.e. P_A or P_C), the final direction vector ($\vec{d}_A$ or $\vec{d}_C$) is in the direction of the adjacent face's slope (uv-space) which most closely matches the most perpendicular vector $\vec{x}$ (xyz-space), such that the transition surface will be slope continuous where it meets the adjacent faces.

In order to construct the transition surface. we need some sense of an "intersection" between corresponding direction vectors found in the previous step. To do this we, we compute the points on each of the lines formed by these direction vectors that minimizes the distance between them and take the average. If we express these lines parametrically as $P_A + t\vec{d}_A$ and $P_C + s\vec{d}_C$, the points on each line which minimize the distance between them can be computed as [6]

$$C_A = P_A + \frac{(P_C - P_A)\cdot(\vec{d}_C \times (\vec{d}_A \times \vec{d}_C))}{\vec{d}_A \cdot (\vec{d}_C \times (\vec{d}_A \times \vec{d}_C))}\vec{d}_A \quad \text{and} \quad C_C = P_B + \frac{(P_A - P_C)\cdot(\vec{d}_A \times (\vec{d}_C \times \vec{d}_A))}{\vec{d}_C \cdot (\vec{d}_A \times (\vec{d}_C \times \vec{d}_A))}\vec{d}_C,$$

respectively. We use the average of these points as the faux intersection point, that is $P_B = (C_A + C_C)/2$.

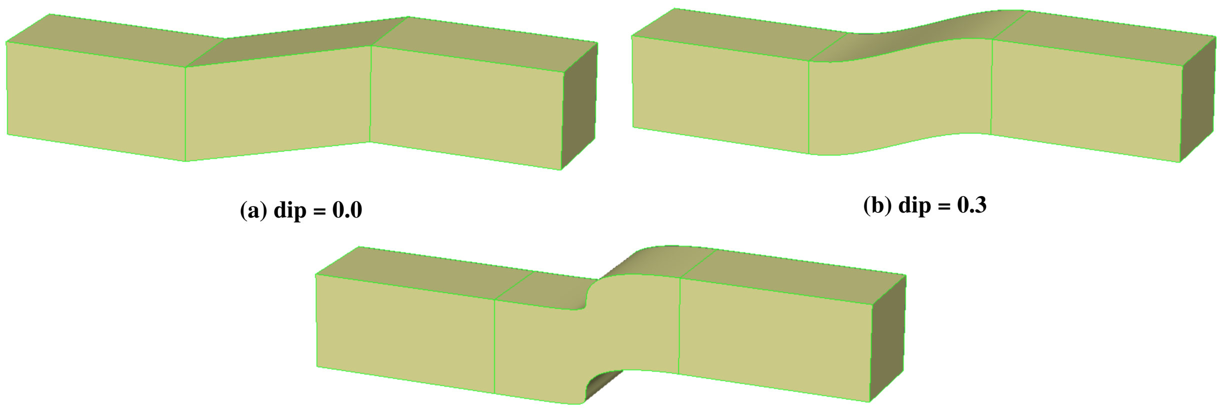
Once we have this faux intersection point, we apply the methodology for creating the control meshes discussed previously. Thus, we have a method for creating transition surfaces between two disjoint bodies which allows the user to parametrically control the resulting curvature. Note that we have assumed the number of edges on each of the bodies being flended to be equivalent. Furthermore, we assume that the corresponding edges on each of the faces are known (or can be determined using additional logic).

B. Demonstrations & Comparisons

In each of the cases shown, the original bodies are those on the left and right and the blending surfaces are those in the center (joining the original bodies). Fig. 11 demonstrates this method being used on two boxes of the same dimensions where one is translated slightly in space. Fig. 12 demonstrates this method on two cylinders of the same dimensions where one is translated and rotated.

IV. Conclusion & Future Work

We have developed methods that allow the creation of transition surfaces (dubbed "flends") that are more flexible and robust than traditional methods. Moreover, we first constructed a method for creating transition surfaces between two faces which takes as input the desired level of dip. This parameter allows the the user to control curvature of the resulting surface, namely the discontinuity in curvature where the transition surface meets the original surfaces. We further expanded on this method by describing how the locations at which the transition surface meet the original surfaces can be chosen directly and independently of one another. Lastly, we developed a supplemental method that allows the creation of transition surfaces between disjoint. Through several examples, implemented in ESP, we demonstrated the

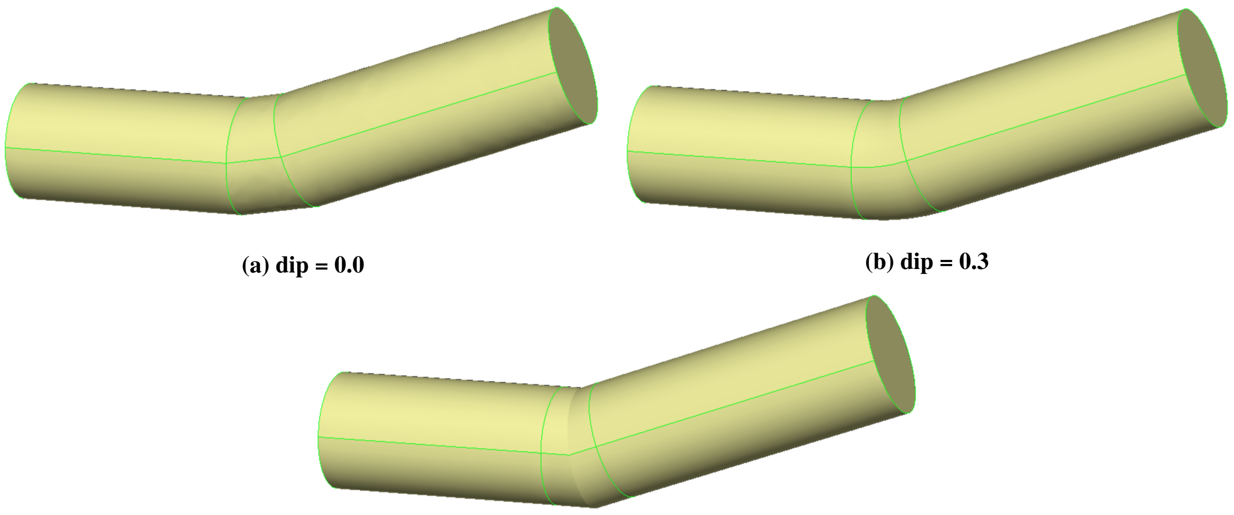


(a) dip = 0.0

(b) dip = 0.3

(c) dip = 1.0

Fig. 11 Flend between two boxes.



(a) dip = 0.0

(b) dip = 0.3

(c) dip = 1.0

Fig. 12 Flend between two cylinders.

flexibility and robustness of these methods in comparison to the ESP's methods. We reiterate that these methods are not specific to any solid modeler and can be implemented for any system with the supporting functions used herein.

There are several improvements to these methods that we wish to explore. First, C^2 continuity where the transition surfaces meet the original surfaces would be strongly desirable for certain applications. We would like to give the user the ability to enforce this. Next, the methods considered in this manuscript are only for creating transition surfaces between two surfaces. Also, a user may wish to create a flend about three faces (e.g. the corners of a box). We are currently working on a concise supplement to these methods that allows for this. Lastly, for the method discussed in Sec. II, we require the intersection of the bodies to be flended to be computed a priori. For complex geometries, this can be computational expensive and error-prone. Thus, we would like to be able to create the flend between the two un-intersected bodies.

References

[1] Haimes, R., and Dannenhoffer, J., "The engineering sketch pad: A solid-modeling, feature-based, web-enabled system for building parametric geometry," *21st AIAA Computational Fluid Dynamics Conference*, 2013.

[2] Goldman, R., “INTERSECTION OF THREE PLANES,” *Graphics Gems*, edited by A. S. GLASSNER, Morgan Kaufmann, San Diego, 1990, p. 305. doi:https://doi.org/10.1016/B978-0-08-050753-8.50065-6, URL `http://www.sciencedirect.com/science/article/pii/B9780080507538500656`.

[3] Piegl, L., and Tiller, W., *The NURBS Book*, Springer-Verlag, Berlin, Heidelberg, 1995.

[4] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Numerical Recipes in C (2nd Ed.): The Art of Scientific Computing*, Cambridge University Press, New York, NY, USA, 1992.

[5] O’Rourke, J., *Computational Geometry in C (2nd Ed.)*, Cambridge University Press, New York, NY, USA, 1998.

[6] Weisstein, E. W., *Skew Lines*, MathWorld - A Wolfram Web Resource, 2018. URL `http://mathworld.wolfram.com/SkewLines.html`.