DE-Vol. 65-1, Advances in Design Automation — Volume 1
ASME 1993

AN AREA PRESERVING TRANSFORMATION ALGORITHM FOR PRESS FORMING BLANK DEVELOPMENT

Nirmal K. Nair and James H. Oliver
Department of Mechanical Engineering
Iowa Center for Emerging Manufacturing Technology
Iowa State University
Ames, Iowa

ABSTRACT

An efficient algorithm is presented to determine the blank shape necessary to manufacture a surface by press forming. The technique is independent of material properties and instead uses surface geometry and an area conservation constraint to generate a geometrically feasible blank shape. The algorithm is formulated as an approximate geometric interpretation of the reversal of the forming process. The primary applications for this technique are in preliminary surface design, assessment of manufacturability, and location of binder wrap. Since the algorithm exhibits linear time complexity, it is amenable to implementation as an interactive design aid. The algorithm is applied to two example surfaces and the results are discussed.

INTRODUCTION

In many industries, the design, analysis, development and production of sheet metal surfaces comprise a substantial portion of component manufacture. The most prevalent manufacturing process for sheet metal components is press forming. In spite of its widespread use, press forming practice has remained somewhat of an art, typically handled by experienced tooling engineers and designers. This is due to the fact that the physical process of press forming is not well understood. Complex interacting mechanisms and features such as friction, metal flow, material properties and boundary conditions make the press forming process difficult to analyze and predict.

In press forming, the initial flat sheet of material used to develop the final shape is called the blank. The main components of the press forming assembly are the punch, die, and the draw binder mechanism which controls the flow of the blank material inward to form the product. Design of this forming assembly is directly dependent on the surface definition of the final product. For a surface design to be manufactured without defects, the blank should be uniformly deformed by the descending punch without thinning or wrinkling. This process is influenced, to a large extent by the draw binder ring, which is placed outside the trim line, i.e., the boundary between the formed surface and surrounding scrap material. To locate the trim line, the designer must determine the boundary of the area on the blank which is affected by the forming process. This process is referred to as blank development. Besides binder-wrap design, the developed blank is also used for punch contact analysis, press forming layout, and as an indicator of material flow during the process.

Iterative redesign of a product which fails in production is very expensive in terms of time and capital investment. It would therefore, be highly useful for product designers to quickly determine the formability of the surface in the early stages of the surface design process. Detailed analysis techniques such as the finite element method are computationally demanding and not amenable for an interactive design environment. Thus, there is a need for quick and qualitative tools which can guide a designer toward a successful design. This paper presents a technique to bridge the gap between final design analysis and initial surface construction.

RELATED RESEARCH

Sculptured surface models are employed in a wide variety of applications in the automotive, aerospace and appliance industries. Such surfaces can be broadly classified as developable or non-developable. A developable surface is characterized by the ability to form the shape by bending a plane without creasing or tearing, i.e., the surface can be generated by sweeping straight lines or generators along a curve in space. Mathematically, a surface is developable if its Gaussian curvature (the product of the principal normal curvatures) is zero everywhere (Mortenson, 1985). This property is often exploited in algorithms to map a developable surface onto a plane (Redont, 1989). Several methods for the transformation of developable surfaces have been formulated (Clements, 1981; Clements and Leon, 1987; Chu et al., 1985). For example, Clements and Leon (1987) developed an algorithm based on the relationship between the generating and geodesic lines on the surface to get an accurate blank transformation.

Non-developable surfaces encompass the family of surfaces that have non-zero Gaussian curvatures, and thus cannot be generated by simple bending of a plane without distortion. However, this does not preclude the use of these surfaces in manufacturing since they have advantages over developable surfaces in terms of styling, aerodynamics and other functional aspects of design. To investigate blank shape, Chu. et al., (1985) formulate a simplistic constant area transformation approach for the mapping of a non-developable surface onto a plane. The method is unique from other developments in the field in that it does not include material properties of the sheet metal but relies exclusively on the geometric properties of the surface. However, the method is limited by an artificial boundary condition requirement, an approximate area conservation method, and it is ineffective for surfaces with vertical flanges, i.e., areas of the surface that lie in planes perpendicular to the blank plane. Blank development of non-developable surfaces has largely remained a finite element analysis problem. Shimada and Tada (1989, 1991) have formulated methods for such transformations of surfaces using both the finite element method and dynamic programming. In the dynamic programming approach Shimada and Tada (1991) use a two step algorithm to start a multi-stage decision process using a good initial guess, and then refine the solution to get the final two-dimensional shape. The method requires computation of strain energies and solution of stiffness matrix equations. Both the finite element technique and dynamic programming approach are computationally intensive.

MATERIAL PROPERTY INDEPENDENT APPROACH

Any tangible product is ultimately defined by its geometry; i.e., the geometric aspects of an object qualify it for any purpose. For example, an object's viability with respect to a specific functional configuration is possible only if it is geometrically feasible. It follows that, if an object can be substantiated geometrically, then it may be manufacturable. Therefore, prior to detailed design validation incorporating material properties, the geometric feasibility of a product should first be established. Unlike most physical transformations, geometric transformations are reversible. Thus if the geometry of a part and its fabrication process are properly modeled, then the fabrication feasibility of the part can be assessed via reverse geometric transformation before detailed process analyses are attempted.

The research presented in this paper, is motivated by this philosophy. For blank development, a qualitative indicator is sufficient at the preliminary design stage, and can be arrived at using material independent transformations. In particular, an approach similar to Chu (1983) is adopted. The methodology is independent of material properties and relies on basic geometric manipulations to derive the blank shape and other manufacturability properties.

CONSTANT AREA TRANSFORMATION ALGORITHM

In sheet metal press forming tool design, the designer strives to achieve a uniform deformation of the blank to the final surface. The ideal deformation process is one in which the surface undergoes this transformation with no change in thickness. To determine such an ideal transformation, the tool designer would need to know where, exactly, each point on the blank lies after deformation. This requirement combined with the concept of geometric reversibility suggests the interpretation of blank development a geometric transformation problem of mapping the formed surface to a plane such that the area remains constant.

Two fundamental characteristics motivate the algorithm: 1) a procedural reversal of the forming process from the formed to the unformed state and 2) a geometric conservation of area between the two states. In effect, the algorithm transforms or "unstamps" the formed three-dimensional geometry into an initial planar shape. Variational geometry principles (Light and Gossard, 1981; Lin et al., 1981) are employed to derive a robust and efficient algorithm for the mapping. Area constraint equations limiting the degrees of freedom of points on the deformed surface are used to map each point to a feasible location in the plane containing the blank. Since the formulation is linear, the results are accurate and computational effort increases in linear proportion to the number of surface elements that are being transformed. A geometrically feasible solution is obtained to give the designer an assessment of initial blank shape, trim line and material flow during the forming process.

The basic assumptions underlying the constant area transformation algorithm are summarized as follows:

- *The surface is represented by a grid of points (vertices) which are considered as the elemental surface entities.*
- *Elemental area entities (triangles) are formed from any three mutually adjacent non-collinear vertices.*
- *The surface is subjected to a state of plane stress only.*
- *The surface is continuous, homogenous and isotropic.*
- *Only plastic deformation is considered.*
- *The surface has no thickness.*

SURFACE GEOMETRY REPRESENTATION

Contemporary computer-based design tools provide several methods for generating parametric sculptured surface models. The most common representation scheme is the non-uniform rational B-splines (NURBS) surface. A B-spline surface can be represented as

$$S(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} P_{i,j} N_{i,k}(u) N_{j,l}(v) \qquad u,v \in [0,1] \qquad (1)$$

where $S(u,v)$ is a three dimensional vector function of control points P_{ij} arranged in an $(m+1) \times (n+1)$ topologically rectangular grid and $N_{i,k}(u)$ and $N_{j,l}(v)$ are the degree k and l B-spline basis functions, respectively (Piegl and Tiller, 1987).

A well-defined homogenous triangulation of the surface is required as the input to the algorithm. Any surface $S(u,v)$ can be approximated by a faceted polyhedron, defined by a $(M \times N)$ set of three dimensional vertices $V_{ij}=S(u_i,v_j)$, $i=1,\ldots, M$ and $j=1,\ldots, N$. The resulting polyhedron approximates the actual surface. Such a polyhedral approximation can be constructed with a specific topological structure to algorithmically exploit vertex adjacency relationships. The nature of this topology depends on the method of surface discretization. In this application, the surface is triangulated in uniform parametric intervals to form a topologically rectangular mesh, which, after mapping through $S(u, v)$ generates a uniform network. The topology of the network is constructed such that any internal vertex has exactly eight surrounding vertices as shown in Figure 1. The polyhedral surface model is stored in a data structure which distinguishes the topological and geometric information. In particular, a vertex adjacency list is created to establish connectivity between each vertex and its neighbors.

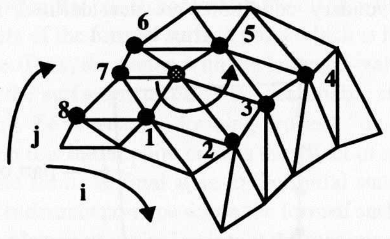

Figure 1 . DISCRETIZED SURFACE MODEL WITH VERTEX ADJACENCY RELATIONSHIPS AND NUMBERING SYSTEM.

One useful characteristic of an underlying parametric surface representation is the inherent separation of the topological and geometric information. Any surface vertex in Euclidean space has a dual in the parametric domain (i.e., the *uv*-space). Since the topology of the discrete surface approximation is defined in the parameter space, operations which require adjacency information are simplified. The corresponding geometric information is thus referenced primarily for the area calculations. This separation leads to simple and efficient algorithmic implementation.

DATA STRUCTURE

The information content for the algorithm is reduced to a vertex basis. The topological information for each vertex is stored in the form of a linked list containing pointers to the addresses of its neighbors. The data structure storage requirements are as follows:

Topological information		
Vertex to Vertex adjacency list	8	records
Geometric information		
Three-dimensional coords of each vertex	3	records
Two-dimensional coords (to be generated)	2	records
Visit Flag		
Vertex transformation status indicator	1	record
Total (for each vertex) =	14	records

ALGORITHM BASIS

Any triangle in three dimensions encloses an area on a plane. The constant area transformation is formulated such that both the topology and the area of a triangle are conserved when it is mapped from a three dimensional Euclidean space (E^3) to a two dimensional planar space (E^2). In particular, let

$$V^3 = \{V_1, V_2, V_3 | V_i \in E^3, V_1 \neq V_2 \neq V_3\} \tag{2}$$

be a set of 3-tuples which define unique triangles in E^3, and

$$P^3 = \{V'_1, V'_2, V'_3 | V'_i \in E^2, V'_1 \neq V'_2 \neq V'_3\} \tag{3}$$

be a set of 2-tuples which define triangles on a plane in $E^2 \subset E^3$. The constant area transformation is defined as a mapping:

$$CAT = V^3 \rightarrow P^2 \text{ such that for any } \alpha \in V^3,$$

$$\begin{aligned} CAT(\alpha) = \beta \in P^3 | \, & Area\beta = Area(\alpha), \\ & Top(\beta) = Top(\alpha) \end{aligned} \tag{4}$$

where *Area*() and *Top*() represent the area and topological state, respectively, of any triangle. A geometric interpretation of the constant area transformation can be summarized by the following principles:

• Given the location of two vertices V'_i and V'_j, a family of triangles $T(V'_i, V'_j, V'_k)$ of area A is defined by the area locus l of the point V_k which is a line parallel to $V'_iV'_j$ at a distance $h = 2A/|V'_iV'_j|$ from $V'_iV'_j$. (See Figure 2.)

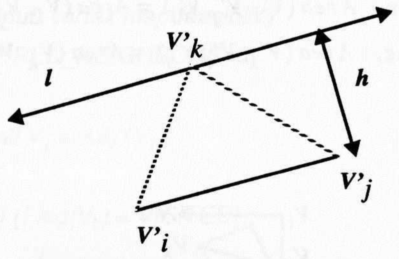

Figure 2 . LOCUS OF THE THIRD VERTEX OF A FAMILY OF CONSTANT AREA TRIANGLES.

• Given two adjacent triangles $T_a(V_i, V_j, V_l)$ and $T_b(V_k, V_i, V_l) \in E^3$ and corresponding projected locations of any three of these vertices in a plane in E^2, say V'_i, V'_j, and V'_k, as shown in Figure 3, if the unknown common vertex $V_l' \in E^2$ is located at the intersection of area loci $l_a \leftarrow T_a(V'_i, V'_j, V'_l)$ and $l_b \leftarrow T_b(V'_k, V'_i, V'_l)$, then

$$\begin{aligned} Area(V_i, V_j, V_l) &= Area(V'_i, V'_j, V'_l) \text{ and} \\ Area(V_k, V_i, V_l) &= Area(V'_k, V'_i, V'_l) \end{aligned} \tag{5}$$

These principles, although similar in spirit to those developed by Chu (1983), provide for several enhancements with respect to algorithmic implementation and computational accuracy. For example, as illustrated in Figure 4, Chu's constant area transformation pro-

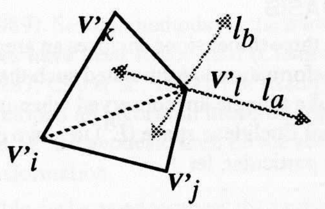

Figure 3 . AREA CONSERVATION PRINCIPLE.

vides an approximate solution for the location of the fourth point of a mapped quadrilateral. Given the location of three vertices V_i, V_j and V_k on adjacent triangles and their images V_i', $V_j'$and V_k' on a plane, Chu's method assumes that

$$Area\,(V_i, V_j, V_k) = Area\,(V'_i, V'_j, V'_k). \qquad (6)$$

Thus the area change due to the mapping of the quadrilateral (V_i, V_j, V_l, V_k) is assumed to be accounted for completely in the image of the triangle (V_k, V_j, V_l). The effect is approximated by constructing the three loci:

$$l_1 \leftarrow T_1\,(V'_i, V'_j, V'_l),$$
$$l_2 \leftarrow T_2\,(V'_k, V'_i, V'_l),$$
$$l_3 \leftarrow T_3\,(V'_k, V'_j, V'_l),$$

assuming, $Area\,(V'_i, V'_j, V'_l) = Area\,(V_k, V_j, V_l)$,

assuming, $Area\,(V'_k, V'_i, V'_l) = Area\,(V_k, V_i, V_j)$

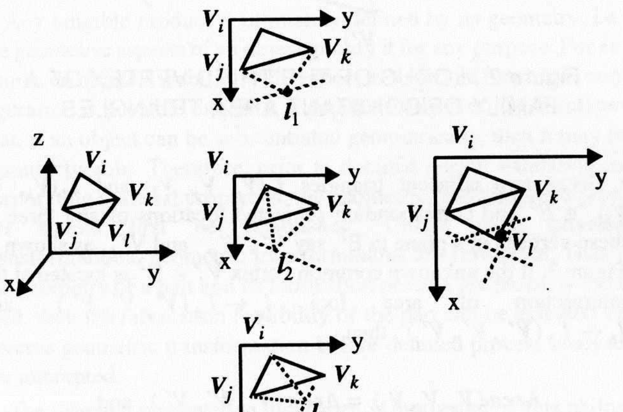

Figure 4 . METHODOLOGY FOR MAPPING THE FOURTH VERTEX OF A QUADRILATERAL (CHU et al., 1985).

as shown in Figure 4b. Vertex V'_l is taken as the centroid of the area enclosed by l_1, l_2 and l_3 as shown in Figure 4c. This solution is obviously approximate due to the assumptions used to construct l_l and l_2, and the fact that l_3 completely neglects any actual area change in the triangle (V_i, V_j, V_k). Since this formulation is this basis of the algorithm to map an entire surface, the error induced by this approximate solution is compounded because, in general, V_i, V_j and V_k are themselves calculated via the same procedure.

Another limiting aspect of Chu's formulation is the requirement of imposed boundary conditions necessary to initialize the algorithm. An orthogonal frame of reference along approximate planes of part symmetry must be established on the surface prior to the mapping. This frame is chosen such that the image of the vertices which lie on them can move only along axes formed from the intersection of the blank plane and the symmetry planes, as shown in Figure 5. Boundary conditions are user-defined, such that the

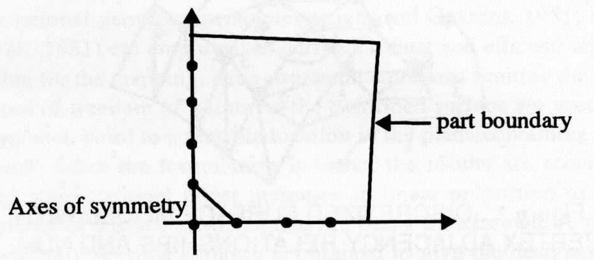

Figure 5 . BOUNDARY CONDITIONS FOR MAPPING (CHU ET AL., 1985).

boundary vertices of any two sides of the surface are fixed and identified along the reference frame. Both the symmetry reference frame and the boundary conditions impose restrictions on the material flow which, in general, do not reflect an accurate model of the forming process. The present work, however, derives its methodology from the forming process directly. The surface is assumed to be constrained equally along all edges to provide a restricted inflow of material. In the geometric context, this equates to a "boundary-less" or free-form deformation, since forces (equal along all edges) are not interpreted geometrically. This characteristic of the algorithm is described in detail on the following two sections.

The following terminology is used in the remainder of the chapter to facilitate the description of the constant area transformation algorithm:

N	Total number of vertices on the surface
V_{ij}	Address of a specific vertex, i and j correspond to surface parametric direction u and v.
$Adj(V_{ij})$	Vertex adjacency list for each V_{ij}
$Vert3D(V_{ij})$	Three-dimensional coordinates of vertex V_{ij}
$Flag(V_{ij})$	1 - if vertex V_{ij} is transformed, 0 - otherwise

***Vert2D*(V_{ij})**	Two dimensional coordinates of the vertex V_{ij}
Primary Neighbors	Vertices in ***Adj***(V_{ij}) which are topologically adjacent to V_{ij} in parameter space, i.e., $V_{i-1,j}$, $V_{i+1,j}$, $V_{i,j-1}$, $V_{i,j+1}$
Secondary Neighbors	Other vertices in the ***Adj***(V_{ij}) list
Visit_List	List of vertices to visit

TRANSFORMATION INITIALIZATION

Without lack of generality, the algorithm assumes that the blank plane is the global *XY* plane and the punch travels parallel to the Z axis. Since thinning is assumed negligible in an ideal forming process, the point of initial punch contact on the blank will most likely lie in the vicinity of the formed surface point which is furthest from the blank plane. Thus, the vertex with the largest Z-value, V_{max}, on the interior of the surface is taken as the reasonable starting point for the mapping. To reverse the forming process, "un-forming" of the surface from this initial point creates the effect of reversing the flow of material from the final state to the initial state. (In actual formed products distinct point peaks on the formed surface may not exist, instead a plateau of surface points at the maximum height can be found.) To mimic the uniform inflow of material to form a final surface, a uniform reverse outflow of area is formulated.

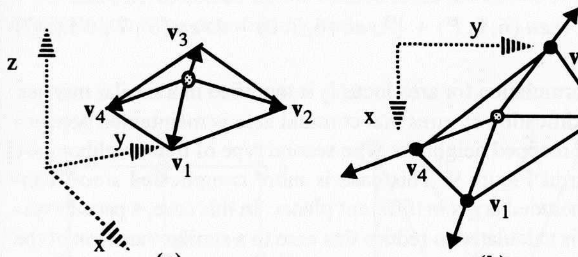

Figure 6 . TRANSFORMATION OF V_{MAX} AND THE SURROUNDING PRIMARY NEIGHBORS BY PRESERVING THE LENGTH OF LINE OF VECTORS v_1, v_2, v_3 AND v_4.

This flow is achieved by allowing the triangular elements to transform in a concentric manner starting with the vertices immediately surrounding V_{max} as shown in Figure 6. The transformation is initiated by projecting V_{max} parallel to the Z-axis, onto the blank plane. To seed the algorithm, the location of at least two additional vertices (adjacent to V_{max}) on the blank plane are required. However, the boundary-less deformation assumption implies a uniform flow of area toward all surface edges. Thus, the locations of the four primary neighbors of V_{max} on the blank plane are required. To map the primary neighbors, area preservation techniques cannot be applied, since V_{max} is the only vertex identified on the plane. Therefore, a length of line preservation procedure is adopted as shown in Figure 6. Unit vectors, formed between V_{max} and each of its primary neighbors are projected on the blank plane parallel to the Z-axis. The locations for the primary neighbors on the blank plane are approximated by scaling the two-dimensional vectors to their original three-dimensional length.

```
Algorithm. Transform initial primary neighbor vertices
   Input: $V_{max}$
   Output: primary neighbor Vert2D images of $V_{ij}$
   {
      calculate the primary neighbor vectors
      calculate the lengths of the vectors
      normalize the vectors and project on the
         blank plane
      scale 2D vectors to 3D lengths
      calculate primary neighbor vert2D point
         images
      return point images
   }
```

The remaining vertices of the surface are then transformed by visiting each vertex and inspecting its neighbors. Each vertex is

```
Algorithm. Vertex map feasibility
   Input: $V_{ij}$
   Output: vertex map feasibility
   MAPPED_NEIGHBORS=0
   {
   for all $V_k \in Adj(V_{ij})$
      {
      if (Flag($V_k$) = VISITED)
         {
         add $V_k$ to Mapped_list ($V_{ij}$)
         increment MAPPED_NEIGHBORS
         }
      }
   if ( MAPPED_NEIGHBORS $\geq 3$)
      {
      if at least three consecutive vertices $V_k$,
         $V_k \in$ Mapped_list ($V_{ij}$) are VISITED
      return vertex CAN BE MAPPED
      else return vertex CANNOT BE MAPPED
      }
   else return vertex CANNOT BE MAPPED
}
```

initially checked for mapping status. If the vertex has not been mapped, the algorithm searches the vertex adjacency list to determine whether at least three adjacent vertices have already been mapped. If the criterion is met, the routine returns that the vertex *CAN BE MAPPED*. Otherwise the function returns that the vertex *CANNOT BE MAPPED*.

A vertex which meets the criterion for mapping is mapped according the area conserving hypothesis. Depending upon the number of starting peaks, there may be more than three mapped neighbors surrounding the vertex which is queried. The solution methodology for the various cases are discussed in the following section.

MOVING FRONT AREA CALCULATION TECHNIQUES

In the general multi-peak implementation, cases will arise in which more than the minimum of three mapped neighboring vertices exist for mapping. In other words, the vertex to be mapped is part of one or more moving fronts and its position is affected by its mapped neighbors due to the area constraint. Extra vertices are eliminated to reduce the problem to a three-vertex case by viewing the vertex and its neighborhood in topological space. The solution procedure for the specific cases are discussed here.

Four adjacent neighbors

Consider the case as shown in Figure 7. For vertex *P*, neighbors *7, 0, 1* and *2* are already mapped. Overhanging neighbor Vertex 7 is eliminated from the area loci calculations since it is not part of any area enclosing triangle in the topological neighborhood of *P*. The elimination of the overhanging neighbor prevents this imbalance. An alternative approach is to consider the area contribution of triangle $T(7, 2, 0)$ in addition to $T(0, 2, 1)$. However, the experimentation with such a formulation revealed that it can induce an undesirable imbalance in the overall map due to the non-uniform attractive and repulsive area components. The problem is thus reduced to the three-vertex case which is solved using the principles explained in the preceding sections.

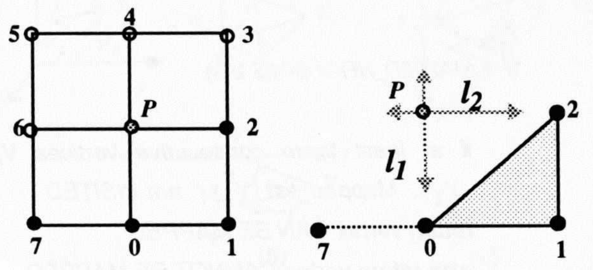

Figure 7 . FOUR NEIGHBOR CASE. VERTICES 7, 0, 1 AND 2 ARE MAPPED.

Five adjacent neighbors

Two types of neighborhood arrangements are possible as shown in Figure 8 and Figure 9. In both cases, all the surrounding mapped neighbors are involved in the area calculation since they constitute area enclosing triangles in the topological neighborhood of *P*. As

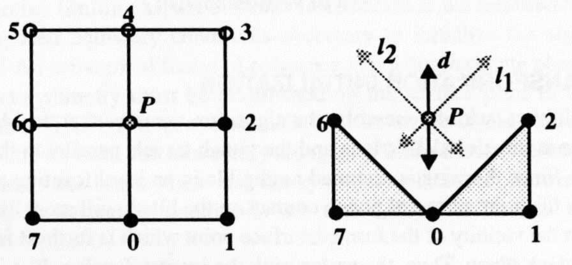

Figure 8 . FIVE-NEIGHBOR CASE. TYPE A, REDUCES TO A THREE-VERTEX CASE BY CONSIDERING VERTICES 6, 0 AND 2.

shown in Figure 8, the five-vertex case reduces to a three vertex case with adjacent triangles $T_1(6, 0, P)$ and $T_2(0, 2, P)$.The nominal area loci formulation is modified to account for the change in total area due to the projection of triangles $T(6,7,0)$ and $T(0,1,2)$. For example, the area used to determine area locus l_2 is taken as,

$$A_{l_2} = Area(6, 0, P) + [Area(6, 7, 0) - Area(6', 7', 0')] \quad (7)$$

The formulation for area locus l_2 is modified in a similar manner. This modification ensures that constant area is maintained between *P* and its mapped neighbors. The second type of five-neighbor case is shown in Figure 9. This case is more complicated since it involves included areas in different planes. In this case, a pseudo vertex (V_p) is calculated to reduce this case to a similar variation of the standard three-vertex problem (Nair, 1993).

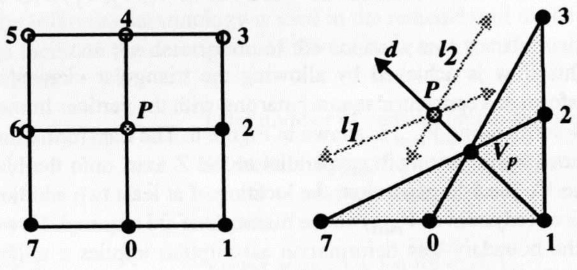

Figure 9 . FIVE NEIGHBOR CASE. TYPE B, REDUCES TO A THREE-VERTEX CASE BY CONSIDERING VERTICES 7, V_P AND 3.

Six adjacent neighbors

This situation is reduced to a five-vertex case by eliminating one overhanging neighbor for the same reason as the four adjacent neighbors case. Then the five-vertex technique is employed as shown in Figure 10.

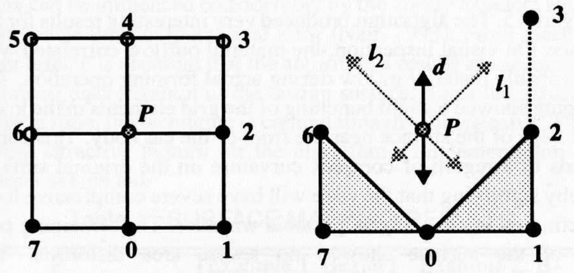

Figure 10 . SIX NEIGHBOR CASE. REDUCED TO A FIVE NEIGHBOR CASE.

Seven adjacent neighbors

This is similar to type A of the five-vertex case shown in the Figure 11. The case is easily reduced to a three vertex case by considering two adjacent triangles enclosed by vertices 5, 0, 3 and sharing *P* as shown in Figure 18. To preserve area, the net area components of the triangles indicated by the shaded areas are added to the area loci calculation for determining vertex *P*.

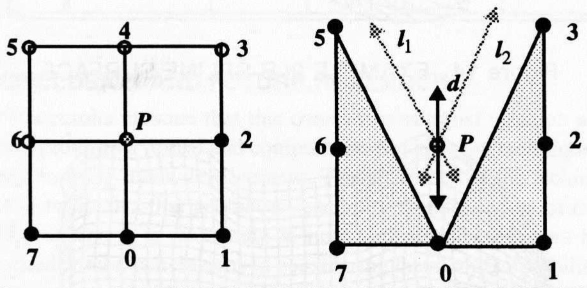

Figure 11 . SEVEN NEIGHBOR CASE. THE PROBLEM IS REDUCED TO A THREE VERTEX CASE BY CONSIDERING VERTICES 5, 0 AND 3.

VERTEX MAPPING

This is the kernel of the algorithm. Prior to this step, all the vertex manipulations are done in the topological space. In this segment of the algorithm, the geometric data is accessed and the area calculations are performed.

```
Algorithm. Map vertex
Input: $V_{ij}$ and three neighboring mapped vertices.
Output: Vert2D image of $V_{ij}$
{
        calculate the area enclosed by the vertices in
            three-dimensional space
        calculate 2D triangle base lengths
        calculate area locus of $V_{ij}$ from each of the two
            adjacent triangles
        calculate Vert2D($V_{ij}$) = intersection of the two
            loci
        return Vert2D($V_{ij}$) coordinates
}
```

MAPPING REMAINING VERTICES

After the first vertex and its primary neighbors are mapped, the remaining vertices are scanned by generating a visit list which is initialized with the addresses of the primary neighbors. Primary neighbors of each element of the visit list are queried for mapping via the *Vertex map feasibility* algorithm. Those which can be mapped, are mapped and appended to the visit list. The algorithm proceeds in this manner until the visit list is exhausted. The algorithm structure follows.

```
Algorithm. Map remaining vertices
Input: Initialized visit_list with primary neighbors of
        ($V_{max}$)
Output: Vert2D coordinates of all vertices of the surface
for all $V_k \in$ visit_list
{
    for all primary neighbors of $V_k$
    {
        query = Vertex Map Feasibility(Adj($V_k$))
        if query = CAN_BE_MAPPED
        {
            Map_the_Vertex(Adj($V_k$))
            add Adj($V_k$) to the visit_list
        }
    }
}
```

MAPPING EXAMPLES

Two example applications are presented which demonstrate constant area transformation of surfaces with single peak vertices. Computation times reported reflect implementation on a Silicon Graphics Indigo workstation with 48MB of RAM.

Example 1: Bezier surface. The fan shaped bicubic Bezier surface shown in Figure 12 was represented by a 40 by 40 parametric subdivision to yield 1600 vertices on the tessellated surface. The mapping was performed and the result is shown in Figure 13. The mapping of the surface was well defined and showed fairly uniform material flow over the entire surface. An increase of the parametric sampling produced the same result in slightly greater detail. The computation times for two surface discretization densities are shown in Table 1. As expected, the computation time grew in linear proportion to the number of vertices used to represent the surface.

Example 2: B-spline surface. This example is a more complicated B-spline surface of a toy model sports car body. This surface is characterized by a single peak, flanged edges and multiple peaks at local maxima as shown in Figure 14. This surface was chosen to study the behavior of the algorithm in the regions of flanges which the previous geometric method developed by Chu (1983), could not handle. A 40 by 40 grid of surface points was generated and the algorithm was applied to this surface definition. The result is shown in Figure 15. The algorithm produced very interesting results for the surface. On visual inspection, the material outflow correlated with the probable material inflow during actual forming operation. The mapping showed a slight bunching of the grid elements in the lower most edge of the surface near the front of the car body. This corresponds to a region of complex curvature on the original surface, thereby signifying that the edge will have severe compressive forces acting on it, resulting in possible wrinkles. The remaining portion of the surface showed no severe area distortion. The computational results for the surface are summarized in Table 1.

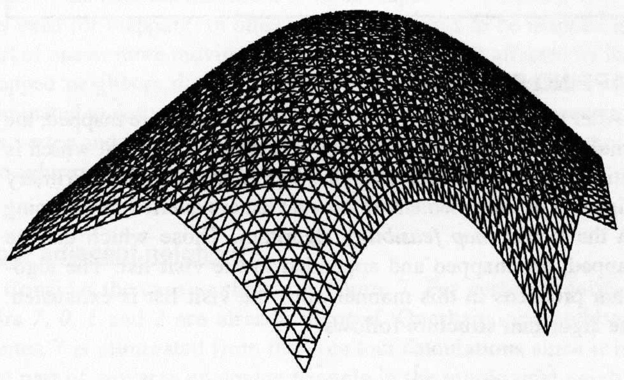
Figure 12 . EXAMPLE 1: BEZIER SURFACE

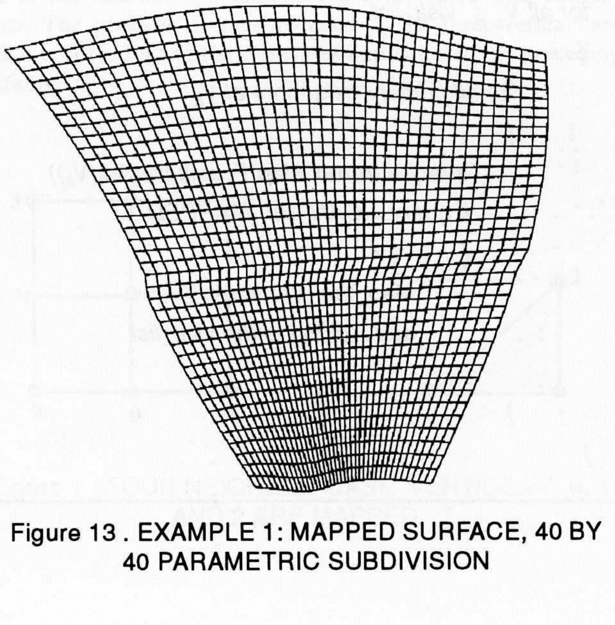
Figure 13 . EXAMPLE 1: MAPPED SURFACE, 40 BY 40 PARAMETRIC SUBDIVISION

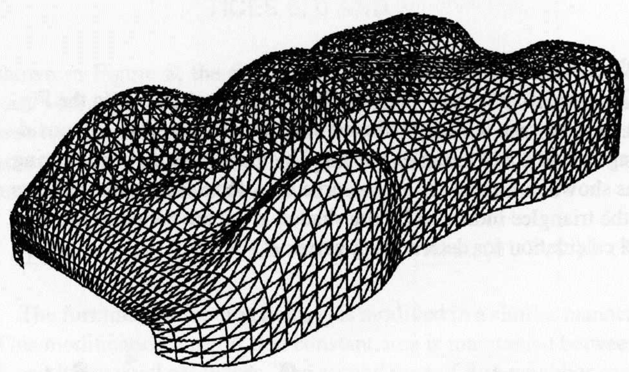
Figure 14 . EXAMPLE 2: B-SPLINE SURFACE

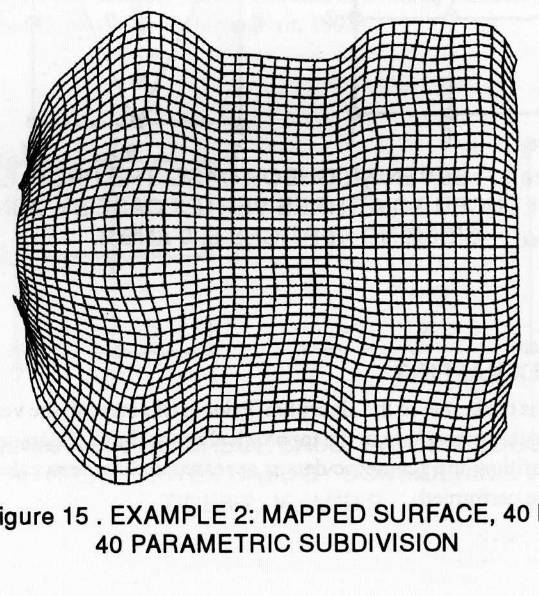
Figure 15 . EXAMPLE 2: MAPPED SURFACE, 40 BY 40 PARAMETRIC SUBDIVISION

ROBUSTNESS ISSUES

Numerical degeneracy arises during computation of an intersection point when the generating area loci are nearly parallel to each other. Since the basic methodology of the constant area transformation algorithm is propagation of areas from the center, any numerical error propagates, resulting in the failure of the algorithm. Although, this is not a deficiency of the algorithm, the resulting blank can be influenced considerably by the ability to detect and accommodate this computational error (Nair, 1993). From these and other tests, it is apparent that the algorithm's results are accurate for increased discretization of the design surface. Since the algorithm is linear time, the technique is computationally inexpensive. This is a very attractive feature for the algorithm's implementation as a general design aid.

Table 1 SURFACE MAPPING RESULTS

		Example 1: Bezier Surface		Example 2: B-spline surface	
	Discretization	40 x 40	60 x 60	40 x 40	60 x 60
Area Calculation	Computation Time (sec)	1.22	1.90	1.35	1.60
	3D Area	8.52	8.52	17.66	17.70
	2D Area	8.52	8.52	17.66	17.70
	Change (%)	0.0012	0.0005	0.0140	0.0072

CONCLUSION AND FUTURE RESEARCH

The results indicate that this constant area transformation algorithm provides a robust and computationally efficient technique for press forming blank development. The efficiency of the technique is due to the fact that geometric feasibility is independent of material property. It is interesting to note that other researchers have reached similar conclusions in considering the manufacturability of parts comprised of layered composite materials (Tam and Gutowski, 1990; Gutowski et al., 1991). Thus, the simplicity of the underlying algorithm and its corresponding linear time complexity make this constant area technique quite suitable for implementation as an interactive design aide.

Research related to the constant are transformation is continuing on several fronts. In fact, the technique developed in this paper was initially conceived as an ancillary function for sculptured surface model synthesis (Oliver et al., 1993). Although the constant area transformation technique has emerged as a useful tool by itself, a major focus has been to develop a general formability constraint to be incorporated into the sculptured surface model synthesis technique (Oliver and Theruvakattil, 1993). As a stand alone design aide, the constant area transformation algorithm will be enhanced in several ways. For example, the method will be extended to accommodate more general surface models to handle merging of multiple fronts from several peaks and stages simultaneously. In addition, other mapping strategies, such as volume conservation and/or minimum energy path, will be investigated. The method can also be extended to provide a qualitative measure of surface strains. It is hoped that these additional capabilities will provide a more accurate assessment of formability for complex surfaces.

ACKNOWLEDGEMENTS

This research was supported by the National Science Foundation (grant number DDM-9111122), the Office of Naval Research (grant number N0014-92-J-4092) and the Iowa Center for Emerging Manufacturing Technology. The authors are grateful for this support. The authors also wish to thank Professor Lin-Lin Chen and Dr. Shou-Yan Chou for their technical assistance.

REFERENCES

Chu, E., 1983, "New Horizons in Computer-Aided Design of Sheet Metal Stampings," *Ph.D. Dissertation*, McMaster University, Montreal.

Chu, E., Soper, D., Gloekl, H., and Gerdeen, J.C., 1985, "Computer-Aided Geometric Simulation of Sheet Metal Forming Processes," *Proceedings of the symposium on Computer Modeling of Sheet Metal Forming Processes*, sponsored by The Metallurgical Society, held at the 12th annual Automotive Material Symposium, Ann Arbor, Michigan, pp. 65-76.

Clements, J.C., 1981, "A Computer System to Derive Developable Hull Surfaces and Tables of Offsets," *Marine Technology*, Vol. 18, No. 3, pp. 227-233.

Clements, J.C., and Leon, L.J., 1987, "A Fast, Accurate Algorithm for the Isometric Mapping of a Developable Surface," *SIAM Journal for Mathematical Analysis*, Vol. 18, No. 4, pp. 966-971.

Light, R.A., and Gossard, D.C., 1982, "Modification of Geometric Models through Variational Geometry," *Computer Aided Design*, Vol. 14, No. 4, pp. 209-214.

Lin, V.C., Gossard, D.C., and Light, R.A., 1981, "Variational Geometry in Computer-Aided Design," *Computer Graphics*, Vol. 15, No. 3, pp. 171-177.

Mortenson, M.E., 1985, *Geometric Modeling*, Wiley, New York.

Nair, N.K., 1993, "Development of Manufacturability Constraints for Press Forming of Sheet Metal Components," *Master's Thesis*, Iowa State University.

Oliver, J.H., and Theruvakattil, P.C., 1993, "Sculptured Surface Model Based on Functional Design Constraints," *To be presented at the ASME Design Automation Conference*, Albuquerque, New Mexico, September, 1993.

Oliver, J.H., Theruvakattil, P.C., Nair, N.K., 1993, "Towards Automated Generation of Sculptured Surface Models," *Proceedings of the 1993 NSF Design and Manufacturing Systems Conference*, Vol. 1, pp. 663-640.

Piegl, L. and Tiller, W., 1987, "Curves and Surface Construction Using Rational B-splines," *Computer-Aided Design*, Vol. 19, No. 9, pp. 485-497.

Redont, P., 1989, "Representation and Deformation of Developable Surfaces," *Computer-Aided Design*, Vol. 21, No. 1, pp. 13-20.

Shimada, T., and Tada, Y., 1989, "Development of Curved Surfaces using Finite Element Method," *Proceedings of the first international conference on Computer-Aided Optimum Design of Structures*, Recent Advances, Springer-Verlag, New York, pp. 23-30.

Shimada, T., and Tada, Y., 1991, "Approximate Transformation of Arbitrary Curved Surface into a Plane using Dynamic Programming," *Computer-Aided Design*, Vol. 23, No. 2, pp. 153-159.

Tam, A.S., and Gutowski, T.G., 1990, "The Kinematics for Forming Ideal Aligned Fibre Composites into Complex Shapes," *Composites Manufacturing*, Vol. 1, No. 4, pp. 219-228.

Gutowski, T.G., Hoult, D., Dillon, G., and Gonzalez-Zugasti, J., 1991, "Differential Geometry and the Forming of Aligned Fibre Composites," *Composites Manufacturing*, Vol. 2, No. 3, pp. 147-152