Development of FE meshes for folded airbags

Chawla A*, Mukherjee S and Sharma A
Department of Mechanical Engineering, Indian Institute of Technology,
Hauz Khas, New Delhi, India

Abstract

A novel algorithm for generating Finite Element meshes of folded airbags for use in crash simulations is proposed and evaluated. The algorithm models the airbag folding process through geometric transformations on a given airbag mesh, followed by a series of numerical optimizations in order to achieve a folded state. An airbag model as a stack of connected planar layers is the input. It is shown that the proposed approach compares favorably with existing airbag folding techniques when evaluated quantitatively with the area and volume of the airbag post folding.

Keywords

airbag folding, cloth model, finite element, crash simulation

Introduction

The paramount consideration in vehicle design today is the safety of passengers and a variety of innovations have emerged to make vehicles increasingly safe. Airbags are one of the vital safety devices used in vehicles and have in fact emerged as a standard safety device in vehicles. Due to their importance, airbag construction, inflation techniques and triggering mechanisms have been widely investigated in literature, and standards on airbags are fairly well established. There is however a paucity of research on computer modeling of airbags. The modeling of the airbag inflation process, essentially a dynamic simulation is made hard by the small duration of airbag expansion. A typical airbag inflates during a crash in about 40 milliseconds after the triggering of the expansion process. The intermediate geometry during the deployment process is modified by the initial folded state of the airbag.

In the nominal design configuration, the airbag system is optimized such that a passenger in a crash should make contact with the middle of the airbag and only after it has inflated fully. After full inflation, the airbag starts deflating through vents provided and acts as an ideal cushion. If the contact is initiated before the ideal time and or is at an undesirable segment of the airbag, i.e. towards the edge of the airbag, adequate cushioning may not be available. Computer simulations are effective in investigating these issues, called the ‘Out of Position’ (OOP) issues.

Airbag modeling is made hard due to the requirements of dynamic simulation and as yet has not been satisfactorily addressed. Though the fully deployed volume of the airbag is of the order of 100 liters, it is systematically folded into a volume of about 0.5 liters in the car. Due to incorrect modeling of the folds, the surface area of the bag, the deployment sequence, the volume and visual shape of the inflated bag may change. The present work intends to fill this gap by proposing an algorithm that is able to model the airbag folding process with sufficient accuracy to generate folded airbag meshes for use in dynamic simulations. The proposed algorithm outputs a mesh of the airbag in the folded position, which can then be input to Finite Element (FE) simulation software for crash simulations.

Commercial software are available for folding airbags, such as PAMCRASH^{TM} suite of software and LDYNA^{TM}. These software treat the airbag as a continuous geometry. The folding of the airbag is then equivalent to transforming the geometry of the airbag while keeping the surface curvature continuous and is meshed after the completion of the folds. Accurate modeling of folding with this approach becomes quite difficult as the folds in airbag are closely packed and leading to complex geometry with sharp curvatures. In order to keep the airbag geometry manageable, software like

PAMCRASH^{TM} make simplifying assumptions, sacrificing the accuracy of the folding process. By viewing the airbag as a mesh, the proposed algorithm controls the geometry near the folds through optimization procedures.

A alternate method reported in literature is to model the real life folding process itself using dynamic simulations. The dynamic simulation approach to model cloth is used extensively in computer graphics and is now a separate branch of computer graphics known as cloth modeling which finds applications in the computer animation industry. Since the airbag is made of cloth, airbag folding becomes a special problem in the wider area of cloth modeling. Cloth surfaces have been modeled in computer graphics literature using spring-mass models by Zhang [1], self collision modeling incorporated by Ng [2], and recently wrinkle modeling by Bridson [5]. Ng [3] presents an extensive survey of computer graphics techniques used to model cloth. Models of cloth represented in terms of physical constitutive laws have of late become popular in computer graphics due to a tremendous increase in computing speeds available to solve the complicated numerical equations of such models. Such models represent the cloth as a very fine mesh in a dynamic simulation, allowing cloth behavior to be governed by self collisions and interaction of cloth with other materials. Bridson [6] presents one such physically based model of the cloth which has been successfully applied for commercial computer animation. Since the emphasis in computer graphics is on obtaining a realistic image of a model rather than accurate time-space-force trajectories and time efficiency required for Finite Element (FE) simulations, the above mentioned techniques are not found suitable for airbag modeling. Such models are either too refined to be of use in FE simulations (the computation time increases exponentially with refinement in a FE mesh), or, are unable to handle very tight squeezing of the cloth, which is required in airbag folding.

A typical driver side airbag is used as a case study in this work to illustrate the efficiency of the presented algorithm. The example airbag cloth when deflated can be laid out as a 600mm diameter membrane and is packed in a casing, which can be approximated by a cuboid of size 130x110x30mm. This results in a close packing of the airbag, with its surfaces in contact with one another. The model of the folded airbag is simulated using a FE simulation package, like PAMCRASH^{TM} in which the computational time is determined by the size and shape of the elements. A more refined the mesh corresponds to smaller element sizes leading to higher the computation time. This work addresses the need for a mesh which can model the tight folds of the airbag, but element sizes to large enough to make dynamic FE simulations computationally feasible.

The deficiencies in the folding process can be quantified by comparing reduction in area post inflation of folded airbag, or reduction in final volume attained by the airbag. Visible distortions in the inflated airbag shape also manifest themselves. In the proposed approach, a given initial 3D mesh of an airbag is folded multiple times and the mesh optimized locally near the folds. The optimization is to minimize the change in geometry and surface area of the airbag. After the final folds, a FE mesh of the airbag is obtained by. This FE model can then be imported into a FE airbag simulator package, physical properties assigned to the airbag material and the inflation simulated. Compared with the SAFE EDITOR^{TM} package for airbag folding, it is observed that the change in area and geometry of the airbag is substantially reduced by this technique.

Problem Definition

Our objective is to generate a well-behaved geometric mesh of a given airbag in a folded position, given the initial geometry of the airbag and the sequence of folds that are defined on it. A well-behaved mesh of an airbag implies a mesh that is amenable to computation using FE simulations. For example, a mesh with very small elements or elements with a very high (or very low) aspect ratio is not well behaved as the time for computation would go up considerably. The initial geometry of the airbag is constrained to be in the form of a set of parallel planar layers, which are connected using a set of elements arranged in a planar configuration (these planes being inclined to the parallel layers and are referred to as inclined layers in this text). This constraint is not a major limitation as a deflated airbag can be easily flattened for measurement.

Airbag folding for FE simulation has some specific issues which need to be highlighted. Naïve folding algorithms tend to generate a fine mesh near the fold to enable the fold line exactly. In case of the airbag, it is not the folded membrane but the final *unfolded* membrane that is of interest. Secondly the unfolded volume needs to expand into the same shape and volume as the actual device. This is a special problem with quadrilateral shell elements as preserving *surface area* under out of plane warping, and preserving element *edge lengths* under warping are important issues. The third issue of significance is that *penetrations between inner layers have* to be avoided. Any internal penetration, though not visible as an artifact on the surface, has catastrophic consequences in FE simulations of airbag unfolding. The final issue is that the element sizes have to be above a minimum threshold to ensure stable simulations with explicit solvers being completed in acceptable time.

Existing tools

PAMCRASH^{TM} software has a module for folding airbags, called SAFE EDITOR^{TM}, which inputs the initial geometry of the airbag and generates the geometry resulting after a sequence of folds. One can also define gas properties and other parameters required for airbag definition in SAFE EDITOR^{TM}, but only the airbag meshing options is of relevance in this description. This module is a geometry-based folder, which takes the initial airbag geometry as input and allows users to define folds on the airbag, giving the mesh of the final geometry as the output. SAFE EDITOR^{TM} takes an IGES geometry file as an input for the initial geometry. In the case study considered later, the IGES file given as input to the airbag folder contains the geometry of a circle. As the SAFE EDITOR^{TM} takes only a planar geometry as the initial input, it is constrained to having the lower layer of the airbag identical to the upper one, both of these layers being connected by a set of elements. After giving the initial geometry, one can proceed to define folds on the airbag. The software supports two kinds of folds: the roll fold and the tuck fold. The parameters to be input for defining a roll fold in SAFE EDITOR^{TM} are location of the folding line, direction of the fold, a fixed point, thickness of the fold and the transient distance for the fold. Thus, one can define a folding line only on one surface of the airbag, which defines the folding location for the rest of the airbag also. The direction of the folding line can be reversed to change the direction of the fold from clockwise to anti-clockwise. The transient distance of an airbag fold (see Figure 1) corresponds to the width of airbag fabric on both sides of a folding line that will participate in the fold. The fixed point is a point that identifies the airbag patch that remains fixed during a folding sequence. And lastly, thickness of the fold defines the desired distance between innermost layers after the fold. The generation of mesh is a separate process from the folding in SAFE EDITOR^{TM}. Thus the folding sequence can be changed quickly, without having to mesh the geometry at each stage, as only the final state can be meshed to obtain a “pc” file, which can then be input for FE simulation in PAMCRASH^{TM}.

There are some restrictions for defining folds in SAFE EDITOR^{TM}. The folding lines have to be parallel or orthogonal to each other for defining a fold. Both the folding lines and transient lines have to be defined by points lying outside the airbag and are always applied to the full stack of layers. Also, a transient patch cannot be split by another parallel folding or transient line, even if only the transient line divides a given layer (since folds are applied to the full stack).

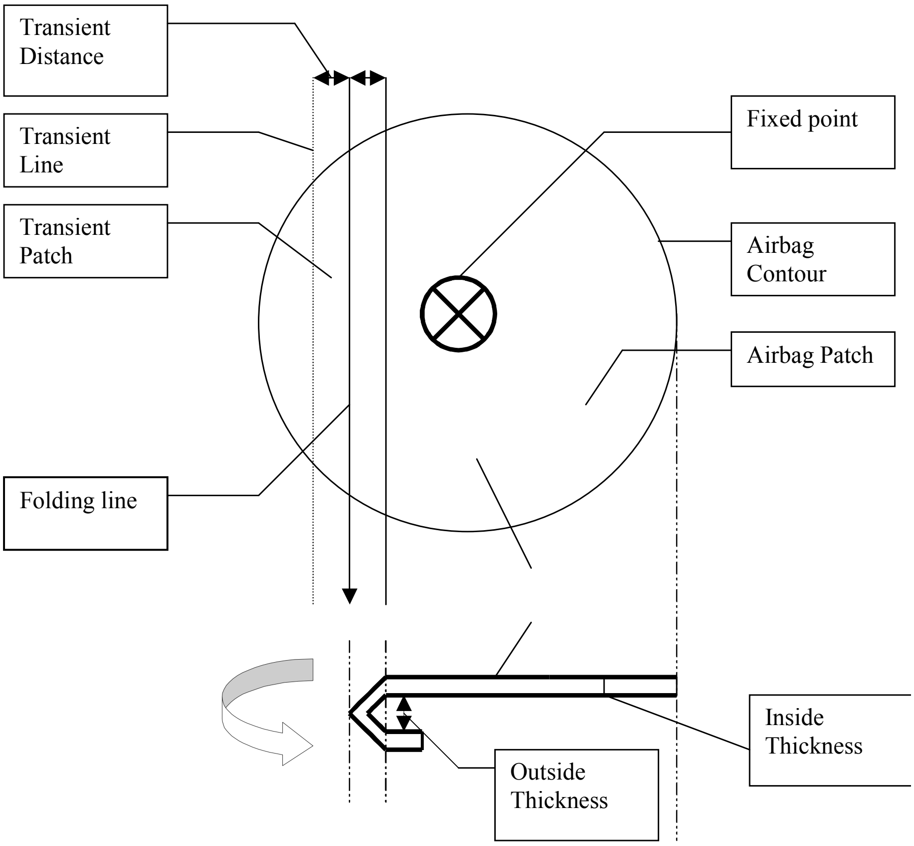

Figure 1 :Illustrations of the parameters for defining a roll fold as required by SAFE EDITOR^{TM} [6]

Proposed tool

As stated earlier, the proposed approach views the folds as a series of geometric transformations applied on the airbag mesh, coupled with an optimization approach to resolve critical points in the mesh. This model of folding takes its inspiration from a real-life model of folding, wherein the cloth is folded physically by locating a folding line and then rotating a part of the cloth such that it lies on the top of the other. Physically when the cloth is folded, wrinkles are generated on the cloth surface. These are computationally expensive to model. But, if wrinkles are not modeled as a part of the folding process, then there is a loss in the area of the airbag surface, resulting in distortion of the inflated geometry of the airbag. In the proposed folding procedure, the area conserving effect of wrinkles has been incorporated, without sacrificing on the computation time.

The initial airbag surface has to be defined as a set of largely planar layers, connected by inclined surfaces. Thus, prior to initiating the folded surface, it has to be collapsed and flattened into a set of connected planar layers. There are several methods to achieve this flattening, a few of them being: physical measurements of the actual airbag, and simulated flattening of the airbag using flat surfaces or using an approximation to the actual shape. In the present case study, an approximate mesh of the shape of a disc of 600mm diameter is being used, which is very close to the actual shape of a commercial driver side airbag. The flattening of the airbag results in several parallel layers (in our case study, the two planar connected layers), which have to be transformed simultaneously to define folds on the deflated airbag, while taking care to avoid mesh self-penetrations.

Each fold is defined by four input parameters:

1. The choice of a folding plane.
2. The desired width of the fold (which is the desired distance between the innermost layers after the fold).

3. The folding part (whether the mesh lying on the left or the right side of the folding plane is to be folded).
4. The folding direction (whether the mesh is to be folded clockwise or anti-clockwise).

Thus, at each step, an airbag mesh is given as input, on which the user (using the above parameters) defines a fold, and the algorithm generates an output mesh of the airbag in the folded position. In the process of folding, the mesh often needs to be readjusted algorithmically to preserve airbag surface areas and edge lengths.

After each fold, the output is a mesh of the airbag in the specified folded position. Usually, after a fold, the mesh size decreases, specially near the fold zone. As a refined mesh becomes computationally expensive for FE simulations, there is a need to coarsen the mesh at the end of the folding process, while keeping the geometry intact, to make the model amenable to simulation using PAMCRASH^{TM}. An algorithm for coarsening the mesh is also incorporated in the software.

Each fold of the airbag generates one or more planar layers, which are added to the already existing set of parallel layers, and connected to the rest of the mesh by inclined layers. The mesh can be folded again and again, as long as the layers are wide enough to support folding, though the time taken for the computation would increase with the number of planar layers. As the folding algorithm is modeled on the real world process of folding, the outer layers in a fold occupy a larger circumference compared to the inner layers in a fold. Thus, each layer that is folded, forks into two layers, both of which are smaller than the original layer, and two inclined layers. Thus, the area of each individual layer keeps on decreasing with each fold, though the sum of all the areas remains almost constant. And after a large number of folds, the layers may become too small to be able to accommodate a transient patch, signaling that no more fold operations are possible on it. This process is intuitively similar to the real world folding process, where a cloth can be folded only a certain number of times, till the layers become too small to be folded again. For the cases studied, the number of folds that can be carried out were quite large and easily satisfy our requirement for airbag folding. Maintaining the sequence of folds is also critical, as different sequences produce distinct folds, thereby affecting the manner in which the airbag deploys on inflation.

The algorithm proposed, unlike SAFE EDITOR^{TM} which handles folds in two orthogonal directions, is capable of handling folds in arbitrary directions, provided they do not intersect some special sections of the mesh termed ‘transient patches’. Transient patches are created when the folding plane intersects the inclined layers. The creation of these patches is necessary to avoid change in the geometry of the airbag. This is not a serious functional limitation as transient patches created in airbag folding are of the order of the thickness of the airbag material and hence very small compared to the airbag area. There is hence added flexibility in trying out various folding sequences over conventional packages.

Case Study

In this section the working of the folder is illustrated by the means of a case study a typical passenger side airbag. This example illustrates the working and demonstrates the efficacy of the algorithm. The generated mesh is compared with the results from SAFE EDITOR^{TM}. The geometric details have been measured from a commercially available airbag. The initial state of the airbag is a closed disc of 640 mm diameter, with two planes of the disc separated by a distance of 0.4 mm, and connected by elements throughout the circumference. This initial state of the fabric of the airbag before the start of the folding process and is shown in Figure 1. A point to be noted is that the geometry of the two layers of the airbag being identical is not a necessity for the proposed method, but is essential for SAFE EDITOR^{TM}. In fact, SAFE EDITOR^{TM} works with a planar geometry as input to the folder and assumes that the second identical layer creates the volume closure.

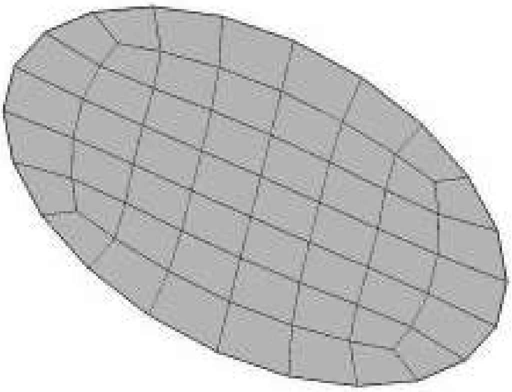

Figure 2
The initial airbag geometry in the form of a disc

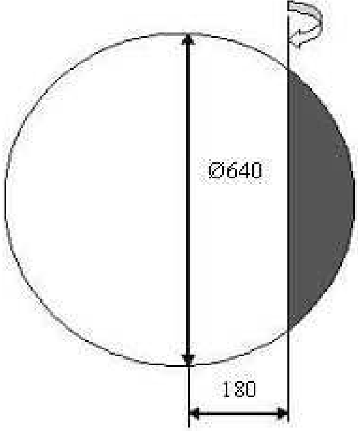

Figure 3 (a) The schematic of the first fold

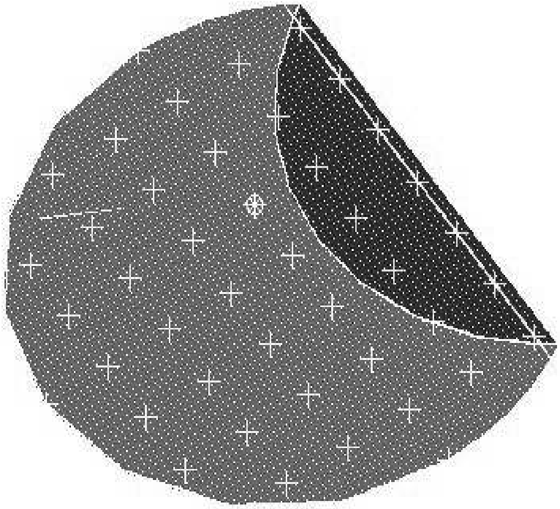

Figure 3 (b) The folded geometry in SAFE EDITOR^{TM}

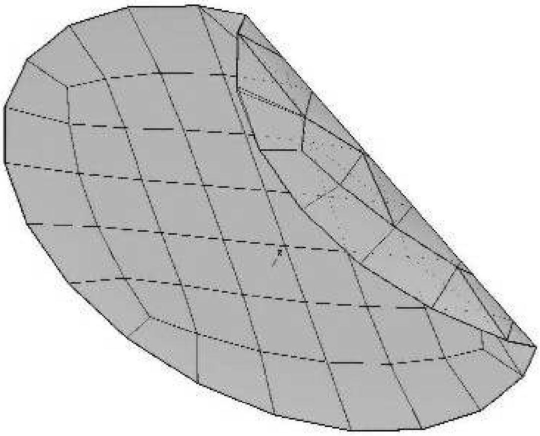

Figure 3 (c) The folded mesh generated by the developed algorithm

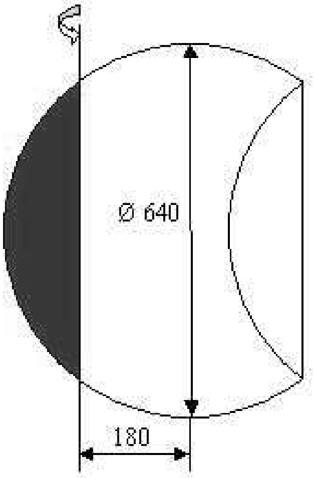

Figure 4 (a) The schematic of the second fold

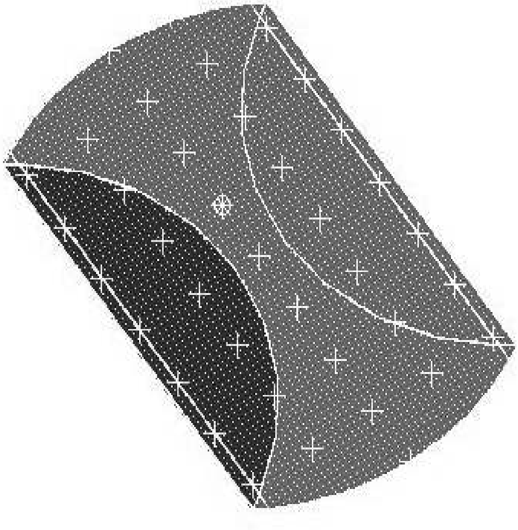

Figure 4 (b) The folded geometry in SAFE EDITOR^{TM}

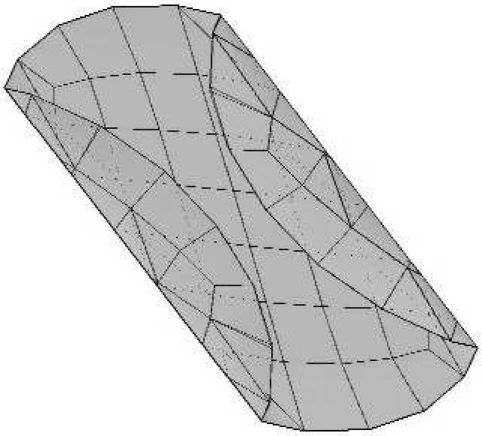

Figure 4 (c) The folded mesh generated by the developed algorithm

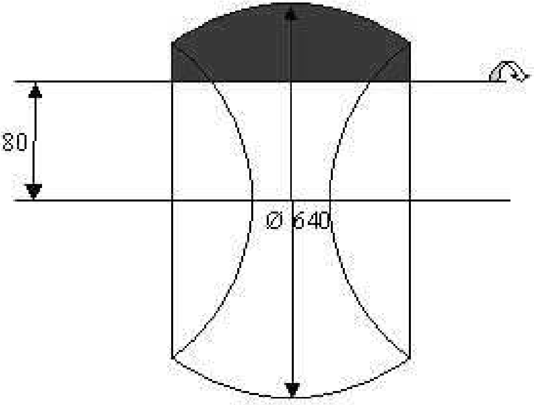

Figure 5 (a) The schematic of the third fold

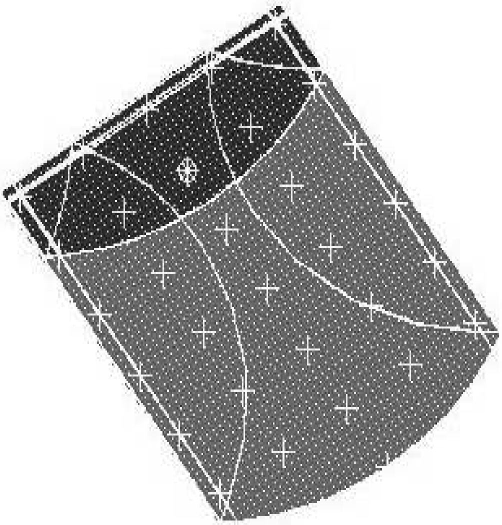

Figure 5 (b) The folded geometry in SAFE EDITOR^{TM}

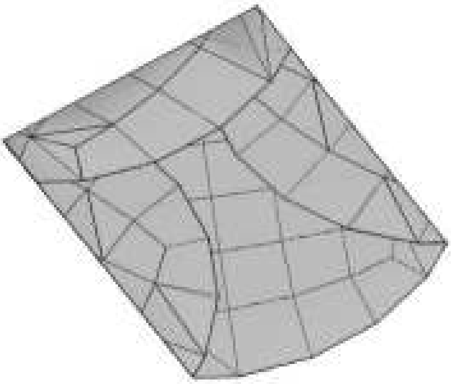

Figure 5 (c) The folded mesh generated by the developed algorithm.

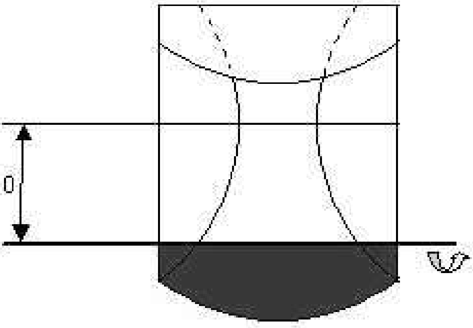

Figure 6 (a) The schematic of the fourth fold

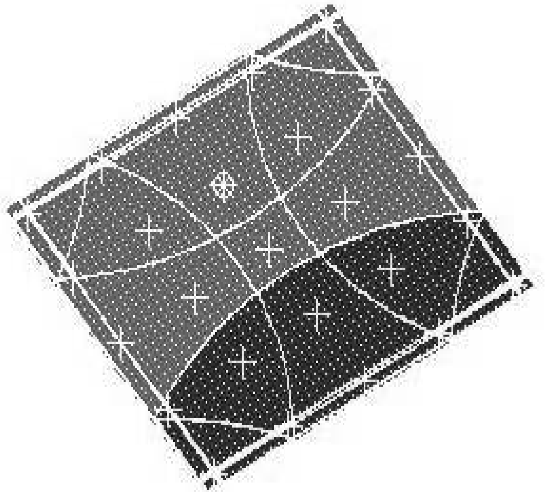

Figure 6 (b) The folded geometry in SAFE EDITOR^{TM}

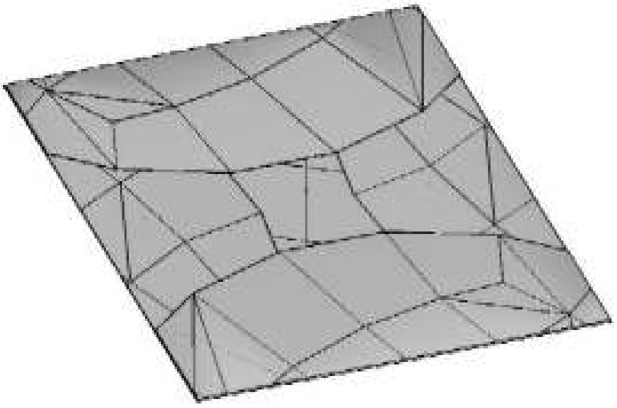

Figure 6 (c) The folded mesh generated by the developed algorithm.

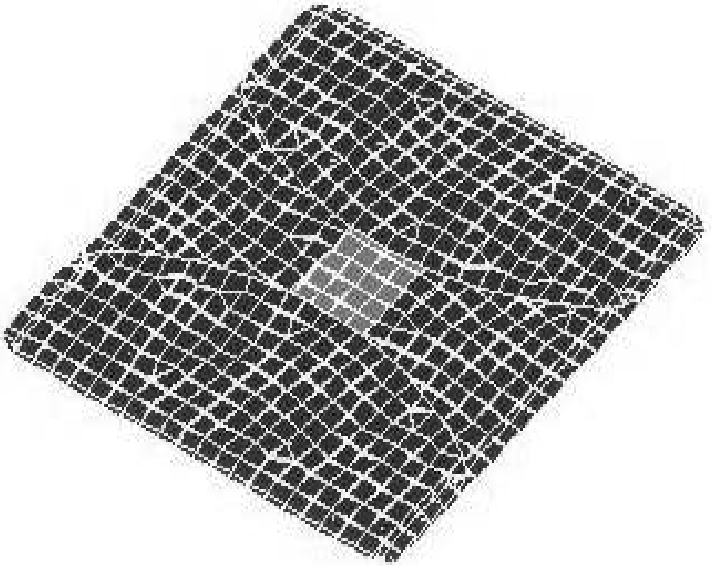

Figure 7
The generated mesh on the airbag folded with SAFE EDITOR^{TM}

Figure 2 shows the initial unfolded circular mesh for the airbag fabric. Figure 3 shows the schematic of the first fold and the folded mesh after the first fold as obtained from SAFEEDITOR^{TM} and from the developed algorithm. Similarly Figures 4, 5 and 6 show the schematic of the second, third and fourth folds and the folded mesh thereafter. The final folded mesh as obtained from SAFEEDITOR^{TM} safe editor after a series of folds is shown in Figure 7.

Results

One of the parameters used to quantify the performance of folding algorithm can be change in surface area of the bag during folding. Inflation of the initially flattened airbag can be simulated and compared to the inflation of the folded airbag. Though the deployment trajectory might differ, ideally the two should have the same end volume and final visual appearance. Change in volume of the inflated bag before and after folding can be used as a measure and the visual shape of the bag can be used as for qualitative comparison. As the third measure, the minimum stable solution time steps for the models can be compared.

The proposed algorithm was very efficient in controlling the change in area of the airbag while folding. The percentage change in surface area with respect to the flattened airbag after four folds is 0.11% (0.637995 m^2 from the initial 0.63874 m^2). Whereas, the surface area in the airbag folded using SAFE EDITOR™ changes by almost 11% during these four folds to 0.570521 m^2. Volume of the unfolded airbag comes to be 48.8 litres after inflation, which reduced to 48.1 litres after four folds using the proposed software. The same airbag outputs a volume of 42.7 litres after being folded using SAFE EDITOR™.

The bag folded using SAFE EDITOR™ undergoes a visible change in shape, while the bag folded by the software retains its similarity to the original shape. Comparing the time steps, there are more than 7000 elements in the airbag folded with the SAFE EDITOR™ with time step of the order of 1×10^{-6}, while all of the elements generated by the proposed algorithm have a time step of the order of 1×10^{-5} or lower. In the airbag modeled using the proposed software, there are about 550 elements, with only about 300 having a time step lower than 1×10^{-5}, leading to much faster simulation than the SAFE EDITOR™ airbag. This difference is due to the fact that this algorithm refines the airbag mesh only where it is required and thus the mesh remains overall well behaved. As is visible in Figure 8 (a), the mesh output from the software remains coarse in zones where refinement is not needed, while the airbag modeled with SAFE EDITOR™ is refined even in places that do not require refinement for adequate representation.

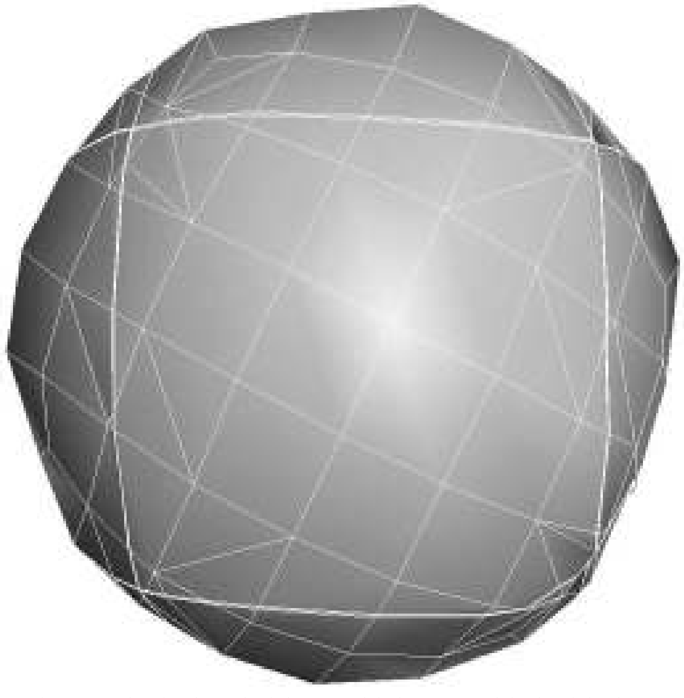

Figure 8 (a) Top view of the inflated airbag, which was folded with the proposed software

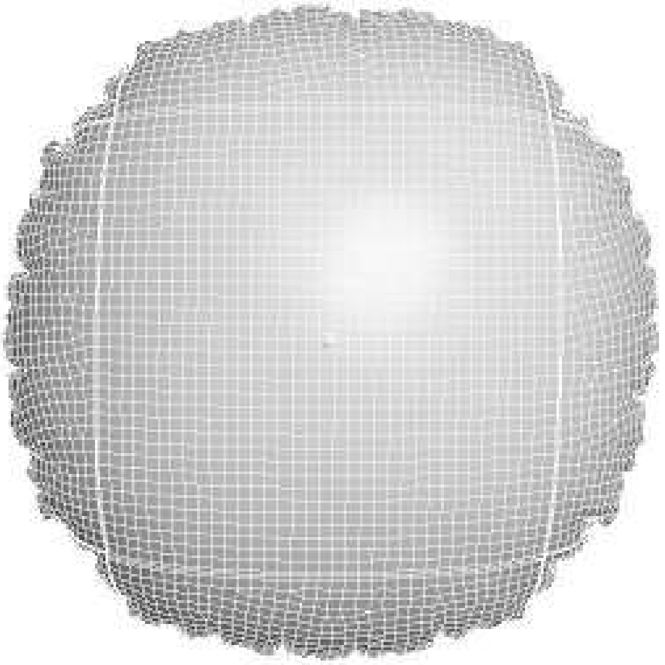

Figure 8 (b) Top view of the inflated airbag, which was folded with SAFE EDITOR^{TM}

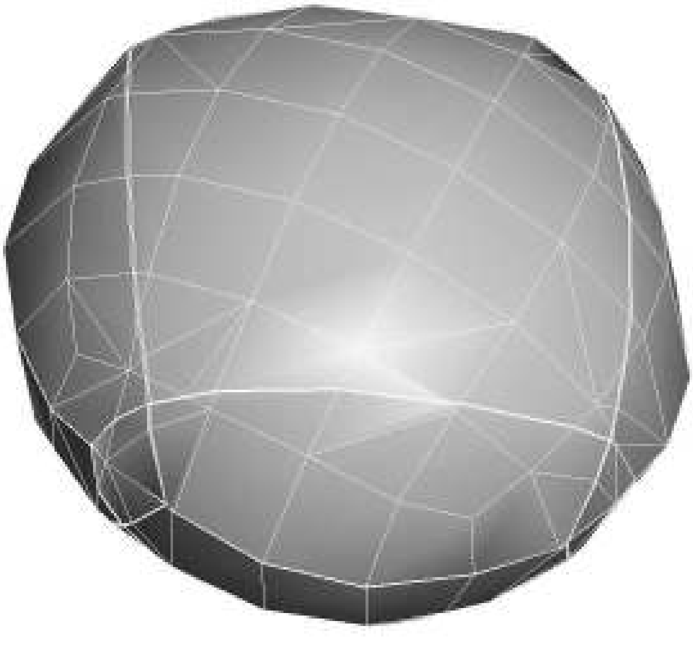

Figure 9 (a) Isometric view of the inflated airbag, which was folded with the presented software

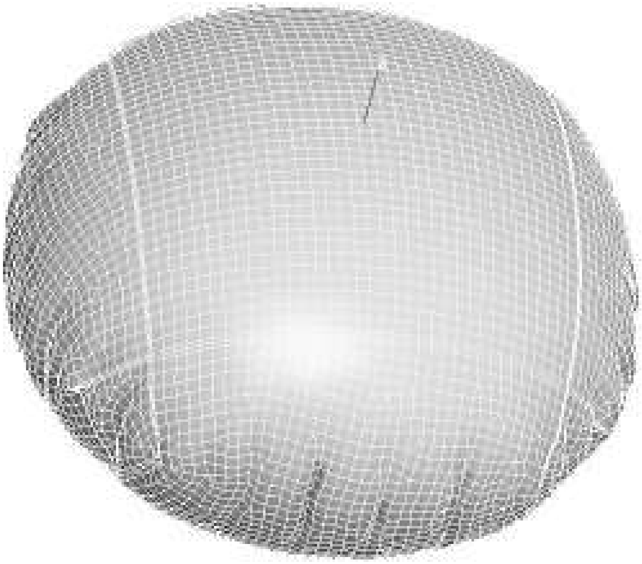

Figure 9 (b) Isometric view of the inflated airbag, which was folded with SAFE EDITOR^{TM}

Figure 10
The original shape of the airbag

Limitations

There are limitations of this approach if it is viewed as a generic cloth folder, as it has been optimized for one specific application, that of airbag modeling for crash simulations. For example, the wrinkles in the cloth have not been modeled explicitly, resulting in a smooth appearance of the airbag surface, which does not appear realistic for a cloth surface. Also, there are certain positions of the folding plane, which are not admissible by the algorithm. This is due to the fact that in certain positions the geometry becomes too complex to be modeled within the limitation of a coarse mesh. A sacrifice of certain positions of the folding plane for a model amenable to FE simulations was adopted. These positions are quite limited and usually a small translation of the folding plane would make it amenable to the software. This does not affect the functionality of the algorithm as far as airbag modeling is concerned.

Conclusions

An alternate approach for modeling airbags for FE simulations has been described. This approach optimizes the mesh during folding to preserve surface area, edge length to minimize distortion of the inflated airbag surface. The minimum stable time step for simulation were kept in bound by controlling the mesh sizes through local remeshing operation. A case study on a typical airbag was conducted by folding using alternate folders and the simulated inflations were compared. The approach compares favorably with existing software in preserving the geometry of the airbag during the folding process and simulation efficiency during inflation.

References:

1. Bridson, R., Fedkiw, R. and Anderson, J., "Robust Treatment of Collisions, Contact and Friction for Cloth Animation", ACM Transactions On Graphics, 2002, 21, 594–603.
2. Bridson, R., Marino, S., and Fedkiw, R., "Simulation of Clothing with Folds and Wrinkles", Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation, San Diego, California, Eurographics Association, 2003.
3. Dongliang Zhang, and Matthew M. F. Yuen, “Cloth simulation using multilevel meshes”, Computers and Graphics, 2001, 25, 3, 383–389.
4. Liu, J., Ko, M., and Chang, R., “A simple self collision avoidance for cloth Animation”, Computers and Graphics, 1998, 22, 1, 117–128.
5. Ng, N., and Grimsdale, R.L., “Computer graphics techniques for modeling cloth”, Computers and Graphics, 1995, 19, 3, 423–430.
6. Ng, N., Grimsdale, R.L., Allen, W., “A system for modeling and visualization of cloth material”, Computers and Graphics, 2001, 25, 3, 383–389.