3D MEMS Design via Matlab Interactive Plots

Nanping R. Lo and K.S.J. Pister*

EE Department, UCLA, Los Angeles CA, USA, nanping@ee.ucla.edu
*Department of EECS, UC Berkeley, Berkeley, CA, USA, pister@eecs.berkeley.edu

ABSTRACT

Matlab is utilized for 3D MEMS design. A preprocessor, written in C, processes the standard, 2D CIF design by applying the topology and spatial information to the design. The Mat3d toolkit then takes the output of the preprocessor and utilizes Matlab's 3D graphics functions to generate a 3D plot of the 2D CIF design. Using the Mat3d toolkit's library of commands as well as the existing Matlab commands, the design within the 3D plot then can be viewed from desired magnifications as well as from different perspectives. Objects within the plot could also be available for manipulations such as rotation and translation. During manipulation of the objects in the plot, a Matlab-based collision detection algorithm can be activated for detection of objects coming in contact. A set of raytracing commands is also available with this toolkit, which will ad in simple MOEM design. The preprocessor is currently capable of processing manhattanized CIF with box geometries. The package is available at http://www.ee.ucla.edu/pub/mat3d.

Keywords: 3D MEMS, CAD, Matlab, Interactive, Plots.

1 INTRODUCTION

Several commercially available MEMS CAD packages perform excellent rendering of microstructures. However, since 3D display is not the primary goal of these packages, other than providing views from different perspectives, the renderings from these MEMS CAD packages are static and do not permit interaction with parts within the design. However, with 3D MEMS, it often requires the designer to imagine and visualize how pieces of initially coplanar structures would eventually have to come off substrate to fit together in 3D space to form a 3D system. Exploring the design parameters of how the various coplanar pieces might fit with hand sketches is a cumbersome process. An interactive tool that would provide an environment to virtually manipulate pieces within a design would expedite the process. Previous works by the authors attempted to address the interactivity issue with a custom software package and Java/VRML solution [1]. It was not widely used because it was viewed as for visualization only and the installation procedure was complicated.

In this paper, the authors extend the solution for interactive 3D MEMS design by using Matlab as the underlying analysis and displaying engine. Using commands in the toolkit's command library, as well as existing Matlab commands [2], pieces within the design are no longer static but can now be manipulated. The designer can fold structures away from substrate directly from their 2D, flat CIF design, and see how multiple pieces in a design might interact with various movement of each piece. During manipulation of objects in the plot, a Matlab-based collision detection algorithm can be activated for detection of objects coming in contact. The visualization is now simpler with the capability to view the 3D plot from desired perspective and magnification by adjusting the graphics display commands. The toolkit is customizable. Raytracing, for basic MEOM design is have already been incorporated in this toolkit.

Figure 1 outlines the overall process flow for using Matlab's 3D plot for interactive MEMS design. The

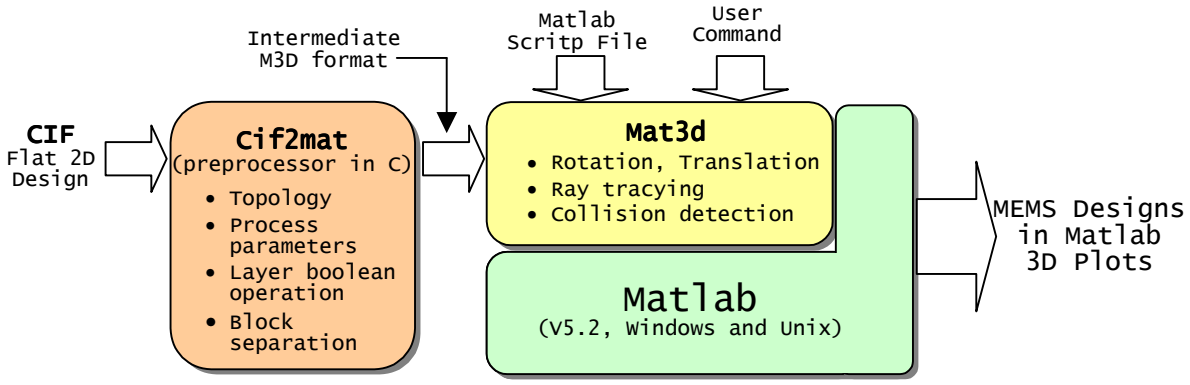

Figure 1. Process flow of plotting 3D MEMS designs with Matlab.

package is composed of two major components, a preprocessor, and a Matlab toolkit. This paper will describe the various necessities for using the toolkit, pieces of the preprocessor, and the components of the toolkit. Sample plots will be included in the last section.

2 DESIGN PRELIMINIARIES

In order to automatically obtain the linkage and reflector information from a CIF file, the following rules should be observed.

1. Substrate hinges. Use simple substrate hinge as shown in Figure 2(a).

2. Scissor hinge. Use a typical scissor hinge, with the addition of a 1-um wide strip of BPMTL layer running along the hinge axis of the 2 joining structures as shown in Figure 2(b). The BPMTL layer strip should be drawn on top of both POLY1 and POLY2 layers.

3. Reflector pseudo layer. Place the BPMTL layer over MTL layer. The reflective pseudo layer is defined as the intersection area of BPMTL and MTL.

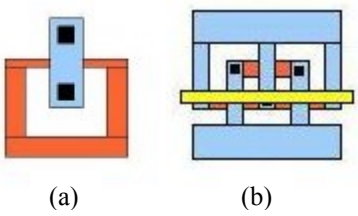

Figure 2. (a) Sample substrate hinge. (b) Sample scissor hinge with 1μm stripe of pseudo layer.

3 CIF2MAT - PREPROCESSOR

Figure 3 further details functionalities and sequence of operations for the Cif2mat preprocessor. Layers within the CIF files are subjected to boolean operations [3][4] with other layers to determine the intersections and unions between layers. In this step, temporary reflector pseudo layer and temporary pivot pseudo layers are determined. The reflector pseduo layer is passed to the Matlab toolkit to use for raytracing. The pivot pseudo layers, for both substrate hinges as well as scissor hinges, are used in the linkage determining step. Figure 4 illustrates the placement of the pseudo pivot layers. In a later preprocessor step, the top pivot pseudo layers are matched up with the bottom pivot pseudo layers to from linkage between spatially disjointed objects.

Once layer boolean operations are completed, all layers, real and pseudo, are processed by adding the process dependent topology information. The side view in Figure 4 also illustrates the inclusion of height and thickness information for a substrate hinge. Once the layers in the design have been transformed from the flat, 2D CIF layers into the corresponding 3D, box-like representation, all spatially jointed boxes are sorted into corresponding structures. The spatial sorting is accomplished by performing exhaustive intersection tests of all boxes in the design. Intersection tests against the bounding box of each structure are performed to increase efficiency of the spatial sort.

If the interconnects between structures, i.e. hinges, are drawn in the specified fashion, the preprocessor can detect the interconnection between structures by matching the top pivot pseudo layers and the bottom pseudo layers. Thus, although all structures are spatially disjointed, linkage information, connected via hinges, between structures, as well as rotational data can be gathered. The topologized design, together with information such as, spatially disjointed structures, linkage between structures, rotational information, etc., is written to an intermediate text data file to be read by the Mat3d toolkit.

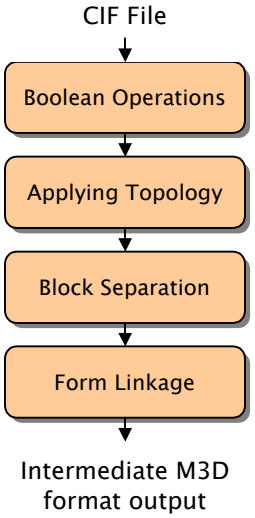

Figure 3. Detailed flow of the Cif2mat preprocessor.

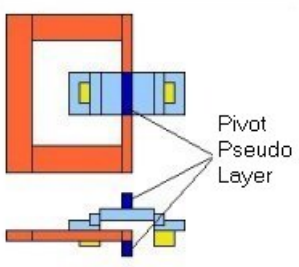

Figure 4. Top and side views of a substrate hinge showing temporary pseudo pivot layer applied. In the side view, top and bottom pseudo layers are shown.

4 MAT3D

The boxes in the intermediate data file are read into Matlab by specifying the vertices of Matlab graph object called "patch" as shown in Figure 5. It is Matlab's underlying plotting functions that render all the patches as 3D plot. Manipulation and movement of structures, such as rotation and translation, are done by applying the spatial delta to the vertices of all the patches involved. After each movement, Matlab's plotting engine updates the plot accordingly. By entering commands in the Matlab console window, or submitting an m-script file, a series of movements can be done such that the design appears to undergo animation. A set of commands exist to do the following tasks:

- Identify, manipulate and view structures.
- Get, set linkage and rotational information (if not already extracted by the preprocessor).
- Specify and commence raytracing parameters.
- Commence collision detection.

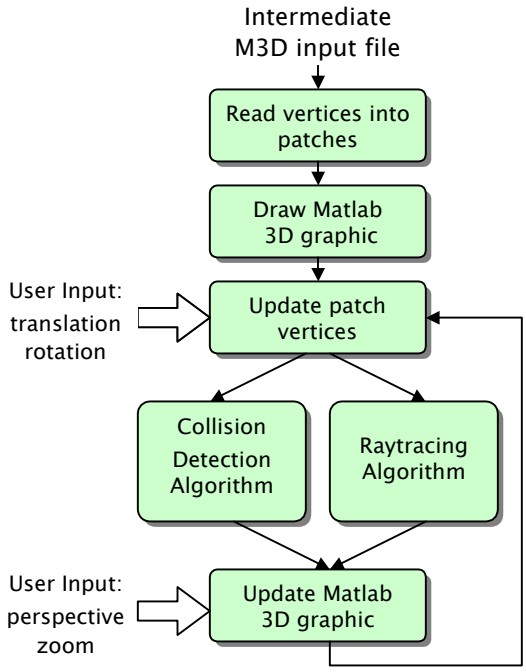

Figure 5. Detailed flow of Mat3d toolkit.

4.1 Collision Detection

There are several collision detection packages available for general use [5][6]. However, in this application, it is more efficient to implement a native Matlab collision detection algorithm since using the other packages requires constant conversions of the patch vertices to and from Matlab's memory space.

In this Matlab implementation, collision between 2 box-shaped objects is determined by 6x6 facet intersection tests. Since each structure is usually composed of multiple boxes, bounding box tests and proximity tests are performed to increase the efficiency of collision detection tests between 2 structures.

4.2 Raytracing

With the reflector pseudo layer included in the layout, it is feasible to perform raytracing with Mat3d. The existence of the incident point of the ray on the reflecting surface can be determined by computing the dot product of the ray vector and the normal vector of the plane [7]. If an incident point exists on reflecting plane, the direction vector of the reflecting ray can be determined by simple vector algebra.

5 LIMITATIONS

There are several limitations in using this package. The current version of the Cif2mat preprocessor only handles Box geometry in the CIF file. Manhattanized and flattened layout is required for the preprocessor. Also, Torsional hinge is not recognized by the preprocessor.

6 SAMPLE PLOTS

Figure 6 shows a hollow triangular beam linkage in 2D design and the linkage and rotational information automatically extracted for assembly. Figures 7 and Figure 8 show a series of plots and a SEM of a XYZ micro-optical stage [8], from its 2D design, to the assembled device with raytracing to illustrate its functionality. Within Figure 10 are plots of a micro mirror [9] and raytracing to demonstrate its operation. A SEM of the micro mirror is included as Figure 11 for comparison.

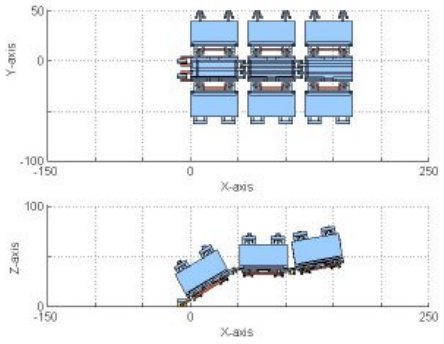

Figure 6. Plots of 2D design (top) in xy-view, and the assembled linkages (bottom) in xz-view.

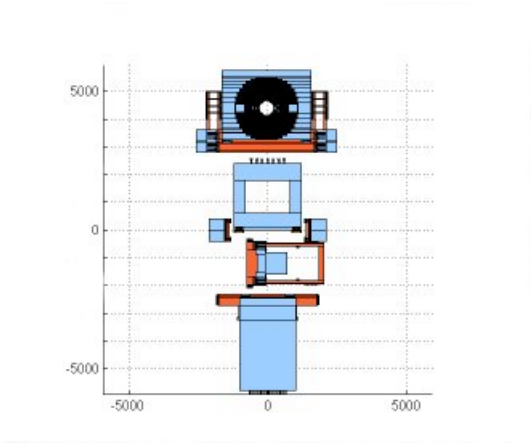

Figure 7. Plot of the flat, 2D design of the XYZ-stage.

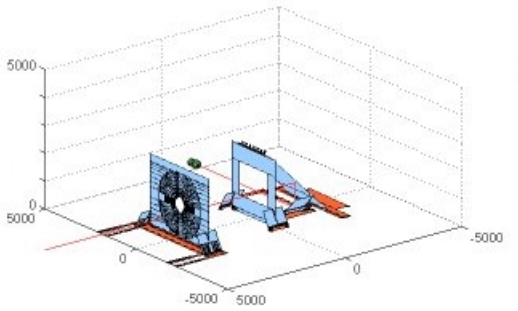

Figure 8. Plot of the assembled XYZ-stage with raytracing demonstrating its functionality.

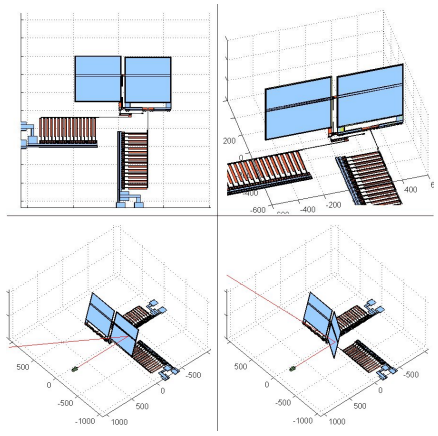

Figure 9. 2D design and 3D plot of an assembled micro mirror (top). Raytracing to illustrate operation (bottom)

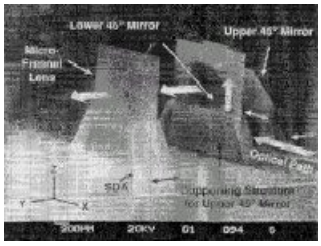

Figure 10. SEM of the XYZ-stage.

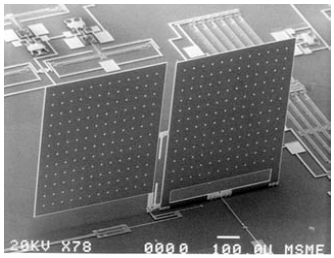

Figure 11. SEM of an assembled micro mirror.

7 DOWNLOAD

The authors invite the readers to visit the Mat3d website, http://www.ee.ucla.edu/pub/mat3d, where more information and plots of Mat3d, its on-line documentation, as well as the entire package itself can be obtained.

REFERENCES

[1] N. Lo and K.S.J. Pister, "3DμV - A MEMS 3-D Visualization Package", *Proceedings of SPIE*, 1995.

[2] Matlab User's Manual, The Mathworks Inc., 3 Apple Hill Drive, Natick, MA 01760.

[3] B.R. Vatti, "A Generic Solution to Polygon Clipping", *Communications of the ACM*, July 1992, pp.56-63.

[4] B. Zalik, M. Gombosi and D. Podgorelec. "A Quick Intersection Algorithm for Arbitrary Polygons", *SCCG98 Conf. on Comput. Graphics and it's Applicat.*, 1998, pp.195-204.

[5] Brian Mirtich, "Impulse-based Dynamic Simulation of Rigid Body Systems," Ph.D. thesis, University of California, Berkeley, 1996.

[6] Alan Paeth, "Graphics Gems V", Academic Press, 1995.

[7] Joseph O'Rourke, "Computational Geometry in C", 2nd Edition, Cambridge University Press, 1998.

[8] L. Y. Lin, J. L. Shen, S. S. Lee, and M. C. Wu "Surface-Micromachined Micro-XYZ Stages for Free-Space Micro-Optical Bench", *IEEE Photonics Technology Letter, Vol. 9, No. 3*, March, 1997.

[9] Matt Last, K.S.J. Pister, "2-DOF Actuated Micromirror Design for Large DC Deflection", *Proc. of MOEMS* '99.