3D MEMS Design via Interactive Plot with Matlab

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

*Department of EECS, 497 Cory Hall, UC Berkeley
Berkeley, CA 94720

Electrical Engineering Department, UCLA
405 Hilgard Avenue, Los Angeles, CA 90095, U.S.A.
email: nanping@ee.ucla.edu

Several commercially available MEMS CAD packages perform excellent 3D rendering of microstructures. However, since 3D display is not the primary goal of these commercial packages, other than providing views from different perspectives, the renderings from these MEMS CAD packages are static and do not permit the designer interaction with parts within the design. The previous works by the authors attempted to address the interactivity issue with custom software and Java/VRML solution. It was not widely used because it was viewed primarily as a visiualization tool and there were installation issues. In this paper, the authors extend the solution for interactive 3D MEMS design by using Matlab as the underlying number crunching and displaying engine. With it, the designers can manipulate and fold structures away from substrate directly from their flat, 2D CIF design. Raytracing, for MEOM design, and collision detection, for micro robotic design, have already been incorporated in this Matlab toolkit.

Figure 1 outlines the process flow for using Matlab's 3D plot for interactive MEMS design. A preprocessor, Cif2mat, which is written in C, processes the standard CIF files by adding the process dependent topology information. The topology is obtained by performing boolean manipulations on layers within the design. Once the layers in the design have acquired the height information, all spatially jointed boxes are sorted into corresponding structures. If the interconnects between structures, i.e. substrate hinges and/or scissor hinges, are layed-out in a specific fashion, the preprocessor is capable of detecting the interconnection between structures. Thus, although all structures are spatially disjointed, linkage information (connection via hinges) between structures, as well as rotational data can be generated. Furthermore, if pseudo layers were used in the design, the preprocessor will treat the areas as the reflecting surfaces for raytracing process. The topologized design along with all the information gathered by the preprocessor are written to an intermediate file in a format called *M3d*.

The boxes in the intermediate data file are read into Matlab's memory space by specifying the vertices of Matlab graph object called "patch". Once the data for the structures in the design are setup, it is Matlab's underlying plotting functions that display all the patches as 3D plot. Manipulation and movement of structures, such as rotation and transition, are done by applying the corresponding spatial delta to the vertices of all the patches involved. After each movement, Matlab's underlying 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 in Mat3d exist to assist the designer with the following tasks:

- identify, manipulate and view structures
- get and set linkage and rotational information (if not already extracted by the preprocessor)
- specify and commence raytracing parameters
- commence collision detection

Figures 2 thru 4 show a series of plots and a SEM of a XYZ micro-optical stage, from its 2D design, to the assembled device in 3D space with raytracing to show its functionality. Figure 5 shows a hollow trianglure beam linkage in 2D design and the linkage and rotational information automatically extracted to assembled the device. With the rich set of analytical tools already in existence for Matlab, additional MEMS analysis tools can be added on top of this toolkit. Thus perhaps one day enable the rapid design and analysis necessary to bring the futuristic bug-on-a-chip design, as seen in Figure 6, to life.

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

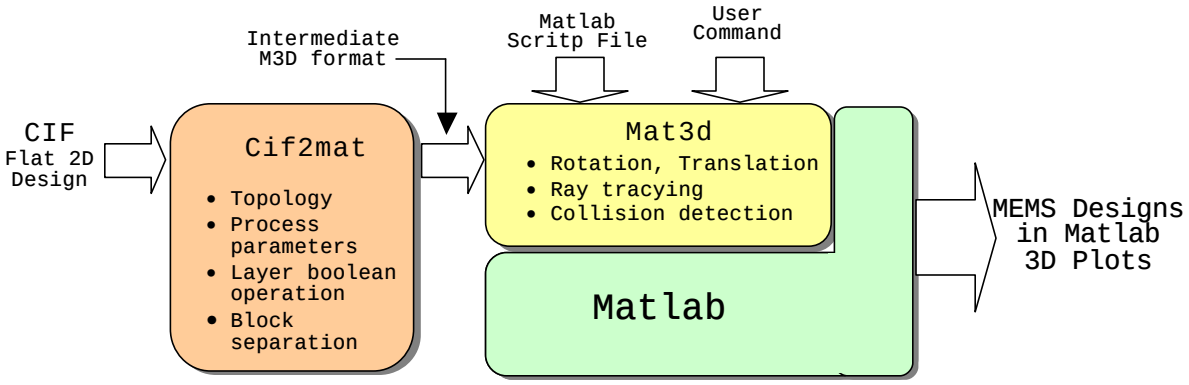

Figure 1. Process flow of plotting 3D MEMS designs with Matlab.

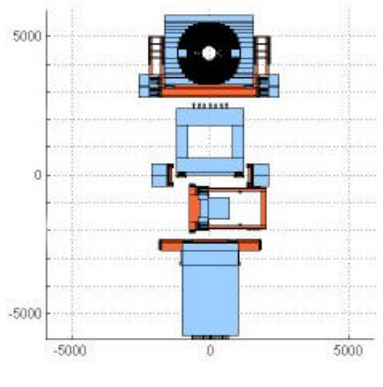

Figure 2. Plot of the 2D flat design of the XYZ-stage.

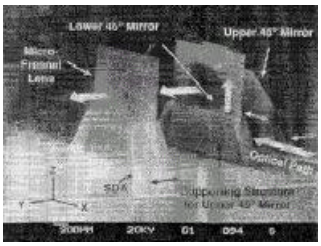

Figure 3. SEM of the XYZ-stage.

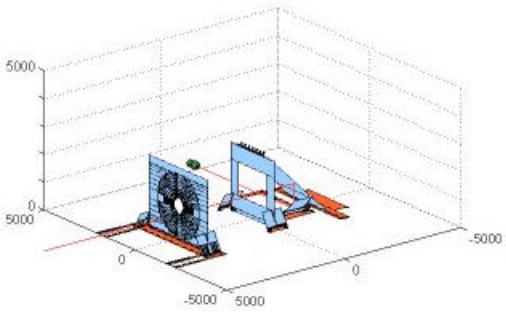

Figure 4. Plot of the assembled XYZ-stage with raytracing demonstrating its functionality.

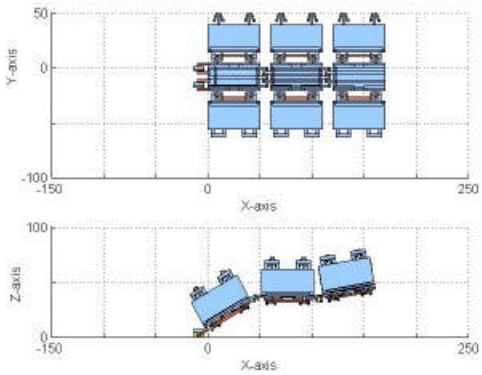

Figure 5. Plots of 2D design (top) in xy-plane, and the assembled linkages (bottom) in xz-plane.

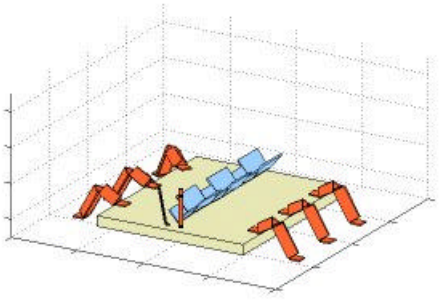

Figure 6. A plot of a futuristic design of micro-bug on a chip.