Fracturing of periodic layouts for photonic crystals

Fracturing of layouts has the objective to optimise a design by creating convenient cuts on it. These cuts define smaller exposure regions since either are required due to the exposure field size or to avoid stitching issues. However, fracturing is challenging since the shapes in a layout can be arbitrarily fragmented, which reduces quality instead of improving it. This application note describes a method to optimally fracture photonic crystals using BEAMER.

INTRODUCTION

Photonic crystals are periodic optical structures used to control the propagation of light (see Fig. 1). Devices compounded with photonic crystals benefit from the periodicity of the layout to manipulate the index of refraction and the absorption of the device. Usually, photonic crystals cover an area of few microns, and thus, they need to be fractured for their fabrication. Accordingly, the precision of the pattern and fabrication process is of utmost importance to reduce scattering losses while warranting a desired optical response.

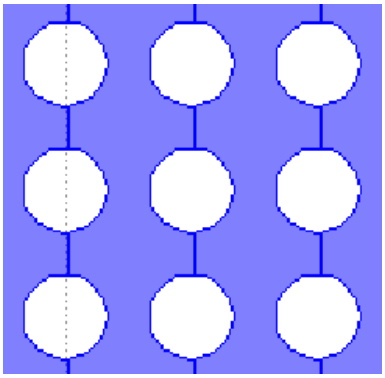

Figure 1. Photonic crystal of circles.

FRACTURING A PHOTONIC CRYSTAL WITH CIRCLES

As an example, let us take a photonic crystal (PC) designed with holes with diameter of 200 nm. Such a device is generated using multiple vertices (see Fig. 2). These vertices increase the resolution of the layout, providing a smooth outline regardless of the shape. However, if the pattern is fractured, each vertex defines the grid for the fragmentation (see Fig. 3). The fracture of the layout changes when varying the number of vertices or the hierarchy of the design. Additionally, the computation time increases with a higher number of vertices.

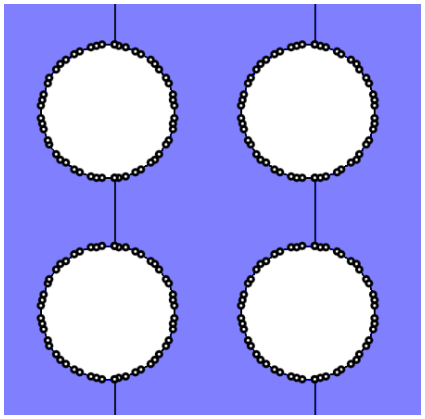

Figure 2. Vertices outlining circles in a PC.

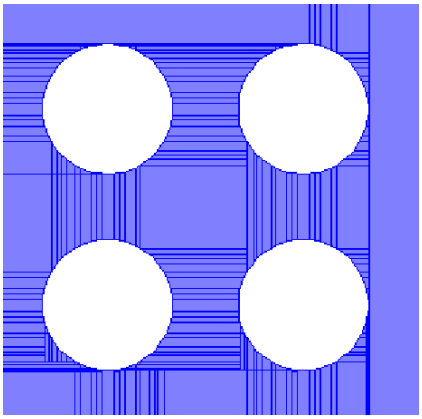

Figure 3. Fractured layout following the vertices of the layout.

Fortunately, BEAMER helps to optimise the fracture of curved shapes. The following steps are recommended to improve the output of circular holes, but they can be extended to other geometries. Figure 4 depicts a basic BEAMER flow with the imported layout at the top and the fractured file exported at the bottom.

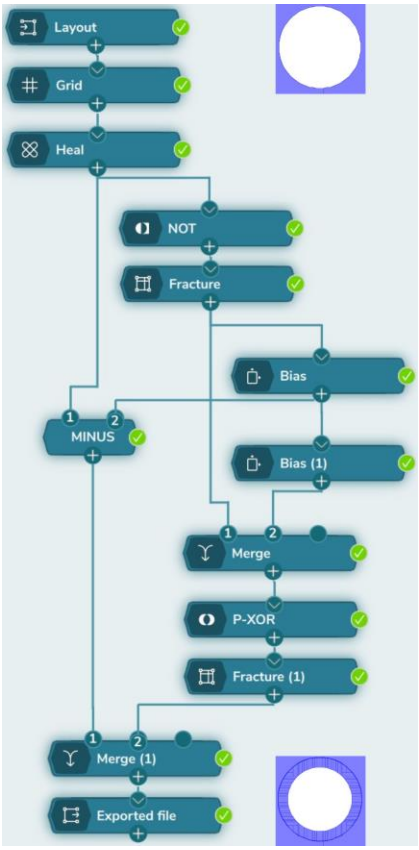

Figure 4. BEAMER flow to optimise curved fractures.

Firstly, the modules *Import*, *Grid* and *Heal* are used to import the layout, lower the resolution of the pattern, and heal possible overlaps, respectively. Particularly, the *Grid* module is used to increase the

design-mesh reducing the number of vertices (see Fig. 5). In our example, the grid is increased from 1 nm to 2 nm, this reduces the number of vertices from 5414 to 4714.

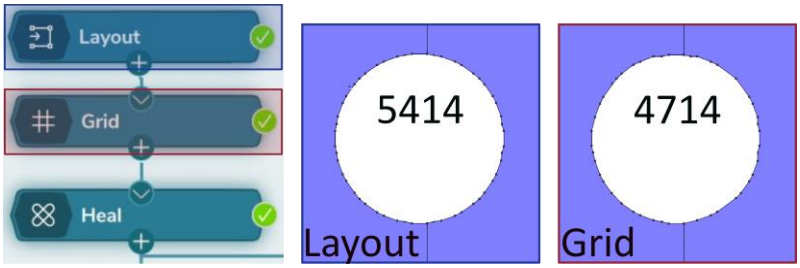

Figure 5. (left) Import, Grid and Heal modules, (middle) imported layout with 5414 vertices, and (right) layout after grid module with 4714 vertices.

Afterwards, the *NOT* module is used to invert the polarity of the pattern, with this, the white-areas or empty-spaces will define the circular shapes. Later, these circles are detected by the *Curved* mode in the *Fracture* module. This mode optimises the circular fragmentation by redistributing the position of the vertices in the shape (see Fig. 6).

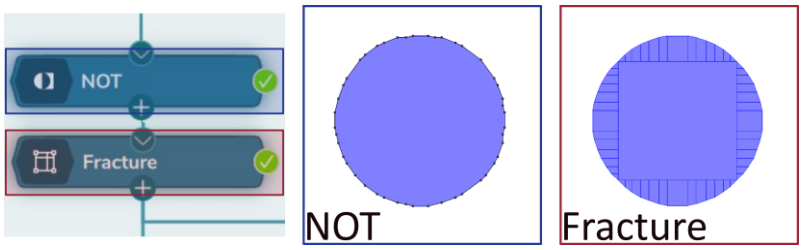

Figure 6. (left) NOT and Fracture modules, (middle) inverted layout obtained after NOT module, and (right) curve mode fragmentation of the circles.

Once the vertices are uniformly spaced, two *Bias* modules are connected in the flow. The first module enlarges the circles by 30 nm around the circumference, and thus, the new diameter is 260 nm. This operation determines an area to place the original holes. The second module adds 2 nm to this area (see Fig. 7) since this will warranty the overlap between the space created with the first bias and the original design (the overlap is clearly seen in Fig. 10). Then a *Minus* module subtracts from the *Heal* module the first *Bias* module (see Fig.4). As a result, the layout is like the initial photonic pattern but with larger holes in the design. Figure 8 depicts the overlap of the *Heal* module (blue) with the *Minus* module (yellow).

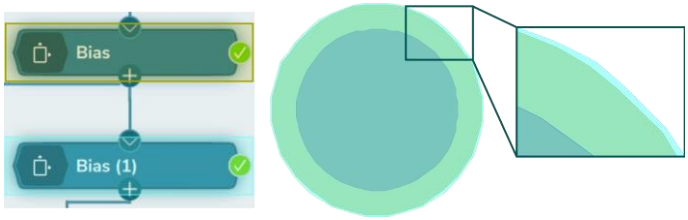

Figure 7. (left) two Bias modules to give spaces to the photonic crystal holes, (right) overlap of the two bias modules with the initial circles from the NOT module.

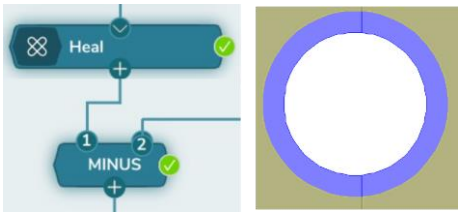

Figure 8. (left) Heal and Minus modules connected and (right) overlap of both results: Heal (blue) and Minus (yellow)

After the second *Bias* module, a *Merge* module combines the *Fracture* and the second *Bias* results (see Fig. 9a). Afterwards the *P-XOR* module has the function to preserve odd overlaps in a layout; therefore, removes the region where the *Bias* module intersects with the *Fracture* module leaving a ring of 32 nm thick (see Fig. 9b). Then, the *Fracture* module fragmentates the resulting ring (see Fig. 9c).

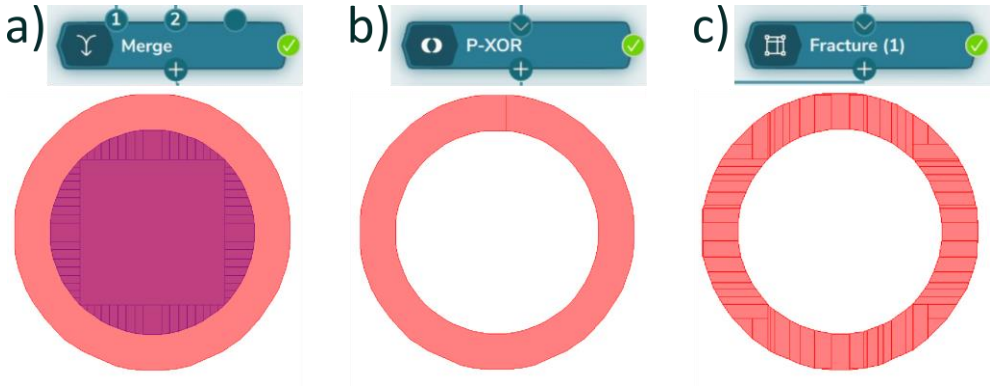

Figure 9. a) Merge module combines the Bias and Fracture results, b) the P-XOR module remove even overlaps from the Merge module, and c) the Fracture module optimises curved fragmentation on the resultant ring.

Figure 10 depicts the result of adding a *Merge* module that combines the *Minus* (red) and the *Fracture* (blue) modules. As is shown in the figure, there is ring of 2 nm where the fractured ring overlaps with the sheet of holes. These 2 nm were added using the second *Bias* module mentioned in fig. 7.

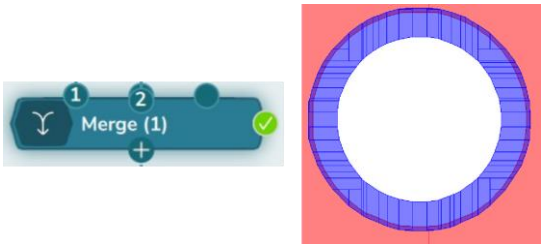

Figure 10. Merge module combines the fractured rings with the rest of the layout.

Finally, the layout is exported to a desired tool format using a *Conventional Fracture Mode* and a *Fixed Field Placement*. These settings in the *Export* module preserve the previous fractures created with the flow (see Fig. 11).

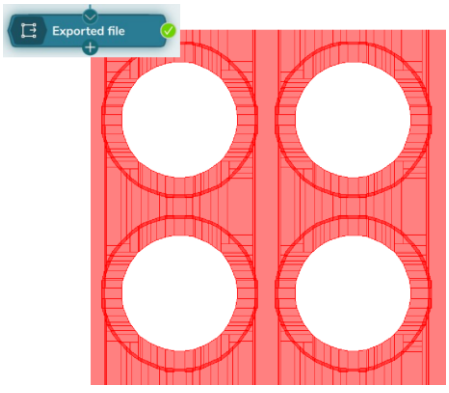

Figure 11. Exported layout.

SUMMARY

- Curved fracturing works for photonic crystals:
    - Positive tone
    - Negative tone
- Decouple vertices from adjacent features to help optimise the fracture.
- Applied Technologies:
    - Grid: Layout Smoothing Tolerance
    - P-XOR
    - Curved Fracturing
- Solution is independent of the design whether it is hierarchical or flattened.
- Since *Curve fracturing* is unique to BEAMER, this can only be done with BEAMER.