Filling arbitrary shapes with tilted gratings

This application note outlines a method for populating an arbitrary shape with a tilted grating using BEAMER. This method only requires a source shape and some grating parameters to generate the desired pattern.

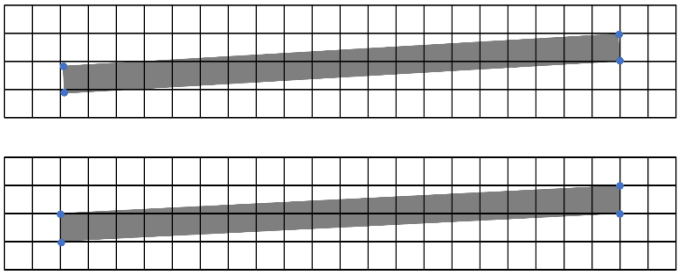
INTRODUCTION

Nowadays, virtual and augmented reality are becoming increasingly important for society. Manufacturing of devices for these purposes always involves the production of gratings in specific areas of the device. Designing these layouts typically does not take the capabilities of the tool into account and can lead to poorly fabricated results. With an algorithmic approach, we can remedy this situation.

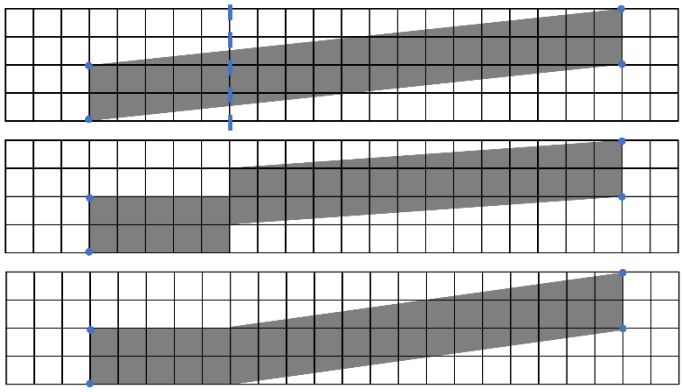
CHALLENGES IN LAYOUT DRIVEN METHOD

A common way to create tilted grating is to create vertical gratings and then rotate them. The problem with this approach is that the design vertices snap to the database grid yielding pattern irregularities.

Since the layout shape must snap to the database grid, the endings of a tilted grating line may not be located at the correct intersection of the database grid. It may be relocated and snapped onto a neighbouring intersection point, so that the grating stored in the database deviates from the target design.

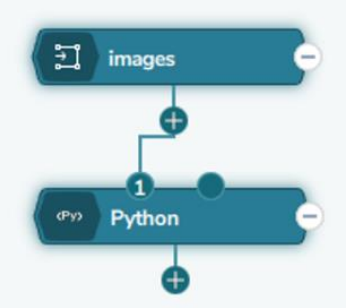


Similarly, when the tilted grating is fractured at field or subfield borders into independent shapes inside the database, the fractured line endings not located precisely at grid intersection points can snap into neighbouring intersection points, causing straight lines to be bent or broken, and leading to a departure from the target design.



In BEAMER, the shape contour to be populated can be extracted from the layout with an Import module. Based on the parameters defined in a Python script, a single line can be created and repeated to form a grating within the subfield. After many repetitions, a main field can be constructed from its subfields and an entire pattern from its main fields. Essentially, the imported pattern acts as a stencil mask to trim the grating into the desired pattern.

BEAMER FLOW USABILITY



When the grating feature and layout filled are known, the user can use a BEAMER flow to effectively populate the grating in layout. The flow includes Import and Python modules.

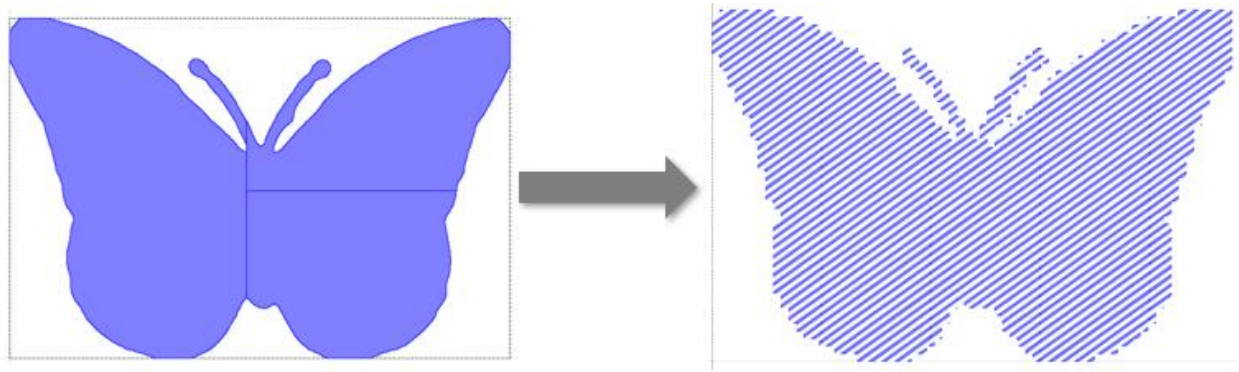
In the Import module, the user can import arbitrary layouts where the grating is populated in.

In the Python module, the user only needs to customize grating features at the beginning of the script as shown below.

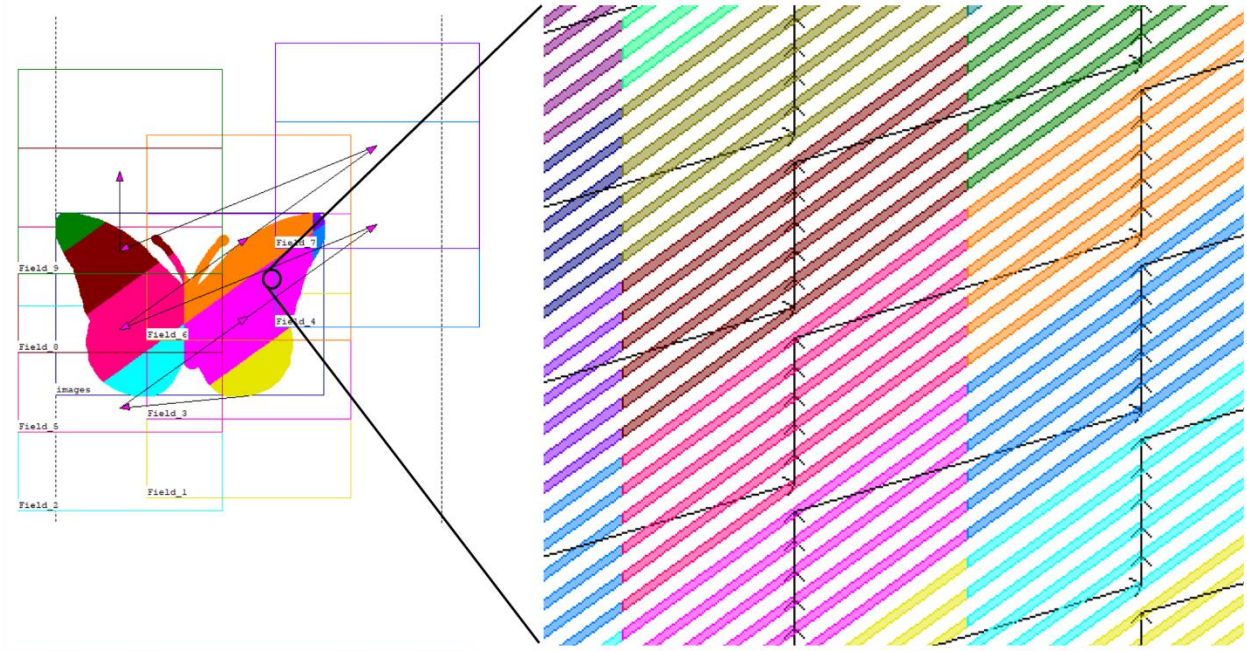
```
from LAYOUTpy import *
import math
import sys
# GRATING PARAMETERS
angle =35.5                      # Angle of grating in [°]
perpendicular_width = 0.070      # Line Width perpendicular [um]
perpendicular_pitch = 0.19321    # Pitch perpendicular [um]
# TOOL SETTINGS
subfieldsize = 4         # Subfield Size in [ um ]
subfield_usage = 0.9     # factor of subfield usage
mainfieldsize = 800      # Mainfield Size in [ um ]
```

The input features include [1] the angle between grating lines and horizontal direction, [2] the single line width in the orthogonal direction, [3] the grating pitch in the orthogonal direction, [4] subfield size, [5] the factor of subfield usage (effective subfield size = subfield_usage * subfieldsize) and [6] the mainfield size.

After importing the layout and defining the grating features, running the BEAMER flow will populate the imported layout with the grating with the specified dimensions as shown below.



User can also view writing order in BEAMER as shown below. In the left part of the image below, the area with different colour in butterfly shape corresponds to different main field. The red arrow trace how the main field is written one-by-one. The right part of the image below is zoom-in view of left image. The grating with different colours corresponds to different subfield. The black arrows indicate how the grating line is written inside subfield and between the different subfields.



Main Field Writing Order

Subfield Writing Order