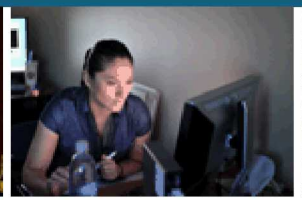
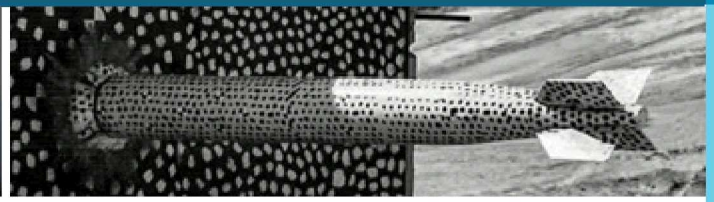
SAND2020-1459C

An Optical Test Simulator Based on the Open-Source Blender Software





IMAC XXXVIII

Dan Rohe

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Outline

Blender Overview

Digital Image Correlation Example Workflow

Uses for Test Planning

Sub-pixel Resolution Analysis (in paper)

Blender Overview

What is Blender?

- "free and open source 3D creation suite. It supports the entirety of the 3D pipeline—modeling, rigging, animation, simulation, rendering, compositing and motion tracking, even video editing and game creation. Advanced users employ Blender's API for Python scripting to customize the application and write specialized tools" – blender.org

Why use Blender?

- Mature, open-source software capable of producing photorealistic images
- Graphical user interface – more intuitive for learning how to use the software
- Python Programming interface – automate the tedious bits

What will we use it for?

- Geometry definition and deformation – Surface meshes of finite element models
- Material definition – local texture of surfaces; how light interacts with the geometry (specular or diffuse, transparency)
- Lighting definition – implications for shadows, specularities, and contrast
- Camera definition – Focal Length, Sensor Size, Aperture, drives field of view, depth of field

What can we do with it?

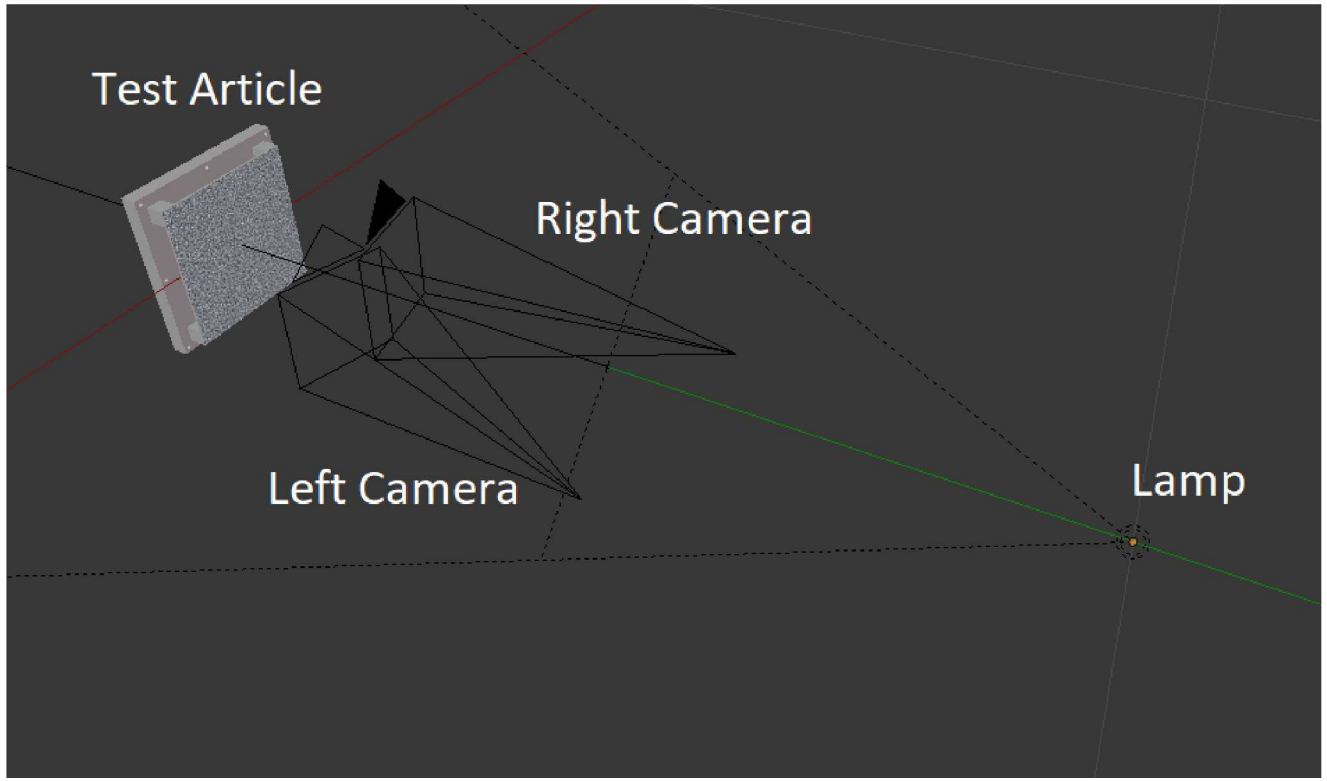
- Virtual optical tests: evaluate image processing algorithms on synthetic datasets where truth responses are known.
- Test planning: examine camera parameters, identify expected pixels of displacement
- Bridge between image and finite element model: provides image degrees of freedom for finite element expansion techniques
  - Rohe and Witt, Predicting 3D Motions from Single-Camera Optical Data, 8105_roh.pdf in conference proceedings.

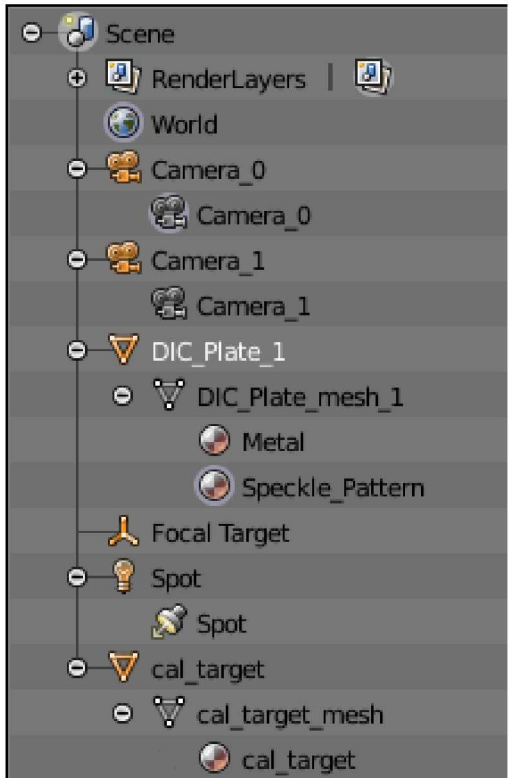
Blender Overview

Key portions of the Blender Data Model:

- Scene – 3D Space in which objects can be positioned
- Object – An abstract "container" that holds other data. Has position, orientation, and scale in the 3D scene
- Mesh – Vertex locations and face connectivity
- Camera – Viewpoint from which the scene will be rendered.
- Lamp – Light source for the scene. Variety of types (spot, point, sun).
- Material – Definition of how light reacts to a mesh's surface.

Example scene that we will create in the workflow:





Preliminary Finite Element Manipulations

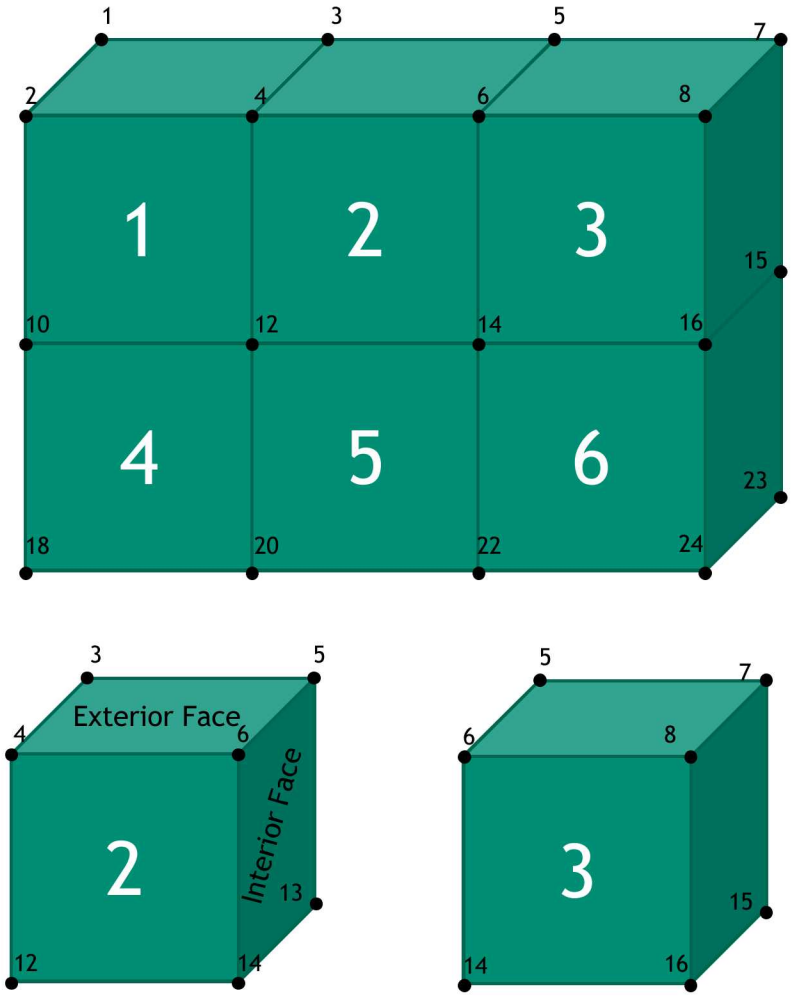
Blender uses surface meshes to define geometry.

Volumetric Finite element models must be "skinned" prior to import.

- Assemble vector of all faces in all elements
- Faces found in multiple elements (face 5,6,14,13 is found in elements 2 and 3) are interior faces
- Faces found in only 1 element (face 3,4,6,5 is found only in element 2) are exterior faces

Face connectivity arrays must be created, which can be imported into Blender using the Python interface.

- Vertices are an $n \times 3$ array where the rows are the vertex indices and columns are (x, y, z) coordinates.
- Faces are $m \times 3$ (triangles) or $m \times 4$ (quads) arrays where the rows are the face number and the columns are indices into the Vertices rows.
- Similar to `patch` Matlab function, except remember Python uses 0-based indexing!



Digital Image Correlation Example Workflow Starting Point

Finite element model has been skinned to make 104458 faces using 104436 nodes.

3D mode shape deformations for 8 elastic modes are extracted at surface nodes.

Saved to .mat file with fields:

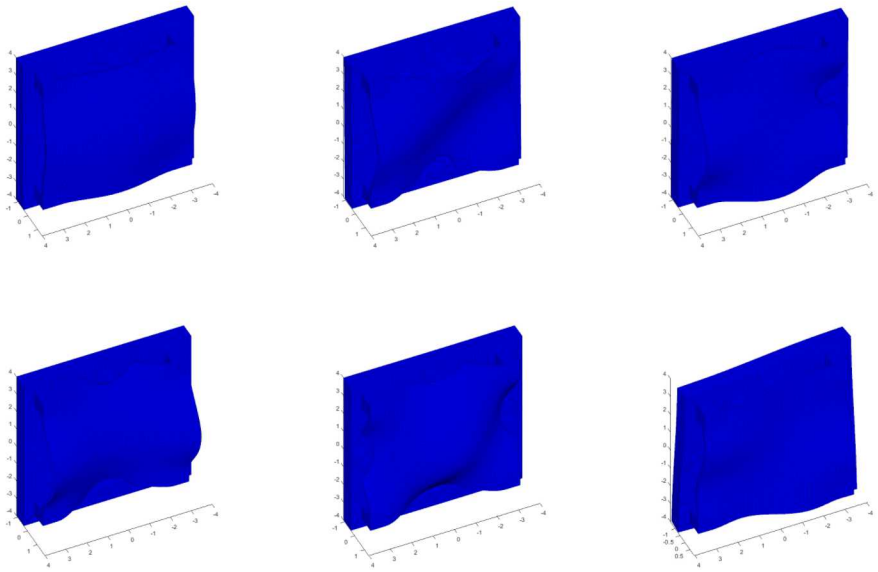
- faces – 104458×4 face connectivity array
- vertices – 104436×3 vertex positions
- frequencies – 1×8 natural frequencies
- shapes – 104436×3×8 shape array

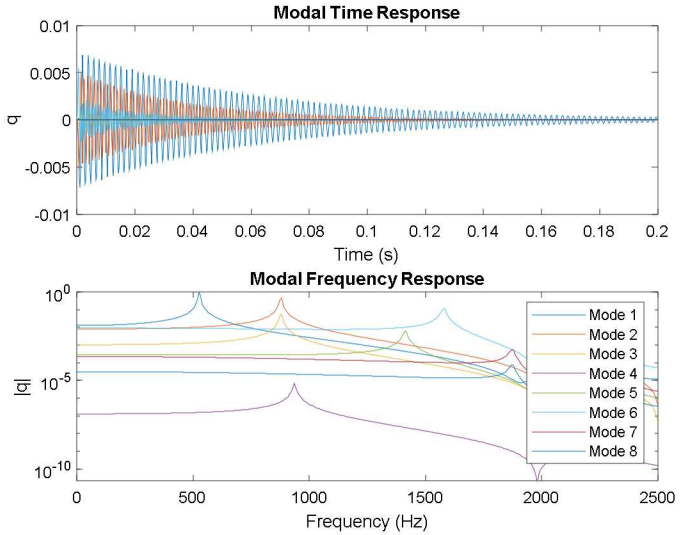
A synthetic hammer impulse has been applied at the corner of the plate, and modal degrees of freedom were integrated for 1000 time steps at 5000 samples per second.

Saved to .mat file with fields:

- q – 1000×8 modal coefficients over time
- t – 1×1000 time steps

These two .mat files were converted to numpy (.npz) files which can be read using the numpy module in Blender.





Digital Image Correlation Example Workflow

Create mesh using python

Add cameras

Add materials

Adjust Render Settings

Add lights

Adjust Speckle Pattern

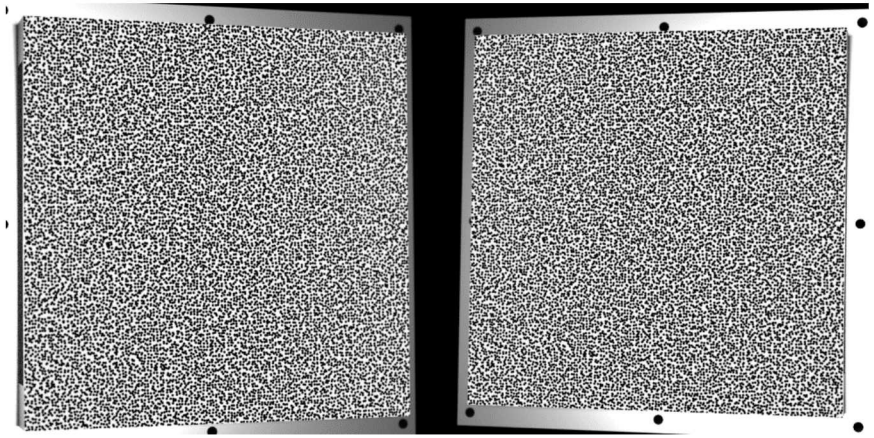
Deform Mesh

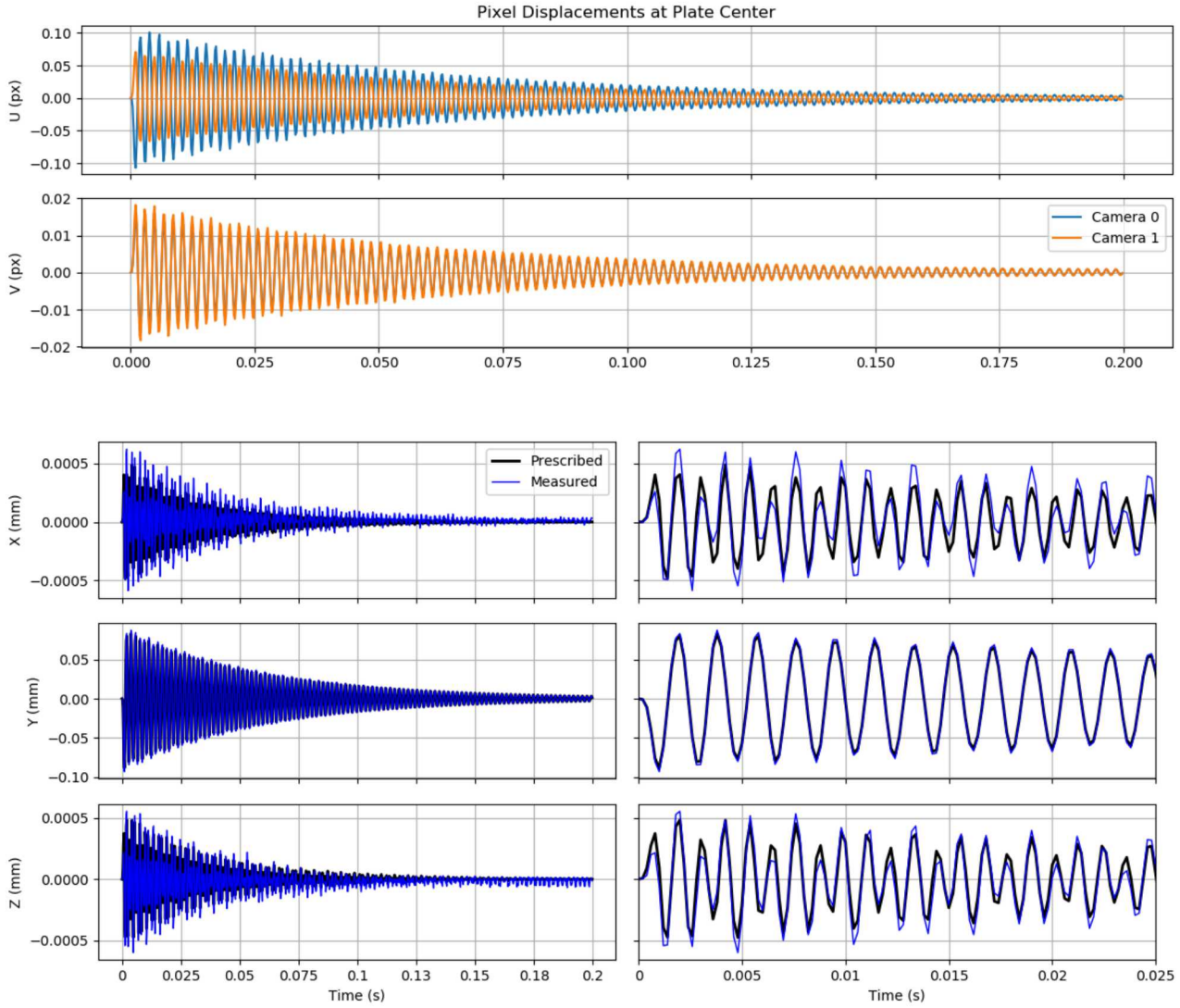
Deform and Render Images

Digital Image Correlation Results

1000 images were rendered with a peak displacement of 0.2 pixels.

Results extracted from stereo DIC agreed very well with the prescribed deformations.





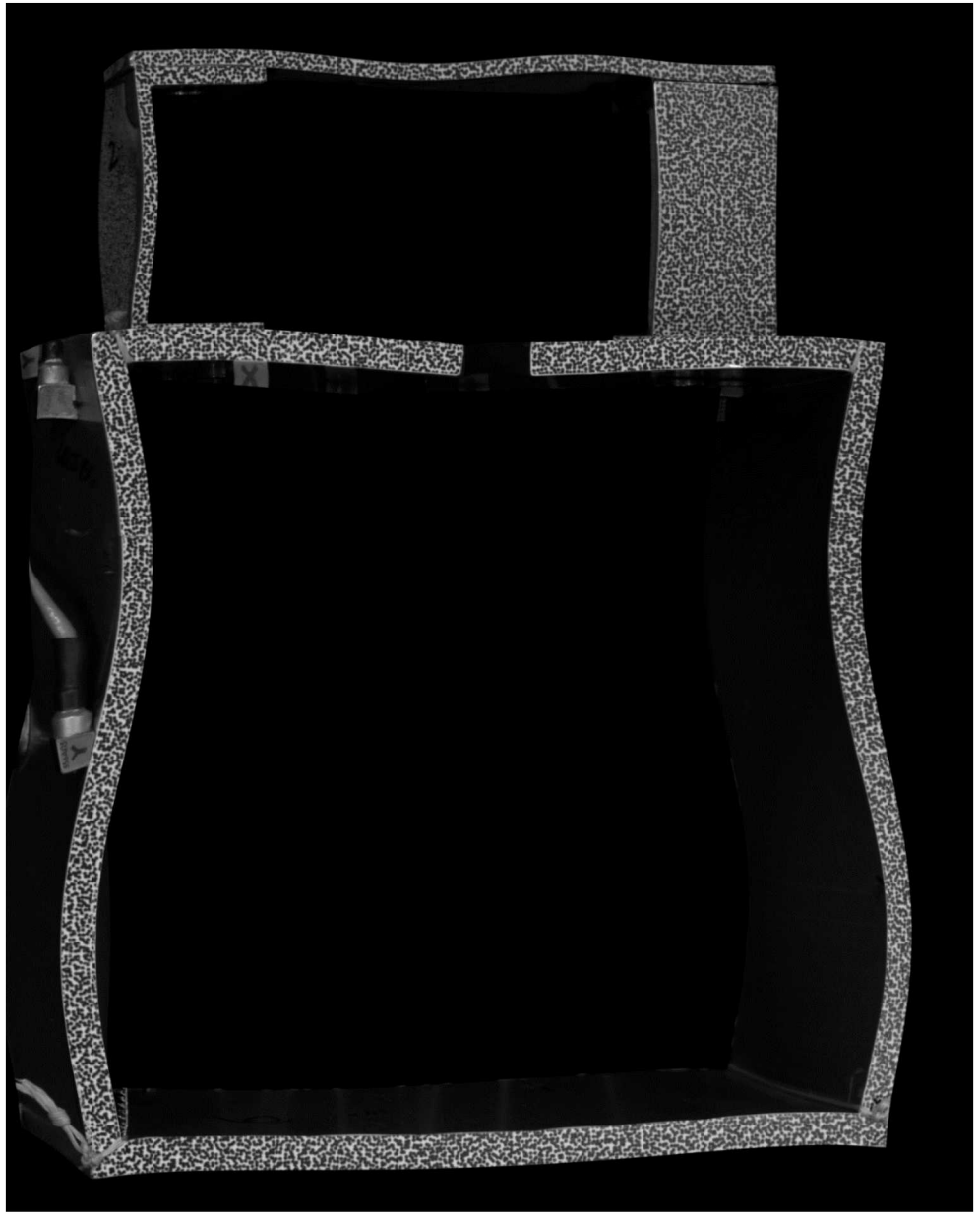
Simulations for Test Planning

Blender has proven useful for test planning; tests can be performed in software to examine things like:

- Excitation level for appropriate pixel response
- Speckle size
- Camera angles and field of view

Speckle patterns produced in software can be exported to graphics programs and printed on stickers

Calibrated test cameras can be reproduced in-software, and test image textures can be projected onto the finite element model



Conclusions

Blender has proven to be a good tool for developing optical techniques

- Ability to create synthetic images
- Truth data available to compare results from processing algorithms

Blender has proven to be a good tool for test planning

- Determine what pixel displacement will be achieved with a given set of cameras
- Determine and create an appropriately sized speckle pattern
- Investigate test setups numerically