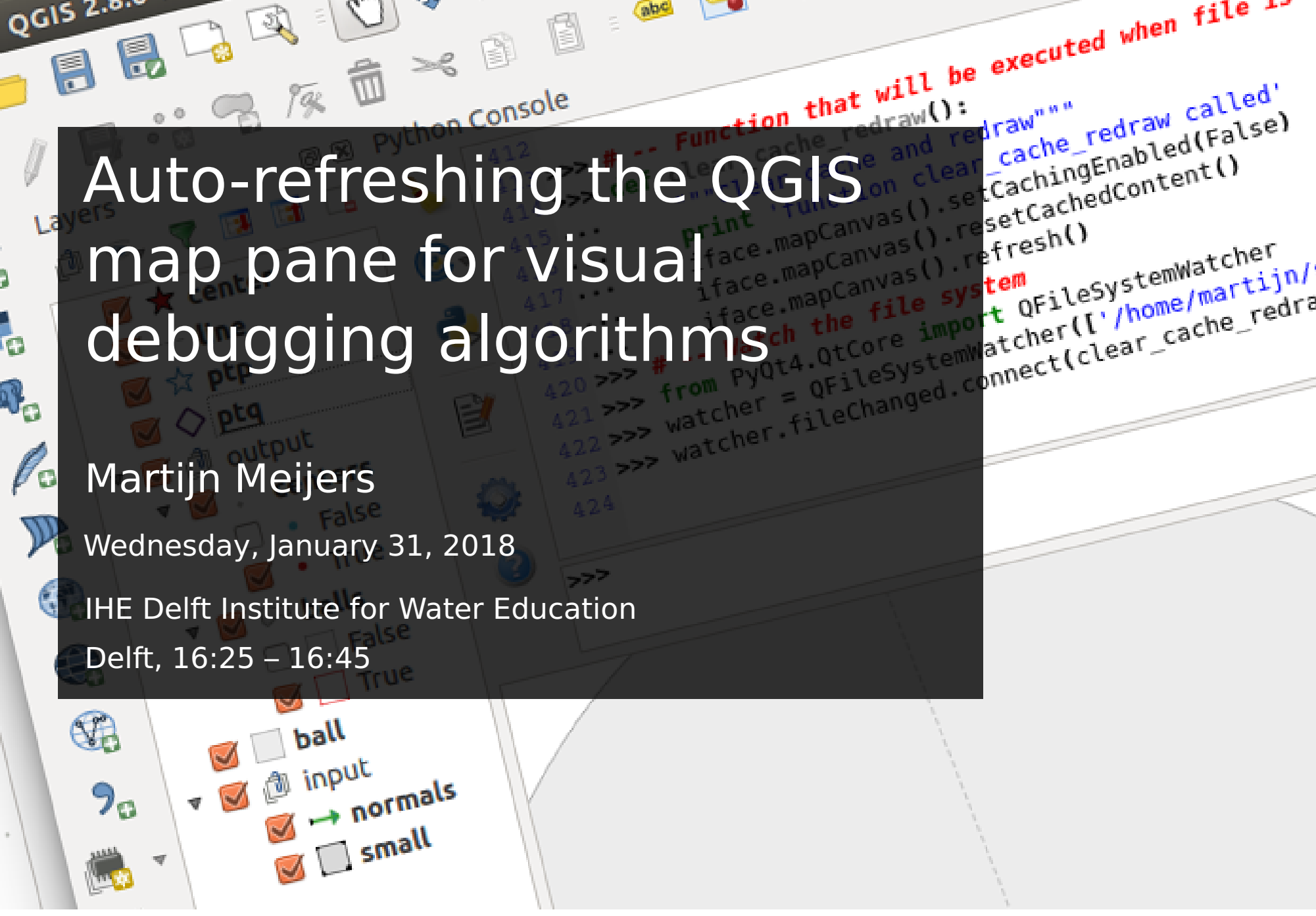


Auto-refreshing the QGIS map pane for visual debugging algorithms

Martijn Meijers

Wednesday, January 31, 2018

IHE Delft Institute for Water Education

Delft, 16:25 – 16:45

TU Delft

Introduction

- Delft University of Technology, GIS technology

Introduction

- Delft University of Technology, GIS technology
- Look under the hood of GIS

Introduction

- Delft University of Technology, GIS technology
- Look under the hood of GIS
- Map generalization: Make vector maps simpler

Introduction

- Delft University of Technology, GIS technology
- Look under the hood of GIS
- Map generalization: Make vector maps simpler
- Develop tools/algorithms/workflows (with Python)

Introduction

- Delft University of Technology, GIS technology
- Look under the hood of GIS
- Map generalization: Make vector maps simpler
- Develop tools/algorithms/workflows (with Python)
- “Share something useful”: Auto-refreshing the QGIS map pane*

*No complete plugin, nor polished solution

Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)

- Given a sufficiently dense sampled polygonal shape

Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)

- Given a sufficiently dense sampled polygonal shape
- Produce points near the approximate center of the shape

Case

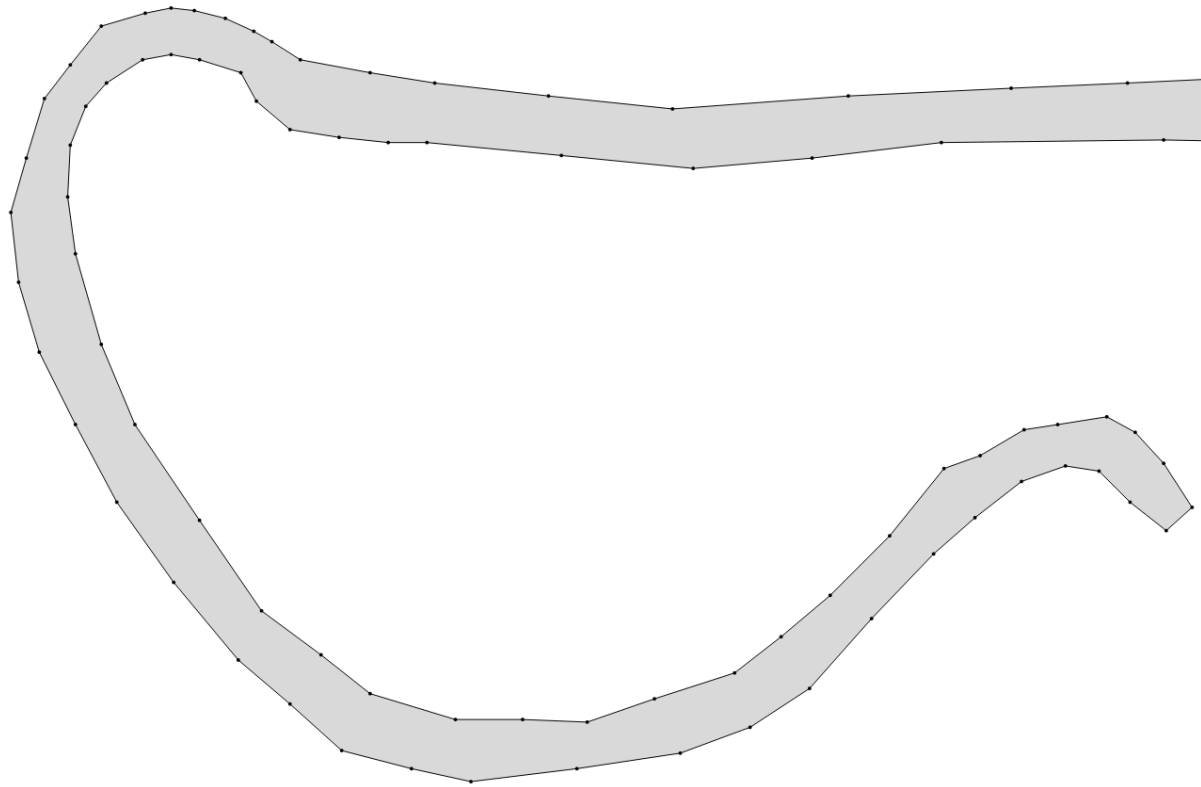
Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)

- Given a sufficiently dense sampled polygonal shape
- Produce points near the approximate center of the shape
- By fitting circles inside the shape

Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)

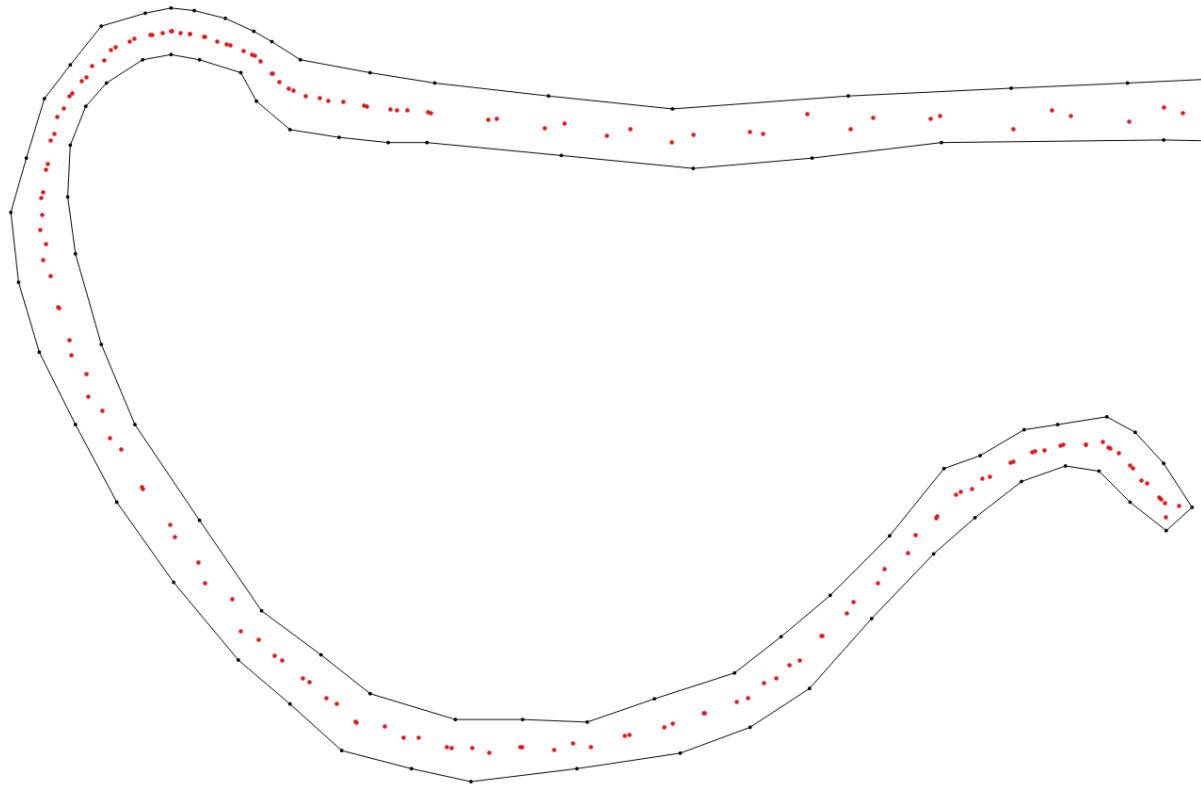
- Given a sufficiently dense sampled polygonal shape
- Produce points near the approximate center of the shape
- By fitting circles inside the shape



Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)

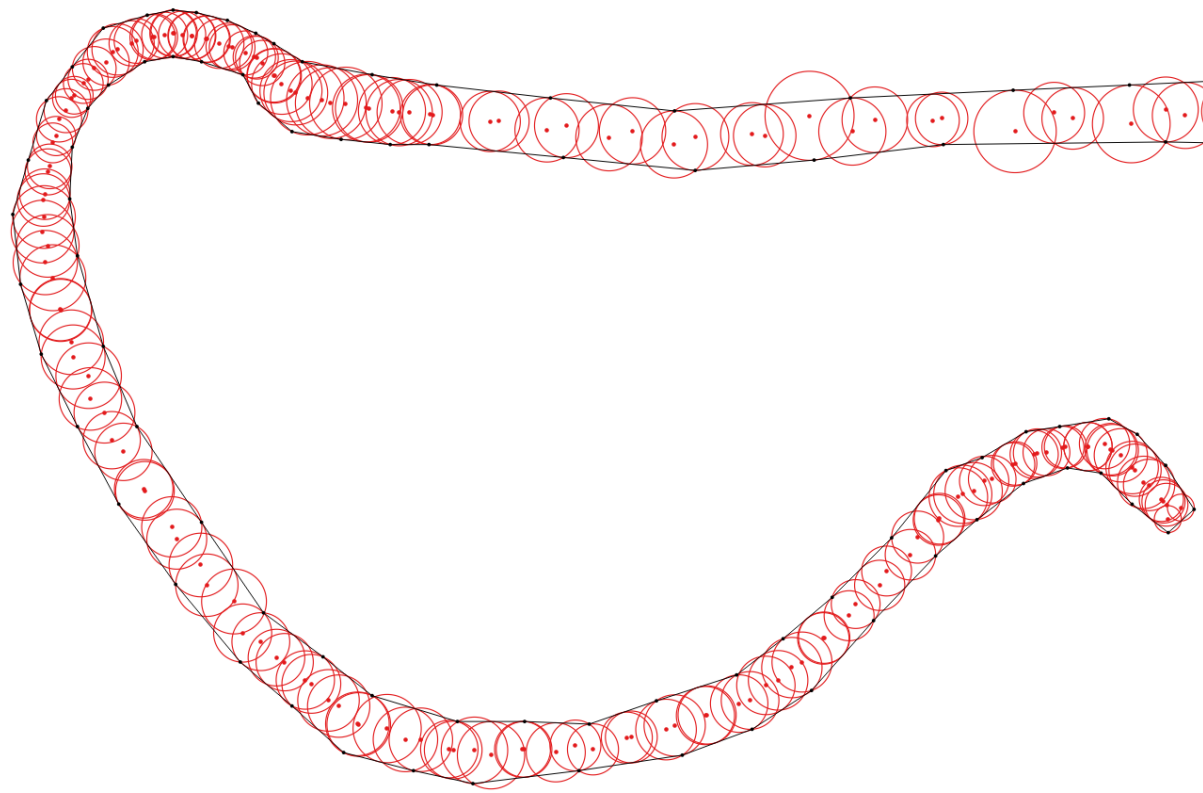
- Given a sufficiently dense sampled polygonal shape
- Produce points near the approximate center of the shape
- By fitting circles inside the shape



Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)

- Given a sufficiently dense sampled polygonal shape
- Produce points near the approximate center of the shape
- By fitting circles inside the shape



Visual Debugging Algorithms

- Input ▷ Algorithm ▷ Output

Visual Debugging Algorithms

- Input ▷ Algorithm ▷ Output
- Algorithm produces intermediate results: Look at these for debugging

Visual Debugging Algorithms

- Input ▷ Algorithm ▷ Output
- Algorithm produces intermediate results: Look at these for debugging
- Graphical debuggers, e.g. Eclipse with PyDev, give tabular view on variables

Visual Debugging Algorithms

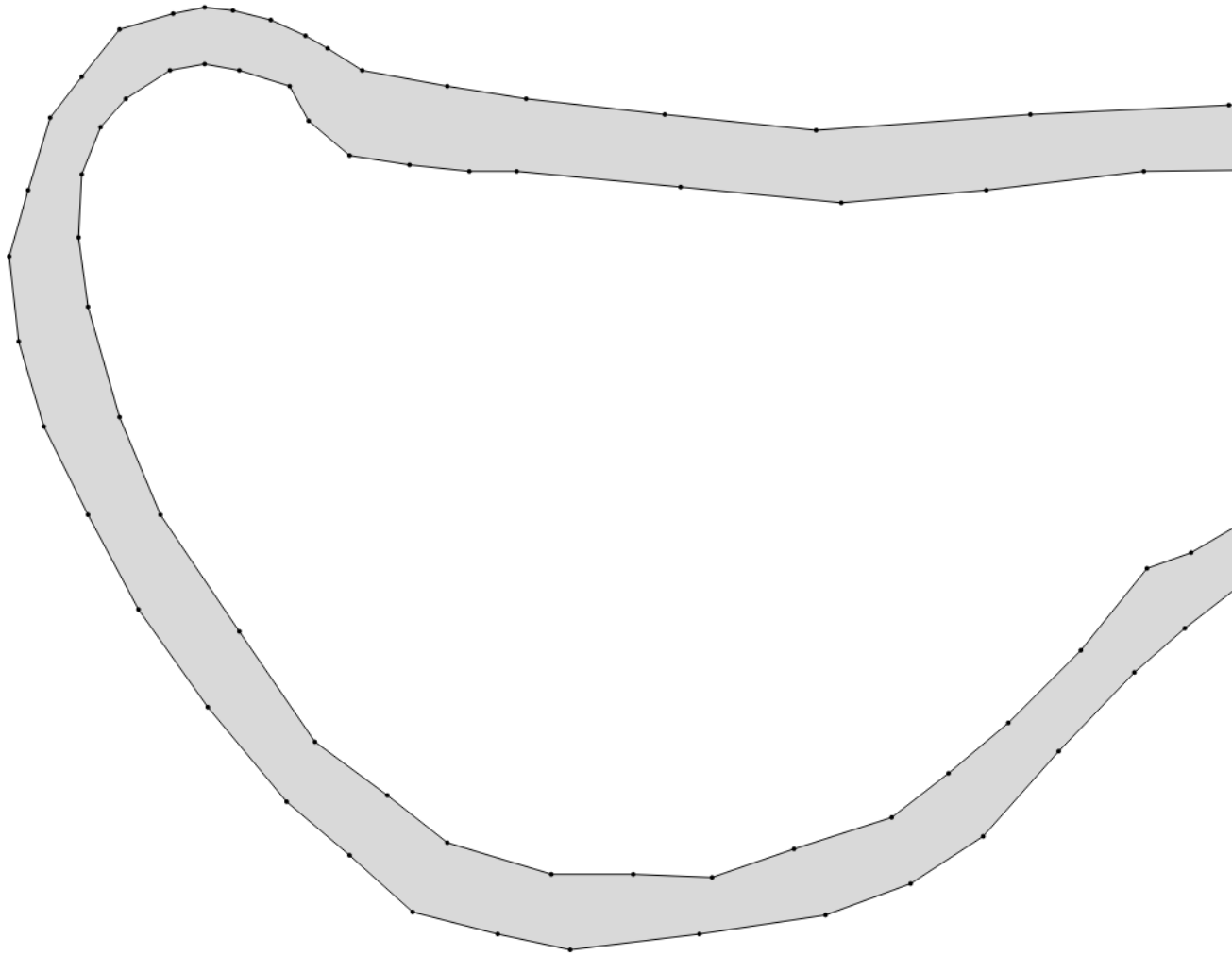
- Input ▷ Algorithm ▷ Output
- Algorithm produces intermediate results: Look at these for debugging
- Graphical debuggers, e.g. Eclipse with PyDev, give tabular view on variables
- No display of geometry of vector data

Visual Debugging Algorithms

- Input ▷ Algorithm ▷ Output
- Algorithm produces intermediate results: Look at these for debugging
- Graphical debuggers, e.g. Eclipse with PyDev, give tabular view on variables
- No display of geometry of vector data
- QGIS to the rescue!
  - Write text file: Tab Separated Values
  - Use Well Known Text (WKT) for geometry (POINT, LINESTRING, POLYGON)
  - Layer ▷ Add Layer ▷ Add Delimited Text Layer...
  - Style layer as usual (e.g. convey direction with marker lines)!

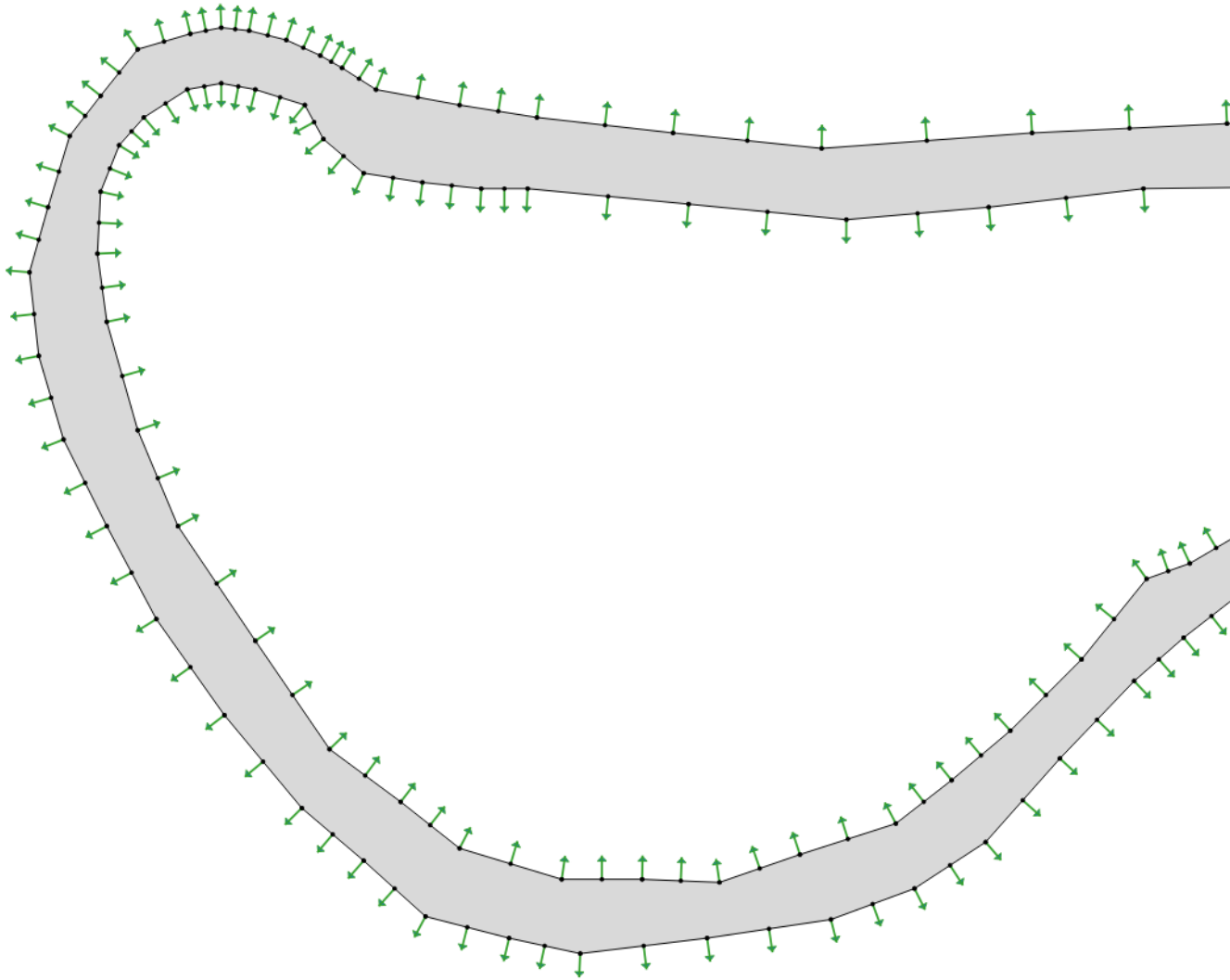
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



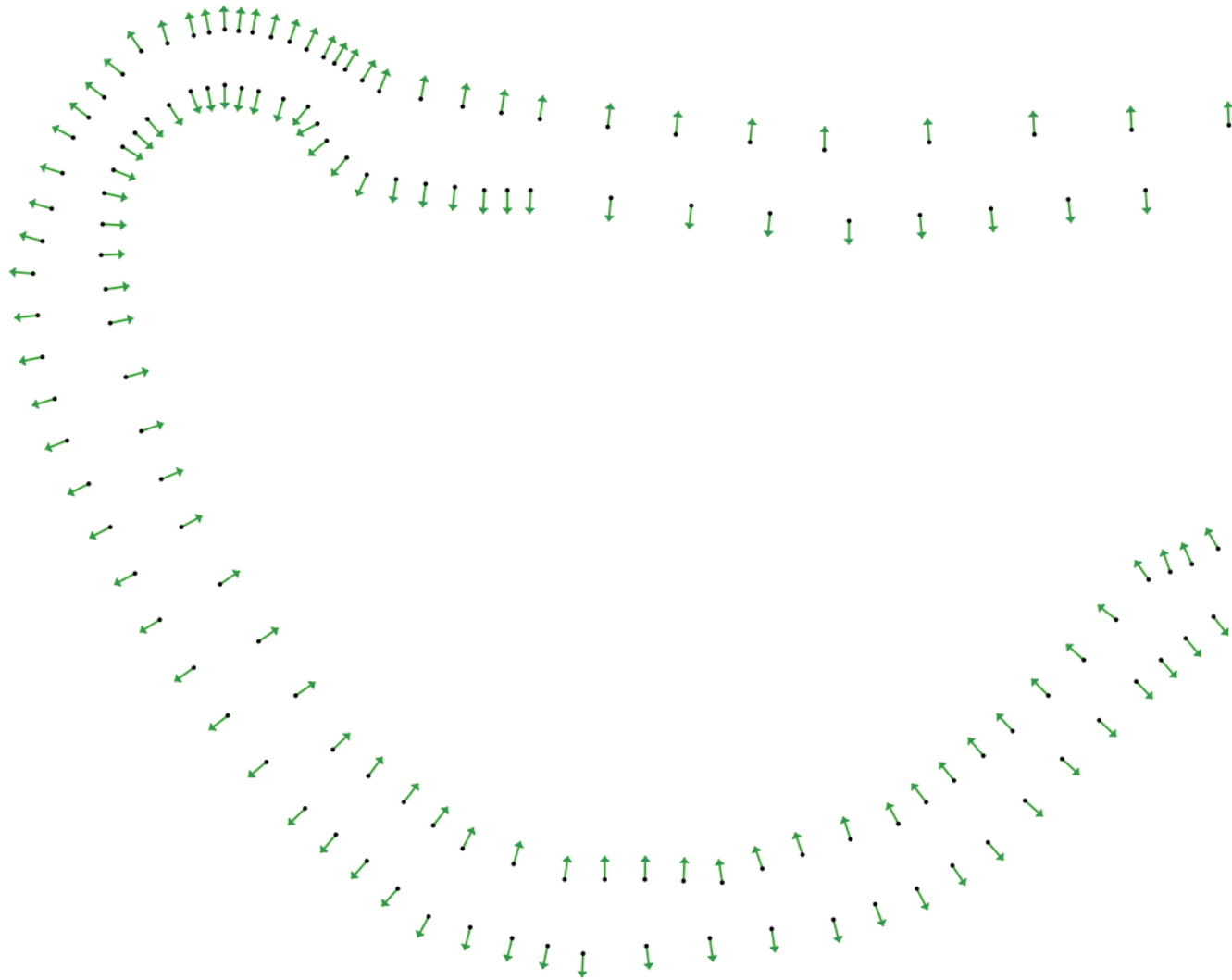
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



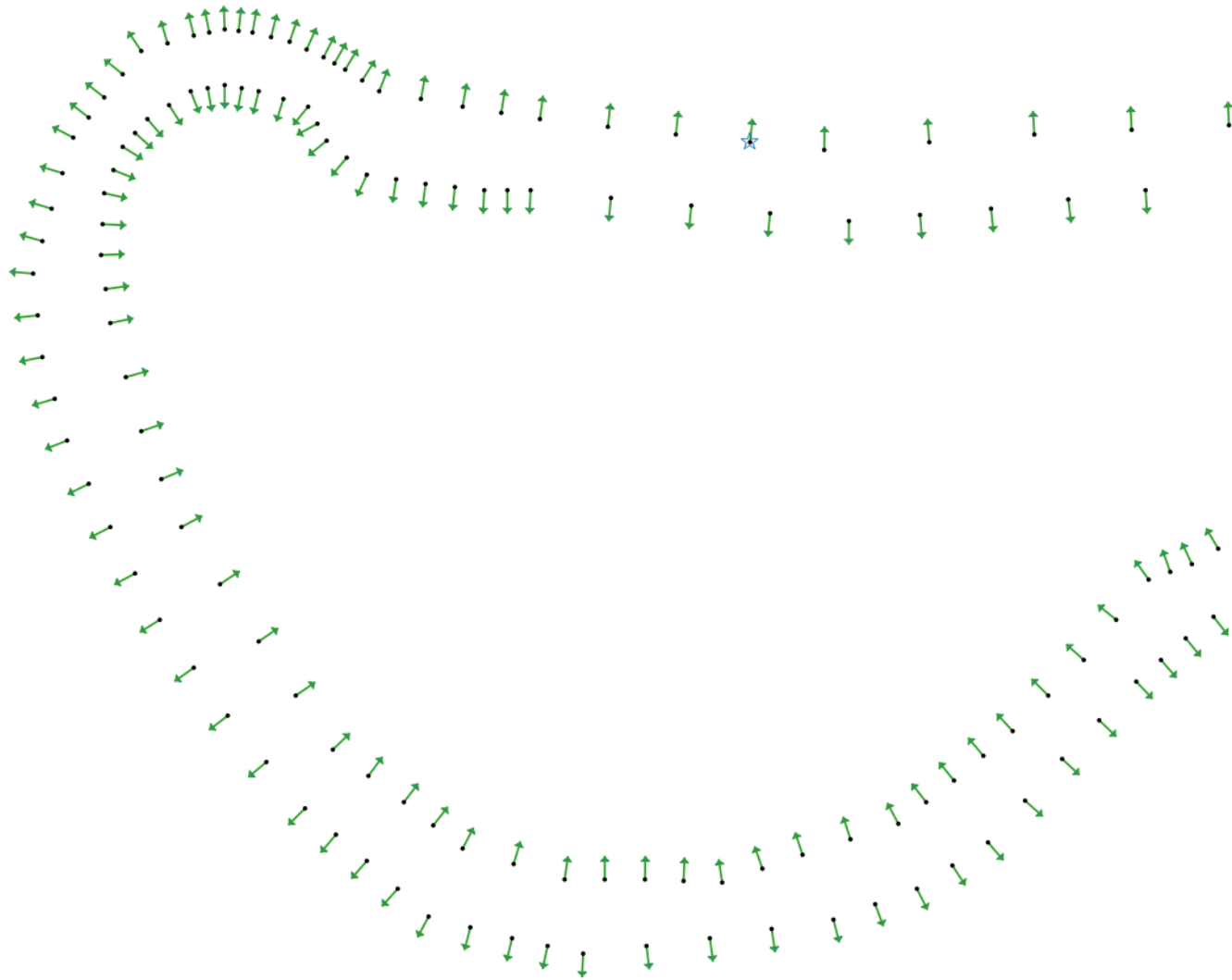
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



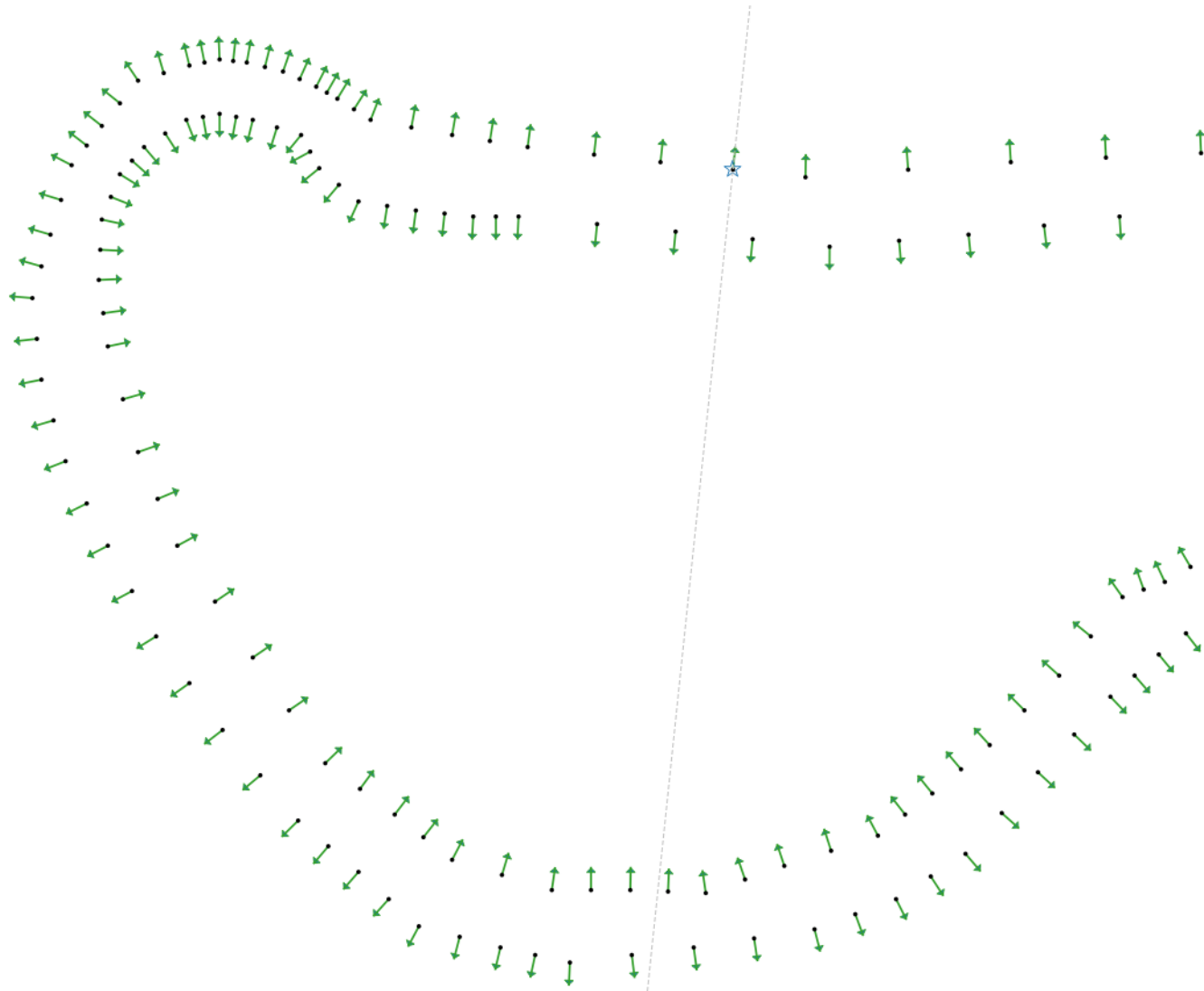
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



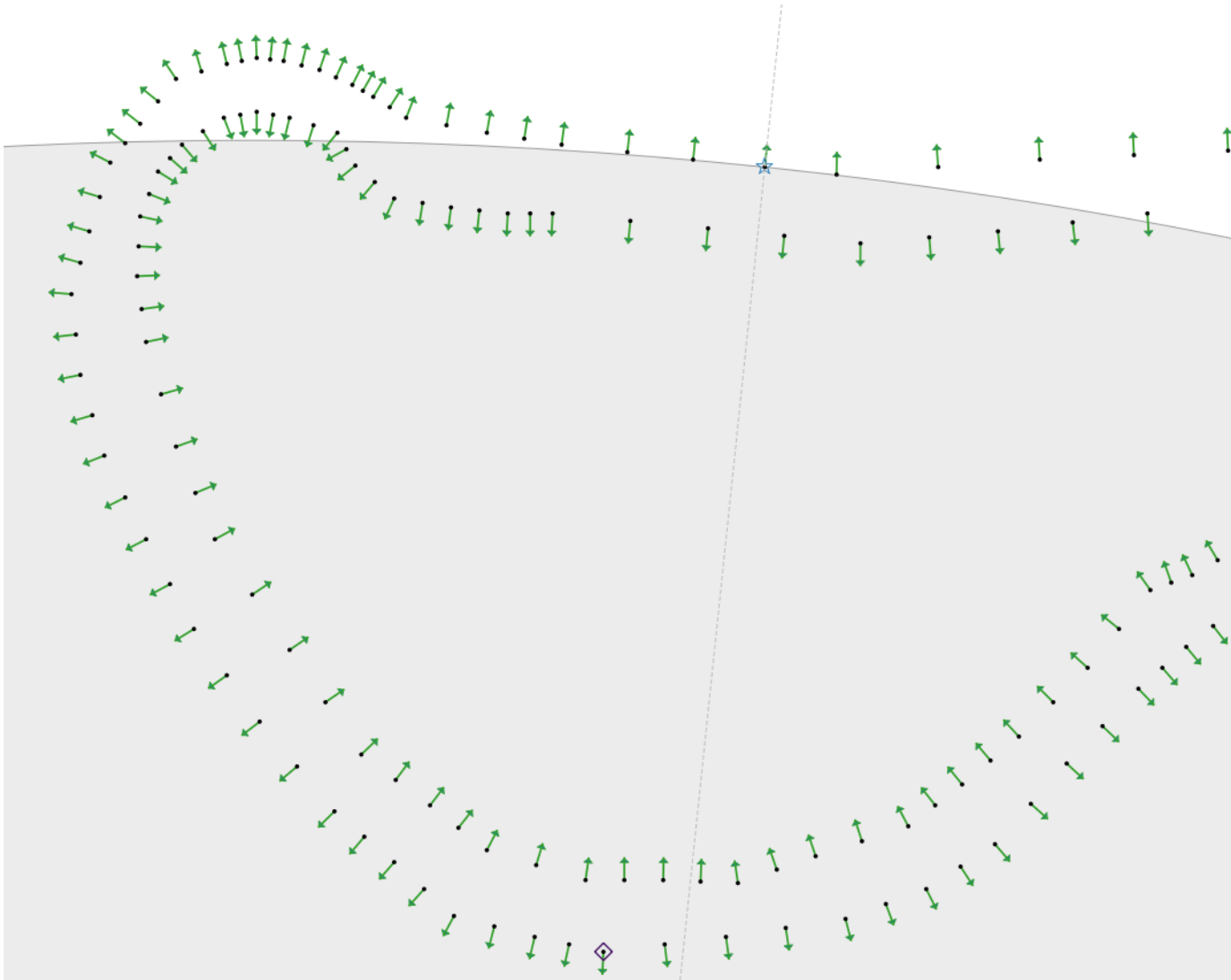
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



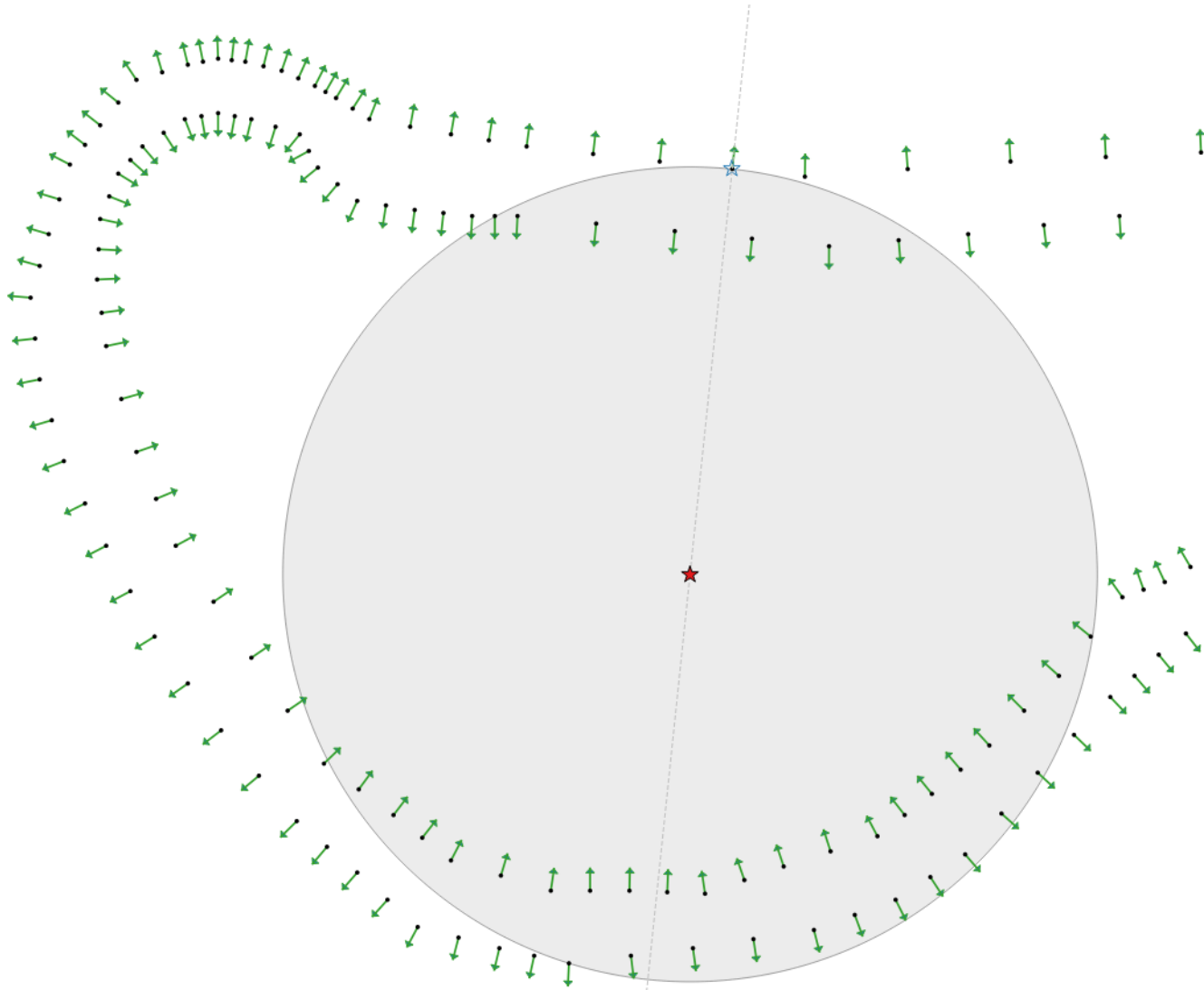
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



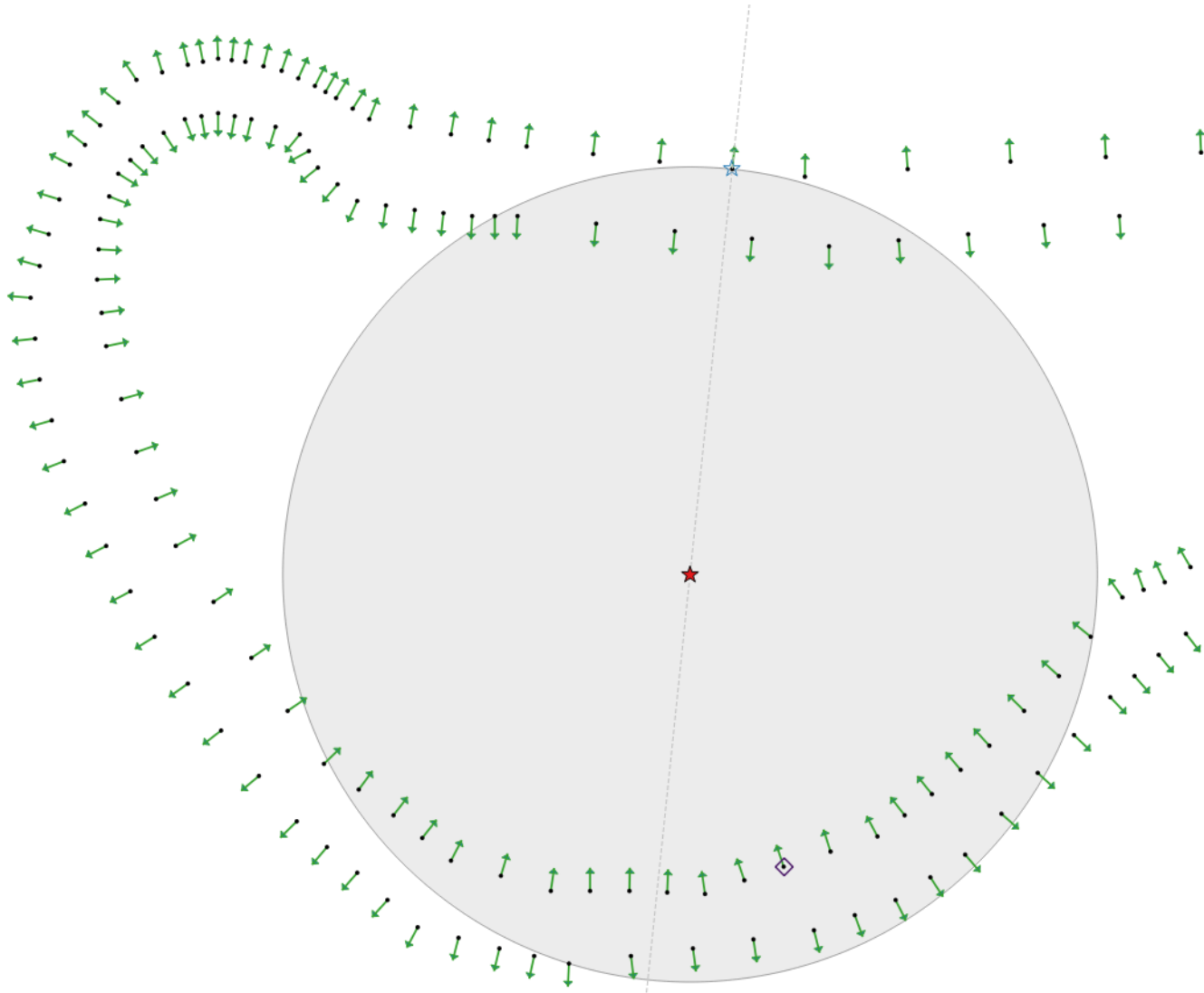
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



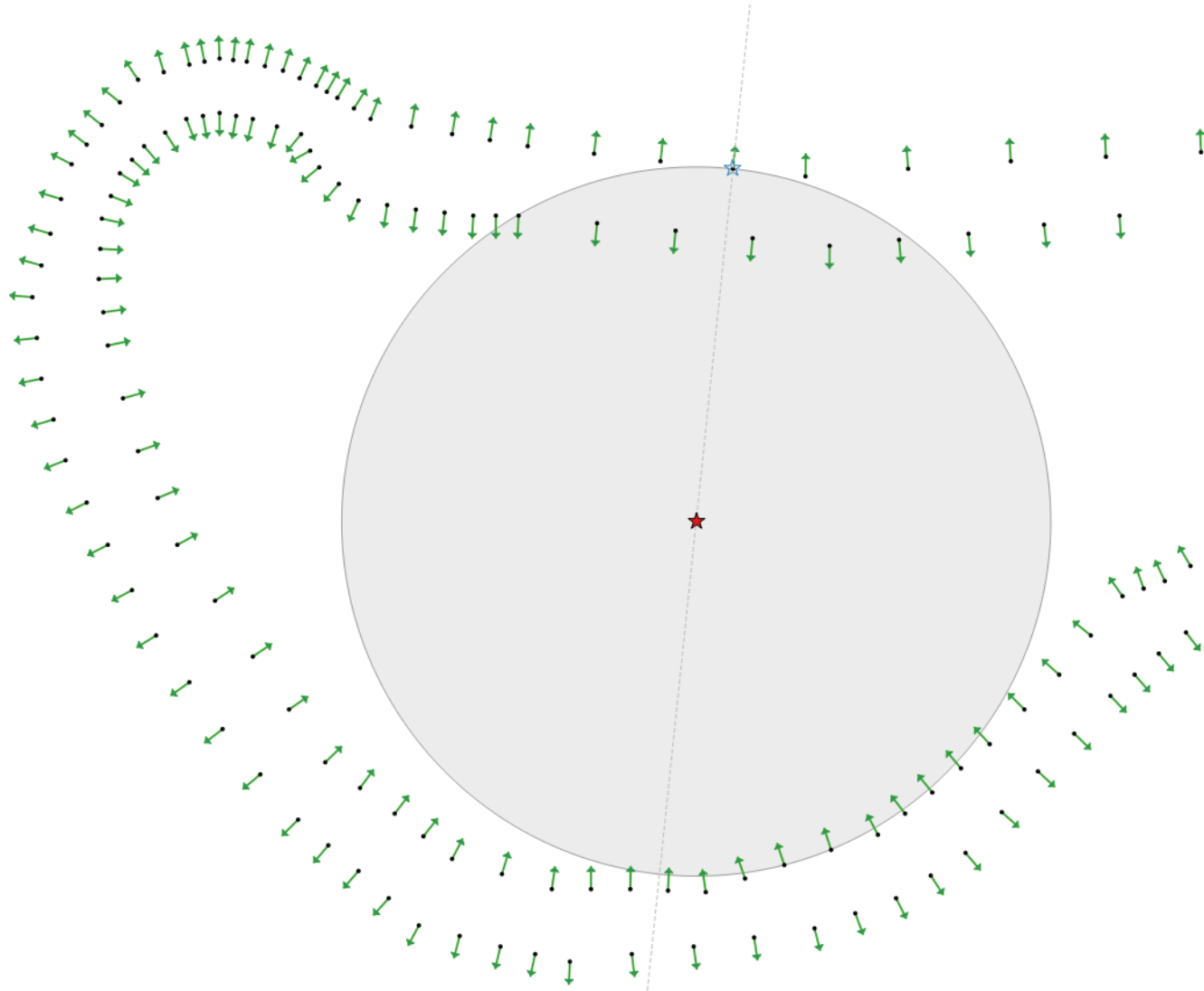
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



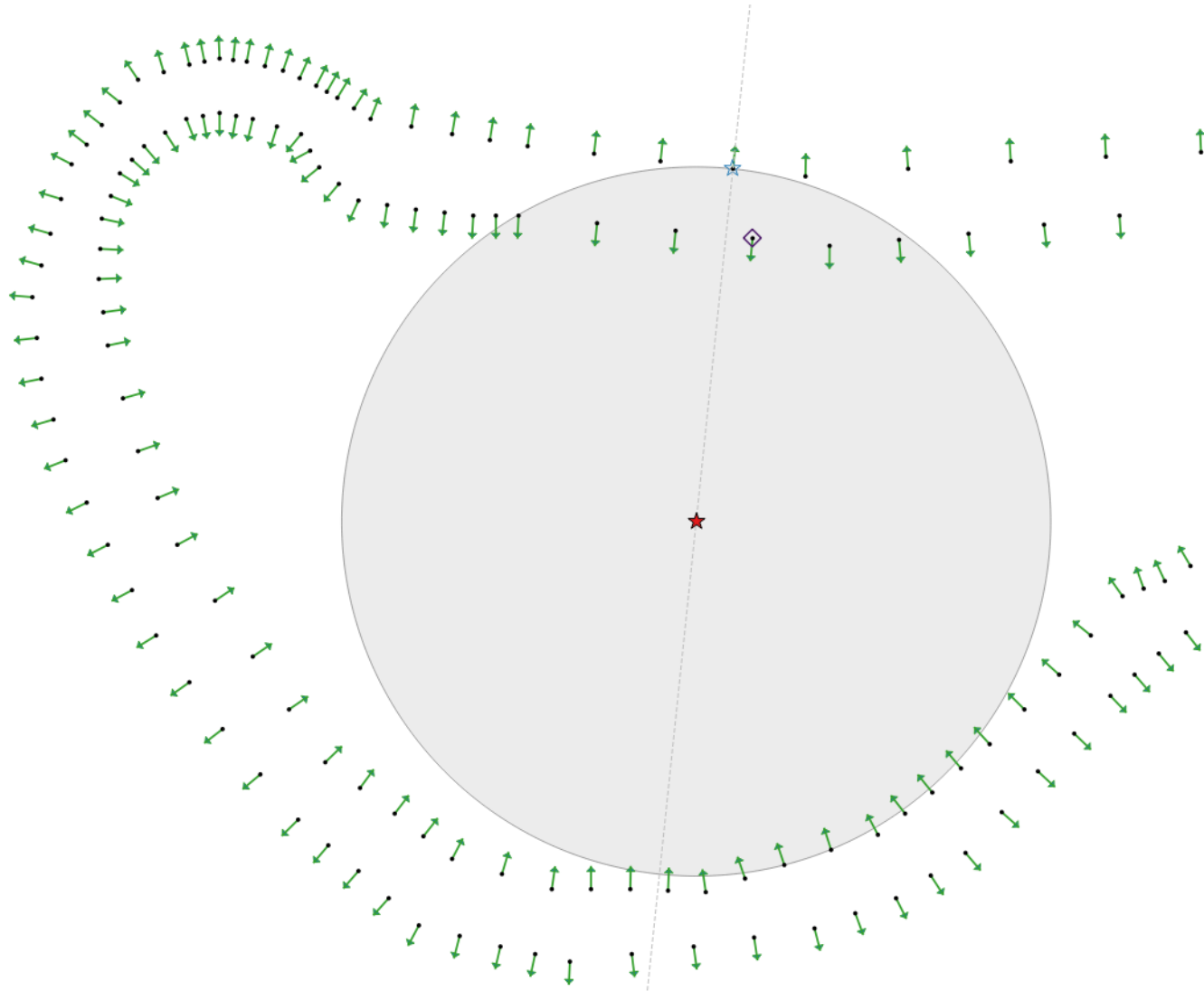
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



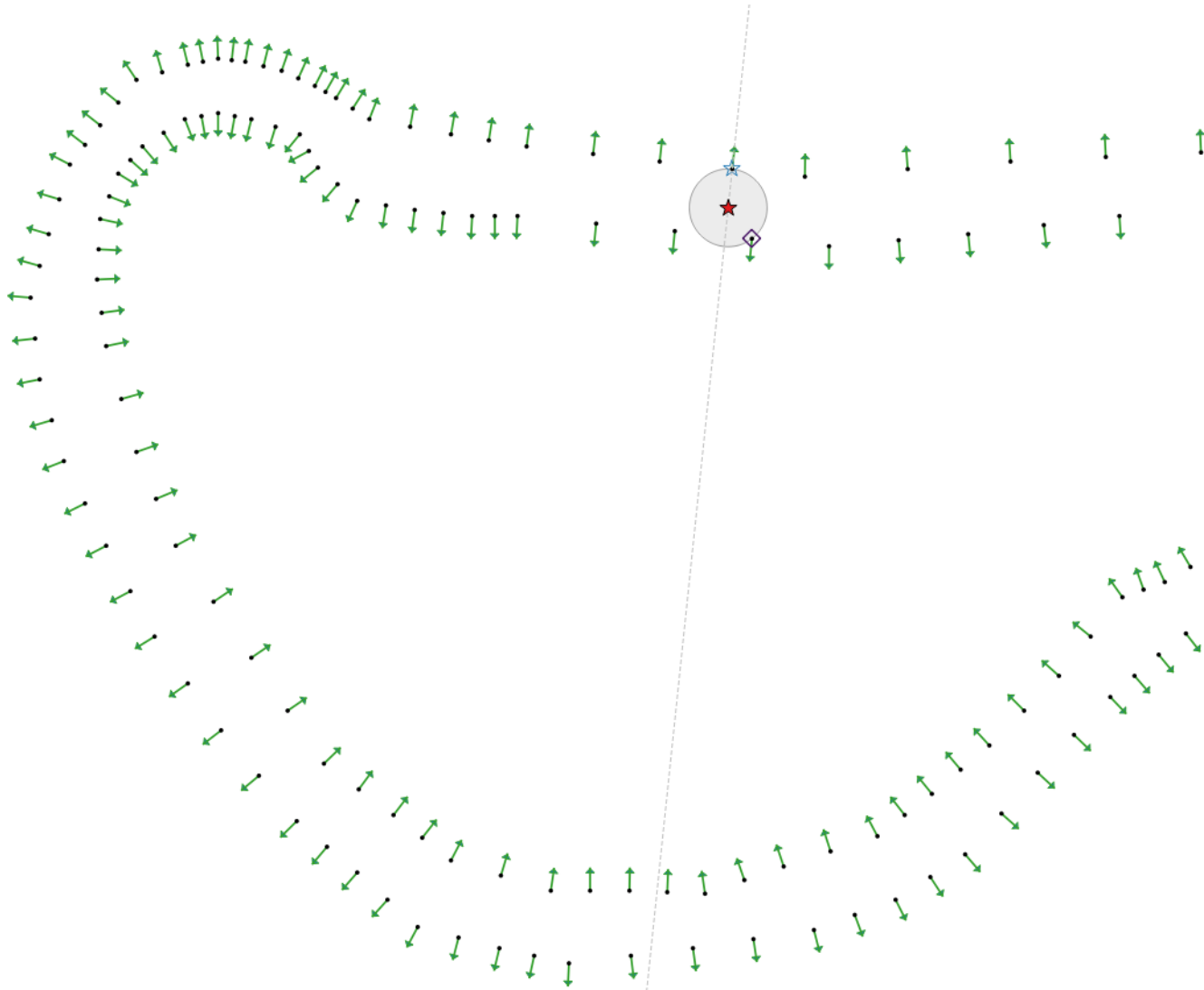
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



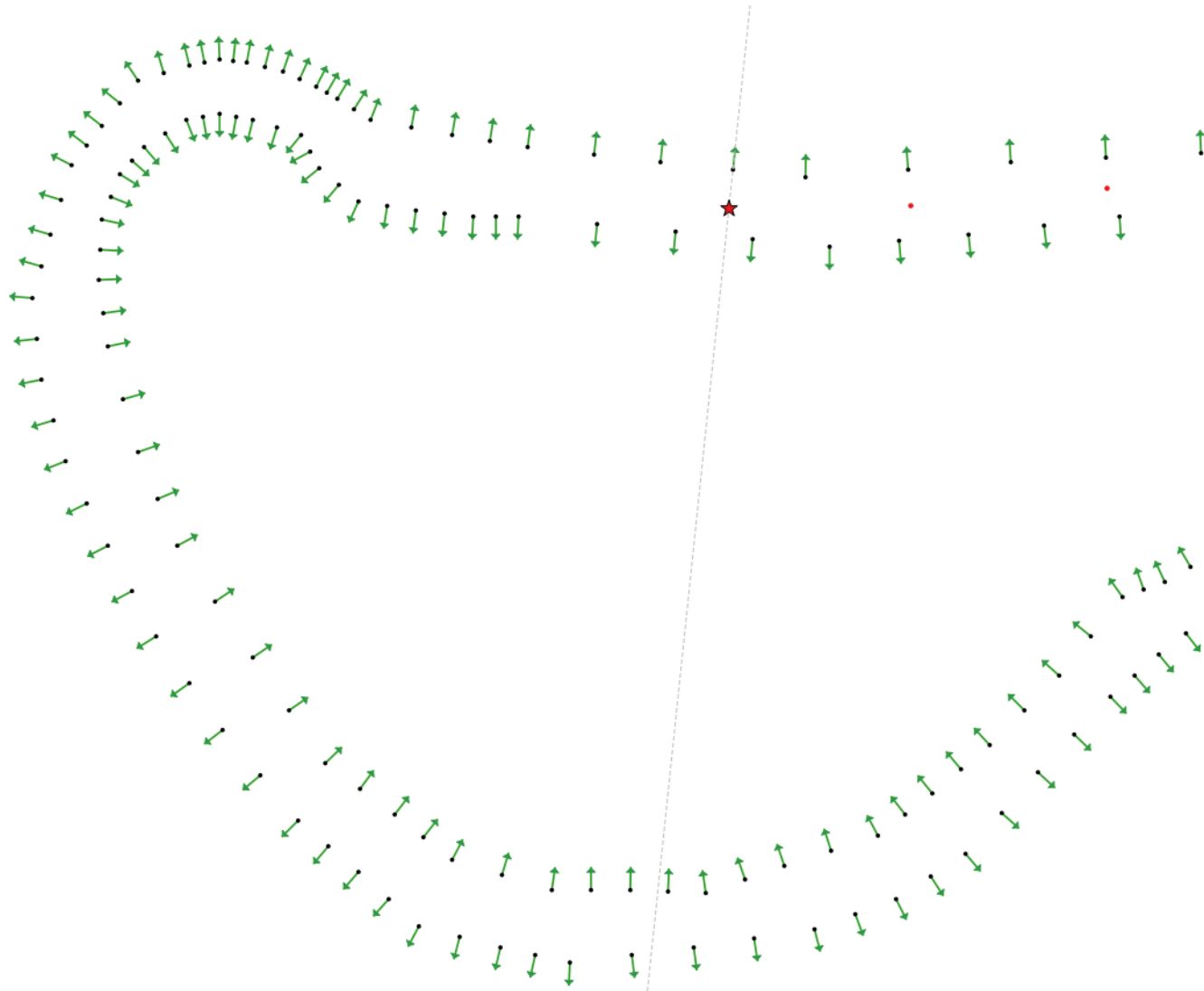
Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



Case

Algorithm (2D version of algo by Ma et al., 2012; Peters and Ledoux, 2016)
intermediate results we are interested in:



Auto-refreshing the QGIS map pane

1. Instruct QGIS to watch for changes on a file (using Python console and `QFileSystemWatcher`)
2. Run program in terminal / Interactive Development Environment (IDE)
3. Program writes intermediate results (as TSV)
4. Program writes random value to ‘watch file’
5. Pause program (using `raw_input()` in Python or break points in IDE)
6. QGIS auto-refreshes the map pane!

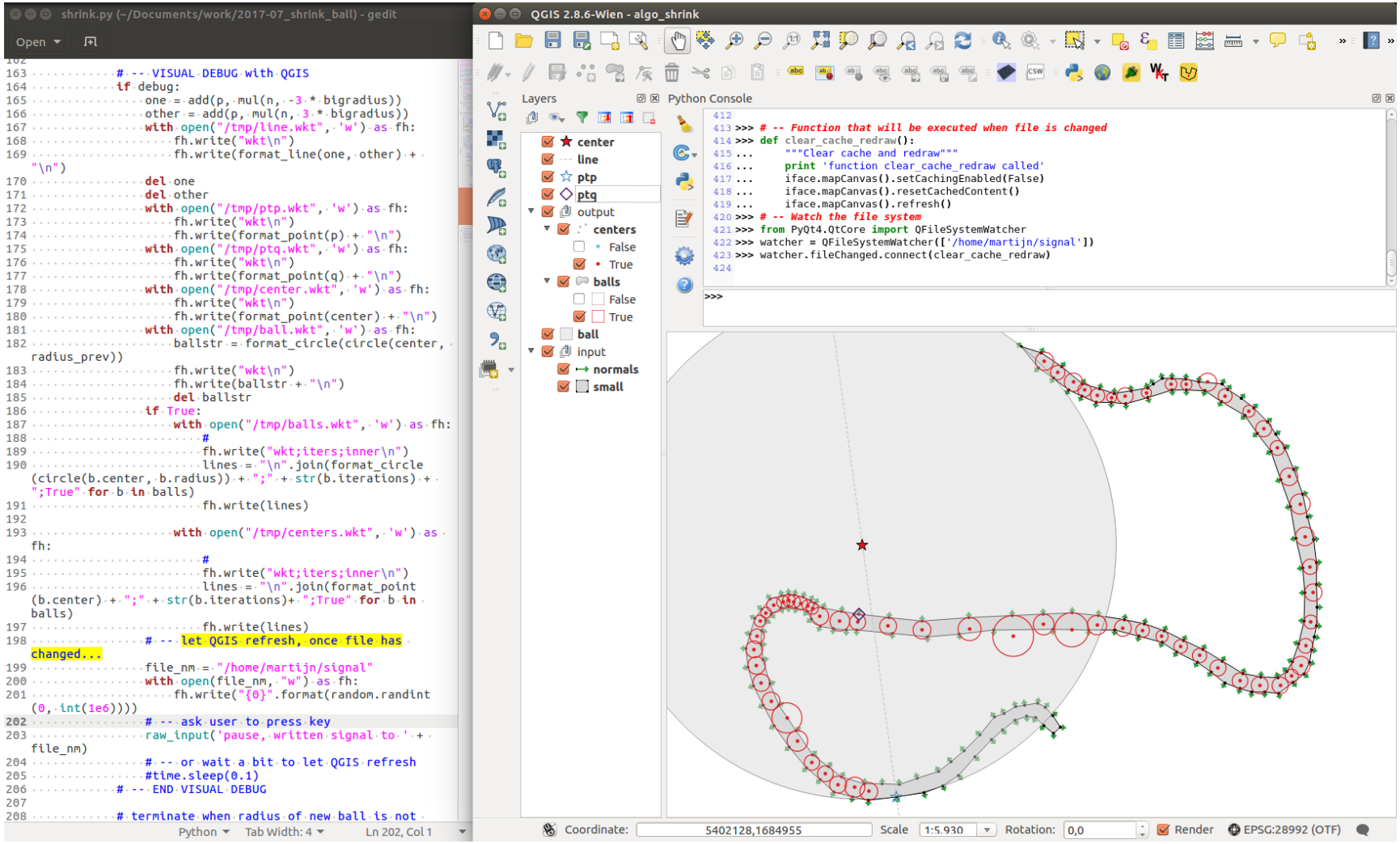
- No need to leave terminal / IDE!

Watching for changes on a file using PyQT

Inside QGIS: Plugins ▷ Python console

```
1  # -- Function that will be executed when file is changed
2  def clear_cache_redraw():
3      """Clear cache and redraw"""
4      print 'function clear_cache_redraw called'
5      iface.mapCanvas().setCachingEnabled(False)
6      iface.mapCanvas().resetCachedContent()
7      iface.mapCanvas().refresh()
8
9  # -- Watch the file system
10 from PyQt4.QtCore import QFileSystemWatcher
11 watcher = QFileSystemWatcher(['/home/martijn/signal'])
12 watcher.fileChanged.connect(clear_cache_redraw)
```

Demo



Python code and QGIS project available on Bitbucket:

https://bitbucket.org/bmmeijers/shrink/src

Auto-refreshing the QGIS map pane

- Also useful for producing a small video:
  1. . . .
  2. QGIS auto-refreshes the map pane
  3. QGIS saves the map pane contents to disk as image
- Use another tool (`ffmpeg`) to process all images to a movie (e.g. for sharing on vimeo, youtube, . . .)

Watching and saving the map pane as image

Plugins ▷ Python console

```
# -- Function that will be executed when file is changed
def clear_cache_redraw():
    """Clear cache and redraw"""
    print 'function clear_cache_redraw called'
    iface.mapCanvas().setCachingEnabled(False)
    iface.mapCanvas().resetCachedContent()
    iface.mapCanvas().refresh()
    global ct  # we update the counter outside the function
    ct += 1
    iface.mapCanvas().saveAsImage(
        '/tmp/shrink{0:05d}.png'.format(ct)
    )

# -- Watch the file system
from PyQt4.QtCore import QFileSystemWatcher
watcher = QFileSystemWatcher(['/home/martijn/signal'])
watcher.fileChanged.connect(clear_cache_redraw)

# -- Initialize counter
ct = 0
```

Producing a small movie

With ffmpeg you can convert the `.png` files to a `.mp4`:

```
ffmpeg -framerate 8 -i shrink%05d.png -c:v libx264 -r 32 out_8_32
    .mp4
```

I hope I ‘shared something useful’

Questions?

- dr.ir. Martijn Meijers
  @bmmeijers
  b.m.meijers@tudelft.nl
  http://www.gdmc.nl/martijn/
  tel. (+31) 15 27 856 42

- Delft University of Technology
  Faculty of Architecture and the Built Environment
  OTB – Research for the built environment
  GIS Technology

Acknowledgements

- Thanks to Ravi Peters, for explaining me the 3D algorithm in detail.
- A 3D version of the algorithm is available from him: `https://github.com/tudelft3d/masbcpp`

References

Ma, J., Bae, S. W., and Choi, S. (2012). 3D medial axis point approximation using nearest neighbors and the normal field. *The Visual Computer*, 28(1):7–19.

Peters, R. and Ledoux, H. (2016). Robust approximation of the Medial Axis Transform of LiDAR point clouds as a tool for visualisation. *Computers & Geosciences*, 90(A):123–133.