# energy2d V2.9 issues

Mark Henning

November 21, 2016

## 1 Introduction

Version 2.9 of energy2d (and presumably some former versions) contains a few bugs which might have a significant impact on simulation results. Most of them have been resolved in version 2.10.

However, as some scientific work has been done using energy2d in the past, I will give some more background information about the issues to help the users to decide whether simulations have to be redone.

## 2 Object mapping bug

Energy2d uses a matrix of $100 \times 100$ finite elements, the geometry of all objects is mapped on. Hence, object dimensions often are 10 elements long or smaller.

To check, whether an object (represented by the java type *Shape*) covers an element, the shape is checked against the position $(x, y)$ representing the element lower left edge. To avoid round-off error in detecting if a point falls within a shape, code similar to the following was used:

```
boolean contains(Shape shape, float x, float y) {
   return shape.contains(x, y)
      || shape.contains(x - 0.01, y)
      || shape.contains(x + 0.01, y)
      || shape.contains(x, y - 0.01)
      || shape.contains(x, y + 0.01);
}
```

Let's take a small object, a $1 \times 1$ rectangle with its lower left corner at $(0, 0)$:

| Shape | $x0$ | $y0$ | $x1$ | $y1$ | width | height |
|---|---|---|---|---|---|---|
| Rectangle | 0 | 0 | 1 | 1 | 1 | 1 |

For the grid points surrounding the rectangle, the following table shows the return values of *contain* and its *shape.contains* calls:

| Grid point | | shape.contains(...) (printed bold, if return value is *true*) | | | | | | | | Function return |
|---|---|---|---|---|---|---|---|---|---|---|
| x, y | | x, y | | x - 0.01, y | | x + 0.01, y | | x, y - 0.01 | | x, y + 0.01 | | value |
| 0 | 0 | **0** | **0** | -0.01 | 0 | **0.01** | **0** | 0 | -0.01 | **0** | **0.01** | 1 |
| 0 | 1 | 0 | 1 | -0.01 | 1 | 0.01 | 1 | **0** | **0.99** | 0 | 1.01 | 1 |
| 1 | 0 | 1 | 0 | **0.99** | **0** | 1.01 | 0 | 1 | -0.01 | 1 | 0.01 | 1 |
| 1 | 1 | 1 | 1 | 0.99 | 1 | 1.01 | 1 | 1 | 0.99 | 1 | 1.01 | 0 |
| Number of grid points contained by rectangle(0, 0, 1, 1): | | | | | | | | | | | | 3 |

It is obvious, that the element is mapped to three finite elements instead of only one making it three times as big as specified. Similar to that, an $10 \times 10$ rectangle will be mapped to 120 finite elements resulting in a 20% surplus size.

This has significant impact on power sources, as their effective heat input is proportional to the number of finite elements covered by them. Hence, the error introduced may be 20%–50%.

Another consequence is, that objects intended to lie close to each other suddenly overlap. Depending on the z-order of the objects (= the order in which they where specified), one of the objects grows at the cost of its neighbour by one element length. The error introduced may be 10%–25%.

As this increase in object size is always one element in each direction, this introduces size effects in simulations, which means, that the finite element length (= the scale) affects the simulation results. The error introduced is in the order of the errors detailed before.

# 3 Asymmetry bug

The heat conductivity solver of energy2d is intended to be an implicit one. However, its implementation is asymmetric. While it acts as an implicit solver in the positive coordinate direction, it acts as an explicit one in the negative direction.

Consider a heat source in the middle of the field. In time step 0, the temperature distribution along the $x$-axis is as shown in figure 1.

The heat distribution of the next time step is given in figure 2.

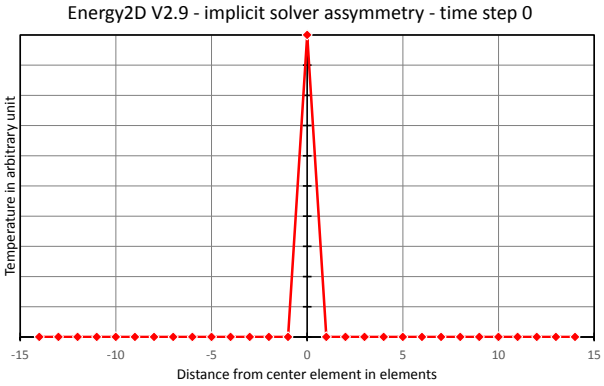


Figure 1:

The asymmetry is obvious. In this example case, the point of maximum temperature is shifted to the left.

Provided the time step fits the requirements for explicit solving, this asymmetry levels out for quasi-static simulations with time. However, it may have an impact on

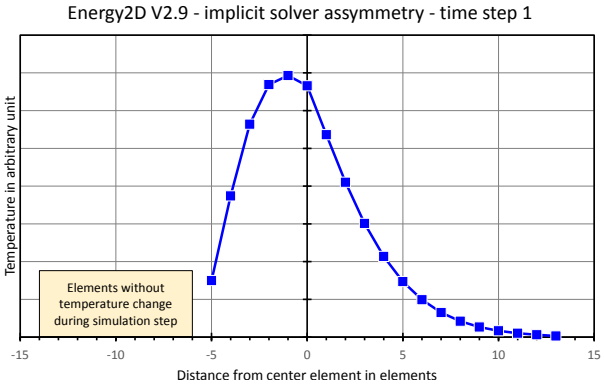


Figure 2:

- dynamic simulations
- simulations, where e.g. temperature as a function of time trends is analyzed: thermometers at one end of the simulation field may exhibit a time lag in the order of 20 time steps compared with thermometers of the opposite end.
- convectivity simulations: an angular momentum may be introduced in the fluid

# 4 Floating point inaccuracy

The data type used in energy2d is *float*, which represents a 32 bit floating point value.

While this can be considered appropriate for large scale simulations such as in the default 10m×10m field, evidence has been found that the accuracy may become insufficient in small-scale simulations.

In the present case, the field size was 15cm × 3cm containing a metal plate being heated at one end. Due to the small element size, the stability criteria required a maximum time step of ≈ 0.001s.

After about 18 minutes (simulated time as shown in energy2d), the temperature profile reached a steady state with a maximum temperature of ≈ 180°C which differed significantly from measurements performed on a real plate where ≈ 250°C where reached.

Heat input by power sources and heat flow between neighboring elements slow down and finally stop, when they become too small compared with the heat currently stored in the elements. Heat develops too lazy, finally a false temperature equilibrium develops.

# 5 Open issues

Note that energy2d V2.10 uses the enhanced floating point arithmetics for heat conductivity simulations only. Other mechanisms such as convectivity might suffer from similar problems as described in section 4.