**MODELING OBJECTS**

We call a hexaflake of the n-th iteration a figure obtained from a hexagon by replacing each of its faces by a Koch curve of the n-th iteration. Each iteration in constructing the Koch curve consists in taking each segment of its constituent polyline, dividing it into three equal parts and replacing the middle interval of a broken line of two segments with a length equal to the length of the seized segment. And the vertex of this broken line will be placed so that it is closer to the center of the figure.

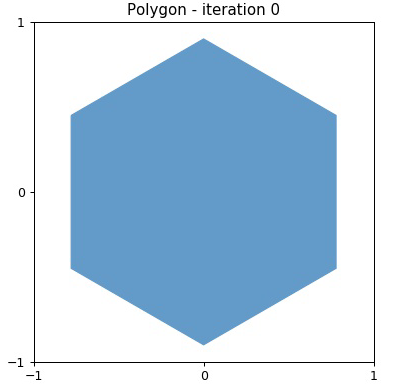


Fig. 1 Original hexagone

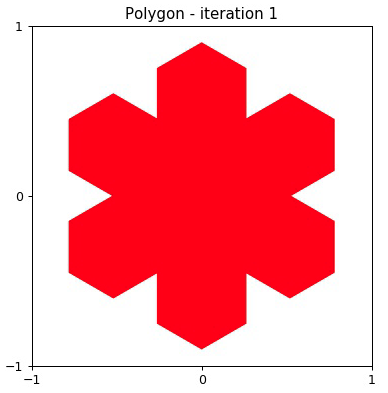


Fig. 2 Hexaflake of 1st iteration

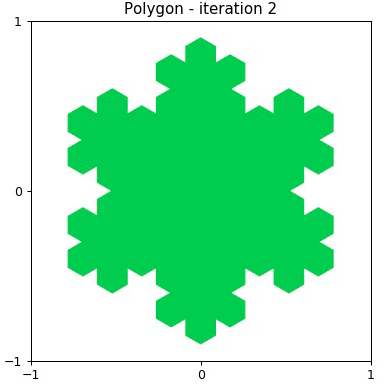


Fig. 3 Hexaflake of 2st iteration

**A**

**B**

Fig. 4 Modeling cell

Figure 4 shows the hexagonal cell in the hexaflex of the second iteration inside (for example). The area of the simulation is close to each other by the flattening of the plane by such cells. The length of the vector AB is taken as the unit of length in the problem.

**GENERAL MODELING PROCEDURE**

For hexaflakes, we perform Monte Carlo simulations using the HOOMD-blue package [1-3]. Simulation takes place under cyclic boundary conditions and the constant area of the system. The simulation protocol is similar to the first protocol from [4]. First, the system is initialized to a large square box with random initial conditions with N = 32 on 32 particles with the target density. Then, a simulation with a number of steps of the order of ~ 10 ^ 8 is performed. Then the final balanced frame from this simulation is taken, replicated in 2 times, and then the simulation continues. After the simulation is over, the replication step is performed again and so on. The final size of the system was chosen 256 by 256 particles. At the finite size of the system, we had to greatly reduce the number of steps compared with the simulation in [5]. For the hexafleks of the first iteration, it was 2 \* 10 ^ 7 steps and 10 ^ 6 steps for the hexafleks of the second iteration. The reason for this is the fact that hexafleks are not convex polygons and for them the check algorithm for the intersection of two polygons is dramatically slower than for convex polygons in [5]. Therefore, despite the use of modern GPU NVidia GeForce 1080 in the simulation, the average number of frames processed per second dropped tens of times compared to convex polygons. However, in [4] results are given, which, it seems to us, convincingly show that the statistics we have accumulated are sufficient. The calculated characteristics were subsequently averaged over the last thousand time steps of the simulation.

We constructed the correlation functions и [7,8] of both the orientation parameter and the translation parameter, which characterized the orientational and translational order [6].

Standard definitions for quantities и [7,8]

where is the position vector of the particle *k*, and ***G*** is the reciprocal lattice vector from the first Brillouin zone of the crystal lattice, is the angle between the vector (0,1) and the vector ,, and the sum over *j* is the sum over the neighbors of the particle *k*.

Correlations functions и definitions are

Here is the pair correlation function. The behavior determines the transition to the solid phase. For this, in the limit there must be c . In turn, similarly, the behavior determines the transition to the hexatic phase at .

In addition, we plotted the histograms of the local particle density for different values of the average density over which the transitions from fluid into hexatic and then into the solid phase are also traced qualitatively (in the form of a histogram).

The figures below show the behavior и for the densities corresponding to the fluid-phase, the transition point in the hexatic-phase (), and the transition point to the solid-phase ().