Convex Hulls on Cellular Spaces: Spatial Computing on Cellular Automata

Luidnel Maignan* and Frédéric Gruau*†‡
* INRIA Futurs Saclay - 4, rue Jacques Monod, 91893 ORSAY Cedex, France
† LRI - Université Paris-Sud 11, bâtiment 490, 91405 Orsay Cedex, France
‡ LIRMM - 31 rue Ada, 34000 Montpellier, France

***Abstract*—In the cellular automata domain, the discrete convex hull computation rules proposed until now only deal with a connected set of seeds in infinite space, or with distant set of seeds in finite space. Taking a spatial computing point of view, we present a cellular automata rule that constructs the discrete convex hull of arbitrary set of seeds in infinite spaces. This is done by characterizing the cellular spaces and the convex hulls by metrical properties. The rule obtained is expressed using intrinsic and general properties of the cellular spaces, considering them as metric spaces. In particular, this rule is a direct application of metric Gabriel graphs. This allows the rule and its components to be used on all common 2D and 3D grids used in cellular automata.**

I. Introduction

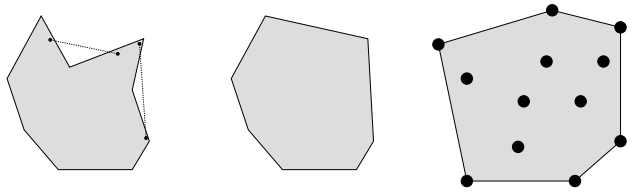
Convex hulls are an important tool of classical, i.e Euclidean, geometry. In this geometry, the convex hull of a set of points is defined as the smallest convex polytope containing these points. Considering the two-dimensional case, Fig. 1 shows a non-convex polygon, a convex polygon, and the convex hull of a given set of points. A common characterisation of convex polygon is that all their internal angles are less than or equal to 180 degrees.

There are many algorithms that compute the Euclidean convex hull of a set of points in sequential and parallel settings. This include Jarvis's gift wrapping [1], Graham's scan [2], Kirkpatrick–Seidel algorithm [3], and Chan's algorithm [4]. These algorithms take a set of point coordinates as input and return the subset of these points that forms the smallest convex polygon.

We are interested in the computation of convex hulls in the framework of cellular automata [5]. In this massively distributed computation framework, the set of processing elements itself forms the space. The considered set of points is therefore indicated by a marker on the corresponding processing elements. Computing their convex hull is to place a special marker on all the processing elements corresponding to the points contained in the convex hull, and only on them.

A. The classical cellular automata approach

Cellular automata consider sets of processing elements forming lattices in Euclidean spaces. Previous works about convex hulls computation on cellular automata mainly study two-dimensional square lattice with the so-called Moore neighborhood consisting of the direct horizontal, vertical, and diagonal neighbors. Because of the discrete nature of the space thus considered, the definition of the convex hull is adapted to only take into account polygons that only consist of horizontal, vertical and diagonal edges . This amounts to consider that the internal angles have to be multiples of 45 degree, as defined by the 45-convexity.



(a) Non-convex set (b) Convex set (c) Convex hull

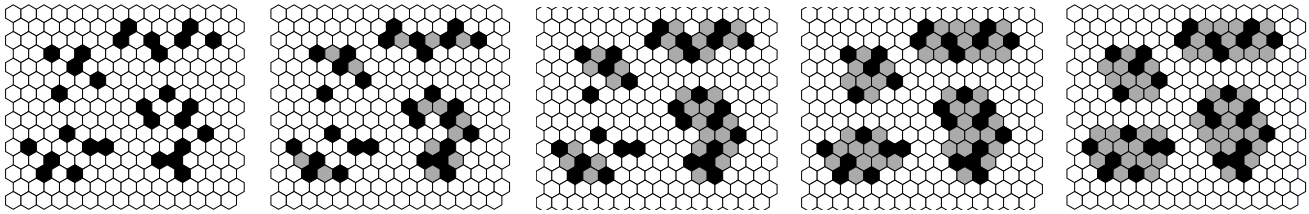
Fig. 1. Euclidean convexity examples. Considered set for convexity in gray, seeds in black. Note that (b) is also the convex hull of (a)

Previous cellular automata that have been proposed to compute this adapted convex hull have important limitation. The first is that most of them are restricted to this specific lattice. Another limitation is that they only consider set of points that are connected, either directly or indirectly by a wrapping pattern. In fact, they are designed by manual characterization of what needs to be done to complete the convex hull construction. We present a way to look at the same problem in a more generic way, without any constraints on the set of points, and without being specific to a particular lattice, or to a particular dimensionality: all common lattices, either 2D or 3D are considered at once by a changing of the point of view on cellular spaces.

B. A spatial computing point of view

This genericity is obtained by taking a spatial computing point of view on cellular automata. Indeed, spatial computing considers a class of massively distributed computing model where the communication time between two processing elements is proportional to their Euclidean distance. This property is called the *locality property*, and taking the spatial computing point of view is to place this property at the center of the framework.

Applying this methodology to the particular case of cellular automata, we propose to consider their intrinsic metric spaces and define the problem and the solution in terms of distances. In particular, we show that the Euclidean and the adapted convex hull are two instances of metric convex hulls applied



(a) t=01 (b) t=02 (c) t=03 (d) t=05 (e) t=08

Fig. 2. The majority rule on the set of seeds (a), generates a set of convex hulls (c)

on different metric spaces. Starting from the metric convex hull definition, which only manipulates distances, we derive a cellular automata rule also expressed in terms of distances. This particularity allows it to be applied on many cellular spaces, including square cellular spaces with 4 or 8 neighbors (Von Neumann or Moore neighborhoods), hexagonal cellular spaces and their analog three-dimensional cellular spaces.

C. Organization of the article

The rest of the article is roughly organized as this introduction. In Sect.II, we present the state of the art before our results. This is done by giving the classical point of view on cellular automata in Sect. II-A, by defining the usual discrete convex hull in Sect. II-B, and by presenting the two pre-existing convex hull cellular automata. In Sect. III, we present our results, firstly by presenting the metrical point of view on cellular spaces in Sect. III-A, then by defining metric convex hulls and how they entirely contain the usual discrete convex hulls. The remaining sections present the corresponding cellular automata rules by incrementally giving the intuition that leads to them. At the end, we notice that this work reveal at the same time an algorithm that build an interesting subgraph of Delaunay graphs: namely metric Gabriel graphs.

II. State of the art

A. Cellular Automata Framework

A cellular automaton is made of an infinite set S of *sites*, i.e. processing elements arranged as a lattice in an Euclidean space. S is called the *cellular space* and is endowed with a neighborhood relation denoted as N. Being processing elements, each site x has a particular state $v_t(x) \in V$ (as value) that changes with time, V being finite. The information accesible by a site x is its current state $v_t(x)$ and the current states $v_t(y)$ of its neighbors $y \in N(x)$. It is also common to consider that a site knows the relative position of all of its neighbors, identifying them as North, South, East or West for example. With this information, all sites x update their state to a new one $v_{t+1}(x)$, all at the same time and all using the same update function.

In this framework, the convex hull problem is formulated this way: Given a set of seeds S_0, represented by a configuration src defined as $src(x) \equiv x \in S_0$, the goal is to find a rule R such that, from some instant t, $R_t(x) \equiv x \in H_{\mathcal{C}}(S_0)$, where $H_{\mathcal{C}}(S_0)$ denotes the convex hull of the set of seeds S_0.

B. Discrete Convex Hulls

As said in the introduction, previous works about convex hulls computation on cellular automata mainly study the square cellular space with Moore neighborhood consisting of the direct horizontal, vertical, and diagonal neighbors. The definition of convex hull used in these works is called 45-convex hull: the smallest convex polygon consisting of horizontal, vertical, and diagonal edges. We recall here that convex polygons have all their internal angles less than or equal to 180 degrees.

Allthough previous works have considered mainly 45-convex hulls, the definition itself is very easy to extend, and one can consider 90-convex hulls for the two-dimensional square lattice with Von Neumann neighborhood (only vertical and horizontal neighbors), or 60-convex hulls for hexagonal cellular spaces which assign 6 neighbors to every sites.

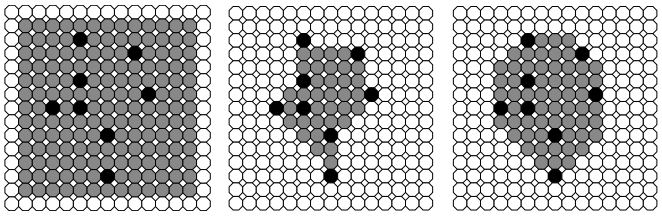
C. Rule for Connected Set of Seeds

In [6], some of the first proposed rules for the convex hull problem are described. The intuition is that any bordering site that does not have a local configuration corresponding to the shape of a convex set boundary has to select itself. After observing that all rejected local configuration have 1, 2 or 3 selected sites, the rule is simplified to a counter that checks whether there are at least 4 selected sites in the neighborhood. This rule can be generalized to the majority rule, since 4 can be interpreted as the half of 8, the number of neighbors. The convex hull behavior of the majority rule is described in [5]. In fact, the set of seeds does not need to be connected, but simply denser enough, since only their quantity matters. Also, as more and more sites are selected by the rule, many local convex hulls can merge to form bigger convex hulls that can also merge, etc. Using our notations, the majority rule can be expressed as follows, and gives the results shown in Fig. 2 for hexagonal grids.

$$majo_{t+1}(x) = \begin{cases} \top \text{ if } src(x) \\ \top \text{ if } card\{\, y \in N(x) \mid majo_t(y) \,\} \geq \frac{card(N(x))}{2} \\ \bot \text{ otherwise.} \end{cases}$$

D. Rule for Wrapped Set of Seeds

In [7], [8], a rule is proposed for non connected set of seeds. However, it requires the seeds to be included in a finite connected pattern. One can also consider that this rule only



(a) input (b) erosion (c) convex hull

Fig. 3. Stages from wrapped seeds to their convex hull: The initial wrapping (a) is shrunk to (b) and is then grown to convex hull (c)

works for finite spaces, considering the space itself as the finite wrapper. The proposed rule consists of two globally successive stages. The first one erodes the wrapper until a minimal isometric set is obtained. Thus between any pair of points of the set, at least one shortest path is in the set. The second stage is the application of a rule to transform this connected set of sites into its convex hull, as done in the previous subsection. The stages are shown in Fig. 3.

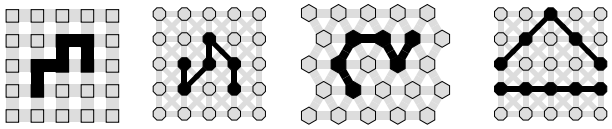
III. Metric space approach

Our goal is to solve the problem with no contrainst on the input, and without depending on one particular cellular space. This is achieved by considering the cellular space as a simple metric space, and by applying the definition of metric convex hulls. In particular, this allows to work formally on a class on cellular spaces, instead of one single cellular spaces. We illustrate this genericity through common two-dimensional cellular spaces, but three or higher-dimensional cellular spaces are also captured by this work.

A. Cellular Spaces and Metric Spaces

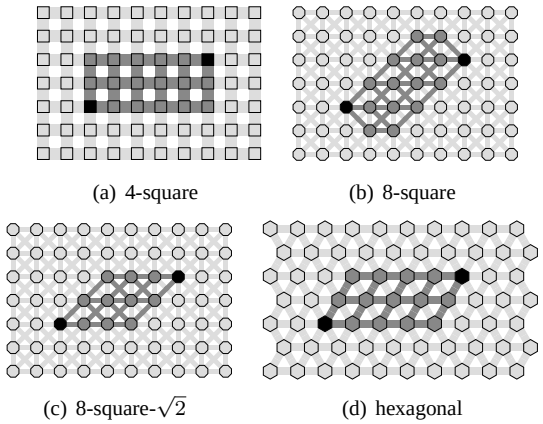
As said in Sect. II-A, the cellular spaces are endowed with a neighborhood relation. It thus forms a graph, usualy called communication graph. The length of the paths in this graph give rise to a distance function, called metric, on the cellular space which is not the Euclidean distance function. Different metrics can be associated to each cellular space. The most commonly used two-dimensional cellular spaces are the square grids, with 4 or 8 neighbors per site, and the hexagonal grids, having 6 neighbors per site. Figure 4 shows these grids, exposing the nodes and the edges of their communication graphs.

For the distance function, we consider the *hop count metric*, which associates to each edge the unitary length. This is a natural choice for the 4-square and hexagonal grids since all edges have the same length in the Euclidean space. However, this is not the case for the 8-square grids, whose diagonal edges are drawn $\sqrt{2}$ times longer than the vertical and horizontal ones. Therefore, we also consider two metrics for the 8-square grids: the hop count metric, and the $\{1, \sqrt{2}\}$ *metric* that associates unitary and $\sqrt{2}$ lengths to non-diagonal and diagonal edges respectively. With the hop count metric, paths represented on Figs. 4(a), 4(b), and 4(c) have length 5. With the $\{1, \sqrt{2}\}$ metric, the path of Fig. 4(b) has length $3 + 2\sqrt{2}$. It is important to note that paths of Fig. 4(d) have the same



(a) 4-square (b) 8-square (c) hexagonal (d) 8-square

Fig. 4. Grids used in this article: the polygons (squares, octagons and hexagons) correspond to the sites, the lines correspond to the edges, and example paths are in black.



(a) 4-square (b) 8-square

(c) 8-square-$\sqrt{2}$ (d) hexagonal

Fig. 5. Intervals for different grids

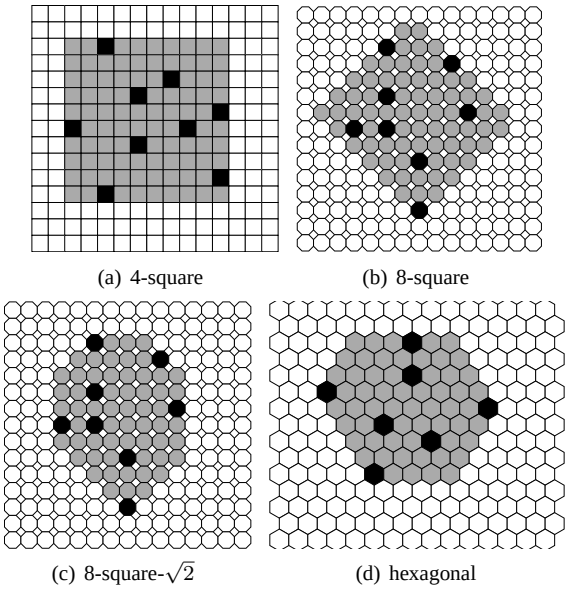
length for hop count and different lengths for $\{1, \sqrt{2}\}$, since this has an effect on the notions of shortest path and convexity.

B. Metric Convex Hulls

As said in Sect. II-B, the convex hull definition needs to be adapted to fit to cellular spaces. Instead of the 45, 90 and 60-convexity definition, we consider the metric convexity one: a subset of the space is metric-convex if it contains all the shortest paths joining two of its points.

The Eculidean, 45, 90 and 60-convexity correspond to the metric convexity applied on different metric space, as we will see. In arbitrary metric space, a point z lying on a shortest path joining two points x and y is said to be *between* the two points. It is denoted as $z \in [x, y]$, where $[x, y]$ is called the *interval* between x and y. Formally, we have $[x, y] = \{\, z \in S \mid d(x, z) + d(z, y) = d(x, y) \,\}$ where d is the metric and S the set of points. In Euclidean spaces, $[x, y]$ corresponds to the segment joining the points x and y, since it is the unique shortest path joining these points. Fig. 1(a) shows that any non-convex polygon has some missing segments. In other metric spaces, including square and hexagonal cellular spaces, there may be many shortest paths between two points, leading to more complex intervals as shown in Fig. 5. In fact, this seems an inevitable artefact when the space is discrete, that is at the heart of the problem difficulty.

Using intervals, metric convexity can be defined as follows. A subset C is metric-convex if and only if $[x, y] \subseteq C$ for any pair of points $(x, y) \in C^2$. If we define the operator $I(P) = \bigcup\{\, [x, y] \mid (x, y) \in P^2 \,\}$ that adds to a set of points the shortest path joining them, we can say that a set C is



(a) 4-square (b) 8-square

(c) 8-square-$\sqrt{2}$ (d) hexagonal

Fig. 6. Each site x is black if $x \in S_0$, or gray if $x \in H_C(S_0)$

metric-convex if and only if $I(C) = C$.

The convex hull $H_C(S_0)$ of a set $S_0 \subset S$ of seeds is the minimal convex set containing S_0. Using $I(\bullet)$, it corresponds to the limit of the sequence $I(S_0), I(I(S_0)), \ldots$, since $I(\bullet)$ adds points that have to be in the convex set, and the limit L verifies $I(L) = L$. Our algorithm also adds points iteratively.

Examples of initial and associated final configuration for the considered cellular spaces are shown in Fig. 6. It is interesting to see how 45, 90 and 60-convexity corresponds to metric convexity.

C. Rule for Local Convexity

Computing the convex hull locally means that each site has to select itself if it belongs to the convex hull of the selected sites present in its neighborhood. The computation of the local convex hull is an easy task due to the simplicity and finiteness of the space in the neighborhood of each site. Indeed, it is enough for a site to check if it lies on a shortest path joining two of its selected neighbors. This gives the following local convexity rule:

$$conv_t(x) = \begin{cases} \top \text{ if } src(x) \\ \top \text{ if } \exists \{y_0, y_1\} \subset \{\, y \in N(x) \mid conv_{t-1}(y) \,\}; \\ \qquad x \in [y_0, y_1] \\ \bot \text{ otherwise.} \end{cases}$$

Let us mention that testing $x \in [y_0, y_1]$ with hop count metric is equivalent to testing $d(y_0, y_1) = 2$ since $x \in [y_0, y_1] \Leftrightarrow d(y_0, x) + d(x, y_1) = d(y_0, y_1)$ and $y \in N(x) \Leftrightarrow d(x, y) = 1$. For the $\{1, \sqrt{2}\}$ metric, the test has to be done explicitly.

Because it follows directly from the convex hull definition, this rule is more precise than the $majo$ rule. It is possible to check that whenever one selects half of the neighbors of a site x, then two of them are joined by a shortest path containing the site x. The reverse is obviously not true, which means the $conv$ rule is able to construct the convex hull in more cases than the $majo$ rule. One can also note that applying $majo$ on an 8-square grid can only give a $\{1, \sqrt{2}\}$-convex hull. The formulation of $conv$ allows choosing the metric to use, and tackles many grid topologies at once, such as the ones considered in this article and their tridimensional counterparts. However, it exhibits roughly the same behavior. Figure 7 shows the evolution of the $conv$ rule with an initial configuration on which $majo$ is stationary.

D. Global Convexity with Only Two Seeds

Since we have seen solutions to transform a connected set of seeds into its convex hull, a natural idea to obtain the convex hull of an arbitrary set of seeds is to connect them in a minimal way that remains in the desired convex hull. We start by studying the simpler case of only two seeds. In this setting, the goal is to select sites of the interval, as shown in Fig. 5.

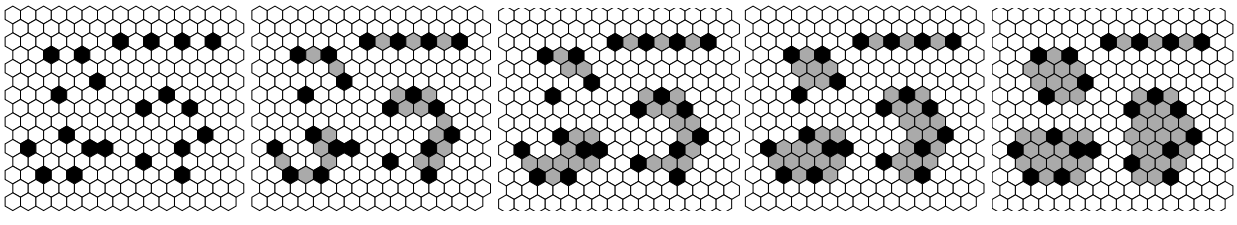
To do so, we compute the distance of each site to the nearest seed, and look at the resulting pattern (Fig. 8(a)). We can notice distinguished sites, namely the middle sites which are exactly the middle of the shortest path joining the two seeds. It turns out that these middles can always be detected by looking at the distance values in a bounded neighborhood. Therefore, they will be the first sites identified as being in the convex hull (Fig. 8(b)). All the other sites of the interval can then be selected by back-propagating from the middles to the seeds, by traveling towards neighbors that are closer to the seeds, again by using the distance values (Fig. 8(c)). This achieves the desired construction, without using any global phase transition but only local interaction. Let us now describe each rule in more details.

1) Distance Field: The distance computation described earlier is what we call a *distance field*. For hop count metrics, it associates to each site an integer and can be expressed by the following rule. The latter converges to $dist_\infty(x) = min\{\, d(x, y) \mid src(y) \,\}$:

$$dist_t(x) = \begin{cases} 0 \text{ if } src(x) \\ 1 \text{ if } \neg src(x) \wedge t = 0 \\ \min\{\, 1 + dist_{t-1}(y) \mid y \in N(x) \,\} \text{ otherwise.} \end{cases}$$

While this rule has an infinite number of possible state, we only need its gradient, i.e. the differences between the distance of neighboring sites. In [9], we have shown how to represent the gradients of some kind of integer fields with a finite number of states, thanks to the modulo operator. Applied on the $dist$ rule, it allows representing it modulo 3 and gives the way to compute the gradient from these modulo values. For brevity and readability, we use directly $dist$ in the rest of the paper and redirect the interested reader to [9]. Finally, we do not have any mean to represent the distance field for $\{1, \sqrt{2}\}$ metric with finite number of states, which is the reason why the final rule can not be directly applied to this metric.

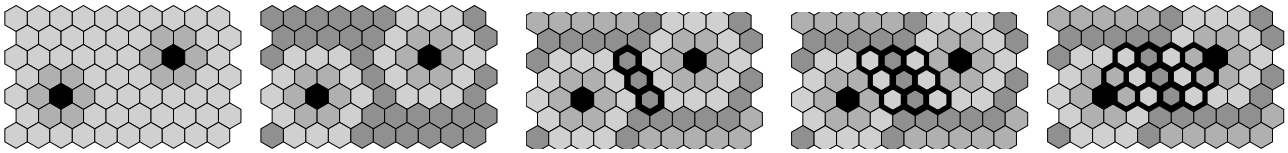
2) From Middles Back to Seeds: As mentioned earlier, the middle sites are detected using the distance values present



(a) t=01 (b) t=02 (c) t=03 (d) t=04 (e) t=06

(f) t=08 (g) t=10 (h) t=15 (i) t=20 (j) t=28

Fig. 7. Evolution of the $conv$ rule. The $majo$ rule is stationary on (a)



(a) Distance (b) Middles (c) Back propagation

Fig. 8. The rules $dist$, $cent$ and $back$ in action

in their neighborhood. The global idea is to detect distance patterns that only happen when there are two nearest seeds for the considered site, in which case, the considered site is the middle of these two nearest sites. Because we solve this detection in our general metric framework instead of a particular grid, we delay the discussion about this detection to the next subsection, and directly use $cent(x)$ to denote that a site x is a middle.

For the back-propagation, each site having a selected site in its neighborhood has to determine if it is closer or not than the selected neighbor. If it is so, it can select itself, since it is between the selected neighbor and the seeds. This is expressed as:

$$back_t(x) = \begin{cases} \top \text{ if } cent_t(x) \\ \top \text{ if } t > 0 \wedge \exists y \in N(x), \\ \qquad back_{t-1}(y) \wedge dist_{t-1}(x) < dist_{t-1}(y) \\ \bot \text{ otherwise} \end{cases}$$

E. Global Convexity and Metric Gabriel Graphs

When considering the general case with many seeds, (instead of just two seeds) some questions naturally arise: does the rule presented for only two seeds do all the work pairwise? Do they produce a connected set? What structure is constructed? The answer is that we produce a connected set, connecting the seeds pairwise to draw a structure related to Delaunay graphs. A complete description is beyond the scope of this article but can be found in [10]. We only give here the material that allows understanding the global structure of the constructed graph.

When computing a distance field, one implicitly associates to each site its nearest seed. It is strongly related to Voronoi diagram [11], the set of sites having the same nearest seed being called the Voronoi region of the seed and the sites having many nearest seeds being the boundaries between the Voronoi regions. In our case, we detect boundary sites that are on a shortest path between the corresponding seeds, to make sure to select only sites that are in the convex hull.

By doing so, we only connect seeds of neighboring Voronoi regions, such that there is a shortest path going trough the boundary between the two regions. Replacing the words "shortest path" by "segment", we obtain one of the defining properties of Gabriel graphs [12], which is a connected subgraph of the Delaunay graph defined for Euclidean spaces. In [10], we generalize the definition of Gabriel graphs to arbitrary metric spaces and obtain *metric Gabriel graphs*, which identify exactly what we need to detect in order to have a connected set of sites. We also explain in detail the rule $cent$ (as metric Gabriel ball *centers*).

By using metric Gabriel graphs, we have, roughly speaking, that $back \circ cent \circ dist$ constructs a connected set of sites that is a subset of the convex hull. In order to complete the convex hull, we simply have to consider $conv \circ back \circ cent \circ dist$. The final cellular automaton thus described has 7 states: (3 distance states) * (2 "in convex hull" states) + 1 special "seed" state. Because of $cent$ rule, it uses a neighborhood of radius 2. The

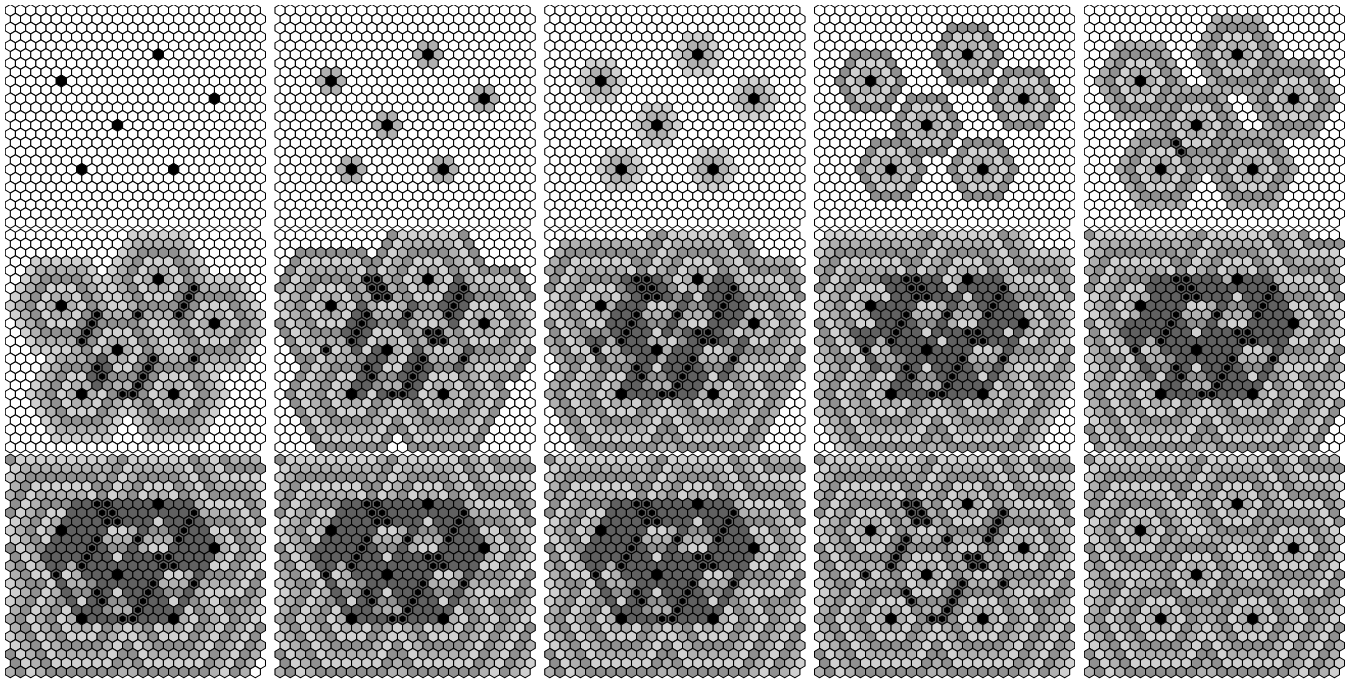


Fig. 9. Snapshots of the computation of the three layers simultaneously. The two last snapshots show the final configuration with, firstly $back$ hidden and then $back$ and $cent$ hidden. Black: generators, light grays: $dist$, small dots: $cent$, dark gray: $back$

evolution of the rule without $conv$ is shown in Fig. 9.

IV. Final remarks

We have presented a cellular automata that computes the convex hull of arbitrary set of seeds for all the common cellular spaces of any dimensionality. This has been done by taking a spatial computing point of view on cellular automata and by considering them as metric spaces. However the solution proposed only works for the hop count metric. Therefore, it may seem that the original 45-convexity has not been tackled, as it does not correspond to a hop count metric, but to a $\{1, \sqrt{2}\}$-metric. However, this is not a problem, since we can still produce the $\{1, \sqrt{2}\}$ convex hull having both diagonal and vertical-horizontal border, by intersecting two convex hulls: the 4-square one, having only vertical-horizontal borders, and the 8-square one with hop count, having only diagonals. This intersection relation can be observed on Fig. 6: Fig. 6(c) is the intersection of Fig. 6(a) and Fig. 6(b).

V. Acknowledgements

We would like to thank Prof. Adamatzky who pointed us firstly to the convex hull problem, which was the starting point of this work, and secondly to Gabriel graphs, when we showed him our particular graph and its properties.

References

[1] R. A. Jarvis, "On the identification of the convex hull of a finite set of points in the plane," *Information Processing Letters*, vol. 2, no. 1, pp. 18 – 21, 1973. [Online]. Available: http://dx.doi.org/10.1016/0020-0190(73)90020-3

[2] R. L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set," *Information Processing Letters*, vol. 1, no. 4, pp. 132 – 133, 1972. [Online]. Available: http://dx.doi.org/10.1016/0020-0190(72)90045-2

[3] D. G. Kirkpatrick and R. Seidel, "The ultimate planar convex hull algorithm?" *SIAM Journal on Computing*, vol. 15, no. 1, pp. 287–299, 1986. [Online]. Available: http://link.aip.org/link/?SMJ/15/287/1

[4] T. M. Chan, "Optimal output-sensitive convex hull algorithms in two and three dimensions," *Discrete & Computational Geometry*, vol. 16, pp. 361–368, 1996.

[5] A. Ilachinski, *Cellular Automata: A Discrete Universe*. River Edge, NJ, USA: World Scientific Publishing Co., Inc., 2001.

[6] A. Adamatzky, *Computing in nonlinear media and automata collectives*. Bristol, UK, UK: IOP Publishing Ltd., 2001.

[7] S. Torbey and S. G. Akl, "An exact and optimal local solution to the two-dimensional convex hull of arbitrary points problem," *Journal of Cellular Automata*, 2008.

[8] A. G. Clarridge and K. Salomaa, "An improved cellular automata based algorithm for the 45-convex hull problem," *Journal of Cellular Automata*, 2008.

[9] L. Maignan and F. Gruau, "Integer gradient for cellular automata: Principle and examples," *Self-Adaptive and Self-Organizing Systems Workshops, IEEE International Conference on*, pp. 321–325, 2008.

[10] ——, "Gabriel graphs in arbitrary metric space and their cellular automaton for many grids," *ACM Trans. Auton. Adapt. Syst.*, 2011, In Press.

[11] F. Aurenhammer, "Voronoi diagrams - a survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, no. 3, pp. 345–405, 1991.

[12] R. K. Gabriel and R. R. Sokal, "A new statistical approach to geographic variation analysis," *Systematic Zoology*, vol. 18, no. 3, pp. 259–278, September 1969.