Combinatorial Properties and Recognition of Unit Square Visibility Graphs*

Katrin Casel[1], Henning Fernau[2], Alexander Grigoriev[3], Markus L. Schmid[4], and Sue Whitesides[5]

1 **Fachbereich 4 – Abt. Informatikwissenschaften, Universität Trier, Germany**
casel@uni-trier.de
2 **Fachbereich 4 – Abt. Informatikwissenschaften, Universität Trier, Germany**
fernau@uni-trier.de
3 **School of Business and Economics, Maastricht University, The Netherlands**
a.grigoriev@maastrichtuniversity.nl
4 **Fachbereich 4 – Abt. Informatikwissenschaften, Universität Trier, Germany**
mschmid@uni-trier.de
5 **Department of Computer Science, University of Victoria, BC, Canada**
sue@uvic.ca

Abstract

Unit square (grid) visibility graphs (USV and USGV, resp.) are described by axis-parallel visibility between unit squares placed (on integer grid coordinates) in the plane. We investigate combinatorial properties of these graph classes and the hardness of variants of the recognition problem, i. e., the problem of representing USGV with fixed visibilities within small area and, for USV, the general recognition problem.

1998 ACM Subject Classification F.2.2 Computations on discrete structures

Keywords and phrases Visibility graphs, visibility layout, NP-completeness, exact algorithms

Digital Object Identifier 10.4230/LIPIcs.MFCS.2017.30

1 Introduction

A visibility representation of a graph G is a set $\mathcal{R} = \{R_i \mid 1 \leq i \leq n\}$ of geometric objects (e. g., bars, rectangles, etc.) along with some kind of geometric visibility relation $\sim$ over $\mathcal{R}$ (e. g., axis-parallel visibility), such that $G = (\{v_i \mid 1 \leq i \leq n\}, \{\{v_i, v_j\} \mid R_i \sim R_j\})$. In this work, we focus on rectangle visibility graphs, which are represented by axis aligned rectangles in the plane and vertical and horizontal axis parallel visibility between them. In particular, we consider the more restricted variant of *unit square visibility graphs* (see [12]), and, in addition, we consider the case where the unit squares are placed on an integer grid (an alternative characterisation of the well-known class of graphs with rectilinear drawings).

The study of visibility representations is of interest, both for applications and for graph classes, and has remained an active research area[1] mainly because axis-aligned visibilities give rise to graph and network visualizations that satisfy good readability criteria: straight

* We acknowledge the support of the first author by the Deutsche Forschungsgemeinschaft, grant FE 560/6-1, and the support of the last author by the NSERC Discovery Grant program of Canada.

[1] The 24th International Symposium on Graph Drawing and Network Visualization (GD 2016) featured an entire session on visibility representation (see [3, 10, 11, 24]), and the joint workshop day of the Symposium on Computational Geometry (SoCG) and the ACM Symposium on Theory of Computing (STOC) included a workshop on geometric representations of graphs in 2016.

42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017).
Editors: Kim G. Larsen, Hans L. Bodlaender, and Jean-Francois Raskin; Article No. 30; pp. 30:1–30:15
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

edges, and edges that cross only at right angles. These properties are highly desirable in the design of layouts of circuits and communication paths. Indeed, the study of graphs arising from vertical visibilities among disjoint, horizontal line segments ("bars") in the plane originated during the 1980's in the context of VLSI design problems; see [16, 30, 29].

Because bar visibility graphs are necessarily planar, this model has been extended in various ways in order to represent larger classes of graphs. Such extensions include new definitions of visibility (e. g., sight lines that may penetrate up to k bars [13] or other geometric objects [4]), vertex representations by other objects (e. g., rectangles, L-shapes [18], and sets of up to t bars [23]), extensions to higher dimensional objects (see, e. g., [8] for visibility representation in 3D by axis aligned horizontal rectangles with vertical visibilities, or [19], which studies visibility representations by unit squares floating parallel to the x, y-plane and lines of sight that are parallel to the z axis). The desire for polysemy, that is, the expression of more than one graph by means of one underlying set of objects, has also provided impetus in the study of visibility representations (see for example [6] and [28]).

Rectangle visibility graphs have the attractive property, for visualization purposes, that they yield right angle crossing drawings (RAC graphs (see [15]), whose edges are drawn as sequences of horizontal and vertical segments forming a polyline with orthogonal bends), which have seen considerable interest in the graph drawing community. Unit square graphs form a subfamily of L-visibility graphs (see [18]) and their grid variant a subfamily of RACs with no bends (note that RAC recognition for 0-bends is NP-hard [2]).

Using visibilities among objects is but one example of the use of binary geometric relations for this purpose; other geometric relations include intersection relations (e.g., of strings or straight line segments in the plane, of boxes in arbitrary dimension), proximity relations (e.g., of points in the plane), and contact relations. In the literature, for the resulting graph classes, combinatorial aspects, relationships to other graph classes, as well as computational aspects are studied (see [20] for a survey focusing on contact representations of rectangles).

Finally, we note that visibility properties among sets of objects have been studied in a number of contexts, including motion planning and computer graphics. In [26] it is proposed to find shortest paths for mobile robots moving in a cluttered environment by looking for shortest paths in the visibility graph of the points located at the vertices of polygonal obstacles. This led to a search for fast algorithms to compute visibility graphs of polygons, as well as to a search for finding shortest paths without computing the entire visibility graph.

We extend the known combinatorial properties of unit square visibility graphs from [12], and proof their recognition problem to be NP-hard (this requires a reduction that is highly non-trivial on a technical level with the main difficulty to identify graph structures that can be shown to be representable by unit square layouts in a unique way to gain sufficient control for designing suitable gadgets). With respect to unit square *grid* visibility graphs, we extend known combinatorial properties and consider variants of its recognition problem.

Due to space constraints, we only provide proof sketches (details can be found in [9]).

2 Preliminaries

A *visibility layout*, or simply *layout*, is a set $\mathcal{R} = \{R_i \mid 1 \leq i \leq n\}$ with $n \in \mathbb{N}$, where R_i are closed and pairwise disjoint axis-parallel rectangles in the plane; the *position* of such a rectangle is the coordinate of its lower left corner. For every $R_i, R_j \in \mathcal{R}$, a closed non-degenerate axis-parallel rectangle S (i. e., a non-empty closed rectangle that is not a line segment) is a *visibility rectangle for R_i and R_j* if one side of S is contained in R_i and the opposite side in R_j. We define $R_i \rightarrow_{\mathcal{R}} R_j$ ($R_i \downarrow_{\mathcal{R}} R_j$), if there is a visibility rectangle S for R_i

and R_j, such that the left side (upper side) of S is contained in R_i, the right side (lower side) of S is contained in R_j and $S \cap R_k = \emptyset$, for every $R_k \in \mathcal{R} \setminus \{R_i, R_j\}$. Let $\leftrightarrow_{\mathcal{R}}$ and $\updownarrow_{\mathcal{R}}$ be the symmetric closures of $\rightarrow_{\mathcal{R}}$ and $\downarrow_{\mathcal{R}}$, respectively. Finally, $R_i \sim_{\mathcal{R}} R_j$ if $R_i \leftrightarrow_{\mathcal{R}} R_j$ or $R_i \updownarrow_{\mathcal{R}} R_j$ ($\sim_{\mathcal{R}}$ is the *visibility relation* (*with respect to* $\mathcal{R}$)). If the layout $\mathcal{R}$ is clear from the context or negligible, we drop the subscript $\mathcal{R}$. We denote $R_i \sim R_j$, $R_i \leftrightarrow R_j$ and $R_i \rightarrow R_j$ also as R_i *sees* R_j, R_i *horizontally sees* R_j and R_i *sees* R_j *from the left*, respectively, and analogous terminology applies to vertical visibilities. For $S, T \subseteq \mathcal{R}$, we use $S \rightarrow_{\mathcal{R}} T$ as shorthand form for $\bigwedge_{R \in S, R' \in T} R \rightarrow_{\mathcal{R}} R'$.

A layout $\mathcal{R} = \{R_i \mid 1 \leq i \leq n\}$ *represents* the undirected graph $\mathsf{G}(\mathcal{R}) = (\{v_i \mid 1 \leq i \leq n\}, \{\{v_i, v_j\} \mid 1 \leq i, j \leq n, R_i \sim R_j\})$, which is then called a *visibility graph*, and the class of visibility graphs is denoted by V. A graph is a *weak* visibility graph, if it can be obtained from a visibility graph by deleting some edges and the corresponding class of graphs is denoted by V_{w}. As a convention, for a visibility graph $G = (V, E)$ and a layout representing it we denote by R_v the rectangle for $v \in V$ and define $R_{V'} = \{R_x \mid x \in V'\}$ for every $V' \subseteq V$. We call layouts $\mathcal{R}_1$ and $\mathcal{R}_2$ *isomorphic* if $\mathsf{G}(\mathcal{R}_1)$ and $\mathsf{G}(\mathcal{R}_2)$ are isomorphic. Furthermore, we call $\mathcal{R}_1$ and $\mathcal{R}_2$ V-*isomorphic* if, for some $x \in \{\rightarrow_{\mathcal{R}_1}, \rightarrow^{-1}_{\mathcal{R}_1}\}$ and $y \in \{\downarrow_{\mathcal{R}_1}, \downarrow^{-1}_{\mathcal{R}_1}\}$, the relational structure $(\mathcal{R}_1, \rightarrow_{\mathcal{R}_1}, \downarrow_{\mathcal{R}_1})$ is isomorphic to $(\mathcal{R}_2, x, y)$ or $(\mathcal{R}_2, y, x)$.[2]

Unit square visibility graphs (USV) and *unit square grid visibility graphs* (USGV) are represented by *unit square layouts*, where every $R \in \mathcal{R}$ is the unit square, and *unit square grid layouts*, where additionally the position of every R is from $\mathbb{N} \times \mathbb{N}$.[3] The weak classes $\mathsf{USV}_{\mathsf{w}}$ and $\mathsf{USGV}_{\mathsf{w}}$ are defined accordingly.

For a graph $G = (V, E)$, $N(v)$ is the *neighbourhood* of $v \in V$, $\vec{E}$ denotes an oriented version of E, i. e., $E = \{\{u, v\} \mid (u, v) \in \vec{E}\}$, and $f \colon \vec{E} \rightarrow E, (u, v) \mapsto \{u, v\}$ is a bijection. Let L, R and D, U be pairs of complementary values (for $X \in \{\mathsf{L}, \mathsf{R}, \mathsf{D}, \mathsf{U}\}$, $\overline{X}$ denotes its complement). An LRDU-*restriction* (for G) is a labeling $\sigma \colon \vec{E} \rightarrow \{\mathsf{L}, \mathsf{R}, \mathsf{D}, \mathsf{U}\}$ and it is *valid* if, for every $(u, v) \in \vec{E}$ with $\sigma((u, v)) = X$ and every $w \in V \setminus \{u, v\}$, $\sigma((u, w)) \neq X \neq \sigma((w, v))$ and $\sigma((v, w)) \neq \overline{X} \neq \sigma((w, u))$. Obviously, LRDU-restrictions only exist for graphs with maximum degree 4. A unit square grid visibility layout *satisfies* an LRDU-restriction σ if $\sigma((u, v)) = \mathsf{L}$ implies $R_v \rightarrow R_u$, $\sigma((u, v)) = \mathsf{R}$ implies $R_u \rightarrow R_v$, $\sigma((u, v)) = \mathsf{D}$ implies $R_u \downarrow R_v$ and $\sigma((u, v)) = \mathsf{U}$ implies $R_v \downarrow R_u$. An HV-*restriction* (for G) is a labeling $\sigma \colon E \rightarrow \{\mathsf{H}, \mathsf{V}\}$ and it is *valid* if, for every $u \in V$ at most two incident edges are labeled H and at most two incident edges are labeled V. A unit square grid visibility layout *satisfies* an HV-restriction σ if $\sigma(\{u, v\}) = \mathsf{H}$ implies $R_v \leftrightarrow R_u$ and $\sigma(\{u, v\}) = \mathsf{V}$ implies $R_v \updownarrow R_u$.

For a class $\mathfrak{G}$ of undirected graphs, the *recognition problem for* $\mathfrak{G}$ (denoted by $\textsc{Rec}(\mathfrak{G})$) for short) is the problem to decide, for a given undirected graph G, whether or not $G \in \mathfrak{G}$. In the following, we shall consider the problems $\textsc{Rec}(\mathsf{USGV})$ and $\textsc{Rec}(\mathsf{USV})$.

We briefly recall some established geometric graph representations relevant to this work. A *rectilinear drawing* (see [17, 25]) of a graph $G = (V, E)$ is a pair of mappings $x, y \colon V \rightarrow \mathbb{Z}$, where, for every $v \in V$, $x(v)$ and $y(v)$ represent the x- and y-coordinates of v on the grid and, for every edge $\{u, v\} \in E$, $(x(u), y(u))$ and $(x(v), y(v))$ are the endpoints of a horizontal or vertical line segment that does not contain any $(x(w), y(w))$ with $w \in V \setminus \{u, v\}$. A graph has *resolution* $\frac{2\pi}{d}$ if it has a drawing in which the degree of the angle between any two edges incident to a common vertex is at least $\frac{2\pi}{d}$. We call such graphs *resolution-*$\frac{2\pi}{d}$ *graphs* and are mainly interested in the case $d = 4$, see [21]. For planar graphs, resolution-$\frac{2\pi}{4}$ graphs are

[2] By $\preceq^{-1}$, we denote the inverse of a binary relation $\preceq$.
[3] Note that in the grid case, if a unit square is positioned at (x, y), then this is the only unit square on coordinates (x', y'), $x' \in \{x - 1, x, x + 1\}$, $y' \in \{y - 1, y, y + 1\}$.

just rectilinear graphs, see [7]. A *bendless right angle crossing* (BRAC) *drawing* of a graph is a straight-line drawing in which every crossing of two edges is at right angles.[4] Note that in a BRAC-drawing or a resolution-$\frac{2\pi}{4}$ drawing, edges are not necessarily axis-parallel (like it is the case for visibility layouts and rectilinear drawings). A graph is called *rectilinear* or *BRAC graph* if it has a rectilinear or BRAC-drawing, respectively.

3 Unit Square Grid Visibility Graphs

The readability of graph drawings is mainly affected by its *angular resolution* (angles formed by consecutive edges incident to a common node) and its *crossing resolution* (angles formed at edge crossings); see the discussion in [1]. In this regard, resolution-$\frac{\pi}{2}$ graphs and BRAC graphs have an angular resolution and crossing resolution of $\frac{\pi}{2}$, respectively, while rectilinear drawings and unit square grid visibility layouts force both resolutions to be $\frac{\pi}{2}$.

The question arises of how these classes relate to each other and in this regard, we first note that USGV and rectilinear graphs coincide. More precisely, a unit square grid layout can be transformed into a rectilinear drawing by replacing every unit square on position (x, y) by a vertex on position (x, y) and translate the former visibilities into straight-line segments. Transforming a rectilinear drawing into a unit square grid layout requires scaling it first by factor 2 and then replacing each vertex on position (x, y) by a unit square on position (x, y) (without scaling, sides or corners of unit squares may overlap). This only results in a *weak* layout, since visibilities may be created that do not correspond to edges in the rectilinear drawing. However, any weak unit square grid visibility graph can be transformed into a unit square grid visibility graph (as formally stated below in Theorem 7).

Since all these graphs except the BRAC graphs have maximum degree 4, we only consider degree-4 BRAC graphs. Obviously, resolution-$\frac{\pi}{2}$ graphs and degree-4 BRAC graphs are both superclasses of USGV (and rectilinear graphs). Witnessed by K_3, the inclusion in degree-4 BRAC graphs is proper, while the analogous question w. r. t. resolution-$\frac{\pi}{2}$ graphs is open. Moreover, K_3 is also an example of a degree-4 BRAC graph that is not a resolution-$\frac{\pi}{2}$ graph; whether there exist resolution-$\frac{\pi}{2}$ graphs without a BRAC-drawing is open.

Due to the equivalence of USGV and rectilinear graphs, results for the latter graph class carry over to the former. In this regard, we first mention that the NP-hardness proof of recognizing resolution-$\frac{\pi}{2}$ graphs from [21] actually produces drawings with axis-aligned edges; thus, it also applies to rectilinear graphs (a similar reduction (for rectilinear graphs and presented in more detail) is provided in [17]). As shown in [17], the recognition problem for rectilinear graphs can be solved in time $O(24^k \cdot k^{2k} \cdot n)$, where k is the number of vertices with degree at least 3. In [25], it is shown that recognition remains NP-hard if we ask whether a drawing exists that satisfies a given HV-restriction[5] or a drawing that satisfies a given circular order of incident edges. However, checking the existence of a rectilinear drawing satisfying a given LRDU-restriction can be done in time $O(|E| \cdot |V|)$. Consequently, by trying all such labellings, we can solve the recognition problem for rectilinear graphs in time $2^{O(n)}$. In this regard, it is worth noting that the hardness reduction from [17] can be easily modified, such that it also provides lower complexity bounds subject to the Exponential-Time Hypothesis (ETH), thereby demonstrating that the $2^{O(n)}$ algorithm is optimal subject to ETH.

---

[4] In the literature (e. g., [15]), the edges of a RAC-drawing are usually allowed to have bends; the investigated questions are on finding RAC-drawings that minimise the number of bends and crossings.

[5] The definition of HV- and LRDU-restriction given above naturally extends to rectilinear drawings.

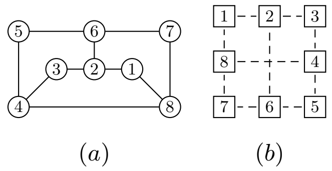

Figure 1 Necessarily non-planar visibility layout for a planar graph.

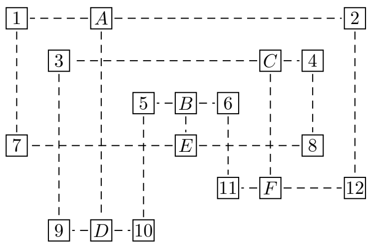

Figure 2 Subdivisions of $K_{3,3}$.

3.1 Combinatorial Properties of USGV

First, we shall see that the class USGV is downward closed w. r. t. the subgraph relation, i. e., if $G \in$ USGV, then all its subgraphs are in USGV (intuitively speaking, deletion of edges can be done by moving unit squares, while deletion of a vertex can be realised by deleting the corresponding unit square and then removing unwanted edges introduced by this operation). This observation will be a convenient tool for obtaining other combinatorial results.

▶ **Lemma 1.** *Let $G = (V, E) \in$ USGV, let $v \in V$ and $e \in E$. Then $(V, E \setminus \{e\}) \in$ USGV and $(V \setminus \{v\}, E) \in$ USGV.*

It is straightforward to prove the following limitations of USGV.

▶ **Lemma 2.** *Let $G = (V, E) \in$ USGV. Then, (**1**) the maximum degree of G is 4, (**2**) for every $u, v \in V$, $|N(u) \cap N(v)| \leq 2$, and, (**3**) for every $\{u, v\} \in E$, $N(u) \cap N(v) = \emptyset$.*

A consequence of Lemma 2 is that no graph from USGV contains $K_{1,5}$, $K_{2,3}$ or K_3 as a subgraph, since they violate the first, second and third condition of Lemma 2, respectively. Obvious examples for graphs from USGV are subgraphs of a grid; as Lemma 1 shows, even non-induced subgraphs of a grid. In this context, notice that the problem of deciding if a given graph is such a *partial grid graph* is equivalent to deciding if it admits a unit-length VLSI layout, which, even restricted to trees, is an NP-hard problem; see [5] for details. Yet, USGV contains more, especially non-bipartite graphs, with the smallest example being C_5.

Next, we discuss planarity with a focus on the relationship between the planarity of graphs from USGV and planarity of their respective layouts (where a layout is called *planar* if it does not contain any crossing visibilities). In this regard, we first note that the planarity of a layout is obviously sufficient for the planarity of the represented graph. Moreover, it is trivial to construct non-planar layouts that nevertheless represent planar graphs. Figure 1(a) is an example of a planar unit square grid visibility graph, which can only be represented by non-planar layouts (e. g., the one of Figure 1(b)):

▶ **Proposition 3.** *There exists no planar unit square grid layout for the graph of Fig. 1(a).*

It is tempting to assume that graphs in USGV are necessarily planar, but, as demonstrated by Figure 2, USGV contains a subdivision of $K_{3,3}$. Hence, with Kuratowski's theorem, we conclude:

▶ **Theorem 4.** USGV *contains non-planar graphs.*

Next, we investigate possibilities to characterise USGV. In this regard, we first observe that a characterisation by forbidden induced subgraphs is not possible (note that under the assumption P ≠ NP, this also follows from the hardness of recognition).

▶ **Theorem 5.** USGV *does not admit a characterisation by a finite number of forbidden induced subgraphs.*

By Lemma 2, the classes of cycles, complete graphs and complete bipartite graphs within USGV are easily characterised: $C_i \in$ USGV if and only if $i \geq 4$, $K_i \in$ USGV if and only if $i \leq 2$, $K_{i,j} \in$ USGV (with $i \leq j$) if and only if ($i = 1$ and $j \leq 4$) or ($i = 2$ and $j = 2$). Furthermore, the trees in USGV have a simple characterisation as well:

▶ **Theorem 6.** *A tree T is in* USGV *if and only if the maximum degree of T is at most four.*

By definition, USGV ⊆ $\mathsf{USGV_w}$ and every $G' \in \mathsf{USGV_w}$ can be obtained from some $G \in$ USGV by deleting some edges. Consequently, by Lemma 1, we conclude the following.

▶ **Theorem 7.** USGV = $\mathsf{USGV_w}$.

3.2 Area-Minimisation

The *area-minimisation* version of the recognition problem is to decide whether a given graph has a drawing or layout of given width and height. The hardness of recognition for USGV and also for HV-restricted USGV carries over to the area-minimisation version, since an n-vertex graph has a layout if and only if it has a $(2n-1) \times (2n-1)$ layout. On the other hand, in the LRDU-restricted rectilinear (or unit square grid) case, recognition can be solved in polynomial time, so the authors of [25] provide a hardness reduction that proves the area-minimisation recognition problem NP-complete even for LRDU-restricted rectilinear graphs. However, this construction does not carry over to USGV, since the non-edges of a rectilinear drawing translate into non-visibilities, which require space as well;[6] moreover, it does not even work for the weak case of USGV, due to the necessary scaling by factor 2 to translate a rectilinear drawing into an equivalent weak unit square grid layout.

Next, we provide a reduction that shows the hardness of the area-minimisation version of Rec($\mathsf{USGV_w}$), which shall also imply several additional results. The problem 3-Partition (3Part) is defined as follows: Given $B \in \mathbb{N}$ and a multi-set $A = \{a_1, a_2, \ldots, a_{3m}\} \subseteq \mathbb{N}$ with $\frac{B}{4} < a_i < \frac{B}{2}$, $1 \leq i \leq 3m$, and $\sum_{i=1}^{3m} a_i = mB$, decide whether A can be partitioned into m multi-sets $A_1, \ldots, A_m$, such that $\sum_{a \in A_j} a = B$, $1 \leq j \leq m$ (note that the restriction $\frac{B}{4} < a_i < \frac{B}{2}$ enforces $|A_j| = 3$, $1 \leq j \leq m$). Given a 3Part instance, we construct a *frame graph* (see Figure 3) $G_f = (V_f, E_f)$ with:

$$\begin{aligned} V_f =& \{u_{i,j}, v_{i,j}, w_{i,1}, w_{i,2} \mid 1 \leq i \leq m, 0 \leq j \leq B\} \cup \{u_{m+1,0}, v_{m+1,0}, w_{m+1,1}, w_{m+1,2}\}, \\ E_f =& \{\{u_{i,j}, u_{i,j+1}\}, \{v_{i,j}, v_{i,j+1}\} \mid 1 \leq i \leq m, 0 \leq j \leq B-1\} \cup \\ & \{\{u_{i,B}, u_{i+1,0}\}, \{v_{i,B}, v_{i+1,0}\} \mid 1 \leq i \leq m\} \cup \{\{u_{i,j}, v_{i,j}\} \mid 1 \leq i \leq m, 1 \leq j \leq B\} \cup \\ & \{\{u_{i,0}, v_{i,0}\}, \{v_{i,0}, w_{i,1}\}, \{w_{i,1}, w_{i,2}\} \mid 1 \leq i \leq m+1\}. \end{aligned}$$

Next, we define a graph $G_A = (V_A, E_A)$ with $V_A = \{b_{i,j}, c_{i,j} \mid 1 \leq i \leq 3m, 1 \leq j \leq a_i\}$ and $E_A = \{\{b_{i,j}, b_{i,j+1}\}, \{c_{i,j}, c_{i,j+1}\} \mid 1 \leq i \leq 3m, 1 \leq j \leq a_i - 1\} \cup \{\{b_{i,j}, c_{i,j}\} \mid 1 \leq i \leq 3m, 1 \leq j \leq a_i\}$. Finally, we let $G = (V, E)$ with $V = V_f \cup V_A$ and $E = E_f \cup E_A$.

[6] In general, this space blow-up cannot be avoided, as witnessed by n isolated vertices which have a $1 \times n$ rectilinear drawing, but a smallest unit square grid layout of size $(2n-1) \times (2n-1)$

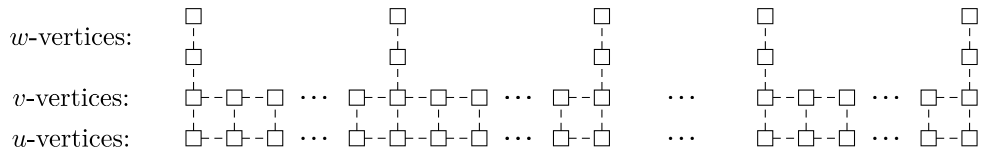

Figure 3 Unit square grid layout for the graph G_f.

The idea is that G_f forms m size-B compartments and the graphs on $b_{i,j}$, $c_{i,j}$ represent the a_i. In a layout respecting the size bounds, the way of allocating the graphs on $b_{i,j}$, $c_{i,j}$ to the compartments corresponds to a partition of A that is a solution for the 3Part-instance.

▶ **Lemma 8.** *(B, A) is a positive 3Part-instance if and only if G has a $(7 \times (2(mB+m+1)-1))$ unit square grid layout.*

Since the reduction defined above is polynomial in m and B, and 3Part is strongly NP-complete (see [22, Theorem 4.4]), we can conclude the following:

▶ **Theorem 9.** *The area-minimisation variant of* Rec(USGV$_w$) *is* NP*-complete.*

The area minimisation variant implicitly solves the general recognition problem, so the question arises whether it is also hard to decide if a graph from USGV$_w$ (given as a layout) can be represented by a layout satisfying given size bounds. Since our reduction always produces a graph in USGV$_w$ (with an obvious layout), independent of the 3Part-instance, it shows that the hardness remains if the input graph is given directly as a layout. Moreover, the problem is still NP-complete for the LRDU-restricted variant (the LRDU-restriction then simply enforces the structure shown in Figure 3).

The reduction also yields a (substantially simpler) alternative proof for the hardness of the area-minimisation recognition problem for LRDU-restricted rectilinear graphs [25] (more precisely, it can be shown that (B, A) is a positive 3Part-instance if and only if G has a $(4 \times (mB + m + 1))$ rectilinear drawing), and the hardness also carries over to the variant where the input graph is already given as a rectilinear drawing.

We conclude this section by pointing out that it is open whether the LRDU-restricted area-minimisation variant of Rec(USGV) can be solved in polynomial-time. Intuitively, reducing the size of a rectilinear drawing is difficult, since space can be saved by placing non-adjacent vertices on the same line, which is not possible for *non-weak* unit square grid layouts. However, computing a size-minimal unit square grid layout includes finding out to what extend the scaling by 2 is really necessary, which seems difficult as well.

4 Unit Square Visibility Graphs

Obviously, a larger class of graphs can be represented if the unit squares are not restricted to integer coordinates (see Figure 4 for some examples). In [12], cycles, complete graphs, complete bipartite graphs and trees in USV are characterised as follows: $C_i \in$ USV, for every $i \in \mathbb{N}$, $K_i \in$ USV if and only if $i \leq 4$, $K_{i,j} \in$ USV with $i \leq j$ if and only if $(1 \leq i \leq 2$ and $i \leq j \leq 6)$ or $(i = 3$ and $3 \leq j \leq 4)$,[7] and a tree T is in USV if and only if it is the union of two subdivided caterpillar forests with maximum degree 3 (note that [23] provides an algorithm that efficiently checks this property).

[7] For the more general question of representing bipartite graphs as rectangle visibility graphs, we refer to [14]. In particular, a linear upper bound on the number of edges, compared to the number of vertices, is known.

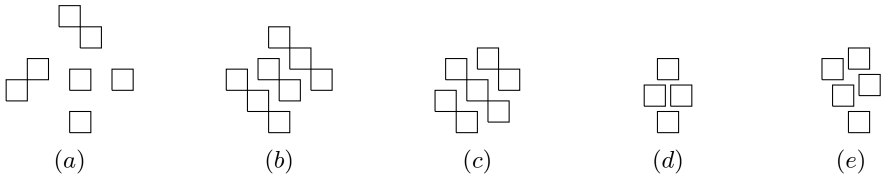

Figure 4 Visibility layouts for $K_{1,6}$, $K_{2,6}$, $K_{3,4}$, K_4 and a K_5 with one missing edge.

Next, we observe that every graph with at most 4 vertices is in USV, while K_5 is not (it is not hard to find layouts for graphs with at most 4 vertices; $K_5 \notin$ USV is shown in [12]).

▶ **Proposition 10.** *Every graph with at most* 4 *vertices is in* USV*.*

A crucial difference between USGV and USV is that for the latter, the degree is not bounded, as witnessed by layouts of the following form: □□□□□□□. However, if a unit square sees at least 7 other unit squares, then these must be placed in such a way that visibilities or "paths" between some of them are enforced (note that any $K_{1,n}$ may exist as induced subgraph, as can be demonstrated by modifying the above example layout such that between each two consecutive neighbours another "visibility-blocking" unit square is inserted). In [12], it is formally proven that in graphs from USV any vertex of degree at least 7 must lie on a cycle. In particular, these observations point out that an analogue of Lemma 1 is not possible for USV.

For the class of trees within USV, as long as we consider trees with maximum degree strictly less or larger than 6, a much simpler characterisation (compared to the one mentioned at the beginning of this section) applies:

▶ **Theorem 11.** *Let T be a tree with maximum degree k. If $k \leq 5$, then $T \in$* USV*, and if $k \geq 7$, then $T \notin$* USV*.*

Figure 5(a) shows an example of a tree from USV with maximum degree 6 and Figure 5(b) its representing layout. It can be easily verified that any node of degree 6 must be represented V-isomorphically to Figure 4(a) (note that this also holds for nodes A and B in Figures 5(a) and (b)). Figure 4(a) also demonstrates that not all trees with maximum degree 6 can be represented: let R denote the square below the central square in the layout, then it is impossible for R to see 5 additional unit squares that exclusively see R. On the other hand, USV contains trees with arbitrarily many degree-6 vertices, e. g., trees of the form depicted in Figure 5(c) (it is straightforward to see that they can be represented as the union of two forests of caterpillars with maximum degree 3). This reasoning shows that not all planar graphs are in USV, while it follows from [30] that all planar graphs are (non-unit square) rectangle visibility graphs (also see [29]).

Finally, we note that USV is a proper subset of USV_w (e. g., $K_{1,7}$ is a separating example):

▶ **Theorem 12.** $\text{USV} \subsetneq \text{USV}_\text{w}$.

4.1 The Recognition Problem

The recognition problem for USV consists in checking whether a given graph can be represented by a unit square layout. We first observe that this problem is in NP (note that this is not completely trivial, since we cannot naively guess a layout) and the main result of this section shall be its hardness (see Theorem 20).

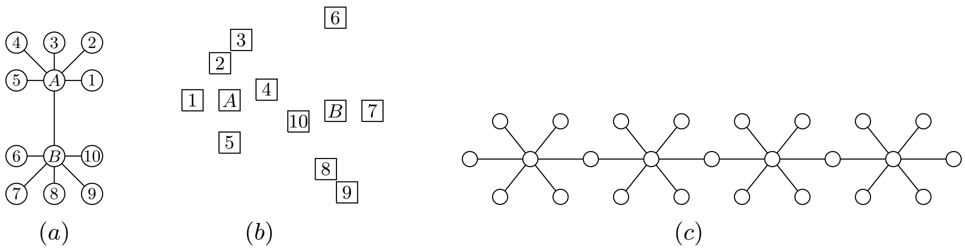

Figure 5 Illustration for trees from USV with maximum degree 6.

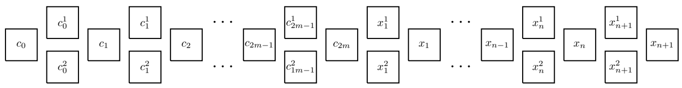

Figure 6 The backbone-gadget.

▶ **Theorem 13.** $\text{Rec}(\mathsf{USV}) \in \mathsf{NP}$.

We prove the NP-hardness by a reduction from NAE-3SAT, i. e., the not-all-equal 3-satisfiability problem [27]. To this end, let $F = \{c_1, \ldots, c_m\}$ be a 3-CNF formula over the variables $x_1, \ldots, x_n$, such that no variable occurs more than once in any clause, and, for the sake of convenience, let $c_i = \{y_{i,1}, y_{i,2}, y_{i,3}\}$, $1 \leq i \leq m$.

The general idea of the reduction is as follows: We identify graph structures that can be shown to have a (more or less) unique representation as a unit square layout. With these main building blocks, we construct a sequence of clause and variable gadgets, called *backbone* (see Figure 6), that can only be represented by a layout in a linear way, say horizontally. Furthermore, every clause gadget is vertically connected to its three literals, two of which are below and the other one above the backbone, or the other way around. The allocation of literal vertices to a variable x_i is done by a path of all literal vertices corresponding to x_i that is connected to the variable vertex for x_i. Such paths must lie either completely above or below the backbone. Interpreting the situation that a path lies above the backbone as assigning *true* to the corresponding literal, yields a not-all-equal satisfying assignment, as it is not possible that all the paths for a clause lie on the same side of the backbone.

We assume that each clause of F contains at most one negated variable, which is no restriction to not-all-equal satisfiability as a clause over literals l_1, l_2, l_3 is not-all-equal satisfied by an assignment if and only if a clause over literals $\bar{l}_1, \bar{l}_2, \bar{l}_3$ is. Furthermore, we also assume that every literal occurs at least three times in the formula. We first transform F into $F' = \{c_1, \ldots, c_{2m}\}$, where $c_{m+i} = c_i$ for $i = 1, \ldots, m$. Then, we transform F' into a graph $G = (V, E)$ as follows. The set of vertices is defined by $V = V_c \cup V_x \cup V_h$, where

$$V_c = \{c_j, c_j^1, c_j^2 \mid 0 \leq j \leq 2m-1\} \cup \{c_{2m}\} \cup \{l_j^1, l_j^2, l_j^3 \mid 1 \leq j \leq 2m\}\,,$$
$$V_x = \{x_i, x_i^1, x_i^2 \mid 1 \leq i \leq n+1\} \cup \{t_i, \overrightarrow{t_i}, \overleftarrow{t_i}, f_i^1, \overrightarrow{f_i^1}, \overleftarrow{f_i^1}, f_i^2, \overrightarrow{f_i^2}, \overleftarrow{f_i^2} \mid 1 \leq i \leq n\}\,,$$
$$V_h = \{h_{t_i}^r, h_{f_i^1}^r, h_{f_i^2}^r \mid 1 \leq i \leq n, 0 \leq r \leq 4\}\,.$$

Vertices c_j and x_i represent the corresponding clauses and variables and the vertices c_j^r, x_i^r, $r \in \{1, 2\}$ are used to enforce the *backbone* structure as described at the beginning of this section. The corresponding edges are implicitly defined, by requiring, for every $0 \leq i \leq 2m-1$ and $1 \leq i \leq n$, the following groups of 4 vertices to form a K_4: $\{c_j, c_j^1, c_j^2, c_{j+1}\}$, $\{x_i, x_{i+1}^1, x_{i+1}^2, x_{i+1}\}$, and $\{c_{2m}, x_1^1, x_1^2, x_1\}$. Also, for every $j \in \{1, 2\}$, the vertices $c_0^j, c_1^j, \ldots, c_{2m-1}^j, x_1^j, x_2^j, \ldots, x_{n+1}^j$ form a path in this order. Consequently, these

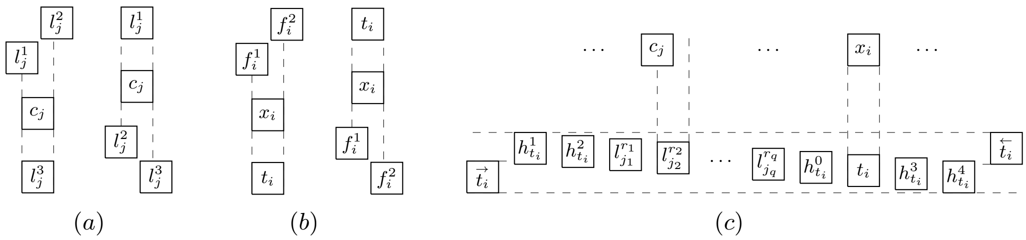

Figure 7 Possible placements of literal vertices, possible placements of assignment vertices, and the clause path for x_i.

vertices form the subgraph represented by the layout in Figure 6, which shall be the *backbone*. Vertices t_i, represent the literal x_i, f_i^1 represent the literal $\overline{x_i}$ in the first m clauses, and f_i^2 represent the literal $\overline{x_i}$ in the remaining clauses. Vertices l_j^1, l_j^2, l_j^3 represent the literals of clause c_j. These roles are reflected with edges $\{x_i, t_i\}$, $\{x_i, f_i^1\}$, $\{x_i, f_i^2\}$ for all $1 \leq i \leq n$ and $\{c_j, l_j^r\}$ for all $1 \leq j \leq 2m$ and $1 \leq r \leq 3$. The connection between literals and variable assignments is build by turning $l_{j,r}$ with $y_{j,r} = x_i$ into a path connected to t_i; analogously, $l_{j,r}$ with $y_{j,r} = \overline{x_i}$ in the first (the last, respectively) m clauses form a path connected to f_i^1 (f_i^2, respectively). More precisely, for every $1 \leq j \leq 2m$, $1 \leq i \leq n$ and $1 \leq r \leq 3$:

- if $y_{j,r} = x_i$, there are edges $\{l_j^r, \overrightarrow{t_i}\}$, $\{l_j^r, \overleftarrow{t_i}\}$,
- if $y_{j,r} = \overline{x_i}$ and $1 \leq j \leq m$, there are edges $\{l_j^r, \overrightarrow{f_i^1}\}$, $\{l_j^r, \overleftarrow{f_i^1}\}$ and $\{l_{j+m}^r, \overrightarrow{f_i^2}\}$, $\{l_{j+m}^r, \overleftarrow{f_i^2}\}$,
- there are edges $\{t_i, \overrightarrow{t_i}\}$, $\{t_i, \overleftarrow{t_i}\}$ and $\{\overrightarrow{t_i}, h_{t_i}^p\}$,$\{\overleftarrow{t_i}, h_{t_i}^p\}$ for all $0 \leq p \leq 4$,
- there are edges $\{f_i^s, \overrightarrow{f_i^s}\}$, $\{f_i, \overleftarrow{f_i^s}\}$ and $\{\overrightarrow{f_i^s}, h_{f_i^s}^p\}$,$\{\overleftarrow{f_i^s}, h_{f_i^s}^p\}$ for all $0 \leq p \leq 4$, $s \in \{1, 2\}$,

Moreover, for every i, $1 \leq i \leq n$,

- if $N(\overrightarrow{t_i}) = \{h_{t_i}^1, h_{t_i}^2, l_{j_1}^{r_1}, l_{j_2}^{r_2}, \ldots, l_{j_q}^{r_q}, h_{t_i}^0, t_i, h_{t_i}^3, h_{t_i}^4\}$ with $j_1 < j_2 < \ldots < j_q$, then these vertices form a path in this order,
- if $N(\overrightarrow{f_i^s}) = \{h_{f_i^s}^1, h_{f_i^s}^2, l_{j_1}^{r_1}, l_{j_2}^{r_2}, \ldots, l_{j_q}^{r_q}, h_{f_i^s}^0, f_i^s, h_{f_i^s}^3, h_{f_i^s}^4\}$ with $j_1 < j_2 < \ldots < j_q$ and $s \in \{1, 2\}$, then these vertices form a path in this order,

Next, we assume that the formula F' is not-all-equal satisfiable and show how a layout for G can be constructed. First, we represent the backbone as illustrated in Figure 6. If a variable x_i is assigned the value *true*, then we place the unit squares $R_{\{x_i, t_i, f_i^1, f_i^2\}}$ as illustrated on the left side of Figure 7(b), and otherwise as illustrated on the right side. The edges for the vertices $t_i, \overrightarrow{t_i}, \overleftarrow{t_i}, h_{t_i}^r$, $0 \leq r \leq 4$, and all l_j^r with $y_{j,r} = x_i$ can be realised as illustrated in Figure 7(c) (either placed above or below the backbone, according to the position of R_{t_i}). An analogous construction applies to the unit squares for l_j^r with $y_{j,r} = \overline{x_i}$, with the only difference that we have two such paths (one for the first m clauses and one for the remaining clauses) and that they both lie on the opposite side of the backbone with respect to R_{t_i}. Moreover, in these paths, the $R_{l_j^r}$ must be horizontally shifted such that they can see their corresponding R_{c_j} from above or from below, according to whether the path lies above or below the backbone (as indicated in Figure 7(c)). As long as not all paths for the three literals of the same clause lie all above or all below the backbone, this is possible by arranging the unit squares as illustrated in Figure 7(a). However, if for some clause all paths lie on the same side of the backbone, then the literals of the clause are either all set to *true* or all set to *false*, which is a contradiction to the assumption that the assignment is not-all-equal satisfiable. Consequently, we can represent G as described.

▶ **Lemma 14.** *If F is not-all-equal satisfiable, then $G \in$* USV.

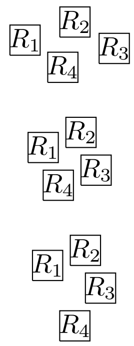

Figure 8 Re- presenting K_4.

Proving that a layout for G translates into a satisfying not-all-equal assignment for F, is much more involved. The general idea is to show that any layout for G must be V-isomorphic to the layout constructed above. However, this cannot be done separately for the individual gadgets, e. g., showing that the backbone must be represented as in Figure 6 (in fact, the structure of the backbone alone does not enforce such a layout) and the literal vertices must form a path as in Figure 7(c) and so on. Instead, the desired structure of the layout is only enforced by a rather complicated interplay of the different parts of G.

A main building stone is that a K_4 can only be represented in 3 different ways (up to V-isomorphism), which are illustrated in Figure 8. This observation is important, since the backbone is a sequence of K_4.

▶ **Lemma 15.** *Every layout for K_4 is V-isomorphic to one of the three layouts of Figure 8.*

We now assume that G can be represented by some layout $\mathcal{R}$. For every j, $1 \leq j \leq m$, we define $L_j = \{l_j^1, l_j^2, l_j^3\}$, for every i, $1 \leq i \leq n$, we define $A_i = \{t_i, f_i^1, f_i^2\}$, and, for every j, $1 \leq j \leq m-1$, we define $C_j^l = \{c_j, c_{j-1}, c_{j-1}^1, c_{j-1}^2\}$, $C_j^r = \{c_j, c_{j+1}, c_j^1, c_j^2\}$ and $C_j = C_j^l \cup C_j^r$.

We shall prove the desired structure of $\mathcal{R}$ by first considering the neighbourhood of c_j; once we have fixed the layout for this subgraph, the structure of the whole layout can be concluded inductively. The neighbourhood of c_j consists of C_j^l and C_j^r (two K_4 joined by c_j) and L_j, where all vertices of the two K_4 (except c_j) are not connected to any vertex of L_j. Intuitively speaking, this independence between L_j and the K_4 of the backbone will force the backbone to expand along one dimension, say horizontally (as depicted in Figure 6), while the visibilities between L_j and c_j must then be vertical (as depicted in Figure 7(a)). However, formally proving this turns out to be quite complicated.

The general proof idea is to somehow place the unit squares of R_{L_j} in such a way that they see R_{c_j} without creating unwanted visibilities. Then, the areas of visibility for the R_{L_j} are blocked for any unit squares from the backbone-neighbourhood R_{C_j}, since these are independent of R_{L_j}. For example, consider the situation depicted in Figure 9. Here, placing unit squares from R_{C_j} in the grey areas implies that they are within visibility of some unit squares from R_{L_j}. This leaves only few possibilities to place the unit squares from R_{C_j} and by applying arguments of this type, it can be concluded, by exhaustively searching all possibilities and under application of Lemma 15, that the only possible layouts have the above described form.

However, this argument is flawed: it is possible to place a unit square R_x within the grey areas, as long as the forbidden visibilities are blocked by other unit squares. This type of blocking would require a path between x and c_j or some vertex from L_j, respectively, which

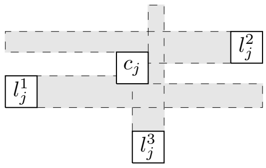

Figure 9 Possible placement of literal vertices for c_j.

does exist as structure in G. Consequently, in order to make the above described argument applicable, we first have to show that the existence of such visibility-blocking unit squares leads to a contradiction. This substantially increases the combinatorial depth of the already technical proof idea described above.

For the next lemma, which is the main tool in proving how the neighbourhood of c_j is represented, we need some notation. Let R_i, R_j, R_k be unit squares. If some (or every) visibility rectangle for R_i and R_k intersects R_j, then R_j is *strictly between* R_i and R_k (or R_j *blocks the view* between R_i and R_k, respectively).

▶ **Lemma 16.** *For all $1 \leq i \leq 2m$ and $r \in \{1, 2, 3\}$ and every $z \in N(c_i) \setminus \{l_i^r\}$ there exists no visibility rectangle for $R_{l_i^r}$ and R_z that is not intersected by R_{c_i}. In particular, this implies: R_z is not strictly between R_{c_i} and $R_{l_i^r}$, $R_{l_i^r}$ is not strictly between R_{c_i} and R_z, and, if R_{c_i} is strictly between $R_{l_i^r}$ and R_z, then R_{c_i} blocks the view between $R_{l_i^r}$ and R_z.*

By applying Lemma 16, we can now show that $R_{C_j^l}$ and $R_{C_j^r}$ cannot all see R_{c_j} from the same side, which can then be used in order to prove that either all R_{L_j} see R_{c_j} vertically or all of them see R_{c_j} horizontally:

▶ **Lemma 17.** *For every j, $1 \leq j \leq 2m - 1$ and $y \in C_j \setminus \{c_j\}$, $R_{c_j} \rightarrow R_{C_j \setminus \{y, c_j\}}$ is not possible.*

▶ **Lemma 18.** *For every j, $1 \leq j \leq m$, either $R_{c_j} \leftrightarrow R_{L_j}$ or $R_{c_j} \updownarrow R_{L_j}$.*

We are now able to combine these lemmas in order to prove that a layout for G translates into a not-all-equal satisfying assignment for the formula F. To this end, we first note that the neighbourhood of a variable vertex x_i has an identical structure as the neighbourhood of the clause vertices, which implies that Lemmas 16, 17 and 18 also apply to this part of the graph. By combining Lemmas 16 and 18, we can show that for each clause c_j, either $R_{c_j} \leftrightarrow R_{C_j \setminus \{c_j\}}$ or $R_{c_j} \updownarrow R_{C_j \setminus \{c_j\}}$. By Lemma 15, this means that the two corresponding induced K_4 are represented as shown in Figure 6, and, furthermore, an inductive application of Lemma 17 forces them to form the shown horizontal or vertical backbone. Due to Lemma 18, the literal vertices and the assignment vertices corresponding to the same variable must all form a path on the same side of the backbone. We can now assign x_i the value *true* if and only if R_{t_i} is below the backbone. As long as, for the variables occurring in some clause c_j, $R_{f_i^1}$ is on the opposite side of R_{t_i}, clause c_j is not-all-equal satisfied, because then literals are set to *true* if and only if they are below the backbone and, due to Lemma 16, it is not possible that they all lie on the same side. However, if $R_{f_i^1}$ lies on the same side as R_{t_i}, which is possible, then $R_{f_i^2}$, again due to Lemma 16, must lie on the opposite side of R_{t_i} and, by the same argument, it follows that c_{j+m}, which is a copy of c_j, is not-all-equal satisfied (note that every clause has at most one negated variable).

▶ **Lemma 19.** *If $G \in$ USV, then F is not-all-equal satisfiable.*

▶ **Theorem 20.** Rec(USV) *is* NP-*complete.*

Since in our reduction the size of the graph is linear in the size of the formula, we can also conclude ETH-lower bounds for Rec(USV).

5 Conclusions

The hardness of Rec($\mathsf{USV_w}$) is still open (note that in our reduction, we heavily used the argument that certain constellations yield forbidden edges, which falls apart in the weak case) and we conjecture it to be NP-hard as well. Two open problems concerning graph classes related to USGV are mentioned in Section 3: (1) are USGV and the class of resolution-$\frac{\pi}{2}$ graphs identical, (2) are there resolution-$\frac{\pi}{2}$ graphs without BRAC-drawing? Note that a positive answer to (2) gives a negative answer to (1).

From a parameterised complexity point of view, our NP-completeness result shows that the number of different rectangle shapes (considered as a parameter) has no influence on the hardness of recognition. Another interesting parameter to explore would be the step size of the grid, i.e., for $k \in \mathbb{N}$, let USGV^k be defined like USGV, but for a $\{\frac{\ell}{k} \mid \ell \in \mathbb{N}\}^2$ grid. We note that these classes form an infinite hierarchy between $\mathsf{USGV} = \mathsf{USGV}^1$ and $\mathsf{USV} = \bigcup_k \mathsf{USGV}^k$, and it is hard to define them in terms of extensions of rectilinear graphs. Another interesting observation is that the hardness reduction for the recognition problem of rectilinear graphs from [17], if interpreted as reduction for Rec(USGV), does not work for USGV^2. The classes USGV^k might be practically more relevant, since placing objects in the plane with discrete distances is more realistic.

Acknowledgements. We thank the organizers of the Lorentz Center workshop 'Fixed Parameter Computational Geometry' in 2016 for the great atmosphere that stimulated this project.

References

1 E. N. Argyriou, M. A. Bekos, and A. Symvonis. Maximizing the total resolution of graphs. In U. Brandes and S. Cornelsen, editors, *Graph Drawing, GD 2010*, volume 6502 of *LNCS*, pages 62–67. Springer, 2011.

2 E. N. Argyriou, M. A. Bekos, and A. Symvonis. The straight-line RAC drawing problem is NP-hard. *Journal of Graph Algorithms and Applications*, 16(2):569–597, 2012.

3 A. Arleo, C. Binucci, E. Di Giacomo, W. S. Evans, L. Grilli, G. Liotta, H. Meijer, F. Montecchiani, S. Whitesides, and S. K. Wismath. Visibility representations of boxes in 2.5 dimensions. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization - 24th International Symposium, GD*, volume 9801 of *LNCS*, pages 251–265. Springer, 2016.

4 M. Babbitt, J. Geneson, and T. Khovanova. On k-visibility graphs. *Journal of Graph Algorithms and Applications*, 19(1):345–360, 2015.

5 S. N. Bhatt and S. S. Cosmadakis. The complexity of minimizing wire lengths in VLSI layouts. *Information Processing Letters*, 25(4):263–267, 1987.

6 T. C. Biedl, G. Liotta, and F. Montecchiani. On visibility representations of non-planar graphs. In S. P. Fekete and A. Lubiw, editors, *32nd International Symposium on Computational Geometry, SoCG*, volume 51 of *LIPIcs*, pages 19:1–19:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

7 H. L. Bodlaender and G. Tel. A note on rectilinearity and angular resolution. *Journal of Graph Algorithms and Applications*, 8:89–94, 2004.

8 P. Bose, H. Everett, S. P. Fekete, M. E. Houle, A. Lubiw, H. Meijer, K. Romanik, G. Rote, T. C. Shermer, S. Whitesides, and C. Zelle. A visibility representation for graphs in three dimensions. *Journal of Graph Algorithms and Applications*, 2(2), 1998.

9 K. Casel, H. Fernau, A. Grigoriev, M L. Schmid, and S. Whitesides. Combinatorial properties and recognition of unit square visibility graphs, 2017. http://arxiv.org/abs/1706.05906.

10 S. Chaplick, G. Guśpiel, G. Gutowski, T. Krawczyk, and G. Liotta. The partial visibility representation extension problem. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization - 24th International Symposium, GD*, volume 9801 of *LNCS*, pages 266–279. Springer, 2016.

11 S. Chaplick, F. Lipp, J.-W. Park, and A. Wolff. Obstructing visibilities with one obstacle. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization - 24th International Symposium, GD*, volume 9801 of *LNCS*, pages 295–308. Springer, 2016.

12 A. M. Dean, J. A. Ellis-Monaghan, S. Hamilton, and G. Pangborn. Unit rectangle visibility graphs. *Electronic Journal of Combinatorics*, 15, 2008.

13 A. M. Dean, W. S. Evans, E. Gethner, J. D. Laison, M. A. Safari, and W. T. Trotter. Bar k-visibility graphs. *Journal of Graph Algorithms and Applications*, 11(1):45–59, 2007.

14 A. M. Dean and J. P. Hutchinson. Rectangle-visibility representations of bipartite graphs. *Discrete Applied Mathematics*, 75(1):9–25, 1997.

15 W. Didimo, P. Eades, and G. Liotta. Drawing graphs with right angle crossings. *Theoretical Computer Science*, 412(39):5156–5166, 2011.

16 P. Duchet, Y. Hamidoune, M. Las Vergnas, and H. Meyniel. Representing a planar graph by vertical lines joining different levels. *Discrete Mathematics*, 46(3):319–321, 1983.

17 P. Eades, S.-H. Hong, and S.-H. Poon. On rectilinear drawing of graphs. In D. Eppstein and E. R. Gansner, editors, *Graph Drawing, 17th International Symposium, GD 2009*, volume 5849 of *LNCS*, pages 232–243. Springer, 2010.

18 W. S. Evans, G. Liotta, and F. Montecchiani. Simultaneous visibility representations of plane st-graphs using L-shapes. *Theoretical Computer Science*, 645:100–111, 2016.

19 S. P. Fekete, M. E. Houle, and S. Whitesides. New results on a visibility representation of graphs in 3D. In F.-J. Brandenburg, editor, *Graph Drawing, Symposium on Graph Drawing, GD'95*, volume 1027 of *LNCS*, pages 234–241. Springer, 1996.

20 S. Felsner. Rectangle and square representations of planar graphs. In J. Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 213–248. Springer, New York, 2013.

21 M. Formann, T. Hagerup, J. Haralambides, M. Kaufmann, F. T. Leighton, A. Symvonis, E. Welzl, and G. J. Woeginger. Drawing graphs in the plane with high resolution. In *31st Annual Symposium on Foundations of Computer Science, FOCS, Volume I*, pages 86–95. IEEE Computer Society, 1990.

22 M. R. Garey and D. S. Johnson. *Computers and Intractability.* New York: Freeman, 1979.

23 E. Gaub, M. Rose, and P. S. Wenger. The unit bar visibility number of a graph. *Journal of Graph Algorithms and Applications*, 20(2):269–297, 2016.

24 E. Di Giacomo, W. Didimo, W. S. Evans, G. Liotta, H. Meijer, F. Montecchiani, and S. K. Wismath. Ortho-polygon visibility representations of embedded graphs. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization - 24th International Symposium, GD*, volume 9801 of *LNCS*, pages 280–294. Springer, 2016.

25 J. Maňuch, M. Patterson, S.-H. Poon, and C. Thachuk. Complexity of finding non-planar rectilinear drawings of graphs. In U. Brandes and S. Cornelsen, editors, *Graph Drawing - 18th International Symposium, GD 2010*, volume 6502 of *LNCS*, pages 305–316. Springer, 2011.

26 N. J. Nilsson. A mobile automaton: An application of artificial intelligence techniques. In D. E. Walker and L. M. Norton, editors, *Proceedings of the 1st International Joint Conference on Artificial Intelligence, IJCAI*, pages 509–520. William Kaufmann, 1969.

27 T. J. Schaefer. The complexity of satisfiability problems. In *Proc. 10th Ann. ACM Symp. Theory of Computing STOC*, pages 216–226. ACM Press, 1978.

28 I. Streinu and S. Whitesides. Rectangle visibility graphs: Characterization, construction, and compaction. In H. Alt and M. Habib, editors, *20th Annual Symposium on Theoretical Aspects of Computer Science, STACS*, volume 2607 of *LNCS*, pages 26–37. Springer, 2003.

29 R. Tamassia and I. G. Tollis. A unified approach to visibility representations of planar graphs. *Discrete & Computational Geometry*, 1(4):321–341, 1986.

30 S. K. Wismath. Characterizing bar line-of-sight graphs. In J. O'Rourke, editor, *Proceedings of the First Annual Symposium on Computational Geometry*, pages 147–152. ACM, 1985.