Morphing Graph Drawings in the Presence of Point Obstacles

Oksana Firman, Tim Hegemann, Boris Klemz, Felix Klesen, Marie Diana Sieper(✉), Alexander Wolff, and Johannes Zink

Institut für Informatik, Universität Würzburg, Würzburg, Germany
{oksana.firman,tim.hegemann,boris.klemz,felix.klesen,marie.sieper,johannes.zink}@uni-wuerzburg.de

Abstract. A crossing-free morph is a continuous deformation between two graph drawings that preserves straight-line pairwise noncrossing edges. Motivated by applications in 3D morphing problems, we initiate the study of morphing graph drawings in the plane in the presence of stationary point obstacles, which need to be avoided throughout the deformation. As our main result, we prove that it is NP-hard to decide whether such an obstacle-avoiding 2D morph between two given drawings of the same graph exists. This is in sharp contrast to the classical case without obstacles, where there is an efficiently verifiable (necessary and sufficient) criterion for the existence of a morph.

Keywords: Graph morphing · Point obstacles · NP-hard · Planar graph · Straight-line drawing

1 Introduction

In the field of Graph Drawing, a (*crossing-free*) *morph* between two straight-line drawings Γ_1 and Γ_2 of the same graph is a continuous deformation that transforms Γ_1 into Γ_2 while preserving straight-line pairwise noncrossing edges at all times. Morphing (beyond the above, strict definition in Graph Drawing) has applications in animation and computer graphics [12]. In this paper, we initiate the study of morphing graph drawings in the presence of stationary point obstacles, which need to be avoided throughout the motion.

Related Work. An obvious necessary condition for the existence of a crossing-free morph in $\mathbb{R}^2$ between two straight-line drawings Γ_1 and Γ_2 is that these drawings represent the same *plane graph* (i.e., a planar graph equipped with fixed combinatorial embedding and a distinguished outer face). It has been established long ago [7,16] that this (efficiently verifiable) criterion is also sufficient, i.e., a crossing-free morph in $\mathbb{R}^2$ between two straight-line drawings of the same plane graph always *exists*.

Work partially supported by DFG grants WO 758/9-1 and WO 758/11-1.

H. Fernau et al. (Eds.): SOFSEM 2024, LNCS 14519, pp. 240–254, 2024.
https://doi.org/10.1007/978-3-031-52113-3_17

More recent work [2,14] focuses on the efficient *computation* of such morphs. In particular, this involves producing a discrete description of the continuous motion. Typically, this is done in form of so-called piecewise linear morphs. In a *linear* morph, each vertex moves along a straight-line segment at a constant speed (which depends on the length of the segment) such that it arrives at its final destination at the end of the morph. The (unique) linear morph between Γ_1 and Γ_2 is denoted by $\langle \Gamma_1, \Gamma_2 \rangle$. A *piecewise linear* morph is created by concatenating several linear morphs, which are referred to as *(morphing) steps*. A piecewise linear morph consisting of k steps can be encoded as a sequence of $k+1$ drawings. Alamdari et al. [2] showed that two straight-line drawings of the same n-vertex plane graph always admit a crossing-free piecewise linear morph in $\mathbb{R}^2$ with $O(n)$ steps, which is best-possible. Their proof is constructive and corresponds to an $O(n^3)$-time algorithm, which was later sped up to $O(n^2 \log n)$ time by Klemz [14].

Other works are concerned with finding crossing-free morphs in $\mathbb{R}^2$ between two given drawings while preserving certain additional properties, such as convexity [3], upward-planarity [10], or edge lengths[1] [9], or with constructing crossing-free morphs in $\mathbb{R}^2$ that transform a given drawing to achieve certain properties, such as vertex visibilities [1] or convexity [13], while being in some sense monotonic, in order to preserve the so-called "mental map" [15] of the viewer.

Quite recent works [4–6] are concerned with transforming two drawings Γ_1, Γ_2 in the plane into each other by means of crossing-free morphs in the space $\mathbb{R}^3$. Such *2D–3D–2D* morphs are always possible [6]—even if Γ_1 and Γ_2 have different combinatorial embeddings—and they sometimes require fewer morphing steps than morphs that are restricted to the plane $\mathbb{R}^2$ [4,5]. Due to connections to the notoriously open Unknot problem, 3D–3D–3D morphs are not well understood and have, so far, only been considered for trees [4,5].

Our Model and Motivation. In this paper, we introduce and study a natural variant of the 2-dimensional morphing problem: given two crossing-free straight-line drawings Γ_1 and Γ_2 as well as a finite set of points P in $\mathbb{R}^2$, called *obstacles*, construct (or decide whether there exists) a crossing-free morph in $\mathbb{R}^2$ between Γ_1 and Γ_2 that *avoids* P, i.e., at no point in time throughout the deformation, the drawing is allowed to intersect any of the obstacles, which remain stationary. In particular, this problem naturally arises when constructing 2D–3D–2D morphs, where it is tempting to apply strategies for the classical 2-dimensional case on a subdrawing induced by the subset of the vertices contained in a plane π. Note that every edge between vertices on different sides of π intersects π in a point, which then acts as an obstacle for the 2-dimensional morph.

Contribution and Organization. We begin by stating some basic observations and preliminary results in Sect. 2. In particular, we observe that the necessary and sufficient condition for the classical case without obstacles is no longer sufficient

[1] In the fixed edge length scenario, the drawings are also known as linkages.

for our model (even when interpreting the obstacles as isolated vertices) and we present a stronger necessary condition (we say that the obstacles need to be "compatible" with the drawings), which is, however, still not sufficient. In fact, as our main result, we show that even if our condition is satisfied, it is NP-hard to decide whether an obstacle-avoiding morph exists (see Sect. 3):

Theorem 1. *Given a plane graph G, a set of obstacles P, and two crossing-free straight-line drawings Γ and Γ' in $\mathbb{R}^2$, it is NP-hard to decide whether there exists an obstacle-avoiding crossing-free morph in $\mathbb{R}^2$ between Γ and Γ'. The problem remains NP-hard when restricted to the case where G is connected, the drawings Γ and Γ' are identical except for the positions of four vertices, and the obstacles P are compatible with Γ and Γ'. (These statements hold regardless of whether the morph is required to be piecewise linear or not.)*

We remark that it is an essential part of the challenge to keep the edges straight-line during the morph – when dropping this requirement (i.e., when allowing edges to be deformed into arbitrary crossing-free curves or polylines), the problem can be solved efficiently [8]. The proof of Theorem 1 (by reduction from 3-SAT) is somewhat unusual from a Graph Drawing perspective: given a Boolean formula Φ, we describe the construction of a set of obstacles P and two (almost identical) drawings Γ_1, Γ_2 that exist irrespectively of the satisfiability of Φ, which instead corresponds to the existence of the obstacle-avoiding morph between the two drawings. In particular, we had to overcome the somewhat intricate challenge of designing gadgets that behave in a synchronous way. We conclude by discussing several open questions in Sect. 4. Claims marked with a clickable "$\star$" are proved in the full version of this article [11].

Conventions. In the remainder of the paper, we consider only morphs in the plane $\mathbb{R}^2$. We write "drawing" as a short-hand for "straight-line drawing in the plane $\mathbb{R}^2$" and, similarly, we write "(*planar*) morph" rather than "(crossing-free) morph in $\mathbb{R}^2$". For any positive integer n, we define $[n] = \{1, 2, \ldots, n\}$.

2 Preliminaries and Basic Observations

Let Γ_1 and Γ_2 be two drawings of the same plane graph G, and let P be a set of obstacles. We say that Γ_1 and Γ_2 are *blocked* by P if there is no planar morph between Γ_1 and Γ_2 that avoids P. Moreover, Γ_1 and Γ_2 are *blockable* if there exists a set of obstacles that blocks them. We start by observing that cycles are necessary to block drawings.

Proposition 1 ($\star$). *Let Γ_1 and Γ_2 be two drawings of the same plane forest F. Then Γ_1 and Γ_2 are not blockable.*

Proof (sketch). We describe how to construct a morph that avoids an arbitrary set of obstacles. Assume for now that F consists of a single tree, which we root at an arbitrary vertex r. We construct an obstacle-avoiding planar morph from Γ_1

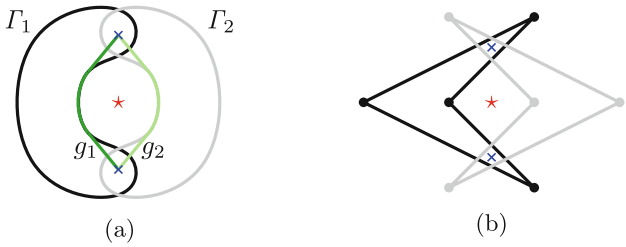


Fig. 1. The two simple closed curves Γ_1 and Γ_2 in (a) cannot be continuously deformed into each other without passing over one of the obstacles and while preserving simplicity. (The corresponding continuous deformation of the geodesic joining the two internal obstacles [blue crosses] within the closed curve would transform the curve g_1 into g_2 while keeping the endpoints fixed and without passing over the external obstacle [red star], which is impossible.) Consequently, the two drawings of a 4-cycle in (b) are blocked by the set of three obstacles; it is not compatible with the two drawings. (Color figure online)

to a drawing Γ_1' located in a disk that (i) is centered on the position of r in Γ_1, (ii) contains no obstacles, and (iii) whose radius is smaller than the distance between any pair of obstacles. This can be done by "contracting" the tree along its edges in a bottom-up fashion (i.e., starting from the leaves). We also morph Γ_2 into an analogously defined drawing Γ_2'. The drawings Γ_1' and Γ_2' can now be translated (far enough) away from the obstacles without intersecting them so that they can be transformed into each other by means of morphing techniques for the classical non-obstacle case [2,7,14,16].

If F contains multiple trees, it is easy to augment Γ_1 and Γ_2 to drawings of the same plane tree by inserting additional vertices and edges, thus reducing to the case of a single tree. □

We now turn our attention to the case when G may contain cycles. Recall that an obvious necessary condition for the existence of planar morph between two drawings is that they represent the same plane graph. Interpreting the obstacles in the set P as (isolated) vertices reveals that a planar morph between Γ_1 and Γ_2 that avoids P is possible only if each obstacle $p \in P$ is located in the same face in Γ_1 and Γ_2. However, as Fig. 1 shows, this condition is not sufficient. We say that P is *compatible* with Γ_1 and Γ_2 if there is a continuous deformation that transforms Γ_1 into Γ_2 while avoiding P and preserving pairwise noncrossing (*not necessarily straight-line*) edges at all times. The compatibility of P with Γ_1 and Γ_2 is obviously a necessary condition for the existence of a planar obstacle-avoiding morph. This condition can be checked efficiently [8]; note that it is violated in Fig. 1.

Compatibility is unfortunately still not sufficient for the existence of obstacle-avoiding morphs—even if the considered graph is just a cycle. In the following, we study and discuss this case in more detail. Let C_n denote the simple cycle with n vertices. Let Γ and Γ' be drawings of a plane C_n such that (i) Γ and Γ' are distinct (as mappings of C_n to the plane), but (ii) the closed curves realizing Γ and Γ' are identical, and (iii) the set of points of $\mathbb{R}^2$ used to represent vertices

is the same in Γ and Γ'. Note that there exists an offset $o \in [n-1]$ such that, for every $i \in [n]$, vertex i in Γ is at the same location as vertex $i+o$ (modulo n) in Γ'. Therefore, we say that Γ' is a *shifted version* of Γ. Due to (ii), *every* set of obstacles is compatible with Γ and Γ'.

Proposition 2 (⋆). *Let $n \geq 6$ be an even integer. Then there exists a drawing Γ of C_n such that, for every shifted version Γ' of Γ, the drawings Γ and Γ' are blockable by seven obstacles that are compatible with Γ and Γ' (for an illustration, see Fig. 2).*

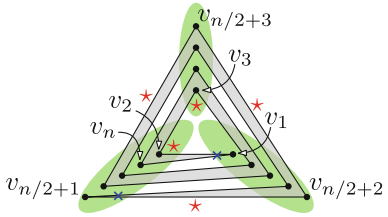


Fig. 2. A schematic drawing of C_n with vertices $v_1, \ldots, v_n$ (here $n = 14$) that cannot be morphed to a shifted version of it in a planar way while avoiding the two internal obstacles (blue crosses) and the five external obstacles (red stars). (Color figure online)

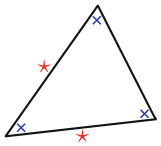


Fig. 3. Five obstacles suffice to block shifted versions of C_3.

It is plausible that Proposition 2 can be strengthened: even three obstacles seem to be sufficient for blocking two shifted drawings of an even-length cycle (we chose to use seven obstacles to simplify the proof). In contrast, mainly due to convexity, more obstacles are needed to block shifted drawings of C_3.

Proposition 3 (⋆). *Two drawings Γ_1 and Γ_2 of C_3 are not blockable by four obstacles that are compatible with Γ_1 and Γ_2.*

Proof (sketch). We perform a case distinction on the number of obstacles that are located in the interior of the cycle. Here, we only sketch the case with two inner obstacles $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$. Assume without loss of generality that $x_1 = x_2$ and $y_1 > y_2$. There exists an $\varepsilon > 0$ such that the rectangle $R = [x_1 - \varepsilon, x_1 + \varepsilon] \times [y_2 - \varepsilon, y_1 + \varepsilon]$ lies in the interiors of Γ_1 and Γ_2 and, hence, does not contain any of the outer obstacles. Then, for any triangle $\Gamma \in \{\Gamma_1, \Gamma_2\}$, there is a planar morph that moves the vertices of Γ onto the boundary of R while avoiding the line segment $\overline{p_1 p_2}$ and the region exterior to Γ. Finally, we show that two triangles with these properties can always be morphed into each other while staying in the closure of R and avoiding $\overline{p_1 p_2}$. □

Our proof for (at most) two inner obstacles does not depend on the number of outer obstacles, which implies a partially stronger statement.

We have a tight upper bound for the number of obstacles needed to block drawings of C_3; see Fig. 3.

Proposition 4 (⋆). *Let Γ be a drawing of C_3, and let Γ' be a shifted version of Γ. Then Γ and Γ' are blockable by five obstacles compatible with Γ and Γ'.*

We now state a sufficient condition for the existence of planar obstacles-avoiding morphs between shifted drawings of a cycle. We call a degree-2 vertex in a drawing *free* if its two incident edges lie on a common line.

Proposition 5 (⋆). *Let Γ be a drawing of C_n, and let Γ' be a shifted version of Γ. If Γ contains a free vertex, then Γ and Γ' are not blockable by obstacles that are compatible with Γ and Γ'.*

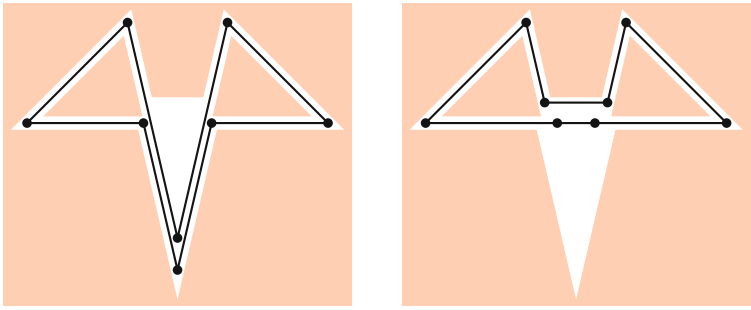


Fig. 4. Two drawings of C_8. If the shaded regions are densely filled with obstacles, the drawing on the left is essentially locked in place—it cannot be morphed planarly to a substantially different drawing without intersecting the obstacle regions. In particular, it cannot be morphed to the drawing on the right, even though this drawing contains two free vertices (and the obstacles are compatible with the two drawings).

Free vertices are helpful in other specific cases as well (in particular, they play a crucial role in our NP-hardness proof), but their usefulness is limited in general: their existence is not a sufficient condition for the existence of obstacle-avoiding morphs even when it comes to plane cycles; see Fig. 4.

Finally, we observe that two obstacles are not enough to block two drawings (with which the obstacles are compatible), regardless of the class of the represented graph.

Proposition 6 (⋆). *Let Γ_1 and Γ_2 be two drawings of the same plane graph G, and let P be a set of obstacles that are compatible with Γ_1 and Γ_2. If $|P| \leq 2$, then there exists a planar morph from Γ_1 to Γ_2 that avoids P.*

Proof (sketch). We interpret the (up to) two obstacles as isolated vertices of our plane graph (by compatibility, each obstacle belongs to the same face in Γ_1 and Γ_2). By known results [2,7,14,16], there is a morph M in which Γ_1 is deformed to Γ_2 without introducing crossings and without intersecting the obstacles. However, since the obstacles are treated as vertices, their positions might

change over time. We can transform M into the desired morph by translating, rotating, and scaling the frame of reference as time goes on to ensure that the obstacles become fixpoints (one can think of this as moving, rotating, and zooming the camera in a suitable fashion). □

In view of the remark after Proposition 2, it seems that Proposition 6 is best-possible. In particular, the proof strategy does not generalize to three obstacles as affine transformations preserve the cyclic orientations of point triples.

3 Proof of Theorem 1

In this section, we show our main result.

Theorem 1. *Given a plane graph G, a set of obstacles P, and two crossing-free straight-line drawings Γ and Γ' in $\mathbb{R}^2$, it is NP-hard to decide whether there exists an obstacle-avoiding crossing-free morph in $\mathbb{R}^2$ between Γ and Γ'. The problem remains NP-hard when restricted to the case where G is connected, the drawings Γ and Γ' are identical except for the positions of four vertices, and the obstacles P are compatible with Γ and Γ'. (These statements hold regardless of whether the morph is required to be piecewise linear or not.)*

Proof. We reduce from the classical NP-hard problem 3-SAT. Given a Boolean formula $\Phi = \bigwedge_{i=1}^{m} c_i$ in conjuctive normal form over variables $x_1, x_2, \ldots, x_n$ whose clauses $c_1, c_2, \ldots, c_m$ consist of three literals each, we construct a plane graph G, two planar drawings Γ and Γ' of G, and a set P of obstacles that are compatible with Γ and Γ'. We show that there exists an obstacle-avoiding planar morph from Γ to Γ' if and only if Φ is satisfiable.

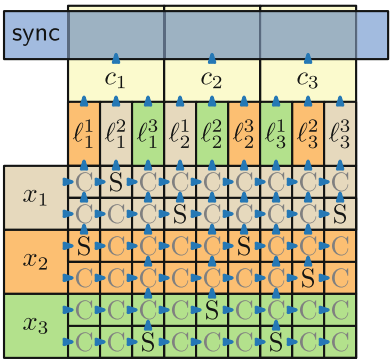


Fig. 5. General grid structure used in our NP-hardness reduction. Here, we use the formula $\Phi = c_1 \wedge c_2 \wedge c_3$, where $c_1 = (\ell_1^1 \vee \ell_1^2 \vee \ell_1^3) = (x_2 \vee x_1 \vee \neg x_3)$, $c_2 = (\ell_2^1 \vee \ell_2^2 \vee \ell_2^3) = (\neg x_1 \vee x_3 \vee x_2)$, and $c_3 = (\ell_3^1 \vee \ell_3^2 \vee \ell_3^3) = (\neg x_3 \vee \neg x_2 \vee \neg x_1)$. There are variable gadgets (left), clause and literal gadgets (top), split gadgets (S), crossing gadgets (C), and a synchronization gadget (sync) spanning over all clause gadgets. The gadgets have various states and orientations; dependencies are marked by triangular arrows.

Overview. In our reduction, we arrange obstacles, vertices, and edges such that we obtain a grid-like structure where we have two rows for each variable (one for each literal of the variable) and three columns for every clause (one for each literal in the clause); see Fig. 5. We then use several gadgets arranged within this grid-like structure. On the left side, the two rows of a variable terminate at a *variable gadget*. A variable gadget is in one of the states true, false, or unset. On the top side, the three columns of a clause are connected via three *literal gadgets* to a *clause gadget*. Each literal gadget and each clause gadget is in one of the states true or false. All clause gadgets are connected by a *synchronization gadget*. Within the column of each literal, we have a *split gadget* in one of the two rows of the corresponding variable x_i – in the upper row if the literal is x_i and in the lower row if the literal is $\neg x_i$. In all other grid cells, we have *crossing gadgets*. Every split gadget and every crossing gadget has a horizontal orientation (left/right) and a vertical orientation (bottom/top).

We first describe the general mechanism of our reduction. Independently of each other, every variable gadget can be morphed to reach any of its three states. In Γ and Γ', all crossing and split gadgets have the orientations left and bottom. A crossing or split gadget α can have orientation right only if the crossing/split gadget to the left of α has orientation right, too, or if α is adjacent to a variable gadget with state true (false) and α is in the upper (lower) row of the corresponding variable. Moreover, a crossing gadget can have orientation top only if the neighboring crossing/split gadget below has orientation top, too, and a split gadget can have orientation top only if it can also have orientation right. A literal gadget can be in the state true only if the gadget below it has orientation top. A clause gadget is in the state true if and only if at least one of its three literal gadgets is in the state true. Moreover, we can reach the final drawing Γ', where only the synchronization gadget differs from its drawing in Γ, if and only if all clause gadgets have state true simultaneously at some point in time. This ensures the correctness of our construction.

We now describe and visualize the geometric realization of our gadgets.

Forbidden Areas. In each gadget, we have *forbidden* areas where the vertices and edges cannot be drawn (henceforth drawn solid red). They are used to create a system of tunnels and cavities in which the edges of our drawings are placed and move; see, e.g., Fig. 6. We achieve this by densely filling the forbidden areas with obstacles, which are placed on a fine grid (as explained in more detail in the paragraph "Number and placement of obstacles" on page 12).

Variable Gadget. The variable gadget has a comparatively simple structure; see the three red boxes on the left side of Fig. 6. It has three vertices enclosed in a straight vertical tunnel of the forbidden area with one exit on the top right and one exit on the bottom right. As it is a straight tunnel, the middle vertex, which we call *decision vertex*, is a free (see Sect. 2 for the definition) vertex (see the thick blue–white vertex of x_3 in Fig. 6 for the arrangement in Γ and Γ'). We say that the variable gadget is in the state true (false) if this decision vertex

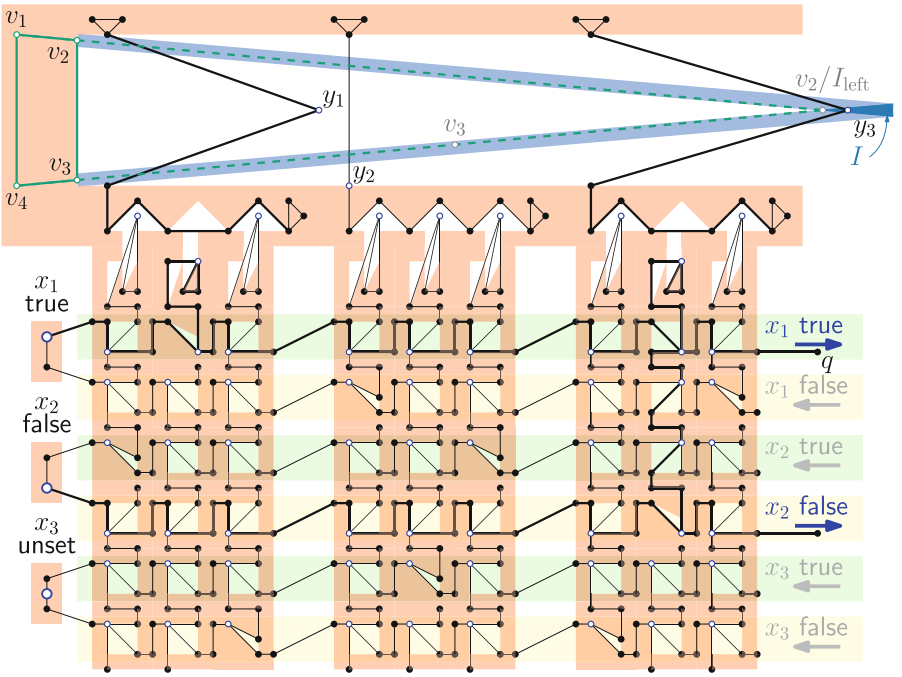


Fig. 6. Full construction for the instance $\Phi = (x_2 \vee x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_2) \wedge (\neg x_3 \vee \neg x_2 \vee \neg x_1)$. Gadgets with orientation right/top/true use thicker strokes.

is moved to the top (bottom) position, and it is in the state unset otherwise. Consequently, we can move the top (bottom) vertex out of the tunnel if and only if we are in the state true (false). This, in turn, yields a free vertex for the adjacent row of crossing and split gadgets, which we describe next.

Split Gadget. A split gadget consists of a central vertex c of degree 3 together with paths to the left, right, and top; see Fig. 7. They are enclosed in a system of tunnels formed by the forbidden area. Figure 7a shows a split gadget σ in the base state as it appears in Γ and Γ'. If the crossing/split/variable gadget to the left of σ, which shares the vertex l with σ, has a free[2] vertex, then l can be moved to the next corner of the tunnel. Then, in turn, c can be moved to the bottom right corner of the white (obstacle-free) triangle in σ (see Fig. 7b), and the two other neighbors of c can be pushed one position to the right and one position up. In this case we say that the horizontal (vertical) orientation of σ is right (top). Otherwise, the horizontal (vertical) orientation of σ is left (bottom).

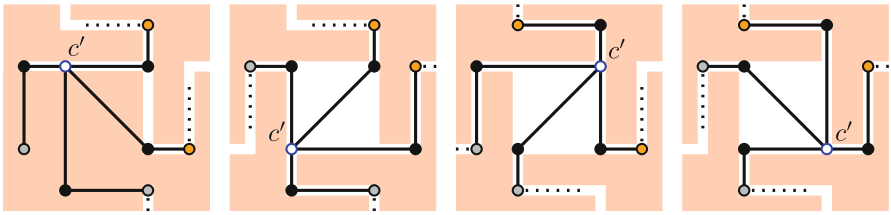
Crossing Gadget. A crossing gadget has a similar structure as a split gadget. However, we now have a central vertex c' of degree 4 with paths to the neigh-

[2] Here, this means that it is a crossing/split gadget that is oriented to the right or it is a variable gadget in its top row with state true or in its bottom row with state false.

(a) Base state with horizontal and vertical orientation left and bottom.

(b) If the gadget on the left has horizontal orientation right, vertices can be pushed one position further and we can reach the orientations right and top.

Fig. 7. Split gadget: from a horizontal row transporting a truth value, we "copy" the same truth value up to a vertical column.

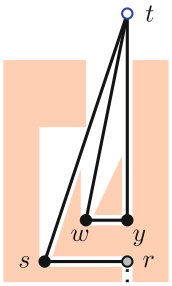


(a) left + bottom (b) right + bottom (c) left + top (d) right + top

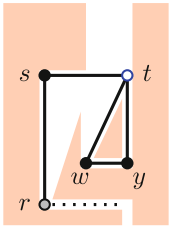
Fig. 8. Crossing gadget: we transport truth values horizontally and vertically without influencing each other. The four possible combinations of orientations are illustrated.

boring gadgets to the left, right, top, and bottom. In the center of the crossing gadget χ in Fig. 8, there is a white (obstacle-free) square. In Γ and Γ', c' is placed in the top left corner of this square (see Fig. 8a). This is the base state of χ. If the gadget on the left of χ has a free vertex, we can push the adjacent vertices to the next corners of the tunnel such that c' can move to the bottom side of the square (see Figs. 8b and 8d). Only then, we can push the vertices of the path leaving the gadget on the right side to the next corner of the tunnel. In this case we say that the horizontal orientation of the crossing gadget is right; otherwise it is left. Symmetrically, if the gadget below χ has a free vertex, we can move c' to the right side of the square (see Figs. 8c and 8d). Only then we can push the vertices of the path leaving χ through the top side to the next corner of the tunnel. In this case we say that the vertical orientation of the crossing gadget is top; otherwise it is bottom. Observe that the states of the gadgets on the left and below χ *independently* determine the horizontal and vertical orientation of χ. This property assures that we can transport information along routes that cross each other, but do not influence each other.

Literal Gadget. Figure 9 shows a literal gadget λ in its base state. It consists of five vertices, one of which, r, is shared with the gadget β below; see Fig. 9. Only if the vertical orientation of β is top, vertex r can move to the original position



(a) State false where vertex t is above the forbidden area of the gadget. This is also the base situation appearing in Γ and Γ'.



(b) State true where vertex t is inside the cavity formed by the forbidden area of the gadget. This state can only be reached if the split or crossing gadget below has orientation top.

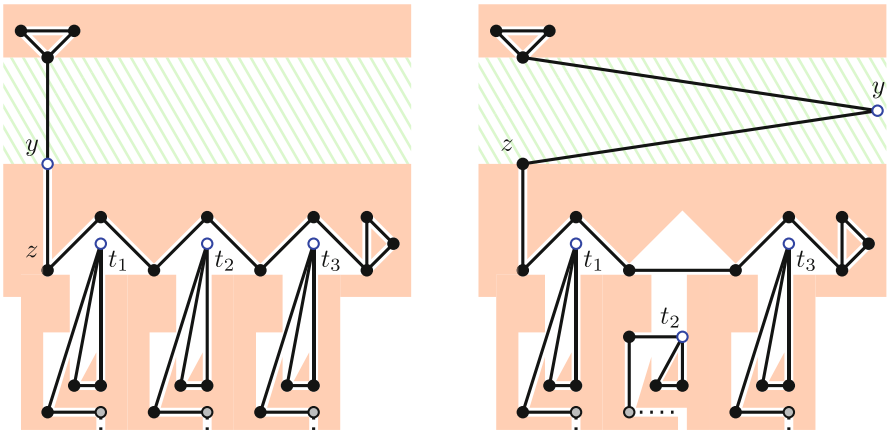
Fig. 9. Literal gadget: Depending on the crossing or split gadget below, it can be in two different states. Only in the state true, vertex t does not pop out of the gadget.

of s and s can move up. This in turn allows vertex t to move into the interior of λ. In this case, we say that λ has the state true; otherwise it has the state false. If a cycle in Γ contains obstacles, we call the cycle an *anchor*. Observe that the anchor $\langle t, w, y \rangle$ restricts the area where we can move t.

Clause Gadget. Each clause c_i ($i \in [m]$) is represented by a clause gadget, which consists of a path of length 9 whose endpoints are anchored by two 3-cycles; see Fig. 10. In the base situation occurring in Γ and Γ', we have exactly one free vertex (denoted by y), which is at the bottom of a large rectangular obstacle-free region, which is also part of the synchronization gadget. Observe that, within this area, we cannot move y (up to a tiny bit) to the left or right due to its neighbors lying at (essentially) fixed positions (see Fig. 10a).

However, if one of the incident literal gadgets is in the state true, we get a second free vertex, due to which we can move z, which is a neighbor of y, onto the base position of y (see Fig. 10b). Now we can move y arbitrarily far to the right within this obstacle-free region (unless y is blocked by the edges of another clause gadget). Only when this is done for all clause gadgets simultaneously, the synchronization gadget (see below) can be morphed as desired. Thus, we say that a clause gadget is in the state true if at least one of its literal gadgets is in the state true; otherwise it is in the state false.

Synchronization Gadget. The synchronization gadget is a 4-cycle $\langle v_1, v_2, v_3, v_4 \rangle$; see Fig. 6. In the base situation occurring in Γ, this cycle is drawn as an isosceles trapezoid T. In Γ', the 4-cycle is drawn as a shifted version (see Sect. 2 for the definition) of T. All sides of T except for the short parallel side are fixed by tunnels of the forbidden area. In particular, v_1 and v_4 can only be moved in an ε-region (for some small $\varepsilon > 0$) around their initial position, while v_2 and v_3 can potentially be moved to the right into the large obstacle-free rectangular



(a) The clause gadget is in the state false. (b) The clause gadget is in the state true.

Fig. 10. Clause gadget together with three literal gadgets: If at least one of the literal gadgets is in the state true, the clause gadget is also in the state true. The obstacle-free region shared with the synchronization gadget is depicted in hatched green (Color figure online).

area (that is shared with all clause gadgets) within a small corridor (see the blue strips in Fig. 6). These corridors extend the legs of T, which have a specific angle depending on the number of clauses m such that the corridors intersect at a region I to the right side of all clause gadgets. Note that we cannot simply "rotate" T, since each of the ε-regions around the positions of v_1 and v_4 in Γ need to contain at least one vertex at any time. To reach a shifted version, there needs to be an intermediate drawing that has a third vertex within the tunnel between these ε-regions. Therefore, only one vertex remains to close the 4-cycle on the right side. That vertex needs to be placed inside I. Hence, we can reach the shifted version if and only if all of the clause gadgets are in the state true: in this case, for each clause gadget, we can move its vertex y into I so that the edges incident to y do not intersect the triangle $\triangle v_1 I_{\text{left}} v_4$, where I_{left} is the leftmost point of I. Thus, we can place v_2 onto I_{left} to make v_3 a free vertex and, therefore, reach the shifted drawing of the 4-cycle (cf. Proposition 5). In contrast, if a clause gadget is in the state false, at least one of the edges incident to its vertex y intersects $\triangle v_1 I_{\text{left}} v_4$ and, thus, v_2 and v_3 cannot reach the corridors of each other.

Correctness. In summary, if Φ admits a satisfying truth assignment, we can describe a (piecewise linear) morph from Γ to Γ' by moving the decision vertices in the variable gadgets according to the truth assignment, which allows to transport these truth values via the split and crossing gadgets to the literal and the clause gadgets. As all clause gadgets can reach the state true at the same

time, we can morph the drawing of the 4-cycle in the synchronization gadget to its shifted version and then move all other vertices back to their original position.

For the other direction, if a morph from Γ to Γ' exists, then we know by our construction that at some point, all clause gadgets were in the state `true` simultaneously (as this is necessary to morph the drawing of the 4-cycle in the synchronization gadget to its shifted version), which means the variable gadgets represent at the same time a satisfying truth assignment for Φ.

Number and Placement of Obstacles. While the grid-like *arrangement* of our gadgets depends on Φ, the *design* of the variable/clause/literal/crossing/split gadgets does not. Therefore, each of these gadgets uses only $O(1)$ obstacles, the overall number of obstacles then is $O(nm)$, and the encoding of their coordinates requires only polynomially many bits in total. Similarly, the synchronization gadget also uses only $O(1)$ obstacles, which, for the most part, can be placed on the grid that is also used by the remaining gadget types. As an exception, two obstacles (in the top-right and bottom-right corner of the obstacles area interior to the 4-cycle) need to be placed on a refined grid to ensure that the area I lies to the right of all clause gadgets. Since vertices v_1 and v_2 have constant (horizontal) distance, we use an obstacle that lies $O(1/m)$ units below the highest possible position of v_1 and on the same x-coordinate as v_2 in Γ in order to bound the slope of the v_1v_2-tunnel from below. We bound the slope of the v_3v_4-tunnel symmetrically from above. As a result, the area I lies $\Omega(m)$ units to the right of the vertical line segment $\overline{v_2v_3}$. Since the width of the whole construction is $O(m)$, this suffices.

Thus, the encoding of the coordinates of the involved obstacles and the coordinates of the drawing of the 4-cycle requires only polynomially many bits.

Connectivity. So far, the graph in our reduction has $\Omega(m)$ connected components: a large connected component comprising all variable, split, crossing and literal gadgets, a connected component for all clause gadgets, as well as another connected component for the synchronization gadget. We can merge these components by adding edges without influencing the behavior of our gadgets: We add the edges v_2y_1, y_1y_2, $\ldots$, $y_{m-1}y_m$ together with a path from y_m to the large connected component of the other gadgets (e.g., in Fig. 6 from y_3 to q). □

4 Open Problems

1. In general, the considered decision problem is NP-hard (Theorem 1), but it can be solved efficiently for forests (Proposition 1). Are there other meaningful graph classes where this is the case? In particular, what about cycles or triangulations? It is conceivable that the latter case is actually easier since the placement of obstacles that are compatible with the two given drawings is quite limited. Regarding cycles, we emphasize that the existence of a free vertex is not a sufficient condition for the existence of a morph (cf. Fig. 4).
2. Does the problem lie in NP? Is it $\exists\mathbb{R}$-hard?

3. Does the problem become easier if there are only constantly many obstacles?
4. The drawings Γ and Γ' produced by our reduction are identical except for the position of four vertices. It seems quite plausible that our construction can be modified so that only three vertices change positions. Does the problem become easier when only up to two vertices may change positions?
5. It is easy to observe that if our reduction is applied to a satisfiable formula Φ with n variables and m clauses, there is a piecewise linear obstacle-avoiding morph between the produced drawings Γ and Γ' with $\Theta(n+m)$ steps, which is also necessary. Note that this number is not independent from the output size of the reduction. This motivates the following family of questions. Let k be a fixed arbitrary constant. Given two planar straight-line drawings Γ and Γ' of the same plane graph and a set of obstacles compatible with Γ and Γ', decide whether there exists a piecewise linear obstacle-avoiding morph from Γ to Γ' with at most k steps. For which values of k can this decision problem be answered efficiently?
6. Given two drawings of the same plane graph, how many compatible obstacles are necessary and sufficient to block them? Can this be computed efficiently?

References

1. Aichholzer, O.: Convexifying polygons without losing visibilities. In: Aloupis, G. Bremner, (eds.) CCCG, pp. 229–234 (2011). http://www.cccg.ca/proceedings/2011/papers/paper70.pdf
2. Alamdari, S., et al.: How to morph planar graph drawings. SIAM J. Comput. **46**(2), 824–852 (2017). https://doi.org/10.1137/16M1069171
3. Angelini, P., Da Lozzo, G., Frati, F., Lubiw, A., Patrignani, M., Roselli, V.: Optimal morphs of convex drawings. In: Arge, L., Pach, J., (eds.) SoCG. LIPIcs, vol. 34 pp. 126–140. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2015). https://doi.org/10.4230/LIPIcs.SOCG.2015.126
4. Arseneva, E., et al.: Pole dancing: 3D morphs for tree drawings. J. Graph Algorithms Appl. **23**(3), 579–602 (2019). https://doi.org/10.7155/jgaa.00503
5. Arseneva, E., Gangopadhyay, R., Istomina, A.: Morphing tree drawings in a small 3D grid. J. Graph Algorithms Appl. **27**(4), 241–279 (2023). https://doi.org/10.7155/jgaa.00623
6. Buchin, K., et al.: Morphing planar graph drawings through 3D. In: Gąsieniec, L. (ed.) SOFSEM 2023. LNCS, vol. 13878, pp. 80–95. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-23101-8_6
7. Cairns, S.: Deformations of plane rectilinear complexes. Am. Math. Monthly **51**(5), 247–252 (1944)
8. Colin de Verdiére, É., de Mesmay, A.: Testing graph isotopy on surfaces. Discrete Comput. Geom. **51**, 171–206 (2014). https://doi.org/10.1007/s00454-013-9555-4
9. Connelly, R., Demaine, E.D., Rote, G.: Straightening polygonal arcs and convexifying polygonal cycles. Discrete Comput. Geom. **30**, 205–239 (2003). https://doi.org/10.1007/s00454-003-0006-7
10. Da Lozzo, G., Di Battista, G., Frati, F., Patrignani, M., Roselli, V.: Upward planar morphs. Algorithmica **82**(10), 2985–3017 (2020). https://doi.org/10.1007/s00453-020-00714-6

11. Firman, O., et al.: Morphing graph drawings in the presence of point obstacles. ArXiv report (2023). http://arxiv.org/abs/2311.14516
12. Gomes, J., Darsa, L., Costa, B., Velho, L.: Warping & Morphing of Graphical Objects. Morgan Kaufmann, Burlington (1999)
13. Kleist, L., Klemz, B., Lubiw, A., Schlipf, L., Staals, F., Strash, D.: Convexity-increasing morphs of planar graphs. Comput. Geom. **84**, 69–88 (2019). https://doi.org/10.1016/j.comgeo.2019.07.007
14. Klemz, B.: Convex drawings of hierarchical graphs in linear time, with applications to planar graph morphing. In: Mutzel, P., Pagh, R., Herman, G., (eds.) ESA, LIPIcs, vol. 204, pp. 57:1–57:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPIcs.ESA.2021.57
15. Purchase, H.C., Hoggan, E., Görg, C.: How important is the "mental map"? – an empirical investigation of a dynamic graph layout algorithm. In: Kaufmann, M., Wagner, D. (eds.) GD 2006. LNCS, vol. 4372, pp. 184–195. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70904-6_19
16. Thomassen, C.: Deformations of plane graphs. J. Combin. Theory Ser. B **34**(3), 244–257 (1983). https://doi.org/10.1016/0095-8956(83)90038-2