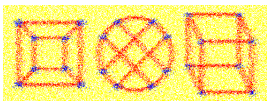

Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 27, no. 4, pp. 241–279 (2023)
DOI: 10.7155/jgaa.00623

Morphing tree drawings in a small 3D grid

Elena Arseneva[1] *Rahul Gangopadhyay*[2] *Aleksandra Istomina*[3]

[1]Università della Svizzera italiana, Lugano, Switzerland
[2]Moscow Institute of Physics and Technology, Dolgoprudny, Moscow Region, Russian Federation
[3]Saint-Petersburg State University, Saint Petersburg, Russia

Submitted: July 2022 Reviewed: November 2022 Revised: January 2023
Reviewed: March 2023 Revised: March 2023 Accepted: April 2023
Final: May 2023 Article type: Regular paper
Communicated by: M.S. Rahman, P. Mutzel, Slamin

Abstract. We study crossing-free grid morphs for planar drawings of trees using the third dimension. A morph consists of morphing steps, where vertices of the tree move simultaneously along straight-line trajectories at constant speeds. The aim is to find a morph between two given drawings of the same tree that has a small number of steps and is *crossing-free*, i.e. no two vertices overlap or no two edges of the tree intersect except in their common endpoints at any moment during the morph. It is already known, that there exists a crossing-free morph between two drawings of an n-vertex planar graph G with $\mathcal{O}(n)$ morphing steps, and that using the third dimension the number of steps can be reduced to $\mathcal{O}(\log n)$ for an n-vertex tree. However, these morphs do not bound one practical parameter, the resolution. Can the number of steps still be reduced substantially by using the third dimension with an additional restriction of keeping the resolution bounded throughout the morph? We answer this question in affirmative by presenting a $3D$ non-crossing morph between two planar grid drawings of an n-vertex tree in $\mathcal{O}(\sqrt{n}\log n)$ morphing steps such that each intermediate drawing is a grid drawing, i.e., vertices of the tree are mapped to the nodes of a $3D$ grid of polynomial volume.

Special Issue on the 16th Int. Conference and Workshops on Algorithms and Computation, WALCOM 2022

Research was mostly conducted while Elena Arseneva and Rahul Gangopadhyay were at Saint-Petersburg State University. Elena Arseneva was partially supported by the Foundation for the Advancement of Theoretical Physics and Mathematics "BASIS". Elena Arseneva and Aleksandra Istomina were partially supported by RFBR, project 20-01-00488. Rahul Gangopadhyay was supported by Ministry of Science and Higher Education of the Russian Federation, agreement no. 075-15-2022-287. A preliminary version of this paper appeared in [12].

E-mail addresses: elena.arseneva@usi.ch (Elena Arseneva) rahulg@iiitd.ac.in (Rahul Gangopadhyay) st062510@student.spbu.ru (Aleksandra Istomina)

1 Introduction

Given an n-vertex graph G, a *morph* between two drawings (i.e., embeddings in $\mathbb{R}^d$) of G is a continuous transformation from one drawing to the other through a sequence of intermediate drawings. One is interested in well-behaved morphs, i.e., those that preserve essential properties of the drawing at any moment. Usually, this property is that the drawing is *crossing-free*; such morphs are called *crossing-free* morphs. This concept finds applications in multiple domains: animation, modeling and computer graphics, etc. [7–9]. A drawing of G is a *straight-line drawing* if it maps each vertex of G to a point in $\mathbb{R}^d$ and each edge of G to the line segment whose endpoints correspond to the endpoints of this edge. In this work, we focus on the case of drawings in the Euclidean plane ($d = 2$) and $3D$ drawings ($d = 3$); a crossing-free drawing of a graph in $\mathbb{R}^2$ is called *planar*.

There is an interest in studying crossing-free morphs of straight-line drawings, where vertex trajectories are simple, in particular, *linear morphs*. In every step of a linear morph, each vertex of G moves along a straight-line segment at a constant speed in each step of a linear morph. In other words, a linear morph transforms one straight-line drawing Γ of a graph G to another such drawing Γ' through a sequence $\langle \Gamma = \Gamma_1, \Gamma_2, \ldots, \Gamma_k = \Gamma' \rangle$ of straight-line drawings; each *morphing step* or *step*, for brevity, is a linear interpolation between the vertices of two consecutive drawings in that sequence, see Fig. 2. A linear morph is said to be *unidirectional* if all vertices move along parallel lines in the same direction. Alamdari et al. [1] showed that for any two topologically equivalent planar drawings of a graph G, there is a linear $2D$ morph that transforms one drawing to the other in $\Theta(n)$ steps. Alamdari et al. [1] also showed the bound to be asymptotically optimal in the worst case by creating an example where a linear $2D$ morph between two planar drawings of an n-vertex path needs $\Omega(n)$ morphing steps. A natural further question is how the situation changes when we involve the third dimension. For general $3D$ graph drawings, the problem seems challenging since it is tightly connected to the *unknot recognition* problem, that is in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ [10, 13], and its containment in $\mathbf{P}$ is wide open. If the given graph is a tree, the worst-case tight bound of $\Theta(n)$ steps holds for $3D$ crossing-free linear morph [2], and the lower-bound example is again a path. If both the initial and the final drawings are planar, then $\mathcal{O}(\log n)$ steps suffice [2].

Both algorithmic results [1, 2] have a drawback crucial from the practical point of view. Their intermediate steps use infinitesimal or very small distances, as compared to distances in the input drawings. This may blow up the space requirements and affect the aesthetical aspect. This issue raises a demand for morphing algorithms that operate on a small grid, i.e., on a grid of size polynomial in the size of the graph and parameters of the input drawings. All the intermediate drawings are then restricted to be *grid drawings*, where vertices map to nodes of the grid, see Fig. 1. Two crucial parameters of a straight-line grid drawing are:

- The *volume* of the required grid in 3D. In case of 2D drawings, the *area* is considered instead.
- The *resolution*, that is the ratio between the maximum edge length and the minimum vertex-edge distance.

If the grid area is polynomially bounded, then so is the resolution [3]. For the $3D$ case, if the grid volume is polynomially bounded, then so is the resolution, see Lemma 1. A point in $\mathbb{R}^3$ is said to be an *integer* point if all its coordinates are integer.

Very recently Barrera-Cruz et al. [3] gave an algorithm that linearly morphs between two planar straight-line grid drawings Γ and Γ' of an n-vertex rooted tree in $\mathcal{O}(n)$ steps while each intermediate drawing is also a planar straight-line drawing in a bounded grid. In particular, the maximum grid length and width are respectively $\mathcal{O}(D^3 n \cdot L)$ and $\mathcal{O}(D^3 n \cdot W)$, where $L = max\{l(\Gamma), l(\Gamma')\}$,

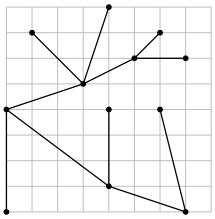


Figure 1: A planar straight-line grid drawing of a 12-vertex tree.

$W = max\{w(\Gamma), w(\Gamma')\}$ and $D = max\{L, W\}$, $l(\Gamma)$ and $w(\Gamma)$ are the *length* and the *width* of the drawing Γ respectively. Note that D is $\Omega(\sqrt{n})$.

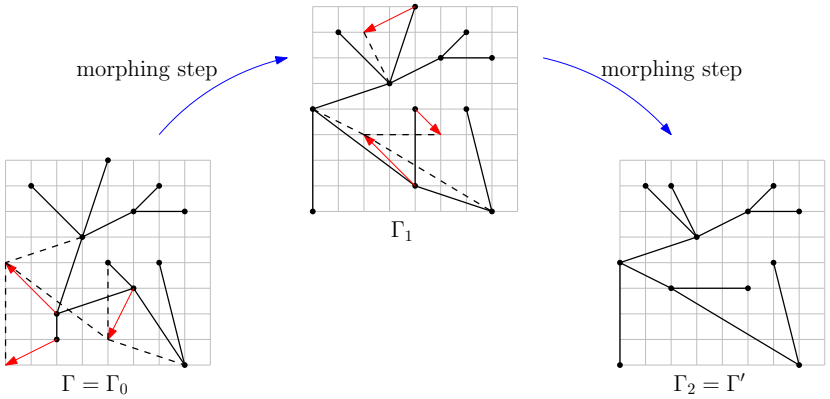


Figure 2: A linear morph consisting of 2 morphing steps. In the drawings Γ_i with $i = 0, 1$, dashed lines in drawing represent the positions of the edges in drawing Γ_{i+1}, the next drawing in the sequence, and the red arrows represent the directions of movement of the vertices during the morphing step $\langle \Gamma_i, \Gamma_{i+1} \rangle$.

Let Γ_0 and Γ' be two planar straight-line drawings of an n-vertex tree T. Throughout this paper, a morph $\mathcal{M} = \langle \Gamma_1, \Gamma_2, \ldots, \Gamma_k \rangle$ of T is a sequence of 3D straight-line drawings of T such that $\Gamma_1 = \Gamma_0$ and $\Gamma_k = \Gamma'$ are respectively the initial and the final drawings, and each step $\langle \Gamma_i, \Gamma_{i+1} \rangle$ is a linear morph, see Fig. 2. Here, we study morphing one straight-line grid drawing Γ of a tree to another such drawing Γ' in *sublinear* number of steps using the third dimension such that the resolutions of the intermediate drawings are bounded. Our algorithm requires the underlying tree T to be rooted, and its worst-case performance does not depend on the exact placement of the root. Thus we fix an arbitrary vertex of T to be its root and from now on treat T to be a rooted tree.

We morph the initial planar drawing of tree T to its $3D$ canonical drawing $\mathcal{C}(T)$ and then analogously morph $\mathcal{C}(T)$ to the final planar drawing. Effectively we solve the same problem as in [2], but with the additional restriction that all drawings throughout the algorithm lie in a small grid. We give an algorithm that requires $\mathcal{O}(\sqrt{n} \log n)$ steps. All the intermediate drawings require a $3D$ grid of length $\mathcal{O}(d^3(\Gamma) \cdot \log n)$, width $\mathcal{O}(d^3(\Gamma) \cdot \log n)$ and height $\mathcal{O}(n)$, where $d(\Gamma)$ denotes

the maximum between the diameters of the initial drawing Γ_0 and the final drawing Γ'. Diameter of a drawing of a tree T is defined as the ceiling of the maximum pairwise (Euclidean) distance between the vertices of the tree in the drawing. We make use of known techniques, e.g., canonical drawing [2] and "Pinwheel" rotation [3] combined with new ideas.

After introducing the necessary definitions and preliminaries in Sec. 2, we describe the tools that are the building blocks of our algorithm: stretching, mapping around the pole, rotating and shrinking subtrees, see Sec. 3. In Sec. 4.1, we introduce a technique to lift a plane tree drawing to $\mathbb{R}^3$ such that the vertices of a certain path of the tree go to their respective canonical positions, their subtrees move along with them, and the morph is linear and crossing-free. We call it *lifting a path*, for brevity. The morphing algorithm in Sec. 4 splits the given tree into disjoint paths that are lifted one by one in specific order. Since lifting each path takes constant number of steps, in the worst case this algorithm takes $\mathcal{O}(n)$ steps to lift a tree. In Sec. 5.1, we show how to lift a set of edges of the given tree simultaneously. This is used in the second morphing algorithm, that lifts the tree by lifting disjoint sets of its edges one after another, see Sec. 5. This algorithm takes $\mathcal{O}(h)$ steps to lift a tree of height h. We then combine two algorithms in Sec. 6 to produce the final algorithm that uses $o(n)$ morphing steps. It first lifts all paths of T of length at most $\sqrt{n}$ using the algorithm of Sec. 5. Since the total number of remaining paths is less than $\sqrt{n}$, it lifts them one after another by using the algorithm of Sec 4.

2 Preliminaries and Definitions

For a point $p = (p_x, p_y, p_z)$ in $\mathbb{R}^3$, let YZ_p, XZ_p, XY_p denote planes $x = p_x, y = p_y, z = p_z$ respectively. Analogously, XZ_p^+ (resp., XZ_p^-) denotes the vertical halfplane $\{(x, y, z) : y = p_y, x \geq p_x (resp., x \leq p_x)\}$ and YZ_p^+ (resp., YZ_p^-) the halfplane $\{(x, y, z) : x = p_x, y \geq p_y (resp., y \leq p_y)\}$. Let XY_0 denote the *horizontal plane* passing through the origin. For brevity, we call it the horizontal plane throughout the paper.
For a point $p = (p_x, p_y, p_z)$ in $\mathbb{R}^3$, consider its vertical projection $p' = (p_x, p_y)$ to the horizontal plane. For the sake of brevity, we denote this vertical projection by $pr()$.

Tree drawings. For a tree T, let $r(T)$ be its root, and $T(v)$ be the subtree of T rooted at a vertex v. Let $V(T)$ and $E(T)$ be respectively the set of vertices and edges of T. The size of a tree T, denoted by $|T|$, is the number of vertices in T. The depth $dpt(v)$ of a vertex v in T is the length of the path from the root $r(T)$ to v.

In a *straight-line drawing* of T, each vertex is mapped to a point in $\mathbb{R}^d$ and each edge is mapped to a closed straight-line segment connecting the images of its end-points. A $3D$- (respectively, a $2D$-) *grid drawing* of T is a straight-line drawing where each vertex is mapped to a point with integer coordinates in $\mathbb{R}^3$ (respectively, $\mathbb{R}^2$). A drawing of T is said to be *crossing-free* if the images of no two edges intersect except, possibly, at common end-points. A crossing-free $2D$-grid drawing is called a *planar grid drawing*, see Fig. 1.

For a crossing-free drawing Γ, let $B(\Gamma(v), r)$ denote the open disk of radius r in the horizontal plane centered at $\Gamma(v)$ that is the image of the vertex v in Γ. Let $l(\Gamma)$, $w(\Gamma)$ and $h(\Gamma)$ respectively denote the *length*, *width* and *height* of the $3D$ drawing Γ of T, i.e., the maximum absolute difference between the x-, y- and z-coordinates of vertices in Γ. Recall that $d(\Gamma)$ denotes the *diameter* of Γ, defined as the ceiling of the maximum pairwise (Euclidean) distance between its vertices. Note that $d(\Gamma)$ estimates the space required by Γ since $M \leq d(\Gamma) \leq \sqrt{3}M$, where $M = \max(l(\Gamma), w(\Gamma), h(\Gamma))$. See Fig 3.

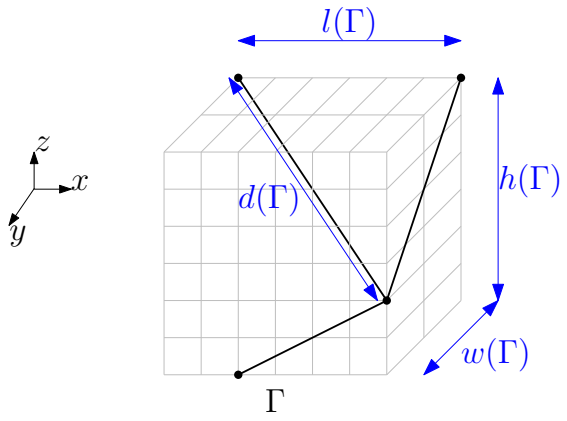


Figure 3: Example of $l(\Gamma), w(\Gamma), h(\Gamma), d(\Gamma)$ in 3D straight-line grid drawing.

We define $dist_\Gamma(v,u)$ to be the Euclidean distance between two points $\Gamma(v)$ and $\Gamma(u)$ that are the images of vertices v,u of T in Γ. Let $dist_\Gamma(v,e)$ be the shortest Euclidean distance between the point $\Gamma(v)$ and the line segment $\Gamma(e)$. Similarly, $dist_\Gamma(e_1,e_2)$ is defined as the shortest Euclidean distance between two line segments $\Gamma(e_1)$ and $\Gamma(e_2)$ which are the images of edges e_1,e_2 of T in Γ, respectively. Precisely we define $dist_\Gamma(e_1,e_2) = \min\{\|x-y\| \mid x \in \Gamma(e_1), y \in \Gamma(e_2)\}$.

For a grid drawing Γ, we define the *resolution* of Γ as the ratio of the distances between the farthest and closest pairs of non-adjacent and non-incident geometric objects of Γ (images of tree vertices and edges). Since our tree T is labeled, when it is clear from the context, we, slightly abusing the notation, refer to the image of elements of T (vertices, edges or subtrees) in a certain drawing simply by the name of these elements, e.g., v instead of $\Gamma(v)$.

Lemma 1 *1. For any vertex v_0 and edge e not incident to v_0 in a 3D crossing-free grid drawing $\Gamma(T)$ of T (also, in a planar grid drawing), $dist_\Gamma(v_0,e) \geq \frac{1}{d(\Gamma)}$.*

2. For a pair of non-adjacent edges e_1,e_2, the distance $dist_\Gamma(e_1,e_2) \geq \dfrac{1}{2\sqrt{3}\,(d(\Gamma))^2}$ in a 3D crossing-free grid drawing $\Gamma(T)$ of T.

Proof:

1. Let v_1 and v_2 be the endpoints of edge e, and v_0 does not lie inside e as $\Gamma(T)$ is non-crossing. If v_0,v_1,v_2 are collinear, then $dist_\Gamma(v_0,e) \geq 1$ since each vertex is mapped to a point with an integer coordinate. Let us assume that the points are not collinear. Let the coordinates of points be $v_0=(x_0,y_0,z_0), v_1=(x_1,y_1,z_1)$ and $v_2=(x_2,y_2,z_2)$. The area of the triangle spanned by the three points v_0,v_1,v_2 is $\dfrac{|\overrightarrow{v_0v_1} \times \overrightarrow{v_0v_2}|}{2}$, where
$$\overrightarrow{v_0v_1} \times \overrightarrow{v_0v_2} = \begin{vmatrix} \hat{\mathbf{i}} & \hat{\mathbf{j}} & \hat{\mathbf{k}} \\ x_1-x_0 & y_1-y_0 & z_1-z_0 \\ x_2-x_0 & y_2-y_0 & z_2-z_0 \end{vmatrix}.$$
Since all the points have integer coordinates and they are not collinear, $|\overrightarrow{v_0v_1} \times \overrightarrow{v_0v_2}| \geq 1$. This implies that the area of the triangle is at least $\dfrac{1}{2}$. On the other hand, the area of

the same triangle is $\frac{1}{2} \cdot dist_\Gamma(v_0, e) \cdot dist_\Gamma(v_1, v_2) \leq \frac{1}{2} \cdot dist_\Gamma(v_0, e) \cdot d(\Gamma)$. This implies that $dist_\Gamma(v_0, e) \geq \frac{1}{d(\Gamma)}$. Note that if $\Gamma(T)$ is a planar grid drawing of T, the values of z_0, z_1 and z_2 are zero, and the same argument holds.

2. Let v_1 and v_2 be the endpoints of edge e_1, and v_3, v_4 be the endpoints of edge e_2. If these four points lie on the same plane, the problem reduces to the previous one. We assume that these four points do not lie on the same plane. Let the coordinates of points be $v_1 = (x_1, y_1, z_1), v_2 = (x_2, y_2, z_2), v_3 = (x_3, y_3, z_3)$ and $v_4 = (x_4, y_4, z_4)$. Then, the volume of the tetrahedron spanned by these four points is given by $\frac{dist(e_1, e_2) \cdot |\overrightarrow{v_1v_2} \times \overrightarrow{v_3v_4}|}{6}$. On the other hand the volume of the same tetrahedron is given by the following determinant.

$$\frac{1}{6} \cdot \left| \begin{vmatrix} x_1 - x_2 & y_1 - y_2 & z_1 - z_2 \\ x_2 - x_3 & y_2 - y_3 & z_2 - z_3 \\ x_3 - x_4 & y_3 - y_4 & z_3 - z_4 \end{vmatrix} \right|$$

This implies that the volume of the tetrahedron spanned by these four vertices is at least $\frac{1}{6}$, since the points have integer coordinates. This also implies that $dist_\Gamma(e_1, e_2) \geq \frac{1}{|\overrightarrow{v_1v_2} \times \overrightarrow{v_3v_4}|} \geq \frac{1}{2\sqrt{3}\,(d(\Gamma))^2}$. □

Path decomposition $\mathcal{P}$ of a tree T is a decomposition of its edges into a set of disjoint paths, defined constructively, as follows. Choose some root-to-leaf path in T and store it in an ordered set $\mathcal{P}$ which is empty at the beginning. Remove the edges of this path from T. Note that the vertices on the path whose residual degree remains greater than zero after the removal of the edges are not removed. It may disconnect the tree; recurse on the remaining connected components while there are edges. In the end, $\mathcal{P}$ is an ordered set of disjoint paths whose union is $E(T)$. *Head* of the path P, denoted as $head(P)$, is the vertex $x \in P$ with the minimum depth in tree T. Let the *internal vertices* of the path be all vertices except $head(P)$. Any path decomposition $\mathcal{P}$ of T induces a linear order of the paths: path P' succeeds P, i.e., $P' \succ P$, if and only if P' is deleted before P during the construction of $\mathcal{P}$. Note that the subtree of each internal vertex of a path P is a subset of the union of the paths that precede P in this linear order. There can be many path decompositions since the way of choosing the paths is not predefined.

In the example of *long-path decomposition* [4], the path chosen in every iteration is the longest root-to-leaf path (ties are broken arbitrarily), see Fig. 5b. Let $\mathcal{L} = \{L_1, \ldots, L_m\}$ be the ordered set of paths of the long-path decomposition of T. Since paths in $\mathcal{L}$ are ordered in the reverse order of their deletion, $|L_i| \leq |L_j|$ for $i < j$.

Heavy-rooted-pathwidth decomposition [2]. The *Strahler number*, also known as *Horton-Strahler number* of a tree is a parameter that was introduced by Horton and Strahler [11,14]. The same parameter was recently used by Biedl [5] under the name of *rooted pathwidth* when addressing the problem of computing upward tree drawings with optimal width.

The *rooted pathwidth* of a tree T, which we denote by $rpw(T)$, is defined as follows. If $|T| = 1$, then $rpw(T) = 1$. Otherwise, let k be the maximum rooted pathwidth of any subtree rooted at a child of $r(T)$. Then, $rpw(T) = k$ if exactly one subtree rooted at a child of of the root $r(T)$ of T has rooted pathwidth equal to k, and $rpw(T) = k + 1$ otherwise, i. e., if there is more than one

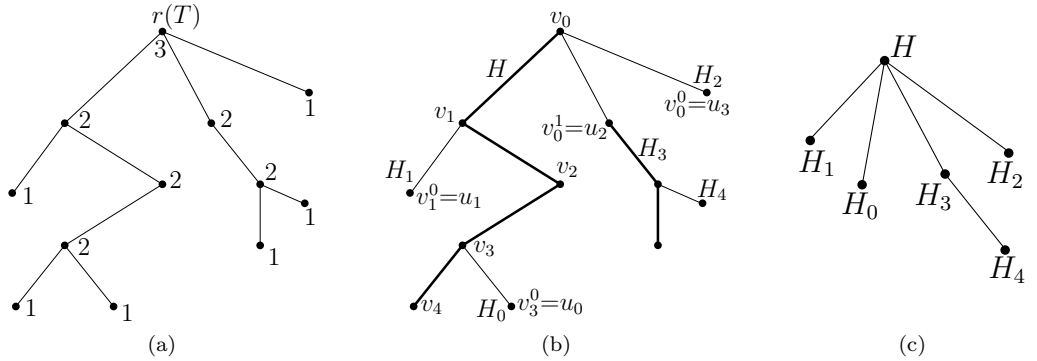


Figure 4: [2] A tree T with (a) the number $rpw(T(v))$ for each vertex v of T. In particular, $rpw(T) = 3$; (b) the heavy edges of T forming the heavy paths $H, H_0, \ldots, H_4$ are shown in bold; (c) the path tree of T

subtree rooted at a child of $r(T)$ having rooted pathwidth equal to k; see Fig. 4a. Clearly, $rpw(T)$ is an integer number.

The *heavy-rooted-pathwidth decomposition* of a tree T is defined as follows, see Fig. 4b. For each non-leaf vertex v of T, let c^* be a child of v in T such that $rpw(T(c^*))$ is maximum (ties are broken arbitrarily). Then (v, c^*) is a heavy edge; further, each child $c \neq c^*$ of v is a *light child* of v, and the edge (v, c) is a *light edge*. Connected components of heavy edges form a set of paths $\mathcal{H}(T) = \{H, H_0, \ldots, H_k\}$, called *heavy paths*, which may have many incident light edges. The *path tree* of T is a tree whose vertices correspond to heavy paths in T; see Fig. 4c. The parent of a heavy path P in the path tree is the heavy path that contains the parent of the vertex of P with the minimum depth. The root of the path tree is the heavy path containing $r(T)$.

For convenience, we also consider the light edge connecting a heavy path to its parent in the path tree to be a part of the path. So heavy-rooted-pathwidth decomposition $\mathcal{H}(T)$ is a path decomposition (see the definition above), when the paths are chosen to be the heavy paths. When it is clear from the context, we will refer to $\mathcal{H}(T)$ simply as $\mathcal{H}$.

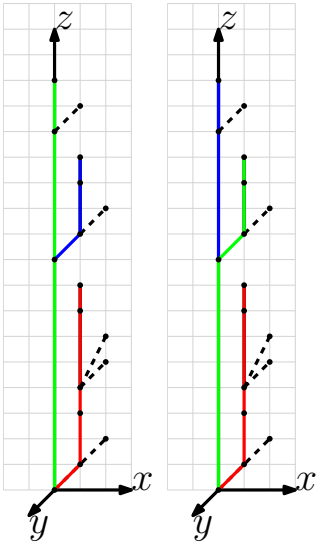
We denote by H the root of the path tree of T; let $v_0, \ldots, v_{m-1}$ be the ordered sequence of the vertices of the path H, where $v_0 = r(T)$. For $i = 0, \ldots, m-1$, we let $v_i^0, \ldots, v_i^{t_i}$ be the light children of v_i ordered arbitrarily. Let $L = u_0, u_1, \ldots, u_{l-1}$ be the sequence of the light children of vertices of H ordered so that: (i) any light child of a vertex v_j precedes any light child of a vertex v_i, if $i < j$; and (ii) the light child v_i^{j+1} of a vertex v_i precedes the light child v_i^j of v_i, see Fig. 4b. For a vertex $u_i \in L$, we denote by $p(u_i)$ its parent; note that $p(u_i) \in H$, see Fig. 4c. The height of the path tree of an n-vertex tree T is $rpw(T)$, for which it is known [5] that $rpw(T) \in \mathcal{O}(\log n)$. Fig. 5a and 5b illustrate the heavy-rooted-pathwidth decomposition and long-path decomposition of the same tree, where heavy paths and long paths are shown in different colors.

Canonical 3D drawing $\mathcal{C}(T)$ of a tree T, introduced in [2], is the crossing-free straight-line 3D drawing of T that maps each vertex v of T to its *canonical position* $\mathcal{C}(v)$ determined by the heavy-rooted pathwidth decomposition of T, as follows.

- First, we set $\mathcal{C}(v_0) = (0, 0, 0)$ for the root v_0 of T.
- Second, for each $i = 1, \dots, k-1$, we set $\mathcal{C}(v_i) = (0, 0, z_{i-1} + |T(v_{i-1})| - |T(v_i)|)$, where z_{i-1} is the z-coordinate of $\mathcal{C}(v_{i-1})$.
- Third, for each $i = 1, \dots, k-1$ and for each $j = 0, \dots, t_i$, we determine $\mathcal{C}(v_i^j)$ as follows. If $j = 0$, then we set $\mathcal{C}(v_i^j) = (1, 0, 1 + z_i)$, where z_i is the z-coordinate of $\mathcal{C}(v_i)$; otherwise, we set $\mathcal{C}(v_i^j) = (1, 0, z_i^{j-1} + |T(v_i^{j-1})|)$, where z_i^{j-1} is the z-coordinate of $\mathcal{C}(v_i^{j-1})$.
- Finally, in order to determine the canonical positions of the vertices in $T(v_i^j) \setminus \{v_i^j\}$, for each $i = 0, \dots, k-2$ and each $j = 0, \dots, t_i$, we recursively construct the canonical 3D drawing $\mathcal{C}(T(v_i^j))$ of $T(v_i^j)$, and translate all the vertices by the same vector so that v_i^j is sent to $\mathcal{C}(v_i^j)$.

Remark 1 *The canonical drawing $\mathcal{C}(T)$ lies in the halfplane XZ_0^+ inside a bounding box of height $|T|$ and width $rpw(T)$, and the root $r(T)$ of T is mapped to the left-bottom point of the box, that is the origin.*

Since tree T never changes throughout our algorithm, we refer $rpw(T)$ as rpw.



(a) (b)

Figure 5: Canonical drawing of a tree with (a) heavy and (b) long paths, where paths are colored with different colors, paths that consist of one edge are dashed. The root of the tree lies in the origin.

For any vertex v of T, let $T'(v)$ be a portion of the subtree $T(v)$. We define the canonical drawing of $T'(v)$ with respect to v to be the drawing of $T'(v)$ obtained by cropping from $\mathcal{C}(T)$ and translating the obtained drawing of $T'(v)$ so that v is mapped to the origin. If $T'(v) = T(v)$, we call it the *relative canonical drawing* of $T(v)$ and denote it by $\mathcal{C}_{T_v}$.

3 Tools for the morphing algorithms

In this section, we define stretching, mapping, rotation and shrinking of subtrees and prove some useful properties of them. Each of these are essential building blocks of our algorithms in Section 4 and 5.

3.1 Stretching by a constant $\mathcal{S}_1$

Let a planar drawing Γ of a tree T lie in the XY_0 plane. During *stretching morph* $\langle \Gamma, \Gamma_1 \rangle$ each coordinate of each vertex in Γ is multiplied by the same positive integer constant $\mathcal{S}_1$ to obtain the drawing Γ_1. Thereby, it is a linear morph that "stretches" the vertices of Γ apart.

Lemma 2 *Stretching is a crossing-free morph.*

Proof: For each $t \in [0,1]$ during the morphing step $\langle \Gamma, \Gamma_1 \rangle$, we denote the drawing of T by $\Gamma^t(T)$. In $\Gamma^t(T)$, the image of a vertex v of T is $\{t\Gamma_1(v) + (1-t)\Gamma(v)\}$. This implies it is the same drawing as Γ, but scaled by $t\mathcal{S}_1 + (1-t)$. Since the original drawing Γ is crossing-free, so is the drawing Γ^t. □

Recall that $B(\Gamma(v), r)$ denotes the open disk of radius r in the horizontal plane centered at $\Gamma(v)$.

Lemma 3 *Let $\langle \Gamma, \Gamma_1 \rangle$ be the stretching morph, that is stretching by $\mathcal{S}_1$. Then the following holds:*

1. *For any pair v_i, v_j of vertices, disks $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2})$ and $B(\Gamma_1(v_j), \frac{\mathcal{S}_1}{2})$ do not intersect in XY_0 plane.*
2. *For a vertex v_i, disk $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2 \cdot d(\Gamma)})$ does not enclose any other vertices or any part of edges non-incident to v_i in Γ_1.*
3. *For every vertex v and every edge $e = (v, u)$ in Γ_1, there is an integer point z such that $z \in e$ and $z \in B(\Gamma_1(v), d(\Gamma))$.*

Proof:

1. Disk $B(\Gamma_1(v_i), \mathcal{S}_1)$ does not contain other vertex $v_j, j \neq i$ since disk $B(\Gamma(v_i), 1)$ contains no other vertices. This implies that $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2})$, i.e., the disk with radius $\frac{\mathcal{S}_1}{2}$, does not intersect with any $B(\Gamma_1(v_j), \frac{\mathcal{S}_1}{2})$ for $j \neq i$.
2. Due to Lemma 1, no edges non-incident to v_i intersect the disk of radius $\frac{1}{d(\Gamma)}$ around v_i in Γ. That means that in Γ_1, where all distances are multiplied by $\mathcal{S}_1$, no non-incident edges intersect the disk of radius $\frac{\mathcal{S}_1}{d(\Gamma)}$ around v_i. Also, other vertices do not lie in $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2 \cdot d(\Gamma)})$ as $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2 \cdot d(\Gamma)}) \subset B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2})$.
3. Every edge $e = (u, v)$ in Γ has been stretched by $\mathcal{S}_1$. Let the images of vertex u and v be $\Gamma(u) = (u_x, u_y)$ and $\Gamma(v) = (v_x, v_y)$. Then, $\Gamma_1(u) = (u_x \cdot \mathcal{S}_1, u_y \cdot \mathcal{S}_1)$ and $\Gamma_1(v) = (v_x \cdot \mathcal{S}_1, v_y \cdot \mathcal{S}_1)$. The integer point $z \in e$ and $z \in B(\Gamma_1(v), d(\Gamma))$ is $(u_x \cdot \mathcal{S}_1 + v_x - u_x, u_y \cdot \mathcal{S}_1 + v_y - u_y)$, see Fig. 6. □

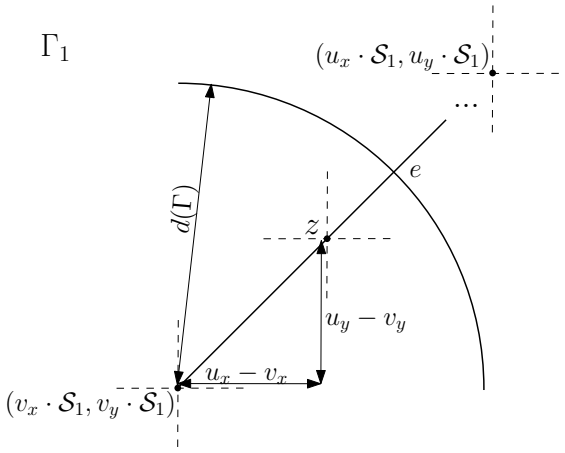


Figure 6: The integer point z for vertex v and edge $e = (v, u)$.

3.2 Shrinking lifted subtrees

Let v be a vertex of a tree T. Assume that the image $\Gamma(T(v))$ of subtree $T(v)$ coincides with the canonical drawing $\mathcal{C}_{T_v}$ of $T(v)$, in particular, it lies in $\overline{h} = XZ_v^+$. Let $\mathcal{C} = \{v_1, \ldots, v_l\}$ be the sequence of children of v, ordered according to their z coordinates in $\mathcal{C}_{T_v}$. Let $\mathcal{C}' = \{v_{i_1}, \ldots, v_{i_k}\}$ be some subsequence of $\mathcal{C}$. Let us consider the new subtree $T'(v)$ which is obtained by deleting the vertices in $\mathcal{C} \setminus \mathcal{C}'$ and their subtrees from $T(v)$. Note that, for each j with $1 \le j \le k$, $T'(v_{i_j})$ still lies inside a box of height $|T(v_{i_j})|$ and width $rpw(T(v_{i_j}))$ on h.

We define the *shrink subtree* procedure on T'_v as follows. We move each vertex v_{i_j} along with its subtree from $\mathcal{C}_{T_v}(v_{i_j})$ to $(\mathcal{C}_{T_v}(v_{i_j})_x, \mathcal{C}_{T_v}(v_{i_j})_y, \mathcal{C}_{T_v}(v) + h_j)$, where h_j equals to the number of vertices that were lying below v_{i_j} in $\mathcal{C}_{T_v}(T')$, see Fig. 7. Let us denote the shrunken subtree by $\mathcal{C}'_{T'_v}$. The height of the shrunken subtree $\mathcal{C}'_{T'_v}$ is equal to the number of vertices in $T'(v)$. Also, note that shrinking is a crossing-free unidirectional morph.

The following observation will be used to prove Lemma 10, which establishes the correctness of a part of our algorithm.

Observation 1 *The height of the shrunken subtree $\mathcal{C}'_{T'_v}$ is equal to the number of vertices in $T'(v)$.*

Proof: For $1 \le j \le k$, the subtree rooted at v_{i_j} has a canonical drawing with respect to v_{i_j} in $\mathcal{C}_{T'_v}$. Each $T'(v_{i_j})$ lies inside a box of height $|T'(v_{i_j})|$. Also, note that the z-coordinate of $\mathcal{C}'(v_{i_j})$ is equal to $\mathcal{C}_{T_v}(v_j)_z$. This implies that in $\mathcal{C}'_{T'_v}$, $T'(v)$ lies inside a box of height $\sum_{j=1}^{k} |T(v_{i_j})| + 1 = |T'(v_i)|$ and the down-left corner of the box coincides with $\mathcal{C}_{T_v}(v) = \mathcal{C}'_{T_v}(v)$. Also note that $|T'(v)| \le |T(v)|$. □

The following proposition from Alamdari et al. [1] is needed to prove that shrinking is a crossing-free morph, see Lemma 4.

Proposition 1 *[1, Corollary 15] Let $\langle \Gamma, \Gamma' \rangle$ be a unidirectional morph between two planar straight-line drawings Γ and Γ' of T. Let u be a vertex of T, let vw be an edge of T and, for any drawing of T, let l_{vw} be the line through the edge vw oriented from v to w. Suppose that u is to the left of l_{vw} both in Γ and in Γ'. Then v is to the left of l_{vw} throughout $\langle \Gamma, \Gamma' \rangle$.*

Lemma 4 *Shrinking is a crossing-free morph.*

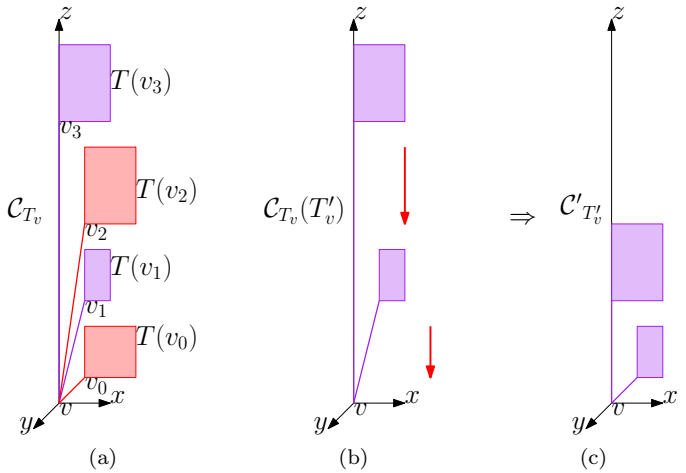


Figure 7: An example of a shrinking morph. (a) The original tree T_v with 4 subtrees $T(v_i), i \in \{0,1,2,3\}$. (b) Subtree $T'(v)$ of the original tree and the direction of movement for vertices in $T(v_1), T(v_3)$. (c) Resulting drawing after the Shrinking morph.

Proof: Note that the x and y coordinates of each vertex in $\mathcal{C}_{T'_v}$ and $\mathcal{C}'_{T'_v}$ are the same and z-coordinate can only decrease. This implies that shrinking is obviously a unidirectional morph in XZ_v plane. Note that throughout the morphing process all vertices in T'_v maintain the relative orders among themselves since they move along parallel vectors. Also, note that shrinking satisfies the conditions of Proposition 1. This implies that shrinking is a crossing-free morph. □

3.3 Mapping around a pole

Let the *pole through* $(x', y', 0)$ be the vertical line in $3D$ through point $(x', y', 0)$. Let α, β be two distinct vertical halfplanes containing the pole l passing through a point with integer coordinates. Suppose $\angle(\alpha, \beta) \notin \{0, \pi\}$ and α, β contain infinitely many points with integer coordinates. Mapping *around the pole* l is a single morphing step to obtain a drawing Γ' which lies in β from Γ which lies in α. Each vertex moves along a horizontal vector between α and β. The direction of these horizontal vectors is common for all vertices of Γ and is defined by α and β. Let us fix the horizontal plane $\overline{h}$ passing through the point $(0, 0, b)$ where b is an integer. Let p_α, p_β be points that lie on $\overline{h} \cap \alpha$ and $\overline{h} \cap \beta$, respectively; such that $dist(l, p_\alpha) = d_\alpha$ and $dist(l, p_\beta) = d_\beta$ be the minimum non-zero distances from the l to the integer points lying in $\overline{h} \cap \alpha$ and $\overline{h} \cap \beta$. The *vector of mapping* is defined as $\frac{p_\beta - p_\alpha}{|p_\beta - p_\alpha|}$. Mapping is a unidirectional morph since all vertices of Γ move along the vectors parallel to the vector of mapping till they reach the halfplane β. See Fig. 9.

We denote by *rotation* a mapping when α, β are halfplanes of planes parallel to XZ_0, YZ_0 respectively or vice versa. Similarly, we define mapping around a horizontal pole, i.e., a pole parallel to the X-axis. For rotation, the vector of mapping is lying in the following set, see Fig. 8.

$$\left\{ \left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, 0 \right), \left(\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, 0 \right), \left(-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, 0 \right), \left(-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, 0 \right) \right\}$$

For rotation around horizontal pole, the vector of mapping is lying in the following set.

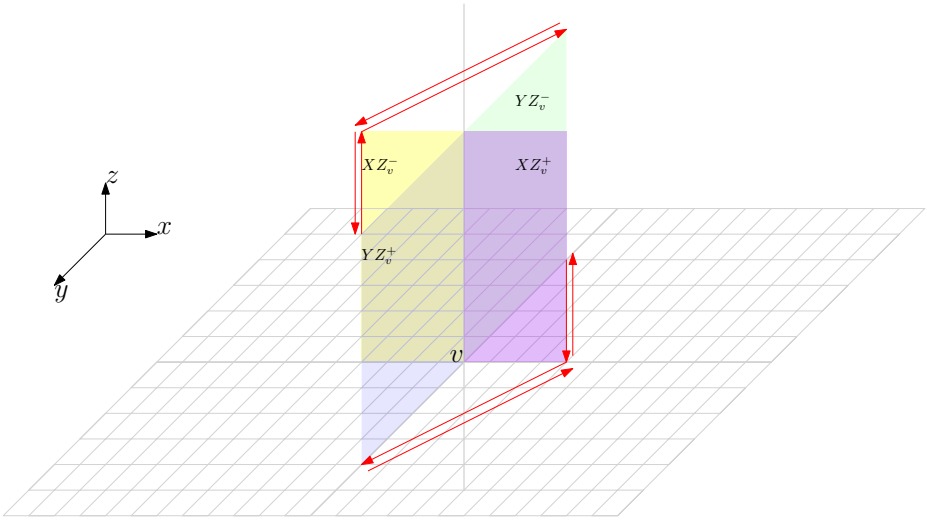


Figure 8: Possible directions of rotations between 4 halfplanes sharing a common vertical pole through vertex v

$$\left\{\left(0,\frac{\sqrt{2}}{2},\frac{\sqrt{2}}{2}\right),\left(0,\frac{\sqrt{2}}{2},-\frac{\sqrt{2}}{2}\right),\left(0,-\frac{\sqrt{2}}{2},\frac{\sqrt{2}}{2}\right),\left(0,-\frac{\sqrt{2}}{2},-\frac{\sqrt{2}}{2}\right)\right\}$$

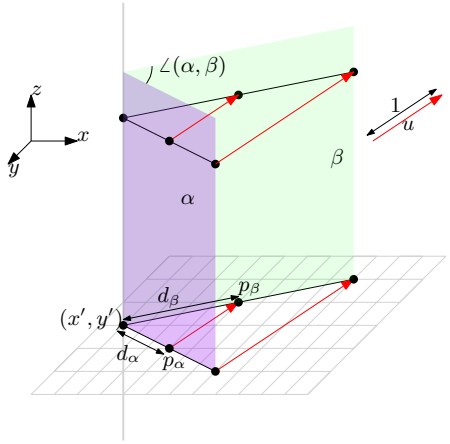


Figure 9: The mapping morph, halfplanes α, β sharing a common pole through point (x', y') and their vector of mapping.

Lemma 5 *Mapping around a pole is a crossing-free morph.*

Proof: We prove this statement for the case of mapping around a vertical pole. The case of horizontal pole is analogous. First, note that we map an integer point to an integer point by construction. Let α, β be vertical halfplanes containing the pole l through the origin. Let us assume that $\angle(\alpha, \beta) = \theta$ and let us define the stretching factor $S_m = \frac{d_\beta}{d_\alpha}$. Then, we can define mapping by the following matrix.

$$\begin{pmatrix} S_m \cos\theta & -S_m \sin\theta & 0 \\ S_m \sin\theta & S_m \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Note that the determinant of the matrix is positive. This implies that the order type of the points does not change during mapping, see [15]. Therefore, the drawing obtained in β is crossing-free since the drawing in α is crossing-free. Let the $(x_\alpha, y_\alpha, z_\alpha)$ maps to $(x_\beta, y_\beta, z_\beta)$. For each $t \in [0, 1]$ during mapping, the position of the point is $(1-t) \cdot (x_\alpha, y_\alpha, z_\alpha) + t \cdot (x_\beta, y_\beta, z_\beta)$. In other words, during mapping the position of a point at the moment $t \in [0, 1]$ is defined by the following matrix.

$$\begin{pmatrix} 1 - t + t \cdot S_m \cos\theta & -t \cdot S_m \sin\theta & 0 \\ t \cdot S_m \sin\theta & 1 - t + t \cdot S_m \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Observe that at a fixed t all points of α lie on the same plane, and the order type of the points does not change. This implies that we do not form any crossing at any t during mapping around the pole. Note that if the pole passes through the point $(a, b, 0)$ instead of the origin, we can first translate the point $(a, b, 0)$ to origin and translate back after mapping, and the same argument applies. □

3.4 Rotating around a point

Let $\Gamma_0(T(v))$ be the canonical drawing of a subtree $T(v)$ on the horizontal XY_v^+ or XY_v^- halfplane obtained by rotating the relative canonical drawing $\mathcal{C}_{T_v}$ around the horizontal pole through v. Let $\Gamma_1(T(v)), \Gamma_2(T(v)), \Gamma_3(T(v))$ be the drawings obtained from $\Gamma_0(T(v))$ by rotating the horizontal plane around the point $\Gamma(v)$ by the angles $\frac{\pi}{2}, \pi, \frac{3\pi}{2}$, respectively, see Fig 10. The drawing $\Gamma_i(T(v))$ can be obtained from the drawing $\Gamma_{i-1}(T(v))$ in one morphing step, see Lemma 6. Note that during the rotation in the horizontal plane, we fix the root to the origin and rotate the drawing of T by a multiple of $\frac{\pi}{2}$ around the origin. On the other hand, we rotate around a pole passing through the root during the rotation around a pole.
The following lemma is a reformulation of *Pinwheel* lemma of Barrera-Cruz et al. [3, Lemma 3].

Lemma 6 *[3, Lemma 3] The drawing $\Gamma_i(T(v))$ can be obtained from the drawing $\Gamma_{i-1}(T(v))$ in one morphing step, for $i = 0, 1, 2, 3$, where indices are taken modulo 4. This morph is crossing-free.*

4 Morphing through lifting paths

Let T be an n-vertex tree and $\mathcal{P}$ be some path decomposition of T into k paths. In this section, we describe an algorithm that morphs a planar drawing $\Gamma = \Gamma_0(T)$ of T to the canonical $3D$ drawing $\Gamma' = \mathcal{C}(T)$ of T in $\mathcal{O}(k)$ steps. It lifts the paths of $\mathcal{P}$ one by one by applying procedure *Lift*(). Note

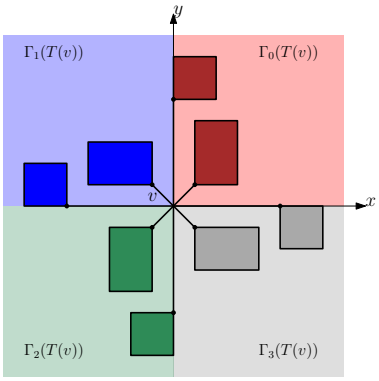


Figure 10: Drawings $\Gamma_i(T(v))$ $i = 0, 1, 2, 3$ obtained from the canonical drawing $\mathcal{C}_{T_v} = \Gamma_0(T(v))$.

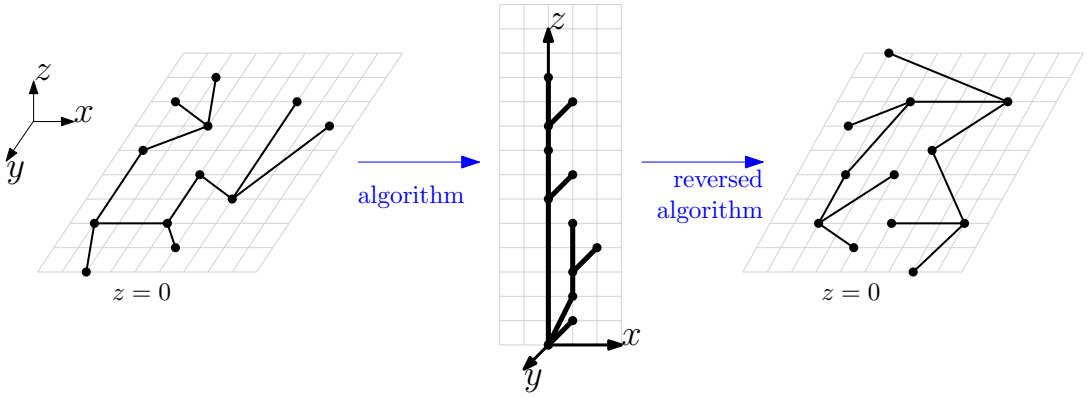


Figure 11: Our algorithms morph between two planar drawings through the 3D canonical drawing.

that the final positions for the vertices in $\mathcal{C}(T)$ are determined by the heavy-rooted-pathwidth decomposition of T, and thus do not depend on $\mathcal{P}$. A morph from $\mathcal{C}(T)$ to $\Gamma_0(T)$ can be obtained by playing the morph from $\Gamma_0(T)$ to $\mathcal{C}(T)$ backwards, see Fig. 11.

Algorithm overview

Step 0: Preprocessing. This step is a single stretching morph $\langle \Gamma, \Gamma_1 \rangle$ with $\mathcal{S}_1 = 2 \cdot (rpw + d(\Gamma))$. This step is crossing-free, see Lemma 2.

After Step 0, for each path in $\mathcal{P}$ the procedure *Lift*(P_i) is performed in the order that is induced by the definition of the path decomposition procedure (see Section 2). We first prove several statements about the algorithm, viewing procedure *Lift*() as a black box and then in Section 4.1 we give a detailed description of *Lift*().

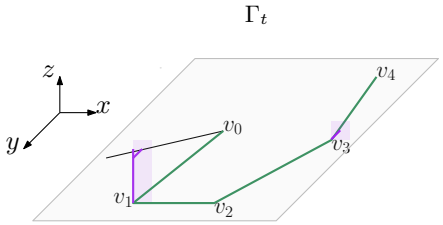


Figure 12: A Drawing Γ_t of T before applying the procedure *Lift*(P); edges of P are shown in green and the lifted subtrees are shown in light purple.

Let $P_i = (v_0, v_1, \ldots, v_m)$ be the first path in $\mathcal{P}$ that has not been processed yet and Γ_t be the current drawing of T, see Fig. 12. We lift the path P_i. For any vertex v, let *lifted subtree* $T'(v)$ be the portion of subtree $T(v)$ that has been lifted during the previous iteration of *Lift*() procedure. The following invariants hold after every iteration of the procedure *Lift*().

(I) The path $P_i \in \mathcal{P}$ is lifted only after all the children of the internal vertices of P_i are lifted. The drawing of the lifted subtree $T'(v)$ in Γ_t is the canonical drawing of $T'(v)$ with respect to v for any $v \in V(T)$.

(II) After the execution of *Lift*(P_i), path P_i moves to its canonical position with respect to *head*(P_i).

(III) For $k > i$, the vertices of path P_k lie in the XY_0 plane.

We now prove several statements that will be useful for proving correctness of our algorithm in Section 4.2. For now we treat *Lift*() as a black box and assume that the above invariants hold. Further in Section 4.1, we give a detailed description of *Lift*().

Let the *processing vertices* be the internal vertices of P_i along with the vertices of their lifted subtrees.

Lemma 7 *The subtrees of all internal vertices v_j in P_i are already lifted.*

Proof: All the paths that precede P_i in $\mathcal{P}$ are exactly the paths that are left after deleting P_i from the forest of subtrees of T in the constructive definition of $\mathcal{P}$ (see paragraph *Path decomposition*

in Section 2). From this fact, it follows that all internal vertices of P_i have become roots of some trees after deleting the edges of P_i and edges of those trees were processed in *Lift*(P_j) procedures for some P_j where $j < i$. □

Lemma 8 *The maximum height of vertices in the drawing Γ_t is strictly less than n.*

Proof: Note that all vertices of the processed paths are the part of the lifted subtrees of their head vertices. By invariant (I), their height is strictly less than the number of vertices in the corresponding subtree, which does not exceed number of vertices in T. □

Let v be a vertex of T. Then the below lemma follows from the definition of the canonical drawing and invariant (I).

Lemma 9 *For any vertex v and its lifted subtree $T'(v)$, the difference between x coordinates for any pair of vertices of $T'(v)$ in Γ_t does not exceed rpw. In other words, the maximum horizontal distance between any pair of vertices of $T'(v)$ in Γ_t is at most rpw.*

4.1 Procedure *Lift*(P)

In this section we describe the procedure *Lift*() for a path P of a path decomposition $\mathcal{P}$. We assume that all paths of $\mathcal{P}$ that precede P were already lifted. The procedure consists of 13 Steps, for each of them we prove that it is crossing-free. Analysis of the size of the grid required for this process is given in Section 4.2.

Step 1: Shrink lifted subtrees. For every internal vertex v_j of the path P, its lifted subtree $T'(v_j)$ morphs into shrunken lifted subtree, see Sec. 3.2. All subtrees are being shrunk simultaneously in one morphing step $\langle \Gamma_t, \Gamma_{t+1} \rangle$. See Fig. 13.

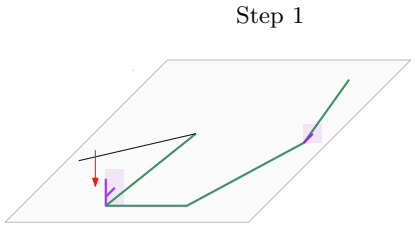


Figure 13: Step 1 of the *Lift*(P) procedure. Red arrows denote the direction of movement for the vertices that move during this step.

Lemma 10 *Step 1 is crossing-free.*

Proof: For a lifted subtree of a particular vertex, this is a shrinking step and by Lemma 4 it is crossing-free. Lifted subtrees of different vertices were not overlapping in projection to the XY_0 plane in the drawing Γ_t due to invariant (I) and Lemmas 3 and 9. Since shrinking of the lifted subtree of v_i takes place entirely in the plane $XZ^+_{v_i}$, all vectors of movement are vertical, and projections of lifted subtrees can not overlap during this morphing step and thus no crossing happens. No vertices or edges that change their position during Step 1 could intersect each other in this step.

In shrunken drawing every vertex of $T'(v_j)$ except v_j has strictly positive z coordinate which means that $T'(v_j) \setminus \{v_j\}$ remains strictly above the XY_0 plane and can not intersect the part of T that is still lying in the XY_0 plane. Internal vertices of P do not change their positions during this morph. □

Step 2: Rotate some lifted subtrees. This step consists of two morphs $\langle \Gamma_{t+1}, \Gamma_{t+2} \rangle, \langle \Gamma_{t+2}, \Gamma_{t+3} \rangle$, see Fig. 14a.

For $0 \leq j < i-1$, if projection $pr(T'(v_j))$ overlaps with $pr((v_j, v_{j+1}))$, we rotate twice the drawing of $T'(v_j)$ around the vertical pole through $\Gamma_t(v_j)$. Since every lifted subtree $T'(v_j)$ lies in halfplane $XZ^+_{v_j}$, after this step each such lifted subtree $T'(v_j)$ lies in $XZ^+_{v_j}$ or $XZ^-_{v_j}$.

Lemma 11 *Step 2 is crossing-free.*

Proof: Two lifted subtrees $T'(v_j), T'(v_i)$ of different vertices v_j, v_i cannot cross because Lemma 9 ensures that $pr(T'(v_j)) \subset B(\Gamma_{t+1}(v_j), rpw), pr(T'(v_i)) \subset B(\Gamma_{t+1}(v_i), rpw)$. On the other hand, due to Lemma 3 disks around vertices with radius $rpw \leq \frac{S_1}{2} = (rpw + d(\Gamma))$ do not cross with each other.

No edges within a single rotating lifted subtree can intersect each other as rotation is a crossing-free morph, see Lemma 5. Due to invariant (II), all edges that do not lie in the lifted subtrees are lying in the XY_0 plane. No lifted subtree $T'(v_j)$ can intersect any edges in the XY_0 plane due to invariant (I), i.e., $\Gamma_{t+1}(T'(v_j))$ is the canonical drawing with respect to v_j and therefore lies strictly above the XY_0 plane, except for the point $\Gamma_{t+1}(v_j)$. Rotation morph does not change z coordinate of points during the movement, so throughout the morph the drawing of $T'(v_j)$ lies above the XY_0 plane; vertex v_j does not move during the rotation as it lies on the pole.

No vertices move in XY_0 plane during Step 2 and thus no crossing can happen within the plane XY_0. □

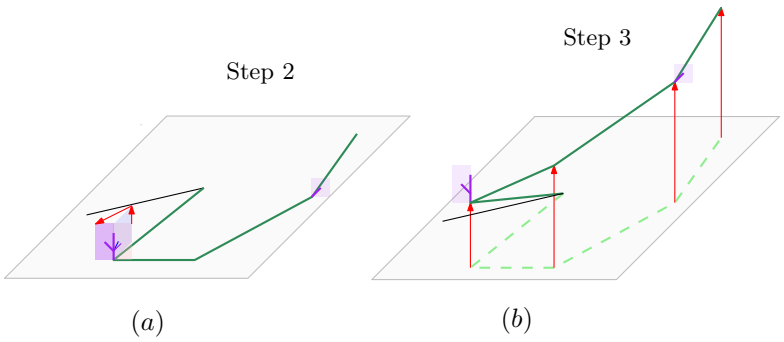


Figure 14: (a) Step 2 of *Lift*(P), (b) Step 3 of *Lift*(P); where the arrows show the directions of movement for the substeps.

Step 3: Lift the path up: This is a single morphing step $\langle \Gamma_{t+3}, \Gamma_{t+4} \rangle$ that moves each internal vertex $v_j, j \geq 1$ of path P vertically to the height defined recursively as follows. For v_1: $\Gamma_{t+4}(v_1)_z = n$; for $v_j, j > 1$: $\Gamma_{t+4}(v_j)_z = \Gamma_{t+4}(v_{j-1})_z + h_{sh}(T'(v_{j-1}))$, where $h_{sh}(T'(v_j))$ is an integer number equal to the height of shrunken lifted subtree $T'(v_j)$. See Fig. 14b.

Lemma 12 *Step 3 is crossing-free.*

Proof: Every internal vertex v_j in the path P moves with its subtree $T'(v_j)$ along a vertical vector. In the beginning of this morphing step, for any pair of distinct subtrees, their projections to XY_0 plane did not intersect. Also, no intersection existed between projections of different subtrees and the projections of path edges in XY_0 plane. Since all the movements are along the vertical vectors, these projections stay intact during this step. Moreover, all movements happen strictly above XY_0 plane implying that the subtrees that are still not lifted and lie in the XY_0 plane do not change their positions and can not form new crossings. Therefore, there can be no crossings during this step of the algorithm. □

Lemma 13 *After Step 3 of the Lift(P), the following holds:*

1. *All internal vertices of P, along with vertices of their subtrees, are separated from the rest of the vertices of the tree by a horizontal plane.*
2. *For any two internal vertices u, v of P, the subtrees of u and v are separated from each other by a horizontal plane.*

Proof:

1. Due to Lemma 8, the z-coordinates of all vertices that do not lie on the path are integers that are strictly less than n. By the definition of height in Γ_{t+4}, internal vertices of P and vertices in their lifted subtrees have z-coordinates at least n. Thus, the horizontal plane $z = n - \frac{1}{2}$ is the plane of separation.
2. For every pair of internal vertices v_j, v_k with $j < k$ of P, let us define the plane of separation as follows. By the definition of the height, $\Gamma_{t+4}(v_k)_z$ the horizontal plane $z = \Gamma_{t+4}(v_k)_z - \frac{1}{2}$ separates lifted subtree of vertex v_j from lifted subtree of vertex v_k for all $k > j$. □

Step 4: Rotate subtrees to horizontal plane. In this single morphing step $\langle \Gamma_{t+4}, \Gamma_{t+5} \rangle$, each lifted subtree $T'(v_j)$ of internal vertices of path P is rotated around a horizontal pole through $\Gamma_{t+4}(v_j)$ to lie in a horizontal plane. The direction of rotation is chosen in such a way that $T'(v_j)$ does not cross with an edge (v_j, v_{j+1}) throughout this morph. That is, we rotate in one of the two possible halfspaces, that does not contain edge (v_j, v_{j+1}). See Fig. 15a.

Lemma 14 *Step 4 is crossing-free.*

Proof: Suppose for the sake of contradiction, a crossing happens during Step 4. It must involve one of the moving edges, i.e., edges of lifted subtrees. For the internal vertices v_j, their subtrees $T'(v_j)$ are horizontally separated from each other due to Lemma 13, thus edges of different lifted subtrees can not cross. Since for each $j \geq 1$, the vertex v_{j-1} lies below the horizontal plane passing through the vertex v_j, an edge of $T'(v_j)$ could not cross the edge (v_{j-1}, v_j). Also, subtrees of internal vertices can not cross with non-processing vertices or edges since they are horizontally separated, Lemma 13.

If for some $j, 1 \leq j \leq m$, the edge (v_j, v_{j+1}) lied on the vertical plane through $T'(v_j)$ there was no crossings in Γ_{t+4} and after the beginning of movement $T'(v_j)$ will no longer lie in that plane and no crossing can happen. If (v_j, v_{j+1}) lied strictly within one of the halfplanes defined by $T'(v_j)$, then by the choice in Step 4, we rotate through the half-space that does not contain (v_j, v_{j+1}) and thus can not cross with this edge either. A contradiction. □

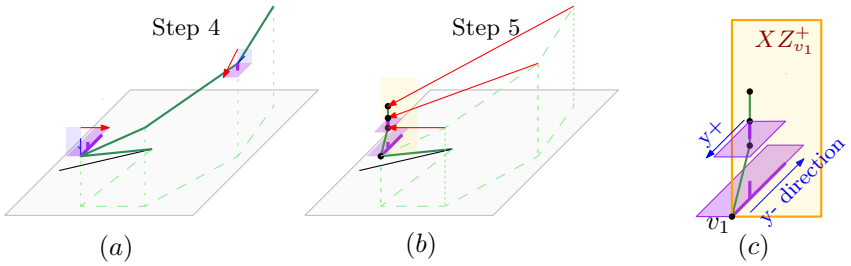


Figure 15: (a) Step 4. (b) Step 5. (c) Portion of the drawing after Step 5. Blue arrows show y-positive and y-negative directions with respect to yellow $XZ^+_{v_1}$ plane that contains $v_1, \ldots, v_m$.

Step 5: Correct the path. In morph $\langle \Gamma_{t+5}, \Gamma_{t+6} \rangle$, each vertex v_j of path P with $j \geq 2$ moves together with its subtree $T'(v_j)$ along the vector $((v_{1_x} - v_{j_x}) + \mathcal{C}(v_j)_x - \mathcal{C}(v_1)_x, v_{1_y} - v_{j_y}, 0)$, where v_{1_x} denotes x coordinate of vertex v_1 in drawing Γ_{t+5}. At the end of this step, the x and y coordinates of each v_j are same as their x and y coordinates in the canonical drawing with respect to v_1. See Fig. 15b and Fig. 15c.

Lemma 15 *Step 5 is crossing-free.*

Proof: First, note that each vertex v_j, $j \geq 2$, moves on the horizontal plane passing through it, that is, z coordinates of the vertices do not change. This step is crossing-free because all edges of the path P and all subtrees of different vertices v_j of the path P are horizontally separated from each other. All moving edges are horizontally separated from already lifted subtrees with the plane $z = n - \frac{1}{2}$ due to Lemma 13. □

Step 6: Go down. During the morphing step $\langle \Gamma_{t+6}, \Gamma_{t+7} \rangle$, every vertex $v_j, j \geq 2$ of the path P moves together with its subtree $T'(v_j)$ along the same vertical vector $(0, 0, (v_{1_z} - v_{j_z}) + \mathcal{C}(v_j)_z - \mathcal{C}(v_1)_z)$, where v_{1_z} and v_{j_z} are the z-coordinates of vertex v_1 and v_j, respectively, in the drawing Γ_{t+6}. At the end of this step, the z coordinate of v_j is the same as its z coordinate in the canonical drawing with respect to v_1. Fig. 16a illustrates the result of this step.

Lemma 16 *Step 6 is crossing-free.*

Proof: First, note that each vertex moves vertically. All moving edges are horizontally separated from already lifted subtrees by the plane $z = n - \frac{1}{2}$. During the morph, the vertical order of internal vertices of the path does not change as P is a portion of a root-to-leaf path. This implies that horizontal separation of $T'(v_j)$ is preserved during the step for different j. □

Step 7: Turn subtrees in horizontal planes. This step consists of two morphing steps $\langle \Gamma_{t+7}, \Gamma_{t+8} \rangle, \langle \Gamma_{t+8}, \Gamma_{t+9} \rangle$, and is illustrated in Fig. 16b. In Step 2 of procedure *Lift*(P), for some internal vertices v_j of P, the lifted subtrees were rotated to lie on $XZ^-_{v_j}$ planes. In Step 4, they were rotated to lie on horizontal planes. More specifically, all such subtrees $T'(v_j)$ lie in negative x-direction with respect to vertex v_j on the horizontal plane passing through v_j. We rotate every such lifted subtree $T'(v_j)$ around the corresponding internal vertex v_j twice by an angle $\frac{\pi}{2}$ in the horizontal plane passing through it. After this rotation the subtree lies in the positive x-direction

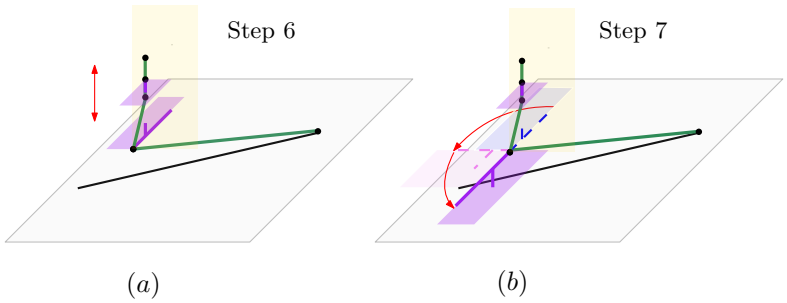


Figure 16: (a) Step 6, (b) Step 7.

with respect to vertex v_j. Note that we execute this morphing step for every subtree that lies in negative x-direction with respect to the corresponding internal vertex in the drawing Γ_{t+7}. See Section 3.4 for the description of rotating a drawing of a tree around a point in a horizontal plane.

Lemma 17 *Step 7 is crossing-free.*

Proof: By Lemma 6, rotating a drawing of a tree in the horizontal plane by the angle $\frac{\pi}{2}$ around the image of its root is a crossing-free morphing step. We perform this rotation twice consecutively in steps $\langle \Gamma_{t+7}, \Gamma_{t+8} \rangle, \langle \Gamma_{t+8}, \Gamma_{t+9} \rangle$. Thus no crossing can happen between the elements of the same subtree. Vertices of different lifted subtrees are horizontally separated from each other and from non-processing vertices and edges, which means no crossings can happen between them. □

Step 8: Stretch subtrees in y direction. This morphing step $\langle \Gamma_{t+9}, \Gamma_{t+10} \rangle$ transforms the lifted subtree of each internal vertex of P from its shrunken size to the original size of the subtree before shrinking. The transformation of the subtree of an internal vertex happens in a horizontal plane passing through the vertex. See Fig. 17a. Note that stretching here is a shrinking morph played backwards, and it is a crossing-free morph for every $T'(v_j)$ by Lemma 4.

Lemma 18 *Step 8 is crossing-free.*

Proof: As mentioned above, stretching here is a crossing-free morph for every $T'(v_j)$. As all subtrees $T'(v_j)$ are pairwise horizontally separated and also horizontally separated from the unprocessed part of the tree. Since the morph is horizontal, no crossings can happen between different subtrees $T'(v_j)$ or between a subtree $T'(v_j)$ and unprocessed part of the tree. □

Step 9: Rotate subtrees to canonical positions. This step is a single morphing step $\langle \Gamma_{t+10}, \Gamma_{t+11} \rangle$ where every lifted subtree $T'(v_j)$, where v_j is an internal vertex of path P, is rotated around the horizontal pole passing through v_j to lie in vertical halfplane $XZ^+_{v_j}$, see Fig. 17b.

Note that after Step 6, each internal vertex v_j with $j \geq 2$ lies in the relative canonical position in the halfplane $XZ^+_{v_1}$. This implies that the subtree $T(v_1)$ is in canonical position with respect to vertex v_1 after Step 9.

Lemma 19 *Step 9 is crossing-free.*

Proof: Let H be the vertical halfplane $XZ^+_{v_1}$; it contains $v_1, \ldots, v_m$.

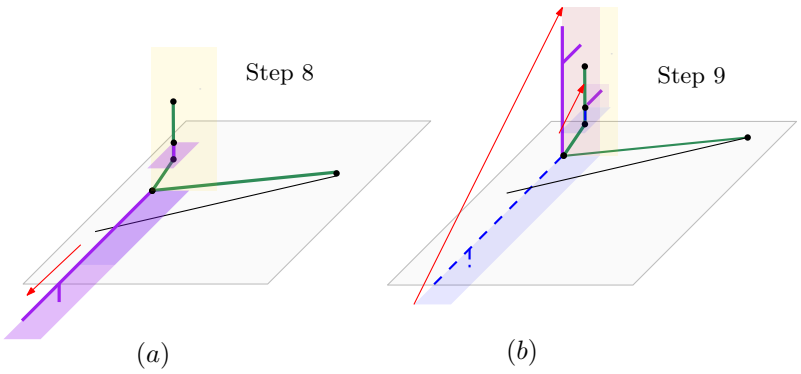


Figure 17: (a) Step 8, (b) Step 9.

For each j, vertices in subtrees $T'(v_j)$ lie on the same side of H in parallel planes at any moment of this morphing step. Note that at the beginning of the step, halfplanes that contain different subtrees are horizontal. Let v_j and v_k be two internal vertices of P, and suppose that their subtrees $T'(v_j)$ and $T'(v_k)$ lie on the same side of H. Since all internal vertices lie on the same vertical plane, rotation of the horizontal plane α containing $T'(v_j)$ and rotation of the horizontal plane β containing $T'(v_k)$ can be viewed as the rotations of two horizontal planes α, β around two horizontal poles contained in $XZ^+_{v_1}$ and passing through v_j and v_k, respectively. Then, at every moment of rotation, the angle between α and $XZ^+_{v_1}$ equals the angle between β and $XZ^+_{v_1}$, i.e., $\angle(\alpha, XZ^+_{v_1}) = \angle(\beta, XZ^+_{v_1})$. This means that these halfplanes are parallel to each other during every moment of the rotation. This implies that edges of $T'(v_j)$ do not cross the edges of $T'(v_k)$ this step.

If $T'(v_j)$ and $T'(v_k)$ lie on the different sides of H, they remain on the different sides of H during the entire movement, and do not cross as well.

At the end of Step 9, for each internal vertex v_j, $T'(v_j)$ is in the canonical position with respect to v_j. Each internal vertex v_j is in the canonical position with respect to the vertex v_1 as a result of Step 6 and does not move after it. Therefore, no two subtrees of distinct internal vertices cross each other at the end of the rotation when they lie on the halfplane $XZ^+_{v_1}$. □

Step 10: Move the subtrees along. In this morphing step $\langle \Gamma_{t+11}, \Gamma_{t+12} \rangle$ every internal vertex v_j of the path, together with its subtree $T'(v_j)$ moves horizontally in the direction $(v_{0_x} - v_{1_x}, v_{0_y} - v_{1_y}, 0)$. If in the canonical drawing $\mathcal{C}(T)$ of T the edge (v_0, v_1) is vertical, vertex v_1 moves along this vector to the point with x, y-coordinates (v_{0_x}, v_{0_y}). Otherwise, vertex v_1 moves along this vector as far as possible to get integer x and y coordinates not equal to (v_{0_x}, v_{0_y}).

By Lemma 3 there is such a point $p = (p_x, p_y, 0)$ in disk $B(\Gamma_{t+11}(v_0), rpw + d(\Gamma))$. Note that $dist_{\Gamma_{t+11}}(p, v_0) \leq d(\Gamma)$. Note that after Step 9, each internal vertex v_j with $j \geq 2$ along with its lifted subtrees lies in the relative canonical position with respect to v_1. This implies that all subtrees $T'(v_j)$ with $j \geq 2$ together take no more than rpw space around the vertical pole through the point (v_{1_x}, v_{1_y}), see Remark 1. Thus, the projection of those subtrees to the XY_0 plane lie within the disk $B(\Gamma_{t+11}(v_0), rpw + d(\Gamma))$. After this step, the projections of all processing vertices lie within $B(\Gamma_{t+11}(v_0), rpw + d(\Gamma))$ in XY_0 plane. See Fig. 18.

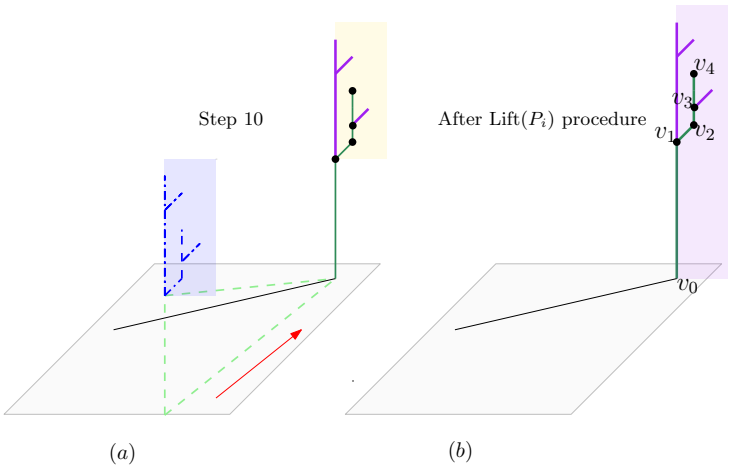


Figure 18: a) Step 10, (b) the drawing after *Lift*(P) procedure.

Lemma 20 *Step 10 is crossing-free.*

Proof: Step 10 is crossing-free because each vertex of the path P is horizontally separated from the subtrees of other vertices in the plane XY_0 (already lifted subtrees have height at most $n-1$ and all processing vertices have z-coordinates at least n). Recall that $\Gamma_1(T)$ is the drawing of T after Step 0. Note that vertex v_1 moves along the edge (v_0, v_1) towards vertex v_0, so the projection of the edge (v_0, v_1) always coincide with the segment $\Gamma_1(v_1)\Gamma_1(v_0)$ and shrink in length. Note that if in $\mathcal{C}(T)$ the edge (v_0, v_1) is vertical, then at the end of this step v_1 lies on the vertical pole through $\Gamma_1(v_0)$ but no other vertex in the subtree of $T(v_0)$ that lies on a different path than P can lie on the vertical pole through $\Gamma_1(v_0)$. □

Steps 11-13 go differently (see the two cases below) depending on whether we rotated $T'(v_0)$ during Step 2 or not.

Case 1: $T'(v_0)$ was not rotated since $pr(v_0, v_1)$ and $T'(v_0)$ did not overlap after Step 1. This implies that in Γ_{t+12} the lifted subtree of v_0 lies in $XZ^+_{v_0}$, see Fig. 19.
Case 2: The overlap happened after Step 1 and $T'(v_0)$ was rotated to lie in $XZ^-_{v_0}$, see Fig. 20.

Step 11: First part of the subtree xy-correction. **Case 1:** If in Step 2, $T'(v_0)$ was not rotated, during $\langle \Gamma_{t+12}, \Gamma_{t+13} \rangle$ all internal vertices of P with their subtrees $T'(v_j)$ move along the same horizontal vector until the edge (v_0, v_1) lies on the halfplane parallel to YZ_0 and $|{v_1}_y - {v_0}_y| = |\mathcal{C}(v_1)_x - \mathcal{C}(v_0)_x|$, see Fig. 19. The direction is chosen so that the angle between $pr((v_0, v_1))$ in Γ_{t+12} and $pr((v_0, v_1))$ in Γ_{t+13} is minimal.
Case 2: During Step 2, $T'(v_0)$ was rotated because the edge (v_0, v_1) was parallel to the X axis. Therefore $pr((v_0, v_1))$ is still parallel to the X axis, since $pr((v_0, v_1))$ always coincide with the segment $\Gamma_1(v_1)\Gamma_1(v_0)$ where $\Gamma_1(T)$ is the drawing of T after Step 0. By definition of Step 2, we know that $T'(v_0)$ in Γ_{t+12} lies in $XZ^-_{v_0}$. We are rotating $T'(v_0)$ around the pole through v_0 to lie in $YZ^+_{v_0}$, see Fig. 20.

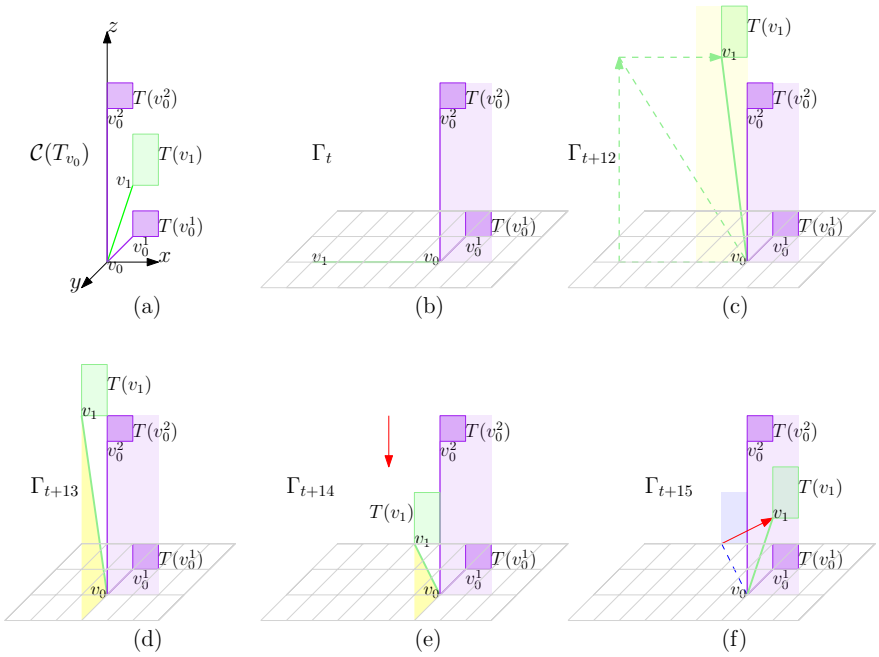


Figure 19: Steps 11-13. Case 1. (a) The canonical drawing of T_{v_0}. Path that will be lifted and other processing vertices are shown in green. (b) Drawing Γ_t before lifting path. (c)-(f) The four morphing steps composing Steps 11-13 in Case 1.

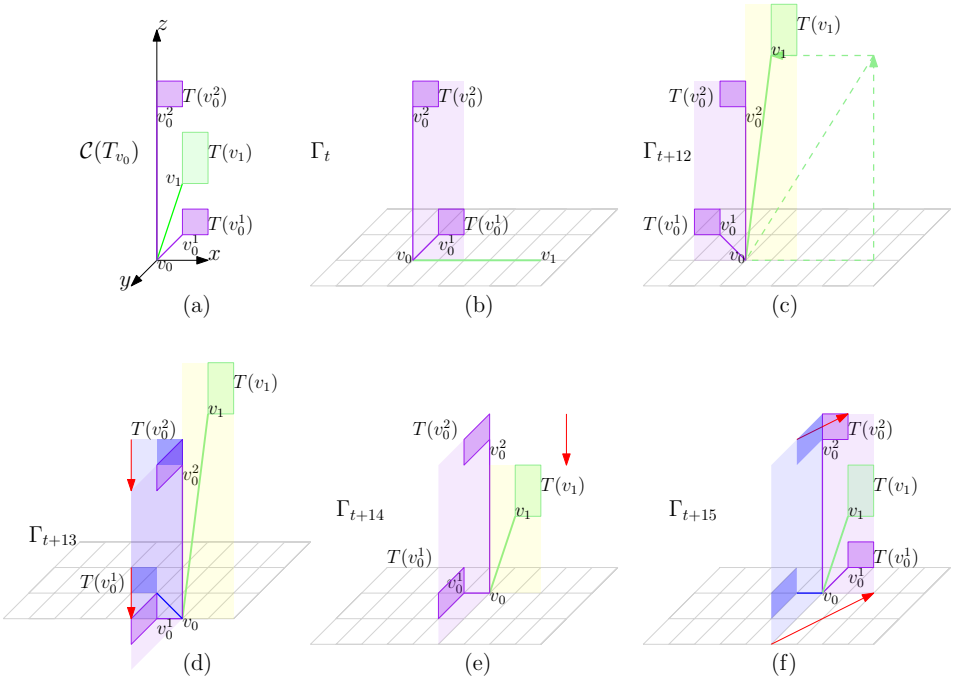


Figure 20: Steps 11-13. Case 2. (a) The canonical drawing of T_{v_0}. Path that will be lifted and other processing vertices are shown in green. (b) Drawing Γ_t before lifting path. (c)-(f) The four morphing steps composing Steps 11-13 in Case 2.

Lemma 21 *Step 11 is crossing-free.*

Proof: Case 1: For all vertices of $T(v_1)$, their vectors of movement are the same during this step. This means no crossing can happen in $T(v_1)$. Also $T(v_1)$ is horizontally separated from the unprocessed part of T, which is motionless.

As for the edge (v_0, v_1), its projection $pr((v_0, v_1))$ lies within the disk $B(\Gamma_{t+12}(v_0), rpw + d(\Gamma))$ throughout this morphing step and therefore, can not cross with any other lifted subtree $T'(v)$ rooted at a non-processing vertices v whose projection lies within $B(\Gamma_{t+12}(v), rpw + d(\Gamma))$. This step ends as soon as the edge (v_0, v_1) lies on YZ_{v_0} plane. Note that we move the edge (v_0, v_1) in such a way that it touches YZ_{v_0} plane before XZ_{v_0} plane. This implies that (v_0, v_1) can not cross with $T'(v_0)$ because (v_0, v_1) moves within one half-space defined by plane XZ_{v_0} and $T'(v_0)$ lies on XZ_{v_0}.

Case 2: Rotation is a crossing-free morph due to Lemma 5. In projection, it happens within the disk $B(\Gamma_{t+12}(v_0), rpw+d(\Gamma))$. Therefore, for the same reason as in Case 1 no crossings happen. □

Step 12: Move the subtrees down. This step consists of a single morphing step $\langle \Gamma_{t+13}, \Gamma_{t+14} \rangle$, and it is a vertical morph. In Γ_{t+13}, the internal vertex of P lies n units above its canonical position with respect to v_0. We decrease the z-coordinates of the internal vertices along with all their subtrees by n.

Lemma 22 *Step 12 is crossing-free.*

Proof: Case 1. This step is crossing-free because $T'(v_0)$ and $T(v_1)$ lie in distinct planes and their projections do not intersect in Γ_{t+13}. Vertical morph does not change the projection of the drawing, so the separation remains. After Step 11, edge (v_0, v_1) and $T'(v_0)$ lie in different planes as well.

Case 2. During Step 12, (v_0, v_1) and $T(v_1)$ are in $XZ^+_{v_0}$. $T'(v_0)$ during the same step is in $YZ^+_{v_0}$. They can not make any crossings because they do not intersect in projection during the entire morph. □

Step 13: Second part of the subtree xy-correction. This step consists of one morphing step $\langle \Gamma_{t+14}, \Gamma_{t+15} \rangle$.

Case 1: All processing vertices move along the same horizontal vector so that after this movement vertex v_1 lies in the canonical position with respect to v_0. As $T(v_1)$ is already in the canonical position with respect to v_1, after Step 13 it is in the canonical position with respect to v_0.

Case 2: Note that $T(v_1)$ and (v_0, v_1) are in the canonical positions with respect to v_0 after Step 12. Indeed in Step 10 we got x, y-coordinates equal to $(v_{0_x} + (\mathcal{C}(v_1)_x - \mathcal{C}(v_0)_x), v_{0_y} + (\mathcal{C}(v_1)_y - \mathcal{C}(v_0)_y)$ because (v_0, v_1) was parallel to X-axis, after Step 12 we have corrected z-coordinates. In this case we rotate $T'(v_0)$ to $XZ^+_{v_0}$, i.e. to its canonical position with respect to v_0. In both cases processing vertices and $T'(v_0)$ lie in canonical position with respect to v_0 in Γ_{t+15}, they all now form new lifted subtree of vertex v_0.

Lemma 23 *Step 13 is crossing-free.*

Proof: Case 1. During the morph, projections of no two elements of (v_0, v_1), $T(v_1)$, and $T'(v_0)$ cross. In the end of the step, $T(v_1)$ and (v_0, v_1) lie in their canonical positions with respect to v_0, From the beginning of the procedure *Lift*(P), $T'(v_0)$ was already in canonical position with respect to v_0 by invariant (I), so at the end of the morph they also can not cross.

Case 2. Rotation is a crossing-free morph and in the end of this step $T'(v_0), T(v_1)$ and (v_0, v_1) are in the canonical position with respect to v_0 and $\mathcal{C}_{T_{v_0}}$ does not contain any crossings. □

Observe that in the end the entire sequence of steps (Steps 1-13) all the internal vertices of P along with their subtrees are placed in the canonical position with respect to v_0.

4.2 Correctness of the algorithm

In this section we first prove that the algorithm is correct, and then analyse its complexity.

Correctness of the algorithm. To prove that our algorithm is correct, first we prove that the invariants, stated in the beginning of Section 4, hold after each iteration of *Lift*(P). Suppose that the paths P_j with $j < i$ are already lifted, and P_i is the path for which the lifting procedure is not started yet. We repeat the invariants here for convenience.

(I) Path $P_i \in \mathcal{P}$ is lifted only after all the children of the internal vertices of P_i are lifted. Recall that $T'(v)$ is a portion of the subtree $T(v)$ which is already lifted. Let Γ_t be the intermediate drawing of T obtained after the execution of *Lift*(P_{i-1}). The drawing of $T'(v)$ in Γ_t is the canonical drawing of $T'(v)$ with respect to v for any $v \in V(T)$.

(II) After the execution of *Lift*(P_i), path P_i moves to its canonical position with respect to *head*(P_i), i.e., all the vertices of the path P_i together with their children are in the canonical position with respect to *head*(P_i).

(III) After the execution of *Lift*(P_i), the vertices of path P_k with $k > i$ lie in the XY_0 plane until the next iteration of *Lift*() procedure begins.

Lemma 24 *The above mentioned invariants hold after performing Step 0 and after Lift*(P_i) *for each* $1 \leq i \leq m$.

Proof: First, we prove by induction that invariant (II) and invariant (III) hold. Then, we argue that invariant (II) and (III) together imply invariant (I).

Let P_i consists of the vertices $\{v_0, v_1, \ldots, v_k\}$ where v_0 is the *head*(P_i).

Invariant(II). The drawing of $T'(v)$ in Γ_t is the canonical drawing of $T'(v)$ with respect to v for any vertex $v \in V(T)$.

Note that after performing the stretching step the whole T lies in the XY_0 plane. Since none of the paths are lifted, invariant (II) trivially holds after the Step 0.

Suppose that we have lifted $i-1$ paths for some $i \geq 1$ and invariant (II) holds, i.e., for any internal vertex v of P_i, the drawing of $T'(v)$ is the canonical drawing of $T'(v)$ with respect to v before the execution of the procedure *Lift*(P_i). We now prove that invariant(II) will hold after the execution of the procedure *Lift*(P_i).

After Step 5, each internal vertex v_j with $j \geq 2$ has canonical x, y-coordinates with respect to vertex v_1 and after Step 6 has canonical z-coordinates with respect to v_1. Recall that we have shrunken lifted subtrees in Step 1. Steps 7 and 8 guarantee that each of the lifted subtrees $T'(v_j), j = 1, \ldots, k$ is in canonical position with respect to its root. They all lie in the positive x-direction from their respective roots in the horizontal plane passing through their respective roots. For every $T'(v_j), 1 \leq j \leq k$, the canonical positions with respect to the roots get restored after Step 8 but in horizontal direction.

Then, Step 9 lifts subtrees of internal vertices into (vertical) canonical position after rotating them. That is $T(v_1)$ is in canonical position with respect to vertex v_1. Steps 10-13 move $T(v_1)$ along with v_1, so after the end of procedure *Lift*(P_i), the subtree $T(v_1)$ is in canonical position with respect to v_0. Note that Steps 10-13 also guarantee that old $T'(v_0)$, i.e., the portion of the subtree rooted at v_0 that was already lifted before the beginning of the execution of *Lift*(P_i), is placed back to the canonical position with respect to v_0 in case it was moved in Step 2. That means that after Step 13, new $T'(v_0)$, which consists of the old $T'(v_0)$ along with new edge (v_0, v_1) and subtree $T(v_1)$, is in canonical position with respect to v_0.

As for the other vertices in XY_0, their lifted subtrees had not moved during *Lift*(P) procedure and are in canonical positions with respect to their roots by induction hypothesis.

Invariant(III). Let us prove that vertices of paths $P_k, k > i$ are lying in XY_0 plane. We again use induction on i.

After Step 0 the invariant (II) holds since the entire tree T lies in the XY_0 plane.

Suppose we have lifted $i-1,\ i \geq 1$, paths and before the *Lift*(P_i) procedure all vertices of non-processed paths lie in XY_0 plane — invariant (III) holds. Internal vertices of the path P_i can not lie in other non-processed paths than P_i by definition of path decomposition. In *Lift*(P_i), we only move processed paths, i.e. internal vertices of P_i along with the lifted subtrees. All vertices that lie on non-processed paths still are in XY_0 plane. That means that after the execution of the *Lift*(P_i) procedure invariant (III) holds too.

Invariant(I). Before the start of the *Lift*(P_i) procedure, by the induction hypothesis all the subtrees of the internal vertices are lifted. Also, note that the lifted subtree $T'(v_0)$ is in the canonical position with respect to v_0. Since invariant (II) is true, in the end of the *Lift*(P_i) procedure path P_i moves to its canonical position with respect to v_0. This implies that the lifted subtree of v_0 is in the canonical position with respect to it at the end of the *Lift*(P_i) procedure. Note that only the subtrees of the internal vertices of path P_i take part in morphing steps during the *Lift*(P_i) procedure and each of them is in the canonical position with respect to their roots in the *Lift*(P_i) procedure. By invariant (III), the vertices of path P_k for $k > i$ lie in XY_0 plane. Note that they also are not moved during *Lift*(P_i). □

We now prove that our morph is a grid morph, which completes the proof of correctness of our algorithm.

Lemma 25 *All morphing steps in the algorithm are grid morphs.*

Proof: Let us prove this by induction on number of lifted paths.

Consider the base case where none of the paths is lifted. After the stretching morph, i.e., Step 0, all coordinates of all vertices are integer because the constant of stretching is integer and in the input drawing $\Gamma = \Gamma_0$ all vertices had integer coordinates.

Suppose $i-1, i \geq 1$ paths are lifted and all performed morphs were grid morphs. That implies that in the beginning of procedure *Lift*(P_i), all vertices lie on integer points of the grid. During the *Lift*(P_i) procedure non-processed vertices and vertex v_0 do not change coordinates at all. Rotation and shrinking morphs move points with integer coordinates to points with integer coordinates, see Lemma 5 and Lemma 4, respectively. Turning of the subtrees in horizontal planes in Step 7 is a grid morph, see Lemma 6. In all other steps the lifted subtrees of internal vertices are moved along with their respective roots by the same vector to the points with integer coordinates. Thus,

after the procedure *Lift*(P_i) all vertices still have integer coordinates since all coordinates at the beginning of the procedure *Lift*(P_i) were integer.

This implies that all morphing steps in the algorithm are grid morphs. □

Complexity of the algorithm. Let us estimate the size of a grid required by the algorithm. Let us recall that $l(\Gamma)$, $w(\Gamma)$ and $h(\Gamma)$, respectively, denote the *length*, *width* and *height* of the $3D$ drawing Γ of T, i.e., the maximum absolute difference between the x-, y- and z-coordinates of vertices in Γ, respectively. Also, recall that $d(\Gamma)$ denote the *diameter* of Γ, defined as the ceiling of the maximum pairwise (Euclidean) distance between its vertices.

Note that the initial drawing $\Gamma = \Gamma_0$ is planar. At the beginning of procedure *Lift*(), our tree $T = (V, E)$, $|V| = n$ has a drawing $\Gamma = \Gamma_0$ that requires a two dimensional grid of size $l(\Gamma) \times w(\Gamma)$.

First step of the algorithm multiplies the required grid size by $\mathcal{S}_1 = 2 \cdot (rpw + d(\Gamma))$. During our algorithm, already lifted subtrees can take space in x, y-directions, but no more than $rpw(T)$. In z-direction every lifted subtree takes no more than n, while during *Lift*() procedure we may get vertices at height at most $2n$.

Therefore, the space required throughout the algorithm can be estimated as follows:

$$(x \times y \times z): \ \mathcal{O}(l(\Gamma) \cdot 2 \cdot (d(\Gamma) + rpw) + 2 \cdot rpw) \times \mathcal{O}(w(\Gamma) \cdot 2 \cdot (d(\Gamma) + rpw) + 2 \cdot rpw) \times \mathcal{O}(n) =$$

$$\mathcal{O}(d^2(\Gamma)) \times \mathcal{O}(d^2(\Gamma)) \times \mathcal{O}(n)$$

Recall that $rpw(T) = \mathcal{O}(\log n)$ [2, 5]. Note that for a grid drawing Γ of an n vertex tree T, $d(\Gamma) \geq \sqrt{n}$. This implies that for a sufficiently large n, $rpw(T)$ is asymptotically less than $d(\Gamma(T))$ for any grid drawing $\Gamma(T)$.

The lemmas proved in this section along with the space and time complexity bounds imply the following theorem, that is the main result of this section.

Theorem 1 *For any two planar straight-line grid drawings Γ, Γ' of a tree T with n vertices and any path decomposition $\mathcal{P}$ of T, there exists a crossing-free 3D morph $\mathcal{M} = \langle \Gamma = \Gamma_0, \ldots, \Gamma_l = \Gamma' \rangle$ that consists of $\mathcal{O}(k)$ steps where k is the number of paths in $\mathcal{P}$. In this morph, every intermediate drawing $\Gamma_i, 1 \leq i \leq l$ is a straight-line 3D grid drawing lying in a grid of size $\mathcal{O}(d^2) \times \mathcal{O}(d^2) \times \mathcal{O}(n)$, where d is maximum of the diameters of the input drawings Γ and Γ'.*

Since any path decomposition has at most n paths, we can derive the following.

Corollary 1 *For every two planar straight-line grid drawings Γ, Γ' of a tree T with n vertices there exists a crossing-free 3D-morph $\mathcal{M} = \langle \Gamma = \Gamma_0, \ldots, \Gamma_l = \Gamma' \rangle$ that has $\mathcal{O}(n)$ morphing steps, and require grid size $\mathcal{O}(d^2) \times \mathcal{O}(d^2) \times \mathcal{O}(n)$, where d is maximum of the diameters of the input drawings. In this morph every intermediate drawing $\Gamma_i, 1 \leq i \leq l$ is a straight-line 3D grid drawing lying in a grid of size $\mathcal{O}(d^2) \times \mathcal{O}(d^2) \times \mathcal{O}(n)$.*

5 Morphing through lifting edges

In this section, we describe another algorithm that morphs a planar grid drawing Γ of an n-vertex tree T to the canonical drawing $\mathcal{C}(T)$ of T, see Section 2 for the definition of $\mathcal{C}(T)$. The algorithm, similarly to the algorithm from Section 4, depends on a given path decomposition $\mathcal{P}$. This time one iteration of our algorithm lifts simultaneously a set of edges of T, that contains at most one edge from each path of decomposition $\mathcal{P}$. Let us set the drawing Γ_0 to be equal to the initial drawing Γ of T.

Step 0: Preprocessing. This step $\langle \Gamma, \Gamma_1 \rangle$ is a stretching morph with $\mathcal{S}_1 = 2{\cdot}rpw{\cdot}d(\Gamma){\cdot}(4{\cdot}d(\Gamma){+}1)$. Recall that stretching is a crossing-free morph due to Lemma 2.

Let m denote the depth of T. For an edge e of T, let the start vertex $st(e)$ (respectively, the end vertex $end(e)$) be the vertex of e with smallest (respectively, largest) depth. Let $\mathcal{K} = \{K_1, \ldots, K_m\}$ be the partition of edges of T into disjoint sets such that $e \in K_i$ if and only if the depth of $st(e)$ in T equals $m - i$. For every vertex v, let $End(v)$ be a set of edges that have v as their start vertex. All edges of $End(v)$ are contained in the same set K_i because their depth is determined by $dpt(v)$. We lift up sets K_i one by one from $i = 1$ to $i = m$ by running the procedure $\overline{Lift}(K_i)$, until we obtain the canonical drawing of T.

Let Γ_t be the drawing of T before lifting set K_i. See Fig. 22a. The details of the procedure $\overline{Lift}(K_i)$ are presented in Section 5.1. For $j < i$, let *lifted subtree* $T'(v_j)$ be the portion of subtree $T(v_j)$ which is already lifted by the execution of $\overline{Lift}(K_j)$ for some $j < i$. Invariants that we maintain throughout the algorithm are the following:

(I) The drawing of $T'(v_j)$ in Γ_t is the canonical drawing of $T'(v_j)$ with respect to v_j.

(II) The edges that are part of a not yet processed set $K_l \in \mathcal{K}$, $l \geq i$, lie in the XY_0 plane, together with the vertices incident to those edges.

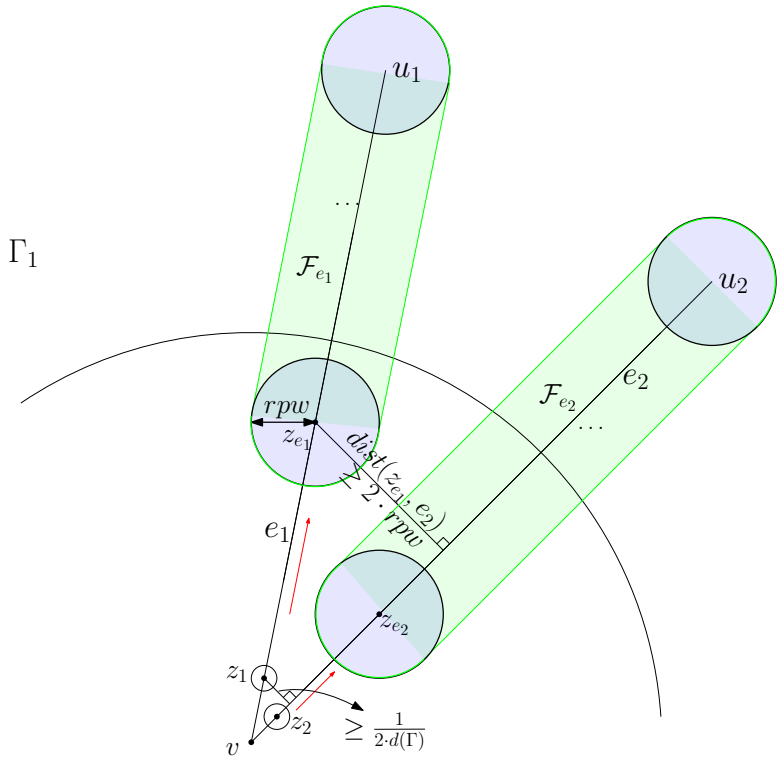


Figure 21: Finding points z_{e_1}, z_{e_2} for edges e_1, e_2 with common start endpoint v.

Lemma 26 *For any edge $e = (v, u)$ with $st(e) = v$ in the drawing Γ_1 of T, obtained by performing Step 0 on the initial drawing Γ_0, there is an integer point $z_e \in e$ such that the following conditions hold:*

1. *Disk $B(\Gamma_1(z_e), rpw \cdot d(\Gamma))$ is enclosed in the disk $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$.*

2. *For any pair of distinct edges $e_1, e_2 \in K_i$ for any $i \in \{1, \dots, m\}$, the disks $B(\Gamma_1(z_{e_1}), rpw)$ and $B(\Gamma_1(z_{e_2}), rpw)$ are disjoint.*

3. *For any pair of distinct edges $e_1, e_2 \in K_i$ for any $i \in \{1, \dots, m\}$, regions $\mathcal{F}_{e_1}, \mathcal{F}_{e_2}$ are disjoint, where $\mathcal{F}_e = \{x \in XY_0 : dist_{\Gamma_1}(x, (z_e, u)) \leq rpw\}$, i.e., $\mathcal{F}_e$ is the locus of all points in the plane XY_0 within distance rpw from the line segment (z_e, u) where $u = end(e)$, see Fig 21.*

Proof: Let us fix an edge $e = (v, u)$. By Lemma 3, there is an integer point z lying on e in $B(\Gamma_1(v), d(\Gamma))$. Let z_e be the point with coordinates $(v_x + (z_x - v_x) \cdot 4 \cdot rpw \cdot d(\Gamma), v_y + (z_y - v_y) \cdot 4 \cdot rpw \cdot d(\Gamma), 0)$. Let us show that it satisfies items (1)-(3).

1. Item (1) holds because $dist_{\Gamma_1}(z_e, v) \leq d(\Gamma) \cdot (4 \cdot rpw \cdot d(\Gamma))$.

2. **Case 1:** For edges $e_1, e_2 \in K_i$ with different start vertices v_1, v_2, disk $B(\Gamma_1(z_{e_1}), rpw)$ is disjoint from $B(\Gamma_1(z_{e_2}), rpw)$ since $B(\Gamma_1(z_{e_j}), rpw) \subset B(\Gamma_1(v_j), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ and disks $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ are disjoint for different vertices, due to Lemma 3.

   Case 2: For edges $e_1, e_2 \in K_i$ with a common start vertex v, let z_1, z_2 be the points in Γ_1 provided by Lemma 3. We have $dist(z_1, z_2) \geq 1$, because z_1, z_2 are integer points. Then by the choice of z_{e_1}, z_{e_2} we have $dist_{\Gamma_1}(z_{e_1}, z_{e_2}) \geq 4 \cdot rpw \cdot d(\Gamma) > 2 \cdot rpw$. This implies that for edges $e_1, e_2 \in K_i$ disks $B(\Gamma_1(z_{e_1}), rpw)$ and $B(\Gamma_1(z_{e_2}), rpw)$ do not intersect each other.

3. Since the drawing Γ_1 is crossing-free and planar, the minimum distance between the line segments $(z_{e_1}, u_1), (z_{e_2}, u_2)$ is realized at one of the endpoints of the segments. Note that two distinct edges either share a common endpoint or they are disjoint. Let us consider these two cases.

   Case 1: Suppose that edges $e_1 = (v, u_1)$ and $e_2 = (v, u_2)$ share a common endpoint v. Note that two points v, z_2 lie on the edge e_2 and are integer points. Also, note that z_1 is an integer point. $dist(z_1, e_2)$ is at least the distance between the point z_1 and the line spanned by the points v, z_2. By using an argument similar to Lemma 1, we conclude that $dist(z_1, e_2) \geq \frac{1}{2 \cdot d(\Gamma)}$ since the length of the line segment vz_2 is at most $2 \cdot d(\Gamma)$. Then, $dist(z_{e_1}, e_2) \geq (4 \cdot rpw \cdot d(\Gamma)) \cdot \frac{1}{2 \cdot d(\Gamma)} = 2 \cdot rpw$. For vertices $u_j, j = 1, 2$ by Lemma 3 disk $B(\Gamma_1(u_j), 2 \cdot rpw)$ does not intersect any edges non-incident to u_j or contain any other vertices than u_j. That means that all segment (z_{e_1}, u_1) is at distance at least $2 \cdot rpw$ from segment (z_{e_2}, u_2) and $\mathcal{F}_{e_1}, \mathcal{F}_{e_2}$ do not intersect. See Fig. 21.

   Case 2: Suppose now that edges e_1, e_2 with different start vertices and end vertices. Let $e_1 = (v_1, v_2), e_2 = (v_3, v_4)$, and for the same reasons as above (by Lemma 3) the corresponding regions are disjoint.

   Both the cases together imply item (3). □

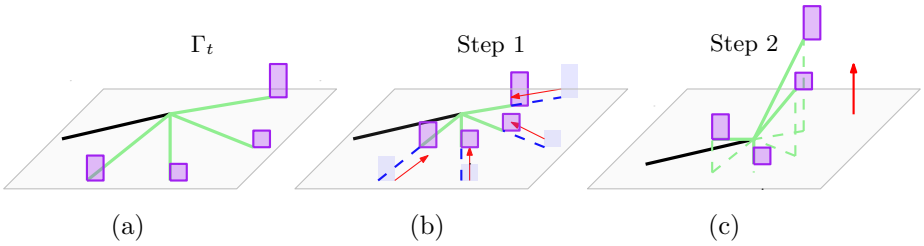


Figure 22: Drawing Γ_t, Steps 1, 2. (a) Drawing Γ_t before execution of the procedure $\overline{Lift}(K_i)$; lifted subtrees are shown in violet, K_i consists of green edges. (b) Step 1 of $\overline{Lift}(K_i)$. (c) Step 2 of $\overline{Lift}(K_i)$

5.1 Procedure $\overline{Lift}(K)$

This section describes the procedure $\overline{Lift}()$ for a set $K \in \mathcal{K}$ of edges of T. The procedure consists of five steps, for each of which we prove that it is crossing-free.

Step 1: Move subtrees towards starting vertices. In the single step $\langle \Gamma_t, \Gamma_{t+1} \rangle$, for each edge $e \in K$ we simultaneously move the vertex $end(e)$ along with its lifted subtree towards vertex $st(e)$ until $end(e)$ reaches point z_e determined for edge e by Lemma 26. See Fig. 22b.

Lemma 27 *Step 1 is crossing-free.*

Proof: Vertices that lie in the horizontal plane XY_0 plane do not move or move along their incident edges, so no intersections can happen in XY_0.

Lemma 26 guarantees that the moving subtrees do not intersect because each lifted subtree in projection lies in the corresponding region $\mathcal{F}_e$ throughout the whole morph and these regions do not intersect for different edges $e \in K$. This implies that no intersections happen during this morphing step. □

Step 2: Lift the edges up. In this morphing step $\langle \Gamma_{t+1}, \Gamma_{t+2} \rangle$, we move $end(e)$ together with its subtree $T'(end(e))$ by the vector $(0, 0, (\mathcal{C}(end(e))_z - \mathcal{C}(st(e))_z))$ for each edge $e \in K$, where $\mathcal{C}(v)$ is the canonical position of v. See Fig. 22c.

Lemma 28 *Step 2 is crossing-free.*

Proof: Recall that all edges of the set $End(v)$ defined above are contained in the same set $K \in \mathcal{K}$ because *dpt* is equal to their depth in the tree. This implies that v does not have any lifted subtrees in the beginning of $\overline{Lift}(K)$ procedure. Let $e_1 = (v, u_1)$ and $e_2 = (v, u_2)$ be two edges in K that both belong to the set $End(v)$. In Γ_{t+1},the projections of the lifted subtrees of u_1 and u_2 to XY_0 plane do not intersect. Let $e_1 = (v_1, u_1)$ and $e_2 = (v_2, u_2)$ be two edges in K where $v_1 \neq v_2$. Then, the projections of the lifted subtrees of u_1 and u_2 to the plane XY_0 also do not intersect in Γ_{t+1}.

During this morphing step, we do not change the projections and move every lifted subtree by the same vertical vector, so no intersections can happen between different subtrees and between the edges of the same subtree. Vertices of unprocessed sets lie in XY_0 plane and do not change their positions. It implies that they can not create any intersection as well. □

Step 3: Mapping of the lifted subtrees. Morphing step $\langle \Gamma_{t+2}, \Gamma_{t+3} \rangle$ is a mapping morph, see Section 3.3. For every lifted subtree $T'(v_j)$, where $v_j = end(e), e \in K$, we define the halfplanes of the mapping morph as follows: halfplane α is $XZ^+_{v_j}$, halfplane β is part of the vertical plane containing the edge e in such direction that $e \notin \beta$, and the common vertical pole of α and β is a pole through v_j. Such mappings are done simultaneously for all subtrees of end vertices of the edges of K. See Fig. 23a.

Lemma 29 *Step 3 is crossing-free.*

Proof: By Lemma 5 mapping is a crossing-free morph and thus no intersections happen within the same subtree $T'(v_j)$. Note that in projection to XY_0 plane, for every edge e, the movement of $T'(v_j)$ for $v_j = end(e)$ happens within the region $\mathcal{F}_e$ defined for e and $st(e)$ in Lemma 26. Also by Lemma 26, e contains at least rpw integer points between z_e and the boundary of the disk $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ and thus between points z_e and $end(e)$. Note that in Γ_{t+1} and in Γ_{t+2} projection of vertices of $T'(v_j)$ lies in $\mathcal{F}_e$. This implies that the projection of $T'(v_j)$ lie in $\mathcal{F}_e$ during the entire morphing step.

Different lifted subtrees do not intersect as they do not intersect in projection to XY_0 during this step since the regions $\mathcal{F}_e$ are pairwise disjoint for different edges. Other vertices do not move and also make no intersections. □

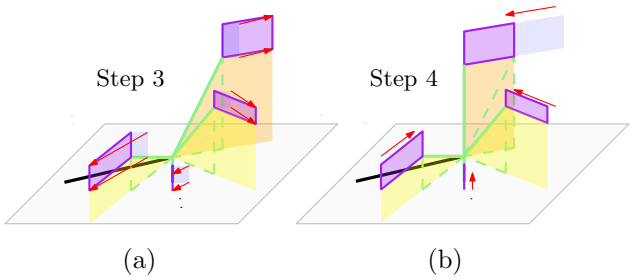


Figure 23: (a) Step 3, (b) Step 4.

Step 4: Move subtrees to the pole. The morphing step $\langle \Gamma_{t+3}, \Gamma_{t+4} \rangle$ is a horizontal morph. See Fig. 23b.

For each $v_j = end(e), e \in K$ we define a horizontal vector of movement depending whether edge e is a vertical edge in the canonical drawing or not.

Case 1: If edge e is a vertical edge in the canonical drawing, we move the vertex $end(e)$ with its subtree $T'(end(e))$, along the vector $(\Gamma_{t+3}(st(e))_x - \Gamma_{t+3}(end(e))_x, \Gamma_{t+3}(st(e))_y - \Gamma_{t+3}(end(e))_y, 0)$, until the image of the edge e becomes vertical. This is the canonical position of $end(e)$ with respect to $st(e)$.

Case 2: If e is not a vertical edge in the canonical drawing, then $\mathcal{C}(end(e))_x - \mathcal{C}(st(e))_x = 1$ and we move $end(e)$ together with the whole subtree $T'(end(e))$ along the vector $(\Gamma_{t+3}(st(e))_x - \Gamma_{t+3}(end(e))_x, \Gamma_{t+3}(st(e))_y - \Gamma_{t+3}(end(e))_y, 0)$ until $end(e)$ reaches the last point with integer coordinates before $(\Gamma_{t+3}(st(e))_x, \Gamma_{t+3}(st(e))_y, \Gamma_{t+3}(end(e))_z)$.

Lemma 30 *Step 4 is crossing-free.*

Proof: Every lifted subtree $T'(end(e_j))$ moves inside the vertical halfplane containing e_j with the pole through $st(e_j)$. That means that the projections to XY_0 of different subtrees $T'(end(e_j))$ plane with the same $st(e_j)$ do not pairwise intersect.

The projections of subtrees with different $st(e_j)$ lie in disjoint disks $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ by Lemma 26, and thus do not intersect during this morph.

Since the projections of any pair of subtrees do not intersect during this morphing step, neither do the subtrees. □

Step 5: Collide planes. Let k be $\Delta(T)$ - the maximum degree of a vertex in the tree T. During the following steps $\langle \Gamma_{t+4}, \Gamma_{t+5} \rangle, \ldots, \langle \Gamma_{t+5+\lceil \log k \rceil}, \Gamma_{t+5+\lceil \log k \rceil+1} \rangle$ for each $v = st(e), e \in K$ we iteratively divide halfplanes around v that contain $T'(end(e)), e \in K$ in pairs. These pairs are formed by neighboring halfplanes in clockwise order around the pole through $st(e)$, see Fig 24. If in some iteration there are odd number of planes around some pole, the plane without a pair does not move in this iteration. In every iteration we map the drawing of one plane of the pair to another simultaneously for each pair. As around each vertex we can have at most k halfplanes, we need at most $\mathcal{O}(\log k)$ mapping steps to collide all planes in one and to rotate the resulting image to $XZ^+_{st(e)}$.

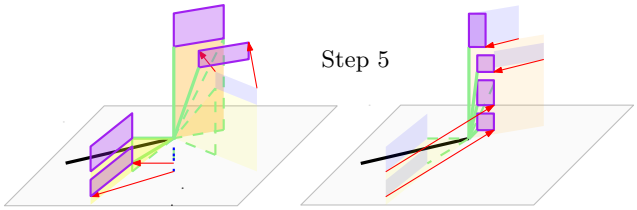


Figure 24: Step 5 in this example consists of two morphing steps.

Lemma 31 *Step 5 is crossing-free.*

Proof: Mapping is a crossing-free morph due to Lemma 5, thus no intersection happen within any single lifted subtree in every iteration.

After any iteration, the subtree $T'(end(e))$, for each edge $e \in K$ is in the canonical position with respect to $st(e)$ that is mapped from $XZ^+_{st(e)}$ to some vertical halfplane with the pole through $st(e)$. This is true in Γ_{t+4} and remains true through all these morphing steps as we do not change the set of halfplanes. In every mapping of one plane to another by the definition of mapping, see Section 3.3, we keep the invariant that the closest to the pole integer point goes to the closest to the pole integer point, other integer points are mapped proportionally.

That means that after mapping one subtree from the plane α to the plane β containing another subtree, we have in plane β a drawing of both subtrees. This drawing is equivalent to their canonical drawing with respect to $st(e)$ which can be obtained by mapping them from the halfplane $XZ^+_{st(e)}$ to β. This implies that during each morphing step and at the end of each morphing step no intersections can happen.

For every $st(e)$, every mapping step is happening inside a region whose projection to the XY_0 plane is contained in its disk $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ where $v = st(e)$. This implies that mappings for different $st(e)$ can not intersect too.

All other, non-processed, vertices are lying in the plane and can not make any intersections. □

5.2 Correctness of the algorithm

In this section, we prove that the algorithm described above is correct, i.e., the following invariants hold.

(I) The drawing of $T'(v_j)$ in Γ_t is the canonical drawing of $T'(v_j)$ with respect to v_j.

(II) The edges that are part of a not yet processed set $K_l \in \mathcal{K}$, $l \geq i$, lie in the XY_0 plane, together with the vertices incident to those edges.

Lemma 32 *Invariants (I) and (II) hold after Step 0 and after performing* $\overline{Lift}(K_i)$ *for each* $1 \leq i \leq m$.

Proof:

(I) We use the induction on the index of the set K_i to prove that the drawing of $T'(v)$ in Γ_t with respect to v for any $v \in V(T)$ is the canonical drawing of $T'(v)$.

Note that after performing the stretching step, before $\overline{Lift}(K_1)$, the whole T lies on the XY_0 plane. Since none of the sets of edges are lifted, condition (I) trivially holds.

Suppose condition (I) holds after $\overline{Lift}(K_{i-1})$, $i \geq 1$, i.e. every $T'(v)$ is in the canonical position with respect to v for every vertex v which is an end vertex of e for some e in K_i before $\overline{Lift}(K_i)$ procedure. After Step 2 of $\overline{Lift}(K_i)$ all end vertices of the edges of K_i have canonical z-coordinates with respect to the start vertices of their edges. After Step 3 of $\overline{Lift}(K_i)$, every lifted subtree $T'(v_j)$, where $v_j = end(e), e \in K_i$ lies on the vertical halfplane that contains e.

Consider the canonical drawing of the subtree $T'(v_j)$ with respect to $st(e)$, where $v_j = end(e), e \in K_i$, along with the edge e, both of them lie on the halfplane $XZ^+_{st(e)}$. Suppose we map the canonical drawing of $T'(v_j)$ together with the edge e to the vertical halfplane that contains e in our current drawing (after Step 3). After Step 4, for any edge e in K_i, we get this equivalent canonical drawing of the tree $T'(end(e))$ along with e on the vertical halfplane that contains e. This implies that for each subtree $T'(end(e))$, its drawing differs from the canonical drawing with respect to $st(e)$ by horizontal mapping from $XZ^+_{st(e)}$.

After Step 5 all vertices $end(e), e \in K_i$ and their lifted subtrees lie in the corresponding halfplanes $XZ^+_{st(e)}$ and have canonical x, y, z-coordinates with respect to the start vertices of their edges.

Note that start vertices of edges in K_i did not have lifted subtrees in the beginning of the procedure $\overline{Lift}(K_i)$ and after this procedure their lifted subtrees consist of the end vertices of the edges in K_i and their lifted subtrees.

As for the other vertices in XY_0, their lifted subtrees had not moved during the procedure $\overline{Lift}(K_i)$ and are in canonical positions with respect to their roots by the assumption.

(II) Again we use the induction on the index of the set K_i. Vertices that lie on edges in a set that is not yet processed are lying within the XY_0 plane.

The invariant (II) holds aftep Step 0 since the entire tree T lies in the XY_0 plane.

Suppose that before the $\overline{Lift}(K_i)$, $i \geq 1$ procedure all vertices that lie on edges in a set that is not yet processed lie in XY_0 plane. End vertices of the edges in K_i can not lie in $K_j, j > i$ by definition of the partition $\mathcal{K}$. Thus after the $\overline{Lift}(K_i)$ in which we move only end vertices

of the edges in K_i and their lifted subtrees, all vertices that lie on edges of sets $K_j, j > i$ will still lie in the XY_0 plane. □

Lemma 33 *Each morphing step during the algorithm maps a grid drawing of tree T to another grid drawing of T.*

Proof: Let us prove this by induction on number of lifted sets K_i.

After the stretching morph all coordinates of all vertices are integer because the constant of stretching is integer and in the given drawing Γ all vertices had integer coordinates, see Lemma 2.

Suppose that $i-1, i \geq 1$ sets are lifted and before the execution of the $\overline{Lift}(K_i)$ procedure all vertices lie on the nodes of the grid. During the $\overline{Lift}(K_i)$ procedure vertices that are not ends of the edges of K_i do not change coordinates at all. Mapping morphs move points with integer coordinates to points with integer coordinates, see Lemma 5. In all other morphing steps the lifted subtrees of end vertices of the edges of K_i are moved along with their roots by integer vectors.

As all coordinates at the beginning of the $\overline{Lift}(K_i)$ were integer and all vectors of movement of all vertices in every step were integer, after the execution of the procedure $\overline{Lift}(K_i)$ all vertices of T still have integer coordinates.

Based on all of the above, each morphing step during the algorithm maps a grid drawing of a tree to another grid drawing of it. □

5.3 Complexity of the algorithm

The estimation on the size of the required grid is similar to the previous algorithm (see Section 4.2):

In the beginning of the algorithm the given drawing $\Gamma = \Gamma_0$ of graph $T = (V, E)$, $|V| = n$ lies in a grid of size $l(\Gamma) \times w(\Gamma) \times 1$.

First step of the algorithm multiplies the required grid size by $\mathcal{S}_1 = 2 \cdot rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1)$. During our algorithm the height of lifted subtrees does not exceed their height in relative canonical drawing with respect to their roots, i.e. does not exceed n. Lifted subtrees take no more than rpw space in x, y-directions from their root after each iteration of $\overline{Lift}()$ procedure and during Steps 1-2 of it. Note that in Step 1 maximum x, y-coordinates of vertices in lifted subtree can not exceed maximum x, y-coordinates of vertices that are the start and the end of the edge along which this lifted subtree is moving. During Steps 3-5 of $\overline{Lift}()$ procedure lifted subtrees lie in disks $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ for some v that is the start of edge in K_i.

Therefore, the grid size required by the algorithm is:

$$(x \times y \times z): \ \mathcal{O}(l(\Gamma) \cdot 2 \cdot rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1) + 2 \cdot rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1)) \times$$

$$\times \mathcal{O}(w(\Gamma) \cdot 2 \cdot rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1) + 2 \cdot rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1)) \times \mathcal{O}(n) =$$

$$\mathcal{O}(d^3(\Gamma) \cdot \log n) \times \mathcal{O}(d^3(\Gamma) \cdot \log n) \times \mathcal{O}(n)$$

There are l sets K_i, where l is the depth of T which is denoted by $dpt(T)$. For every execution of procedure $\overline{Lift}(K_i)$, we need $\mathcal{O}(\log k)$ morphing steps, where $k = \Delta(T)$, the maximum degree of a vertex in T. This implies that the total number of steps in the algorithm is $\mathcal{O}(dpt(T) \cdot \log \Delta(T))$. Note that the whole analysis does not take into account that the path decompositions can be different.

The lemmas proved in this section along with the space and time complexity bounds result in the following.

Theorem 2 *For every two planar straight-line grid drawings Γ, Γ' of a tree T with n vertices, there exists a crossing-free 3D morph $\mathcal{M} = \langle \Gamma = \Gamma_0, \dots, \Gamma_k = \Gamma' \rangle$ that requires a grid $\mathcal{G}$ of size $\mathcal{O}(d^3 \cdot \log n) \times \mathcal{O}(d^3 \cdot \log n) \times \mathcal{O}(n)$, and that consists of $\mathcal{O}(dpt(T) \cdot \log \Delta(T))$ morphing steps, such that every intermediate drawing $\Gamma_i, 0 \leq i \leq k$ is a straight-line 3D grid drawing on the grid $\mathcal{G}$. Here d is the maximum of the diameters of the given drawings Γ, Γ', and $\Delta(T)$ is the maximum degree of a vertex in T.*

Since the maximum degree $\Delta(T) = \mathcal{O}(n)$ for an n vertex tree T, the corollary follows.

Corollary 2 *For every two planar straight-line grid drawings Γ, Γ' of a tree T with n vertices, there exists a crossing-free 3D morph $\mathcal{M} = \langle \Gamma = \Gamma_0, \dots, \Gamma_k = \Gamma' \rangle$ that requires a grid $\mathcal{G}$ of size $\mathcal{O}(d^3 \cdot \log n) \times \mathcal{O}(d^3 \cdot \log n) \times \mathcal{O}(n)$, and that consists of $\mathcal{O}(dpt(T) \cdot \log n)$ morphing steps, such that every intermediate drawing $\Gamma_i, 0 \leq i \leq k$ is a straight-line 3D grid drawing on the grid $\mathcal{G}$. Here d is the maximum of the diameters of the given drawings Γ, Γ'.*

6 The trade-off algorithm

Recall that $\mathcal{L}(T)$ is the set of paths induced by the long-path decomposition of a tree T, see Section 2. Let $Long(T)$ be the subsequence of paths from $\mathcal{L}(T)$, consisting of the paths whose length is at least $\sqrt{n}$, i.e. $Long(T) = \{L_i \in \mathcal{L}(T) : |L_i| \geq \sqrt{n}\}$, let the order in $Long(T)$ be induced by the order in $\mathcal{L}(T)$. We denote by $Short(T)$ the set of trees that are left after deleting from T edges of $Long(T)$, see Fig. 25.

Lemma 34 *1. $|Long(T)| \leq \sqrt{n}$.*

2. For every tree T_i in $Short(T)$, depth of T_i is at most $\lfloor \sqrt{n} \rfloor$.

Proof:

1. Every edge in the tree lies in exactly one path of the long-path decomposition. In a tree T with n nodes there are $n - 1$ edge.

$$n - 1 \geq |E(T)| = |\cup Long(T)| \geq |Long(T)| \cdot \sqrt{n}$$

   Therefore, $|Long(T)| \leq \sqrt{n}$.

2. If there exists a tree T_i which depth is at least $\sqrt{n}$, then long edges from its root create a path that lies in long-path decomposition and has length at least $\sqrt{n}$. That means that this path from the root of T_i should lie in $Long(T)$ and not in $Short(T)$. We have came to a contradiction. □

We divide edges in $Short(T)$ into disjoint sets $Sh_1, \dots Sh_{\lfloor \sqrt{n} \rfloor}$. An edge (v_i, v_j) in tree T_k lies in the set Sh_l if and only if $max(dpt(v_i), dpt(v_j)) = \lfloor \sqrt{n} \rfloor - l + 1$, where $dpt(v)$ is the depth of vertex v in the corresponding tree T_k. Since the maximum depth of any trees T_k is at most $\sqrt{n}$, $Sh_1, \dots Sh_{\lfloor \sqrt{n} \rfloor}$ contain all the edges of these subtrees.

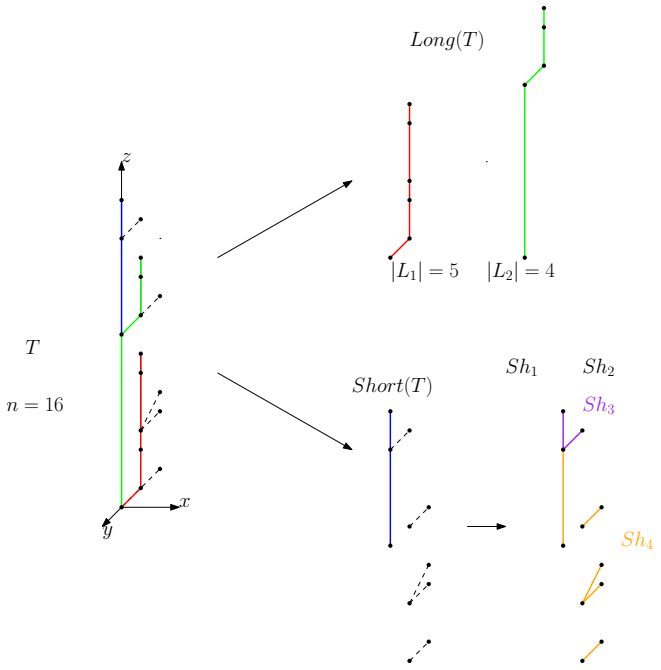


Figure 25: Partition of the edges of T with 16 vertices into the set of paths $Long(T)$ and sets of edges $Sh_i, i \in \{1, \ldots, \sqrt{n} = 4\}$. Sets Sh_1, Sh_2 are empty, as trees in $Short(T)$ have depth at most 2.

The trade-off algorithm. In the beginning we perform a stretching step with $\mathcal{S}_1 = 2 \cdot rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1)$ as described in Section 5. The constant $\mathcal{S}_1$ is large enough to perform *Lift*() procedure described in Section 4. Then, we lift edges from sets Sh_1 to $Sh_{\lfloor \sqrt{n} \rfloor}$ by $\overline{Lift}(Sh_i)$ procedure. This portion of the algorithm takes $\mathcal{O}(\sqrt{n} \cdot \log \Delta(T))$ steps in total by Theorem 2. After that, we lift paths in $Long(T)$ in the reverse order of the order of their appearance in $Long(T)$. Since $|Long(T)| \leq \sqrt{n}$ and each execution of the *Lift*() procedure for a single path consists of a constant number of morphing steps, this portion of the algorithm requires $\mathcal{O}(\sqrt{n})$ steps as well. We conclude with the following theorem, that states the main result of this paper.

Theorem 3 *For every two planar straight-line grid drawings Γ, Γ' of a tree T with n vertices there exists a crossing-free 3D-morph $\mathcal{M} = \langle \Gamma = \Gamma_0, \ldots, \Gamma_l = \Gamma' \rangle$ that requires $\mathcal{O}(\sqrt{n} \cdot \log \Delta(T))$ steps and a grid G of size $\mathcal{O}(d^3 \cdot \log n) \times \mathcal{O}(d^3 \cdot \log n) \times \mathcal{O}(n)$, where d is the maximum of the diameters of the given drawings, $\Delta(T)$ is the maximum degree of vertices in T. In this morph every intermediate drawing $\Gamma_i, 1 \leq i \leq l$, is a straight-line 3D grid drawing on the grid G.*

Corollary 3 *As $\log \Delta(T) \leq \log n$, it is possible to morph between Γ, Γ' using $\mathcal{O}(\sqrt{n} \log n)$ steps.*

Corollary 4 *It is possible to morph between Γ, Γ' using $\mathcal{O}(\sqrt{n})$ steps if maximum degree of a vertex of T is a constant.*

7 Conclusion

In this paper, we presented an algorithm that morphs between two planar grid drawings of an n-vertex tree T in $\mathcal{O}(\sqrt{n}\log n)$ steps such that all intermediate drawings are crossing-free 3D grid drawings and lie inside a 3D grid of polynomial size. Arseneva et al. [2] proved that $\mathcal{O}(\log n)$ steps are enough to morph between two planar grid drawings of an n-vertex tree T where intermediate drawings are allowed to lie in $\mathbb{R}^3$ but they did not guarantee that intermediate drawings have polynomially bounded resolution. Several problems are left open in this area of research. We mention a few of them here.

1. It is interesting to prove a lower bound on the number of morphing steps if intermediate drawings are allowed to lie in $\mathbb{R}^3$.
2. It is also interesting to prove a lower bound for this problem along with the additional constraint of polynomially bounded resolution.
3. Is it possible to morph between two planar grid drawings in $o(n)$ morphing steps for a richer class of graphs (e.g. outer-planar graphs) than trees if we are allowed to use the third dimension? Recently, Buchin et al. gave an algorithm that uses the third dimension to morph between any two, possibly topologically non-equivalent, planar drawings of the same graph in $\mathcal{O}(n^2)$ linear morphing steps but there is still no nontrivial lower bound [6].
4. Is there a trade off between the number of steps required and the volume of the grid needed for the morph?

References

[1] Soroush Alamdari, Patrizio Angelini, Fidel Barrera-Cruz, Timothy M Chan, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Penny Haxell, Anna Lubiw, Maurizio Patrignani, Vincenzo Roselli, Sahil Singla, and Bryan T Wilkinson. How to morph planar graph drawings. *SIAM Journal on Computing*, 46(2):824–852, 2017. doi:10.113716M1069171.

[2] Elena Arseneva, Prosenjit Bose, Pilar Cano, Anthony D'Angelo, Vida Dujmovic, Fabrizio Frati, Stefan Langerman, and Alessandra Tappini. Pole dancing: 3d morphs for tree drawings. *J. Graph Algorithms Appl.*, 23(3):579–602, 2019. doi:10.7155/jgaa.00503.

[3] Fidel Barrera-Cruz, Manuel Borrazzo, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Vincenzo Roselli. How to morph a tree on a small grid. *Discrete & Computational Geometry*, 67(3):743–786, 2022. doi:10.1007/s00454-021-00363-8.

[4] Michael A Bender and Martín Farach-Colton. The level ancestor problem simplified. *Theoretical Computer Science*, 321(1):5–12, 2004. doi:10.1007/3-540-45995-2_44.

[5] Therese Biedl. Optimum-width upward drawings of trees. *arXiv preprint arXiv:1506.02096*, 2015.

[6] Kevin Buchin, Will Evans, Fabrizio Frati, Irina Kostitsyna, Maarten Löffler, Tim Ophelders, and Alexander Wolff. Morphing planar graph drawings through 3d. In *48th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM '23)*, pages 80–95. Springer, 2023. doi:10.1007/978-3-031-23101-8_6.

[7] Craig Gotsman and Vitaly Surazhsky. Controllable morphing of compatible planar triangulations. *ACM Transactions on Graphics (TOG)*, 20(4):203–231, 2001. doi:10.1145/502783.502784.

[8] Craig Gotsman and Vitaly Surazhsky. Guaranteed intersection-free polygon morphing. *Computers & Graphics*, 25(1):67–75, 2001. doi:10.1016/S0097-8493(00)00108-4.

[9] Craig Gotsman and Vitaly Surazhsky. Intrinsic morphing of compatible triangulations. *International Journal of Shape Modeling*, 9(02):191–201, 2003. doi:10.1142/S0218654303000115.

[10] Joel Hass, Jeffrey C. Lagarias, and Nicholas Pippenger. The computational complexity of knot and link problems. *Journal of the ACM*, 46(2):185–211, 1999. doi:10.1145/301970.301971.

[11] Robert E Horton. Erosional development of streams and their drainage basins; hydrophysical approach to quantitative morphology. *Geological society of America bulletin*, 56(3):275–370, 1945. doi:10.1130/0016-7606(1945)56[275:EDOSAT]2.0.CO;2.

[12] Aleksandra Istomina, Elena Arseneva, and Rahul Gangopadhyay. Morphing tree drawings in a small 3d grid. In *International Conference and Workshops on Algorithms and Computation*, pages 85–96. Springer, 2022. doi:10.1007/978-3-030-96731-4_8.

[13] Marc Lackenby. The efficient certification of knottedness and thurston norm. *Advances in Mathematics*, 387:107796, 2021. doi:10.1016/j.aim.2021.107796.

[14] A Stahler. Hypsometric (area altitude) analysis of erosional topology. *Geological Society of America Bulletin*, 63:1117–1142, 1952. doi:10.1130/0016-7606(1952)63[1117:HAAOET]2.0.CO;2.

[15] Günter M Ziegler. *Lectures on polytopes*, volume 152. Springer Science & Business Media, 2012.