Санкт-Петербургский государственный университет

ИСТОМИНА Александра Андреевна

Выпускная квалификационная работа

О морфинге изображений графов в плоскости и пространстве

Уровень образования: бакалавриат
Направление: 01.03.01 «Математика»
Основная образовательная программа: СВ.5000.2017 «Математика»

Научный руководитель:
Ph. D. in Informatics,
доцент факультета МКН СПбГУ
Арсеньева Елена Александровна

Рецензент:
PhD in Applied Mathematics
and Computer Science,
Postdoctoral researcher,
Universite libre de Bruxelles
Pilar Cano

Санкт-Петербург
2021

Saint-Petersburg State University

ISTOMINA Aleksandra

Graduation qualification thesis

On morphing tree drawings in 2D and 3D

Bachelor's program
Specialization and code: 01.03.01 "Mathematics"
Cipher of the EP: CB.5000.2017 "Mathematics"

Thesis supervisor:
Associate Professor
Ph. D. in Informatics,
Elena Arseneva

Reviewer:
PhD in Applied Mathematics
and Computer Science,
Postdoctoral researcher,
Universite libre de Bruxelles
Pilar Cano

Saint-Petersburg
2021

Contents

Introduction [1]

Given a graph G with n vertices, 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 family of intermediate drawings. One is interested in well-behaved morphs, i.e., those that preserve essential properties of the initial and final drawing at any moment. Usually this preserved property is that the drawing is *crossing-free*; such morphs are called *crossing-free morphs.* This concept finds applications in multiple domains, such as animation, modeling, and computer graphics [12–15].

A drawing of a graph 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 non-crossing drawing of a graph in the plane is called *planar.*

Although the study of crossing-free graph morphing began long time ago, it is currently an active area of research in computational geometry and graph drawing with fascinating open problems. In 1944, Cairns [11] gave an existential proof that a planar morph exists between any two topologically equivalent planar straight-line drawings of a maximal planar graph. The first algorithmic result was given by Thomassen [21], who proposed a crossing-free morph between any two *topologically* equivalent (with respect to the ordering of edges around vertices) planar straight-line drawings of a planar graph.

Lately there is a line of research studying morphs of straight-line drawings, that are crossing-free and where the trajectories of vertices are simple. Of particular interest is *linear* morph, that 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. The morph $\langle \Gamma_i, \Gamma_{i+1} \rangle$ is known as a morphing step. For brevity, we call morphing steps simply steps in this thesis. In each step of a linear morph all the vertices of G move simultaneously along a straight-line segment at a constant speed.

A series of results [2–4, 7] led up to a seminal paper by Alamdari et al. [1] showing that for any two topologically equivalent planar drawings of a graph there is a linear $2D$ morph that transforms one drawing to the other in $\Theta(n)$ steps. This bound is asymptotically optimal in the worst case even when the graph G is a path.

A natural further question to ask is whether and how the situation changes if the third dimension is involved. Such question for general graphs seems not to be easy. It is tightly connected to the famous problem of *unknot recognition*, that is known to be in **NP**∩**co-NP** [16, 18], and finding a polynomial-time algorithm or proving its **NP**-hardness is a long-standing open problem. If the given graph is a tree, the worst-case tight bound of $\Theta(n)$ steps holds for $3D$ crossing-free linear morph [5] (and the lower-bound example is again a path). Even more remarkably, if both the initial and the final drawing of the tree are planar, then there is a morph with $\mathcal{O}(\log n)$ number of morphing steps [5].

The above algorithmic results [1, 5] have one common drawback that is crucial from the practical point of view. In their intermediate steps, they essentially use infinitesimal or very small distances, as compared to distances in the initial and the final drawings. Thus the amount of space that is needed for storing the numbers arising throughout the algorithm is very large. Having very large and very small distances also affects the aesthetical aspect, that might be important as well. This raises a demand for morphing algorithms with restriction that after each step we get a *grid drawing*, where vertices map to nodes of a grid, and that the grid is small, i.e., whose size is polynomial in the size of the graph and parameters on the input drawings.

For a straight-line planar grid drawing Γ of a graph, there are two crucial parameters. The *resolution* of Γ is the ratio between the maximum edge length and the minimum vertex-edge distance. The second parameter is the *area* required for the drawing. In the $3D$ case, the resolution and the *volume* of a drawing are defined similarly. For a $2D$ straight-line grid drawing of T, it was shown that if the area of the grid is polynomially bounded, then the resolution of the drawing is also polynomially bounded [6]. For a $3D$

[1]Introduction and Section 2.3 were written together with Rahul Gangopadhyay as a part of our joint paper (manuscript in preparation).

straight-line grid drawing of T, one can analogously show that if its volume is polynomially bounded, then its resolution is also polynomially bounded.

Barrera-Cruz et al. [8] gave an algorithm to morph between two straight line planar Schnyder drawings of the same triangulated graph using $\mathcal{O}(n^2)$ linear morphing steps. They also proved that all the intermediate drawings lie in an $\mathcal{O}(n) \times \mathcal{O}(n)$ sized grid. Very recently, Barrera-Cruz et al. [6] gave an algorithm that linearly morphs a planar straight-line grid drawing Γ of an n-vertex rooted tree to another planar straight-line grid drawing Γ' of the same tree in $\mathcal{O}(n)$ morphing steps such that each intermediate drawing is also a planar straight-line grid drawing. Throughout the morphing process, the grid size is polynomially bounded. In particular, during the entire morph the maximum length of the grid is $\mathcal{O}(D^3 n \cdot L)$ and the maximum width of the grid is $\mathcal{O}(D^3 n \cdot W)$, where $L = max\{l(\Gamma), l(\Gamma')\}$, $W = max\{w(\Gamma), w(\Gamma')\}$, $D = max\{L, W\}$ and $l(\Gamma)$ and $w(\Gamma)$ are respectively the length and the width of the drawing Γ. Note that D is $\Omega(\sqrt{n})$.

In this thesis we study a possibility to morph one straight-line grid drawing Γ of a tree to another such drawing Γ' in *sublinear* number of steps by employing the third dimension for intermediate steps. Effectively we ask the same question as in [5], but now with additional restriction for all the drawings throughout the algorithm to live in a small grid. We give two algorithms that require $\mathcal{O}(n)$ steps and $\mathcal{O}(\sqrt{n} \log n)$ steps, respectively. All the intermediate drawings require a $3D$-grid having $\mathcal{O}(d^3 \cdot \log n)$ length, $\mathcal{O}(d^3 \cdot \log n)$ width and $\mathcal{O}(n)$ height, where $d = max(d(\Gamma), d(\Gamma'))$ is maximum of the diameters of the given drawings Γ and Γ'. Note that at the expense of using an extra dimension we significantly decrease the number of morphing steps. Also, at the cost of using linear amount of height in the third dimension, we use much less area than [6]. To the best of our knowledge, our algorithm is the first one that morphs a planar grid drawing of a tree to another such drawing in sublinear number of steps such that intermediate drawings are $3D$ grid drawings of polynomial volume.

1 Preliminaries and Definitions

In this section we introduce some definitions and preliminaries which are used later in the various parts of the thesis. Following definitions and notations are taken from [6, 7].

Tree drawings For a tree T, let $|T|$ denote the number of vertices in T and let $r(T)$ denote the root of T. Let $V(T)$ be the set of vertices of tree T, $E(T)$ be the set of edges of T.

Let $T(v)$ denote the subtree of T rooted at a vertex v. We say $T(v)$ is a *partial subtree* of T rooted at v if it is connected but does not contain all the vertices present in the subtree rooted at v.

In a *straight-line drawing* of a tree T, each node u is represented by a point in the plane and each edge is represented by a straight-line segment connecting its end-points.

In a $3D$*-grid drawing* of T, each vertex is mapped to a point with integer coordinates in $\mathbb{R}^3$ and edges are drawn as line segments connecting its endpoints. A $3D$-grid drawing of T is said to be *crossing-free* if images of no two edges intersect except, possibly, at common end-points.

A $2D$-grid drawing of T is a special case of $3D$-grid drawing when each vertex is mapped to a point with integer coordinates in a 2-dimensional plane. In this thesis, we consider all planar drawings lie on $X0Y$ plane. A crossing-free $2D$-grid drawing is said to be a *planar grid drawing*.

We use Γ to denote both the planar grid drawing and $3D$ grid drawing of T if it is clear from the context.

With a slight abuse of notation, let $B(\Gamma(v), r)$ denote the open disc of radius r in X0Y plane centered at $\Gamma(v)$, i.e., the point in the plane representing the vertex v in the drawing Γ.

Unless stated otherwise, by the *projection* we mean the vertical projection to X0Y plane, i.e. projection of point $p = (x', y', z')$ is a point $pr(p) = (x', y', 0)$. Projection of an edge or the whole tree, etc. is defined analogously.

Parameters of the $3D$ crossing free grid drawing Let us define the *length* of a $3D$ crossing free grid drawing Γ, denoted by $l(\Gamma)$, as the maximum absolute difference between the x-coordinates of a pair of vertices in Γ.

$$l(\Gamma) = \max_{1 \le i,j \le n} (|\Gamma(v_i)_x - \Gamma(v_j)_x|), \ \textit{where } v_i, v_j \in V(T).$$

Let $w(\Gamma)$ denote the *width* of a $3D$ crossing free grid drawing Γ and it is defined as the maximum absolute difference between the y-coordinates of a pair of vertices in Γ.

$$w(\Gamma) = \max_{1 \le i,j \le n} (|\Gamma(v_i)_y - \Gamma(v_j)_y|),, \ \textit{where } v_i, v_j \in V(T).$$

Similarly, $h(\Gamma)$ denotes the *height* of a $3D$ crossing free grid drawing Γ and it is defined as the maximum absolute difference between the z-coordinates of a pair of vertices in Γ.

$$h(\Gamma) = \max_{1 \le i,j \le n} (|\Gamma(v_i)_z - \Gamma(v_j)_z|),, \ \textit{where } v_i, v_j \in V(T).$$

Let us define the *diameter* of a drawing Γ, denoted by $d(\Gamma)$, as the ceiling of the maximum euclidean distance between any pair of vertices in Γ.

$$d(\Gamma) = \lceil \max_{1 \le i,j \le n} (|\Gamma(v_i) - \Gamma(v_j)|) \rceil, \ \textit{where } v_i, v_j \in V(T).$$

Clearly, the diameter of any drawing Γ is always an integer, i.e., $d(\Gamma) \in \mathbb{Z}$.

Given a vertex v and an edge e of the tree T, $dist_\Gamma(v, e)$ denotes the distance between $\Gamma(v)$ and the $\Gamma(e)$ in the drawing Γ of T. Similarly, for a pair of vertices $v, u \in V(T)$, $dist_\Gamma(v, u)$ is the distance between the images of the vertices in the drawing Γ of T.

Observation 1. *Let us consider a $3D$ grid drawing Γ of T. Its diameter $d(\Gamma)$ is at least* $\max(l(\Gamma), w(\Gamma), h(\Gamma))$ *and at most* $\sqrt{3}\cdot\max(l(\Gamma), w(\Gamma), h(\Gamma))$. *If Γ is a planar grid drawing of T, then the diameter $d(\Gamma)$ is at least* $\max(l(\Gamma), w(\Gamma))$ *and at most* $\sqrt{2}\cdot\max(l(\Gamma), w(\Gamma))$ *since $h(\Gamma) = 0$.*

Since Observations 1 holds, in order to estimate the required space, it is enough to estimate $d(\Gamma)$.

The following lemma has already been proved in [6]. For completeness of the work, we give an alternative proof of this result in Appendix.

Lemma 1. *Let every vertex of the tree T lie in the lattice points of the grid in the drawing Γ and let $d(\Gamma)$ be the diameter of Γ. In Γ, the distance between a vertex v and the line that contains an edge e of T, such that e is not incident to v, is at least $\frac{1}{d(\Gamma)}$.*

Lemma 2. *Let Γ be a planar grid drawing of T. Let $\Gamma(v) = (\Gamma(v)_x, \Gamma(v)_y)$ denote the point which is the image of the vertex v in Γ. Let us consider a new grid drawing Γ_1. For each $v \in V(T)$, let $\Gamma_1(v)$ be equal to $(c \cdot \Gamma(v)_x, c \cdot \Gamma(v)_y)$ where c is an integral constant. Then the distance between any two distinct vertices in Γ_1 is at least c.*

Proof. For any pair of distinct vertices v_i, v_j, either $|\Gamma(v_i)_x - \Gamma(v_j)_x|$ or $|\Gamma(v_i)_y - \Gamma(v_j)_y|$ is at least 1 since all vertices have integer coordinates in Γ. This implies that, for each pair of distinct vertices v_i, v_j, either $|\Gamma_1(v_i)_x - \Gamma_1(v_j)_x|$ or $|\Gamma_1(v_i)_y - \Gamma_1(v_j)_y|$ is at least c since each coordinate of each vertex in Γ is multiplied by c to obtain Γ_1. □

Path decomposition *Path decomposition* $\mathcal{P}$ of a tree T is a decomposition of its edges into a set of disjoint paths. Below we describe an algorithm for building different path decompositions by path deletion procedure.

For a fixed tree T, we choose a path P in T based on a certain criteria and store the path in $\mathcal{P}$. Removal of the path creates disjoint components. We recurse on the remaining connected components. The base case of the algorithm is that we have a single path and its removal creates an empty graph. At the termination of the algorithm, the set $\mathcal{P}$ contains disjoint paths whose union is T.

Note that the rule of how to choose the path may differ. In particular, $\mathcal{P}$ can be the long-path decomposition or the heavy-path decomposition of T defined below.

The *depth of a vertex v* in T, denoted by $dpt(v)$, is defined as the length of the path from $r(T)$ to v. *Head* of the path P, denotes as $head(P)$, is the vertex $x \in P$ with the minimum depth in tree T. Let *internal vertices* of the path be all vertices except $head(P)$. Note, that every edge in tree belongs to exactly one path in a path decomposition.

We define a linear ordering among the paths induced by a fixed path decomposition of T. We say that the path P' succeeds P, i.e., $P' \succ P$, if and only if P' is deleted before P during the execution of the path decomposition algorithm. Otherwise we say that P precedes P', i.e., $P \prec P'$. Note that the subtree of each internal vertex of a path P is a subset of the union of the paths that precede P: $\forall v \in P, v \neq head(P) : T(v) \subset \cup_{P' \prec P} P' \cup P$.

Long-path decomposition [9] The *long-path decomposition* $\mathcal{L} = \{L_1, \ldots, L_m\}$ of tree T into paths is defined as follows: let the *depth* of a tree be the maximum depth of its nodes. For each non-leaf node, define a *long edge* to go from the node to its child with the deepest subtree (ties are broken arbitrarily). Preferred paths $L_i, i = 1, \ldots, m$ are defined as the maximal paths in the graph containing only long edges. Order the paths of $\mathcal{L}$ with respect to their length. For an example of long-path decomposition see Fig. 2b.

For convenience, we will also consider the short edge immediately above a preferred path to be a part of the path. Short edges that are incident to leaves are considered as separate paths.

Let us consider operation of deleting edges of the longest path in T. After every iteration of deleting the path, we get a set of disjoint trees in which we look for the next longest path. Note that, if ties are broken in the same way, the set of paths that we get from path deletion algorithm along with their ordering are equal to $\mathcal{L}$.

Heavy-rooted-pathwidth decomposition [5] The *Strahler number* or *Horton-Strahler number* of a tree is a parameter which was introduced by Horton and Strahler [17, 19, 20]. The same parameter was recently rediscovered by Biedl [10] with the name of *rooted pathwidth* when addressing the problem of computing upward tree drawings with optimal width.

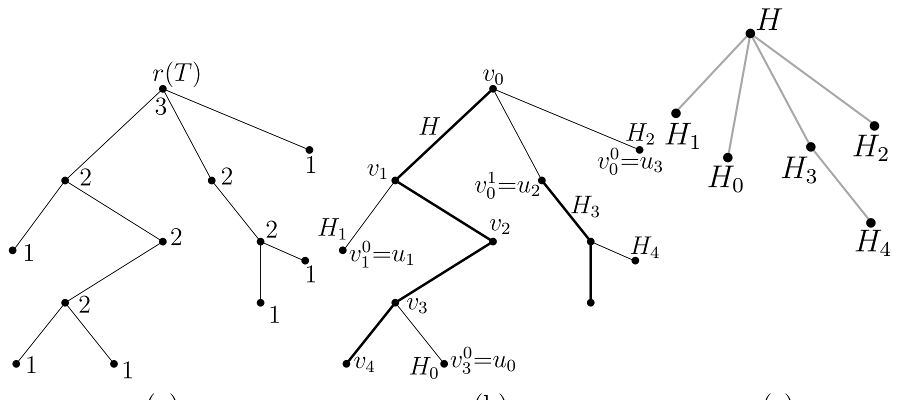

Figure 1: [5] The illustration (a) shows, for each vertex v of a tree T, the number $rpw(T(v))$. In particular, $rpw(T) = 3$. The illustration (b) shows with bold lines the heavy edges of T forming the heavy paths $H, H_0, \ldots, H_4$. The illustration (c) shows the path tree of T

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 $r(T)$ has rooted pathwidth equal to k, and $rpw(T) = k + 1$ if more than one subtree rooted at a child of $r(T)$ has rooted pathwidth equal to k; see Fig. 1a. Clearly $rpw(T)$ is an integer number.

The *heavy-rooted-pathwidth decomposition* of a tree T is defined as follows, see Fig. 1b. For each non-leaf vertex v of T, let c^* be the 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 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. 1c. The parent of a heavy path P in the path tree is the heavy path that contains the parent of the vertex with the minimum depth of P. The root of the path tree is the heavy path containing $r(T)$.

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 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 in any order. Let $L = u_0, u_1, \ldots, u_{l-1}$ be the sequence of the light children 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. It is known [10] that the height of the path tree of an n-vertex tree T is at most $rpw(T) \in \mathcal{O}(\log n)$.

For convenience, we will later consider the light edge immediately above a heavy path to be a part of the path, see Fig. 2a. With this assumption the heavy-path decomposition can be obtained from path deletion algorithm, when we always choose a heavy path that contains the root of the tree. When it is clear from the context we will refer to $\mathcal{H}(T)$ simply as $\mathcal{H}$.

Canonical drawing of a tree [5]. Canonical 3D drawing of a tree T, introduced in [5], is the crossing-free straight-line 3D drawing of T that maps each vertex v of T to its *canonical position* $\mathcal{C}(v)$ defined as follows; refer to Fig. 2a:

- First, we set $\mathcal{C}(v_0) = (0, 0, 0)$ for the root v_0 of T.
- Second, for each $i = 1, \ldots, k-1$, 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 $i = 1, \ldots, k-1$ and for each $j = 0, \ldots, t_i$, 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 $i = 0, \ldots, k-2$ and each $j = 0, \ldots, t_i$, 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 $\mathcal{C}(v_i^j)$.

Note that $max(v_x | v \in V(T)) = rpw(T) - 1$, where v_x is the x-coordinate of $\mathcal{C}(v)$. Since tree T never changes throughout our algorithm we refer to $rpw(T)$ as to rpw.

Relative canonical drawing of a subtree. Let us consider a subtree $T(v)$ of T rooted at v. We define a canonical drawing of $T(v)$ with respect to the canonical $3D$ drawing of T. Let us denote it by $\mathcal{C}_{T_v}$. Let us map the vertex v to the origin. Then, the vertex v_i which belongs to the subtree $T(v)$ is placed at the point $\mathcal{C}_{T_v}(v_i) = (\mathcal{C}(v_i)_x - \mathcal{C}(v)_x, \mathcal{C}(v_i)_y - \mathcal{C}(v)_y, \mathcal{C}(v_i)_z - \mathcal{C}(v)_z)$. Note that we denote by $\mathcal{C}_{T_v}$ all such $3D$ drawings of $T(v)$ which are obtained by translating all points of $\mathcal{C}_{T_v}$ along a constant vector. Also, note that such a canonical drawing can be defined for any partial subtree rooted at v.

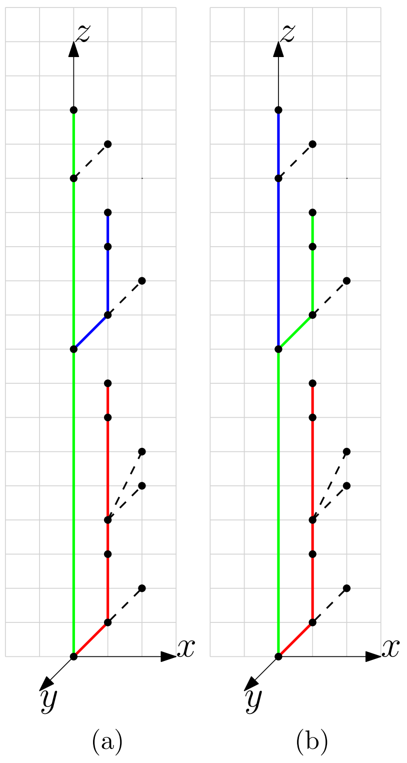

Figure 2: An example of canonical drawing of a tree. In the illustration (a) heavy paths are drawn in different colors, paths consisting of one edge are drawn dashed. In the illustration (b) paths of long-path decomposition are drawn in different colors, paths consisting of one edge are drawn dashed.

Observation 2. *Let us consider $\mathcal{C}_{T_v}$. The entire drawing lies inside a bounding box having height $|T(v)|$ and width $rpw(T(v))$ in the positive XZ (i.e., both X and Z are positive) quarter-plane from vertex v.*

Proof. The proof follows from the construction of $\mathcal{C}_{T_v}$. □

Morph Let Γ and Γ' be two planar straight-line drawings of T. Then a morph $\mathcal{M}$ is a sequence $\langle \Gamma_1, \Gamma_2, \ldots, \Gamma_k \rangle$ of 3D straight-line drawings of T such that $\Gamma_1 = \Gamma, \Gamma_k = \Gamma'$, and $\langle \Gamma_i, \Gamma_{i+1} \rangle$ is a linear morph, for each $i = 1, \ldots, k-1$.

A *linear morph* $\langle \Gamma_i, \Gamma_{i+1} \rangle$ is such that each image of the vertex moves along a straight-line segment at a constant speed; that is, assuming that the morph happens between time $t = 0$ and time $t = 1$, the position of a vertex v at any time $t \in [0, 1]$ is $(1 - t)\Gamma_i(v) + t\Gamma_{i+1}(v)$.

The complexity of a morph $\mathcal{M}$ is then measured by the number of its morphing steps, i.e., by the number of linear morphs it consists of. In the following, a morphing step is sometimes simply called a step.

2 Tools for morphing algorithms

In this section we present four morphing procedures: stretching step with a constant $\mathcal{S}_1$ (2.1), mapping and rotation around a pole (2.2), shrinking lifted subtrees (2.3) and turning in horizontal plane (2.4), that are building blocks of our algorithms: morphing through lifting paths (3) and morphing through lifting edges (4).

2.1 Stretching step with a constant $\mathcal{S}_1$

We assume that the original drawing Γ of the tree is lying in the horizontal plane X0Y. Stretching morph multiplies coordinates of all vertices by a fixed constant $\mathcal{S}_1 > 1$, thereby "stretching" the vertices apart.

During this linear morph each coordinate of each of each vertex in Γ is multiplied by a fixed integer constant $\mathcal{S}_1 > 1$. That is, every vertex v of T moves from $\Gamma(v) = (v_x, v_y, 0)$ to $\Gamma_1(v) = (\mathcal{S}_1 v_x, \mathcal{S}_1 v_y, 0)$.

Lemma 3. *Stretching morph is crossing-free.*

Proof. For each $t \in [0,1]$ the drawing Γ^t, where the image of any vertex v of T is $\{t\Gamma_1(v) + (1-t)\Gamma(v)\}$ is the same drawing Γ scaled by $t\mathcal{S}_1 + (1-t)$. Since the original drawing Γ was crossing-free, so is the drawing Γ^t. $\square$

Lemma 4. *For any pair of distinct vertices v_i, v_j disks $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2})$, $B(\Gamma_1(v_j), \frac{\mathcal{S}_1}{2})$ in X0Y plane are disjoint. In Γ_1 disk $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2\cdot d(\Gamma)})$ for a vertex v_i does not enclose any other vertices or any portion of edges non-incident to v_i.*

Proof. As disk $B(\Gamma(v_i), 1)$ contains no other vertices $v_j, j \neq i$, then $B(\Gamma_1(v_i), \mathcal{S}_1)$ does not contain other vertices too since $\mathcal{S}_1 > 1$. Disk $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2})$ — disk with half that radius — does not intersect with any $B(\Gamma_1(v_j), \frac{\mathcal{S}_1}{2})$ for $j \neq i$.

Due to Lemma 1 no non-incident edges intersected 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. Other vertices do not lie in $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2\cdot d(\Gamma)})$ as $B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2\cdot d(\Gamma)}) \subset B(\Gamma_1(v_i), \frac{\mathcal{S}_1}{2})$. $\square$

Lemma 5. *For every vertex v and every edge $e = (v, u)$ in Γ_1 there is a lattice point z such that $z \in e$ and $z \in B(\Gamma_1(v_i), d(\Gamma))$.*

Proof. Every edge $e = (u, v)$ from Γ_0 has been stretched by $\mathcal{S}_1$.

If $\Gamma_0(u) = (u_x, u_y, 0)$ and $\Gamma_0(v) = (v_x, v_y, 0)$, then $\Gamma_1(u) = (u_x \cdot \mathcal{S}_1, u_y \cdot \mathcal{S}_1, 0)$ and $\Gamma_1(v) = (v_x \cdot \mathcal{S}_1, v_y \cdot \mathcal{S}_1, 0)$.

The integral point $z \in e$ and $z \in B(\Gamma_1(v), d(\Gamma))$ is $(u_x \cdot \mathcal{S}_1 + v_x - u_x, u_y \cdot \mathcal{S}_1 + v_y - u_y, 0)$. $\square$

2.2 Mapping and rotation around a pole

Let *pole* be a vertical line in $3D$. The *pole through* (x', y') is a vertical line through point $(x', y', 0)$.

Recall from the definition that canonical drawing of T lies on the plane X0Z. To morph the given drawing Γ to canonical drawing we get canonical drawings of subtrees of vertices of T starting from leaves. Our goal is to have the non-processed portion of the tree in X0Y plane while the processed portion of subtrees of vertices lie in vertical half-planes containing the pole through the corresponding vertex. We use vertical half-planes and morph the subtrees between them.

Let α, β be half-planes of vertical planes and let l be their vertical line of intersection (the pole of mapping) going through a point (x', y') with integer coordinates. Suppose $\angle(\alpha, \beta) \notin \{0, \pi\}$ and α, β contain infinitely many points with integer coordinates (and infinitely many poles through integer points). Suppose we have some drawing Γ in a half-plane α and we want to move it to a half-plane β (obtaining a drawing Γ') in one morphing step.

During this *mapping* step, vertices move along a horizontal vector from α to β. The direction of this vector is common for all vertices of Γ and is defined by positions of α and β in the following way. Let d_α, d_β be the minimum non-zero distance from the l to the integer points lying in the corresponding half-plane, let p_α, p_β be points for which

$dist(l, p_\alpha) = d_\alpha, dist(l, p_\beta) = d_\beta$ and $p_{\alpha_z} = p_{\beta_z}$. Note that we can choose points p_α and p_β in such way that they have same z-coordinate and lie on the same horizontal plane. *Vector of mapping* u is defined as $\frac{p_\beta - p_\alpha}{|p_\beta - p_\alpha|}$ and is the direction of movement for every vertex in Γ. The length of the vector for every point is defined in such a way that after the step every vertex of Γ lies in half-plane β. See Fig. 3.

Note that the length of the vector of movement of any vertex v in Γ is proportional to its distance to the pole. Let l_v denote the point on the pole l with the same z coordinate as v, once again for each $t \in [0, 1]$ we denote by Γ^t the drawing, where the image of any vertex v of T is $\{t\Gamma'(v) + (1-t)\Gamma(v)\}$. For any two vertices v_1, v_2 from Γ with the same height that are being mapped to the half-plane β, from proportionality of distances, we can cay that at any moment $t \in [0, 1]$ for the intermediate drawing Γ^t we have $\angle((\Gamma^t(v_1), l_{v_1}), \alpha) = \angle((\Gamma^t(v_2), l_{v_1}), \alpha) = \angle((\Gamma^t(v_2), l_{v_2}), \alpha)$. If the angle between the edge $(\Gamma^t(p), l_p)$ and the plane α is the same for all points at some height, then it is the same for all vertices in Γ^t, because points with the same x, y-coordinates had the same vector of movement. From that fact we get the following lemma.

Lemma 6. *For each $t \in [0, 1]$ the drawing $\Gamma^t = \{t\Gamma' + (1-t)\Gamma\}$ lies in one half-plane. Moreover, for each $t \in [0, 1]$ this half-plane has the vertical borderline l, which is a borderline of half-planes α and β.*

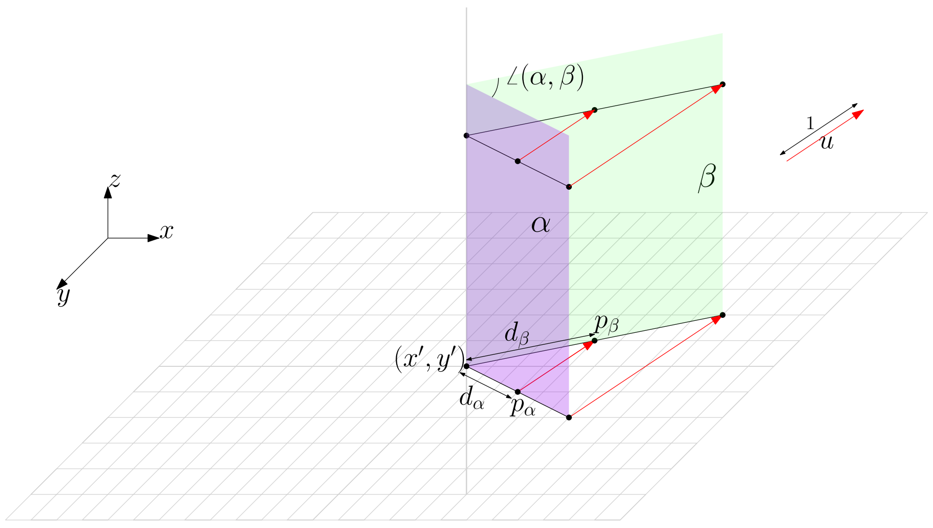

Figure 3: Half-planes α, β (shown in violet and green respectively) sharing a common pole through point (x', y') and their vector of mapping u (shown in red)

For the next lemma we will need the following result from [1]. *A unidirectional morph* is a linear morph in which all the vertices move along parallel lines.

Proposition 1. *[1] 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 7. *Mapping step is a crossing-free morph.*

Proof. z-coordinates of vertices do not change during the rotation, so it can be decomposed as union of 2D morphs each of which is a horizontal slice of 3D mapping at a certain height, where height is some real value. If any crossing happens during mapping it can be described as crossing in 2D morph at some height. For each real value of height we have some points of T: vertices or points of edges.

For height z' we fix a morph $M_{z'}$ for which graph $T_{z'}$ to morph is intersection of T with horizontal plane $z = z'$. Morph $M_{z'}$ moves $\Gamma_{z'} = \Gamma \cap \{z = z'\}$ to $\Gamma'_{z'} = \Gamma' \cap \{z = z'\}$.

Mapping is a unidirectional 3D morph by definition, so $M_{z'}$ is a unidirectional 2D morph. Every two vertices of $\Gamma_{z'}$ lie in one horizontal line l_Γ with the pole-point in $z = z'$, and as they move along the parallel lines not parallel to l_Γ they can not cross throughout the movement.

For every edge vw and vertex u if u was closer to the pole than vw in Γ, then it is closer to the pole in Γ' as order of vertices from the pole does not change in mapping. Then $M_{z'}$ meets the conditions of Cor. 1 and vw and u do not cross during $M_{z'}$. As no crossings happen at any height, mapping is a crossing-free morph. $\square$

Let *rotation* be a mapping in case when half-planes α, β are parallel to planes $X0Z$ and $Y0Z$ respectively, and thus $\angle(\alpha, \beta) = \frac{\pi}{2}$, see Fig. 4. Vector of mapping u of rotation is then lying in the set:

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

XZ_v^+ is a half-plane parallel to X0Z with vertical border-line going through vertex v in X-positive direction from that pole. XZ_v^-, YZ_v^-, YZ_v^+ are defined analogously.

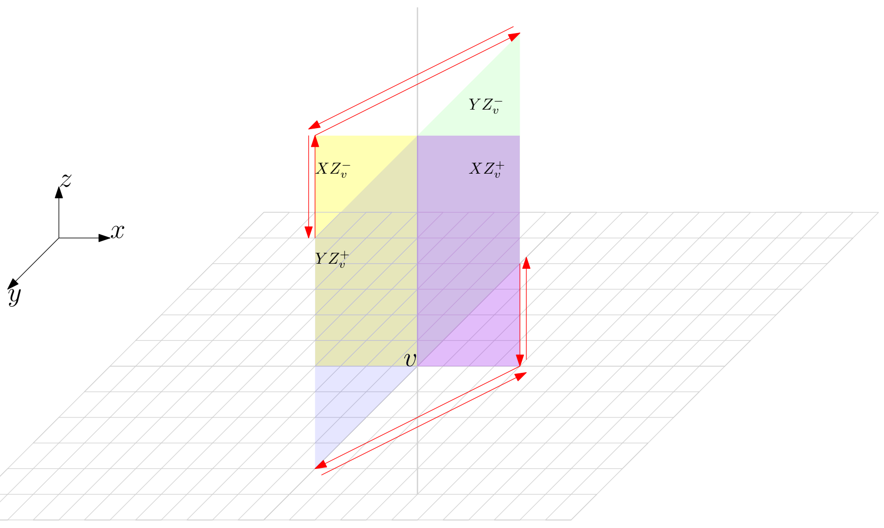

Figure 4: Possible vectors of rotation between 4 half-planes sharing a common pole through vertex v

We define a *horizontal pole* as a line parallel to $0X$ axis. Horizontal pole through the point and *horizontal rotation* are defined analogously. Corresponding set of possible directions of movement are:

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

2.3 Shrinking lifted subtrees

Let v be a vertex of T and $Ch = \{v_1, \ldots, v_l\}$ be the set of its children. Let us consider a $3D$ grid drawing $\Gamma(T)$ of T, where the drawing of the subtree $T(v)$ is the canonical drawing with respect to v. This implies that every $T(v_i), i = 1, \ldots, l$ lies in a quarter plane $h = XZ_v^+$.

Let us assume that the members of $Ch = \{v_1, \ldots, v_l\}$ are ordered according to their z-coordinates in $\mathcal{C}_{T_v}$. Note that for $1 \le i \le l$, each $T(v_i)$ lies inside a box of width

$rpw(T(v_i))$ and height $|T(v_i)|$. Also, note that $T(v)$ lies inside a box of width $rpw(T(v))$ and height $|T(v)|$. Consider a tuple $(i_1, i_2, \ldots, i_k)$, where $1 \leq i_1 < i_2 < \ldots < i_k \leq l$ and the set $Ch' = \{v_{i_1}, \ldots, v_{i_k}\}$. Note that the members of $Ch' = \{v_{i_1}, \ldots, v_{i_k}\}$ are ordered according to their z-coordinates in $\mathcal{C}_{T_v}$ too.

Let us consider the new subtree $T'(v)$ which is obtained by deleting the vertices in $Ch \setminus Ch'$ and their subtrees.

Consider the canonical drawing of $T'(v)$ with respect to v. Even if all the children of v in $Ch \setminus Ch'$ along with the subtrees rooted at them are deleted, the worst case height of the bounding box that contains $\mathcal{C}_{T'_v}$ can still remain the same as the height of the bounding box that contains $\mathcal{C}_{T_v}$. Note that in the drawing $\mathcal{C}_{T'_v}$, the subtrees rooted at the vertices of Ch' are also in a canonical position with respect to their roots.

For each j satisfying $1 \leq j \leq k$, $T'(v_{i_j})$ lies inside a box of height $|T(v_{i_j})|$ and width $rpw(T(v_{i_j}))$ on the plane h such that the down-left corner of the box coincides with $\mathcal{C}_{T'_v}(v_{i_j}) = \mathcal{C}_{T_v}(v_{i_j})$.

In the following, we define the *shrink subtree* procedure on $\mathcal{C}_{T'_v}$. Let us denote the shrinked subtree by $\mathcal{C}'_{T'_v}$. We move each vertex v_{i_j} from $\mathcal{C}_{T'_v}(v_{i_j})$ to $(\mathcal{C}_{T'_v}(v_{i_j})_x, \mathcal{C}_{T'_v}(v_{i_j})_y, \mathcal{C}'_{T'_v}(v_{i_j})_z)$, where $\mathcal{C}'_{T'_v}(v_{i_1})_z = \mathcal{C}_{T_v}(v)_z + 1$ and for $j = 2, \ldots, l$ $\mathcal{C}'_{T'_v}(v_{i_j})_z = \mathcal{C}'_{T'_v}(v_{i_{j-1}})_z + |T(v_{i_{j-1}})| + 1$. Each vertex of the subtree rooted at v_{i_j} also moves along the same vector. x and y coordinates of the vertices of $T'(v)$ in $\mathcal{C}_{T'_v}$ and $\mathcal{C}'_{T'_v}$ are the same.

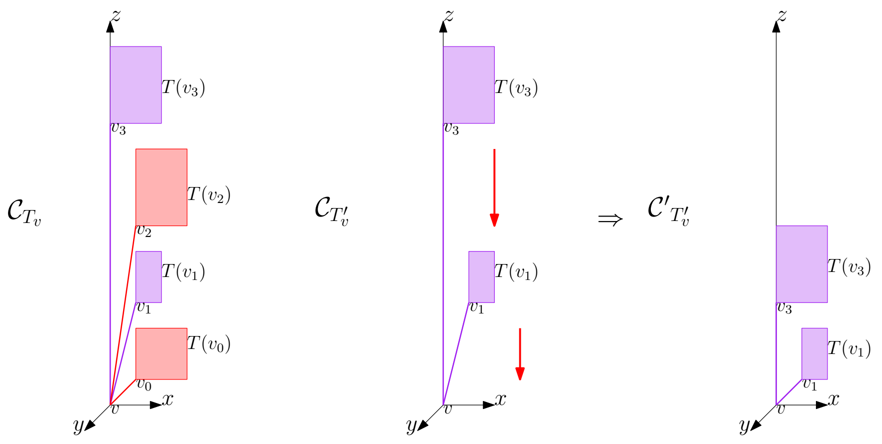

Figure 5: Example of shrinking morph when $l = 4$.

Observation 3. *The height of the shrinked subtree $\mathcal{C}'_{T'_v}$ is equal to the number of vertices in $T'(v)$.*

Proof. Note that the vertices of partial subtree $T'(v)$ lie in canonical positions in $\mathcal{C}_{T'_v}$. This implies that for $1 \leq j \leq k$ the subtree rooted at v_{i_j} also has a canonical drawing with respect to v_{i_j} in $\mathcal{C}_{T'_v}$. Observation 2 implies that 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 $\mathcal{C}_{T_v}(v_j)_z$. From this fact it follows 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 coincide with $\mathcal{C}_{T_v}(v) = \mathcal{C}'_{T_v}(v)$. Also note that $|T'(v)| \leq |T(v)|$. □

Shrinking morph for a subtree $T'(v)$ is a morph in one unidirectional morphing step transforming the drawing $\mathcal{C}_{T'_v}$ into the drawing $\mathcal{C}'_{T'_v}$.

Lemma 8. *Shrinking morph for a subtree $T'(v)$ is a crossing-free 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, shrinking is obviously a unidirectional morph in XZ_v^+. Also, note that throughout the morphing process all vertices in $T'(v)$ maintain the relative order 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. □

2.4 Turning in horizontal plane

Let α be a horizontal plane, $\Gamma_0(T(v))$ be the horizontal canonical drawing of a subtree $T(v)$ in α, i.e. relative canonical drawing $\mathcal{C}_{T_v}$ rotated around horizontal pole through v in any direction. Then $\Gamma_0(T(v))$ lies in XY_v in x-positive, y-positive or y-negative direction. We discuss the case when $\Gamma_0(T(v))$ lies in y-positive direction, the other case is similar.

Let $\Gamma_1(T(v)), \Gamma_2(T(v)), \Gamma_3(T(v))$ be the drawings which are obtained from $\Gamma_0(T(v))$ by α-plane rotation around the point $\Gamma(v)$ by the angles $\frac{\pi}{2}, \pi, \frac{3\pi}{4}$ respectively.

Lemma 9. *(Pinwheel) [6] Let Γ and Γ' be two canonical drawings of a rooted ordered tree T , where $r(T)$ is at the same point in Γ and Γ'. If Γ and Γ' are (i) upward and leftward, or (ii) leftward and downward, or (iii) downward and rightward, or (iv) rightward and upward, respectively, then the morph $\langle\Gamma, \Gamma'\rangle$ is planar and lies in the interior of the right, top, left, or bottom half of the $2n$-box centered at $r(T)$, respectively.*

Lemma 10. *The drawing $\Gamma_i(T(v))$ can be obtained from the drawing $\Gamma_{i-1}(T(v))$ in one morphing step, $i = 0, 1, 2, 3$, where by $\Gamma_{-1}(T(v))$ we mean $\Gamma_3(T(v))$.*

Proof. Proof of this fact is the same as the proof of the Lemma 9 written in [6]. □

3 Morphing through lifting paths

Let T be a tree with n vertices and $\mathcal{P}$ be a fixed path decomposition of T.

In this section we describe an algorithm that morphs a plane drawing $\Gamma = \Gamma_0$ of tree T to the canonical 3D drawing $\Gamma' = \mathcal{C}(T)$ of T in $\mathcal{O}(k)$ steps, where k is the number of paths in $\mathcal{P}$. Note that the final positions for the vertices in $\mathcal{C}(T)$ are defined through heavy-rooted-pathwidth decomposition, and do not depend on the chosen decomposition $\mathcal{P}$.

This is enough to prove that, for any two planar straight-line drawings Γ and Γ' of T, there exists a crossing-free 3D morph from Γ to Γ' with $\mathcal{O}(k)$ steps, since a morph from $\mathcal{C}(T)$ to Γ' can be obtained by playing the morph from Γ' to $\mathcal{C}(T)$ backwards.

Our algorithm lifts the paths in $\mathcal{P}$ one by one starting from the first until the canonical drawing is obtained. That is, for each path $P_i \in \mathcal{P}$ we apply a procedure called *Lift*(). At all times during the algorithm due to the definition of $\mathcal{P}$ the following invariant holds: a path $P_i \in \mathcal{P}$ is lifted only after all the descendants of all of the internal vertices of P_i are already lifted.

Our aim is as follows, during the execution of *Lift*(P_i), path P_i, since it is a portion of the subtree of *head*(P_i), should move to its *canonical position with respect to head*(P_i). We say that vertex v, which lies in the subtree of u, is in canonical position with respect to vertex u in the drawing Γ_t if $\Gamma_t(v) = \Gamma_t(u) + \mathcal{C}_{T_u}(v)$. We consider the given drawing $\Gamma = \Gamma_0$ of the tree T that is lying in $X0Y$ plane. Recall that the parameters of Γ are: the length $l(\Gamma)$, the width $w(\Gamma)$, the height $h(\Gamma)$, and the diameter $d(\Gamma)$.

We proceed to a detailed description of our algorithm. We illustrate its steps in Fig. 10-Fig. 21 for our running example.

Fig. 6 and Fig. 7 give a pseudocode respectively for the entire algorithm and for procedure *Lift*().

Algorithm 1 Lifting paths algorithm

```
    Input: Drawing $\Gamma(T)$
    Output: Canonical drawing $\mathcal{C}(T)$
 1: Let $P_1, \dots, P_k$ be the ordered list of paths from $\mathcal{P}$
 2: Stretch $\Gamma(T)$ with $\mathcal{S}_1 = 2 \cdot (rpw + d(\Gamma))$
 3: for $i = 1$ to $k$ do
 4:     Lift($P_i$)
```

Figure 6: A pseudocode of morphing through lifting paths algorithm

Algorithm 2 *Lift*(P)

```
    Input: Drawing $\Gamma_t$ (after lifting $i-1$ paths), path $P = (v_0, \dots, v_m) = P_i$ in $\mathcal{P}$
    Output: Drawing $\Gamma_{t+14}$ (after lifting path $P_i$)
 1: Shrink: shrink lifted subtrees of vertices of $P$ around vertical pole.
 2: Move others: rotate lifted subtrees of vertices of path $P$ around vertical pole.
 3: Go up: move up vertices of $P$ along with their lifted subtrees.
 4: Rotate lifted subtrees of internal vertices of $P$ to horizontal planes.
 5: Correct the path: move vertices $v_2, \dots, v_m$ horizontally to their canonical $(x, y)$-
    positions with respect to vertex $v_1$.
 6: Go down: move vertices $v_2, \dots v_m$ to their canonical $z$-positions with respect to vertex
    $v_1$.
 7: Turn lifted subtrees of internal vertices of $P$ in horizontal planes.
 8: Stretch in $y$ direction: stretch the lifted subtrees.
 9: Rotate to canonical positions: rotate subtrees of the internal vertices of P to the
    $XZ_{v_i}^{+/-}$.
10: Move along: move $v_1$ with its subtree along horizontal vector towards the pole through
    $v_0$.
11: if lifted subtree of $v_0$ was not rotated in Move others step then
12:     First part of $x, y$-correcting: move $v_1$ with its subtree to $YZ$ plane.
13:     Go down: move $v_1$ with its subtree down to their canonical height.
14:     Second part of $x, y$-correcting: move $v_1$ with its subtree to their canonical $x, y$-
        positions.
15: else
16:     First part of $x, y$-correcting: rotate lifted subtree of $v_0$ to $YZ_{v_0}^{+}$ half-plane.
17:     Go down: move $v_1$ with its subtree down to their canonical height.
18:     Second part of $x, y$-correcting: rotate lifted subtree of $v_0$ to their canonical $x, y$-
        positions.
```

Figure 7: A pseudocode for procedure *Lift*()

Now we proceed with a detailed description of the algorithm.
Step 0: Preprocessing.
This step is a single stretching morph $\langle \Gamma, \Gamma_1 \rangle$ with $\mathcal{S}_1 = 2 \cdot (rpw + d(\Gamma))$, see Section 2.1 for the details. Due to Lemma 3 this Step is crossing-free.

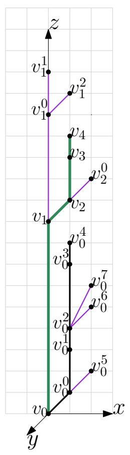

Figure 8: Canonical drawing of the running example

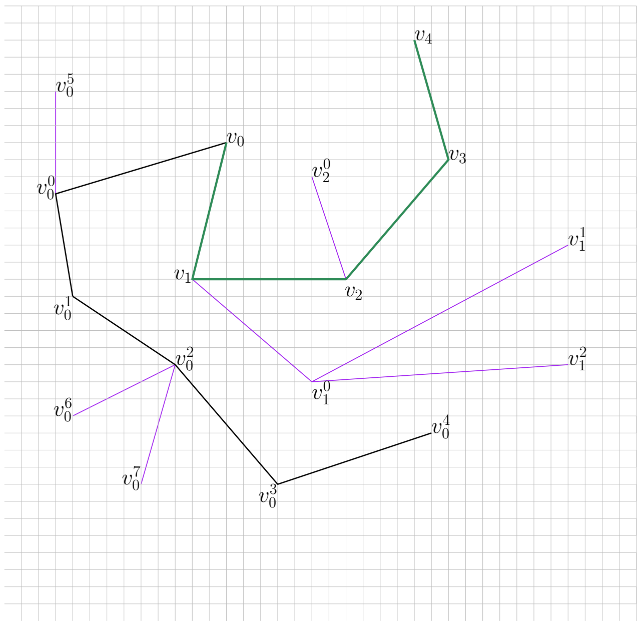

Figure 9: The input of our running example drawing Γ_0 in X0Y plane. Path $(v_0, v_1, v_2, v_3, v_4)$ is lifted in this iteration (all violet edges are already lifted).
After Step 0 every cell is divided into $2 \cdot (rpw + d(\Gamma)) \times 2 \cdot (rpw + d(\Gamma))$ smaller cells, which are not shown in pictures below for more clarity.

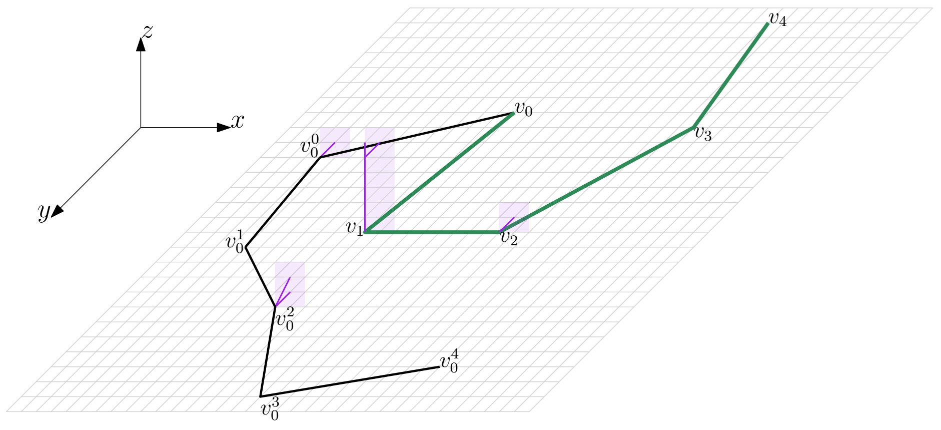

Figure 10: Drawing Γ_t after lifting all paths containing purple edges. All lifted subtrees lie in XZ^+ half-planes and are shown in light purple.

3.1 Procedure ***Lift***(P)

Let $P = (v_0, v_1, \ldots, v_m)$ be the first path in $\mathcal{P}$ that has not been processed yet. See Fig. 9 for an example (Fig. 11 is the drawing before lifting path P shown in green, Fig. 21 is the drawing after lifting this path).

Lemma 11. *The subtrees of all internal vertices v_j in P are already lifted.*

Proof. All the paths that precede P in $\mathcal{P}$ are exactly the paths that are left after deleting P from the forest of subtrees of T (see paragraph *Path decomposition* in Section 1). From this fact it follows that all internal vertices of P have become roots of some trees after deleting the edges of P and edges of that trees were processed in *Lift*(P') procedures for some $P' \prec P$. $\square$

Let Γ_t be the current drawing of T and we lift path $P = P_i$, see Fig. 10.

For any vertex v let *lifted subtree* $T'(v)$ be the portion of subtree $T(v)$ that has been lifted after execution of *Lift*(P_k) where $k < i$. The following invariants hold throughout the algorithm after every iteration of *Lift*() procedure and are proved later by induction.

Conditions:

(I) the drawing of $T'(v)$ in Γ_t is the canonical drawing of $T'(v_j)$ with respect to v for any $v \in V(T)$,

(II) vertices of paths $P_k, k > i$ are lying within X0Y plane.

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

Lemma 12. *The maximum height of vertices in the drawing Γ_t is strictly less than n.*

Proof. As all vertices of the processed paths are part of lifted subtrees of their head vertices, and by (I) have height as in canonical drawing, their height is strictly less than the number of vertices in a corresponding subtree, which does not exceed number of vertices in T. $\square$

Lemma 13. *Let v be any vertex and $T'(v)$ be its lifted subtree. The maximum horizontal distance in $T'(v)$, i.e. the difference between x-coordinates for any pair of vertices in $T'(v)$ in Γ_t, does not exceed rpw.*

Proof. Note that the drawing of $T'(v)$ in Γ_t is the canonical drawing of $T'(v)$ with respect to v. This implies that the maximum horizontal distance between any vertex in $T'(v)$ and v in Γ_t is at most rpw. $\square$

Step 1: Shrink (see Fig. 11).

For every internal vertex v_j of the path P its lifted subtree $T'(v_j)$ morphs into shrinked lifted subtree. All subtrees are shrinked simultaneously in one morphing step. See Section 2.3.

Lemma 14. *Step 1 is crossing-free.*

Proof. For each lifted subtree this is a shrinking step and by Lemma 8 it is crossing-free.

All lifted subtrees were not overlapping in projection to X0Y plane due to Condition (I) and Lemmas 4 and 13 in the drawing Γ_t. As all vectors of movement are vertical, projections of lifted subtrees can not overlap during this morphing step and can not cross with each other.

In shrinked 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 X0Y plane and can not cross with the part of T that is still lying in X0Y. Internal vertices of P do not change their positions during this morph. $\square$

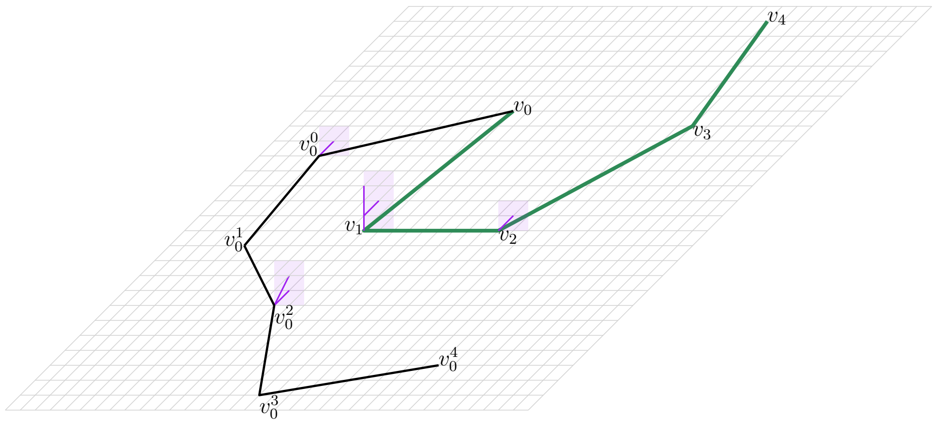

Figure 11: **Step 1.** Lifted subtree of vertex v_1 has morphed into shrinked lifted subtree. All lifted subtrees of other internal vertices of the path are already in shrinked position.

Step 2: Move others (see Fig. 12).

This step consists of two morphs $\langle \Gamma_t, \Gamma_{t+1} \rangle, \langle \Gamma_{t+1}, \Gamma_{t+2} \rangle$.

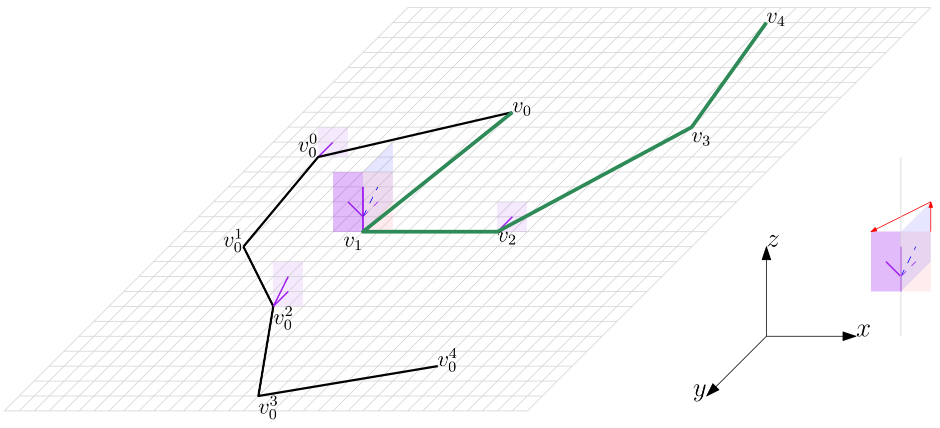

Figure 12: **Step 2.** Shrinked lifted subtree $T'(v_1)$ overlaps with the edge (v_1, v_2) in projection to X0Y plane. $T'(v_1)$ is rotated around the pole through v_1 twice.

Lifted subtrees $T'(v_j)$ of the vertices $v_j, 0 \leq j < m-1$ of P are moved as follows. Recall the definition of projection $pr(T'(v_j))$ from Section 1. If projection $pr(T'(v_j))$ overlaps with $pr((v_j, v_{j+1})), 0 \leq j < m$ we need to rotate twice the drawing of $T'(v_j)$ around the pole through $\Gamma_t(v_j)$, see Section 2.2.

By condition (II) we have: $v_j, v_{j+1} \in X0Y \Rightarrow pr((v_j, v_{j+1})) = (v_j, v_{j+1})$ as v_j, v_{j+1} lie on unprocessed path P.

By condition (I) every lifted subtree $T'(v_j)$ lies in $XZ^+_{v_j}$.

If an overlap happens, we rotate $T'(v_j)$ around vertical pole through $\Gamma_t(v_j)$ twice, so after Step 2 all lifted subtrees lie in $XZ^+_{v_j}$ or $XZ^-_{v_j}$.

Lemma 15. *Step 2 is crossing-free.*

Proof. Two different lifted subtrees $T'(v), T'(v')$ cannot cross as due to Lemma 13: $pr(T'(v)) \subset B(\Gamma_t(v), rpw), pr(T'(v')) \subset B(\Gamma_t(v'), rpw)$ and due to Lemma 4 disks around vertices with radius $rpw \leq \frac{S_1}{2}$ do not cross with each other.

No edges within rotating lifted subtree can intersect as rotating is a mapping and mapping is a crossing-free morph (Lemma 7). Due to condition (II), all edges that do not lie in lifted subtrees are lying in X0Y plane. No lifted subtree $T'(v_j)$ can intersect with any edges in X0Y by condition (I): $\Gamma_t(T'(v_j))$ is the canonical drawing with respect to v_j and lies strictly above X0Y plane except for the point $\Gamma_t(v_j)$. Rotation morph does not change z-coordinate of points during the movement, so throughout the morph drawing of $T'(v_j)$ lies above X0Y plane, vertex v_j does not move during the rotation as it lies on the pole.

No vertices move in X0Y plane during Step 2 and no crossing can happen within the plane X0Y. ☐

Step 3: Go up (see Fig. 13).

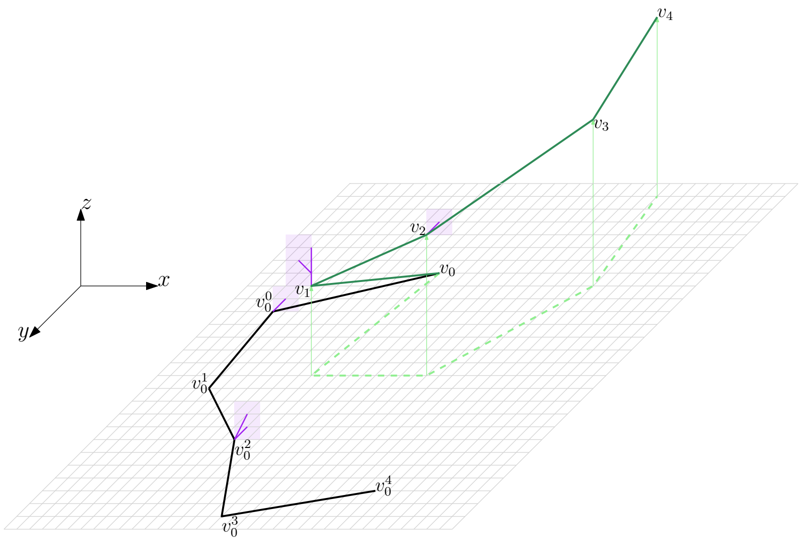

Figure 13: **Step 3.** All internal vertices of the path — v_1, v_2, v_3, v_4 are lifted along with their lifted subtrees $T'(v_i)$.

This step of our procedure consists of one morphing step $\langle \Gamma_{t+2}, \Gamma_{t+3} \rangle$ that moves each internal vertex $v_j, j \geq 1$ of the path P vertically to a certain height defined recursively as follows:

- v_1: $\Gamma_{t+3}(v_1)_z = n$
- $v_j, j > 1$: $\Gamma_{t+3}(v_j)_z = \Gamma_{t+3}(v_{j-1})_z + h_{sh}(T'(v_{j-1}))$, where $h_{sh}(T'(v_{j-1}))$ is the height of shrinked lifted subtree of vertex v_{j-1}

Note that $h_{sh}(T'(v_j))$ is an integer number for all internal vertices of the path.

Lemma 16. *Step 3 is crossing-free.*

Proof. Every internal vertex v_j in the path P moves with its subtree $T'(v_j)$ along the vertical vector. In the beginning of this morphing step no intersections existed in projection to $X0Y$ plane between different subtrees and the path edges. Nothing changes in projection to $X0Y$ plane during this step, all movements happen strictly above $X0Y$ plane. That means that there can be no crossings during this step of an algorithm. □

Lemma 17. *After Step 3:*

1. *All internal vertices of P along with vertices of their subtrees are horizontally separated from the rest of the vertices of the tree, i.e. there exists a horizontal plane that strictly divides these two sets of vertices from each other.*

2. *The subtrees of internal vertices of P are horizontally separated from each other.*

Proof.
1. Due to Lemma 12 z-coordinates of all vertices that do not lie on the path are strictly below n, also they are integer. By the definition of height in Γ_{t+3} internal vertices of P and vertices in their lifted subtrees have z-coordinates at leats n.

   So the plane $(x, y, n - \frac{1}{2}), x, y \in \mathbb{R}$ is the horizontal plane of separation.

2. For every pair of internal vertices of P — $v_j, v_k, j < k$, let us define the plane of separation as follows. From the definition of the height $\Gamma_{t+3}(v_k)_z$ the horizontal plane $(x, y, \Gamma_{t+3}(v_k)_z - \frac{1}{2}), x, y \in \mathbb{R}$ separates lifted subtree of vertex v_j from lifted subtree of vertex v_k.

□

Step 4: Rotate subtrees to horizontal plane (see Fig. 14).

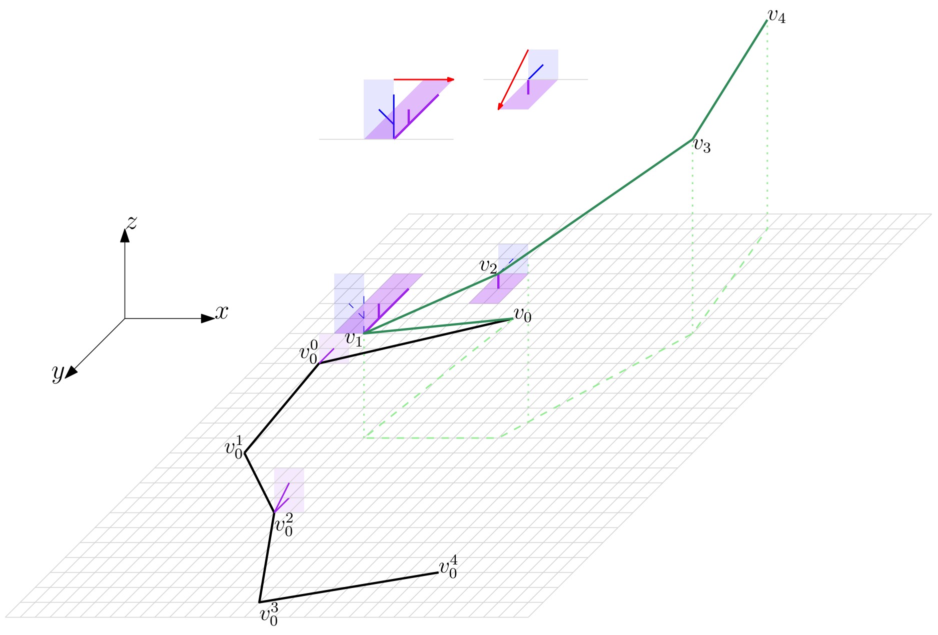

Figure 14: **Step 4.** Lifted subtrees of v_1, v_2 are rotated to horizontal planes.

In this morphing step $\langle \Gamma_{t+3}, \Gamma_{t+4} \rangle$ all lifted subtrees $T'(v_j)$ of internal vertices of path P are rotated around a horizontal pole through the corresponding point $\Gamma_{t+3}(v_j)$ to lie in horizontal plane. The direction of rotation is chosen in such a way that for each $1 \leq j < m$

lifted subtree $T'(v_j)$ does not cross with an edge (v_j, v_{j+1}) throughout this morph, and for v_m is chosen arbitrarily.

More formally, for $T'(v_j), 1 \leq j < m$ consider the two open half-spaces induced by the plane parallel to X0Z and containing $T'(v_j)$. We chose the one of these half-spaces that does not contain edge (v_j, v_{j+1}) in it.

Lemma 18. *Step 4 is crossing-free with the above choice of the direction of rotation.*

Proof. If the crossing happens it must include one of the moving edges, i.e. edges of lifted subtrees.

As subtrees $T'(v_j)$ are horizontally separated from each other after Step 3 by Lemma 17, edges of different lifted subtrees can not cross. Also by Lemma 17, subtrees $T'(v_j)$ can not cross with non-processing vertices or edges between them as they also are horizontally separated in the beginning and in the end of this Step.

If for some $j, 1 \leq j \leq m$ the edge (v_j, v_{j+1}) lied in the vertical plane through $T'(v_j)$ there was no crossings in Γ_{t+3}. After the beginning of movement $T'(v_j)$ will no longer lie on that plane and no crossing can happen. If (v_j, v_{j+1}) lied strictly within one of the half-planes defined by $T'(v_j)$, then by the choice of direction of rotation we rotate through the half-space that does not contain (v_j, v_{j+1}) and thus can not cross with this edge either. □

Step 5: Correct the path (see Fig. 15).

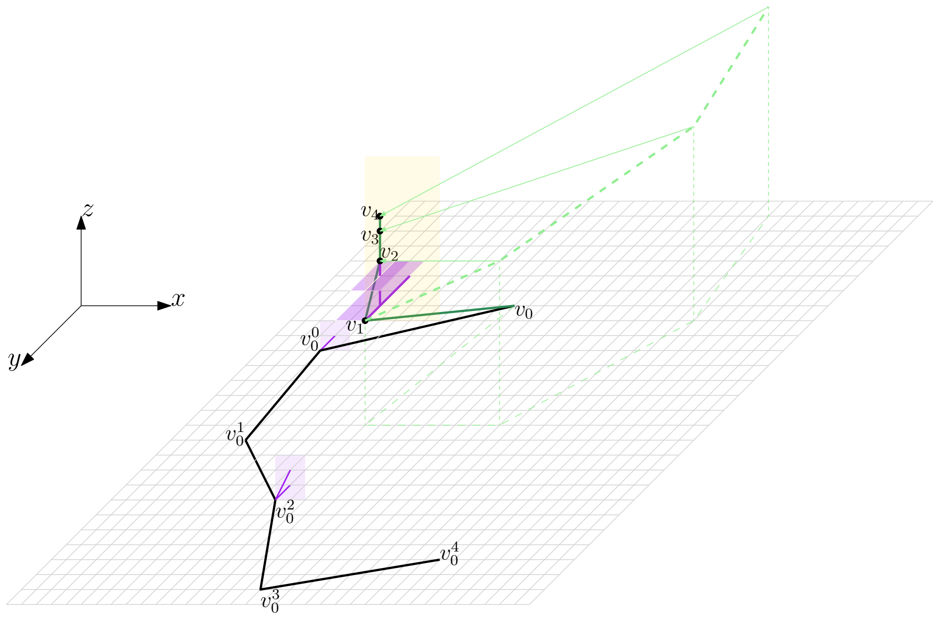

Figure 15: **Step 5.** v_2, v_3, v_4 move horizontally along with their lifted subtrees to get to the canonical x, y-position with respect to v_1.

In morph $\langle \Gamma_{t+4}, \Gamma_{t+5} \rangle$ every vertex $v_j, j \geq 2$ of the path P 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 the x-coordinate of vertex v_1 in drawing Γ_{t+4}. At the end of this step x and y coordinates of $v_j \, \forall j \geq 1$ are the same as x and y coordinates of v_j in canonical drawing with respect to v_1.

Lemma 19. *Step 5 is crossing-free.*

Proof. Step is crossing-free because all edges of the path P and all subtrees of different vertices v_j of the path P (that lie in horizontal planes) are horizontally separated from each other. All moving edges are horizontally separated from already lifted subtrees by the plane $z = n$ due to Lemma 17. □

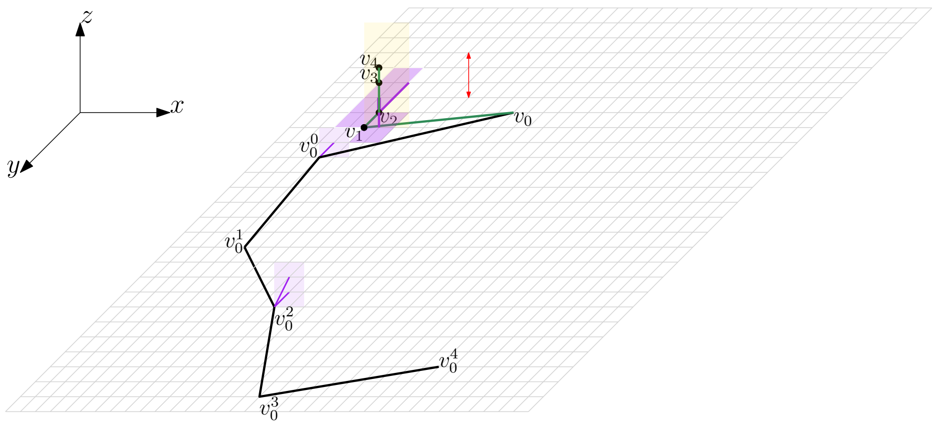

Figure 16: **Step 6.** Vertices v_2, v_3, v_4 move vertically along with their lifted subtrees to get to the canonical z-position with respect to v_1.

Step 6: Go down (see Fig. 16).

This step consists of a single morph $\langle \Gamma_{t+5}, \Gamma_{t+6} \rangle$ and is similar to Step 3. During this morphing step every iternal 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} means z-coordinate of vertex v_1 in drawing Γ_{t+5}. At the end of this step, the z coordinate of v_j, $\forall j > 1$, is the same as the z coordinate of v_j in the canonical drawing with respect to v_1.

Lemma 20. *Step 6 is crossing-free.*

Proof. All moving edges are horizontally separated from already lifted subtrees with the plane $z = n$. 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. From that it follows that horizontal separation of $T'(v_j)$ remains for different j. $\square$

Step 7: Turn subtrees in horizontal planes (see Fig. 17).

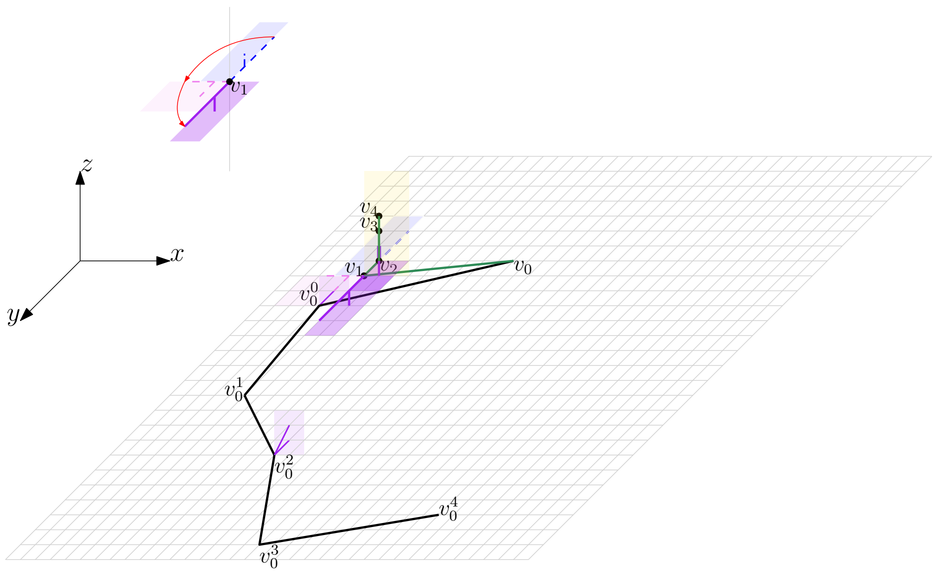

Figure 17: **Step 7.** lifted subtree of v_1 is rotated twice in horizontal plane to lie in X+ direction form v_1.

In this step, consisting of two morphing steps $\langle \Gamma_{t+6}, \Gamma_{t+7} \rangle$, $\langle \Gamma_{t+7}, \Gamma_{t+8} \rangle$, we turn every lifted subtree $T'(v_j)$ of internal vertices of the path to lie in positive x-direction with respect to vertex v_j if needed. For every subtree that lies in negative x-direction in the drawing Γ_{t+6} the result of this step is planar point reflection of $T'(v_j)$ across the point v_j in horizontal plane containing $T'(v_j)$. For turning vertices in horizontal plane we use the morphing step from Lemma 10.

Lemma 21. *Step 7 is crossing-free.*

Proof. Note that for every lifted subtree $T'(v_j)$ of internal vertex Step 7 can be performed by rotating the horizonal quarter plane twice around the root. Lemma 10 implies that such rotations are crossing-free morphs. Since subtrees rooted at internal vertices are horizontally separated from each other, they do not form crossing during the morph. Similarly, no crossing happen with unprocessed part of the tree due to the horizonal separation. $\square$

Step 8: Stretch in y-direction (see Fig. 18).

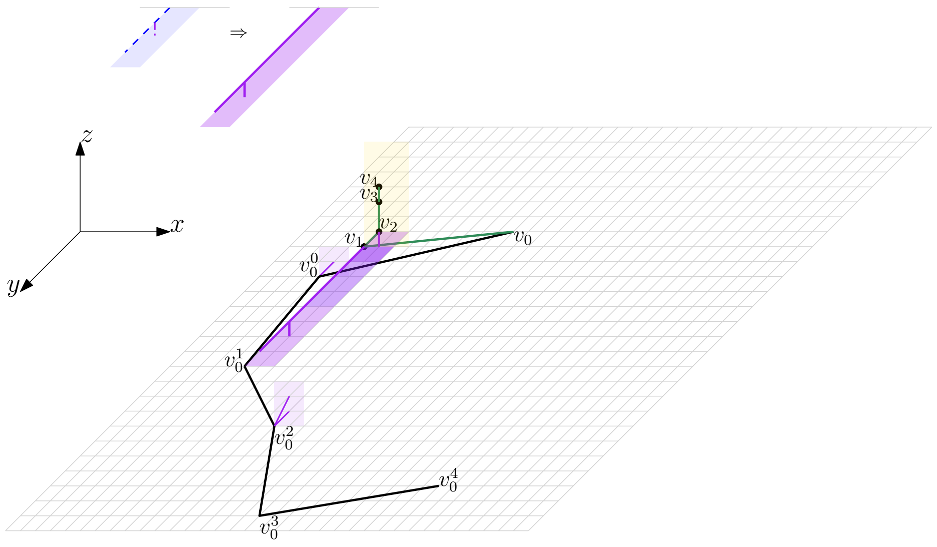

Figure 18: **Step 8.** lifted subtree of v_1 is stretched to canonical width in horizontal plane.

This morphing step $\langle \Gamma_{t+8}, \Gamma_{t+9} \rangle$ transforms lifted subtrees of internal vertices of P in horizontal planes from shrinked to canonical size. All vectors of movement are horizontal.

Lemma 22. *Step 8 is crossing-free.*

Proof. Stretching here is a shrinking morph played backwards and is a crossing-free morph for every $T'(v_j)$ by Lemma 8.

Note tthat all $T'(v_j)$ are horizontally separated from each other and from unprocessed part of the tree. Since the morph is horizontal, no crossings can happen between different $T'(v_j)$. Similarly, due to the horizontal separation, no crossings can happen with the unprocessed part of the tree. $\square$

Step 9: Rotate to canonical positions (see Fig. 19).

In morph $\langle \Gamma_{t+9}, \Gamma_{t+10} \rangle$ all lifted subtrees $T'(v_j)$ of the internal vertices of the path rotate around horizontal axes $(x, v_{j_y}, v_{j_z}), x \in \mathbb{R}$ to lie in vertical plane in positive direction, where v_{j_z} means z-coordinate of vertex v_j in drawing Γ_{t+9}.

Lemma 23. *Step 9 is crossing-free.*

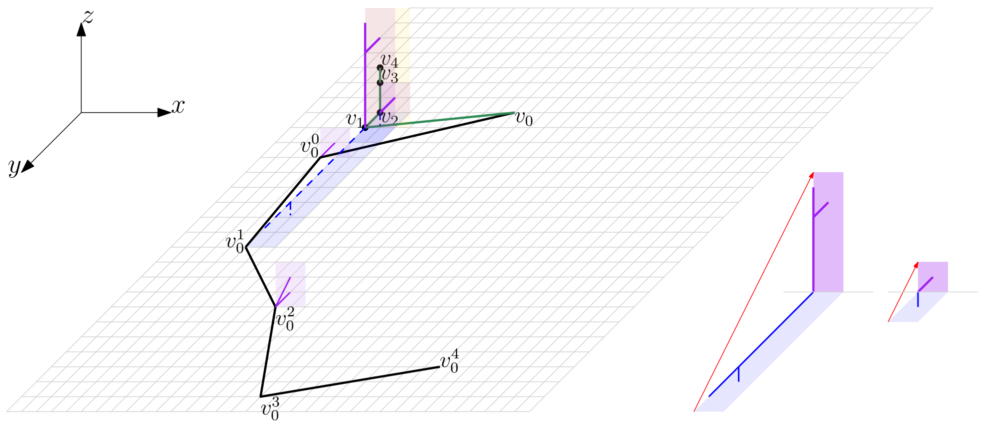

Figure 19: **Step 9.** lifted subtrees of v_1, v_2 are rotated to vertical XZ^+ half-planes

Proof. Let H be the vertical plane parallel to X0Z and containing $v_1, \ldots, v_m$.

Vertices in subtrees $T'(v_j)$ lie from one side of H in parallel planes at any moment of this morphing step by Lemma 6. The half-planes that contain different $T'(v_j)$ are parallel because we can look at the rotation of the half-plane α containing $T'(v_j)$ as at the rotation of the half-plane β containing $T'(v_k), j, k \in \{1, \ldots, m\}$ translated by the vector $\Gamma_{t+9}(v_j)_z - \Gamma_{t+9}(v_k)_z$. Then at every moment angle $\angle(\alpha, XZ) = \angle(\beta, XZ)$ which means that these half-planes are parallel to each other. So the vertices of $T'(v_j)$ do not cross.

$T'(v_j)$ remains in the same half-space from H during all morph for every $j = 1, \ldots, m$. This means that subtrees that lie from different sides of H do not cross too.

At the end of this step subtree of vertex v_1 of the drawing Γ_{t+10} is at its canonical positions with respect to to vertex v_1 and therefore does not have any crossing within itself. □

Note that after this step the whole $T(v_1)$ is in canonical position with respect to vertex v_1.

Step 10: Move along (see Fig. 20).

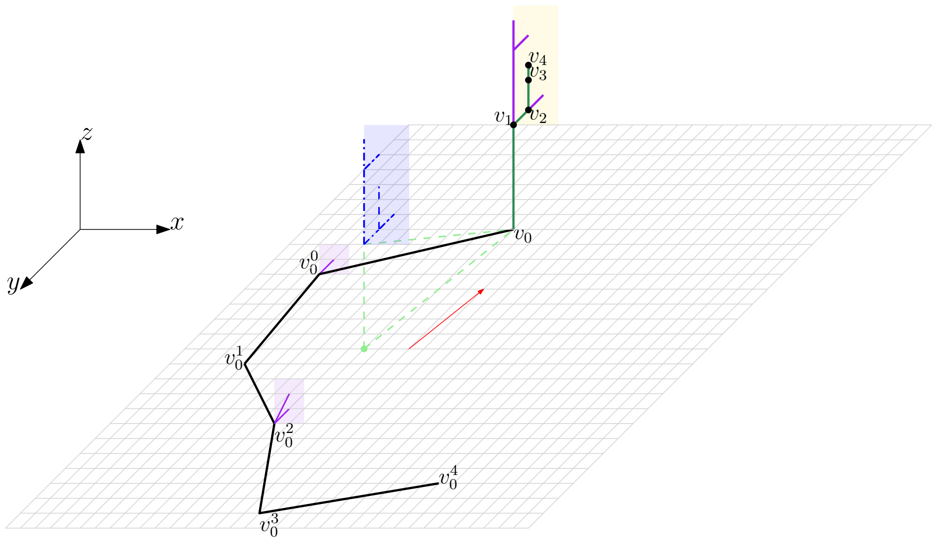

Figure 20: **Step 10.** vertex v_1 with its subtree moves horizontally towards the pole through v_0

In this morphing step $\langle \Gamma_{t+10}, \Gamma_{t+11} \rangle$ every internal vertex v_j of the path with its subtree

$T'(v_j)$ moves horizontally in same direction - $(v_{0_x} - v_{1_x}, v_{0_y} - v_{1_y}, 0)$ - and by vector of the same length defined by v_1.

If in $\mathcal{C}(T)$ the edge (v_0, v_1) is vertical, vertex v_1 moves along this vector to get x, y-coordinates equal to (v_{0_x}, v_{0_y}).

If in $\mathcal{C}(T)$ the edge (v_0, v_1) is not vertical, i.e. $\mathcal{C}(v_1)_x - \mathcal{C}(v_0)_x = 1$, vertex v_1 moves along this vector as long as possible to get integer x and y coordinates not equal to (v_{0_x}, v_{0_y}). From Lemma 5 there is such a point $p = (p_x, p_y, 0)$ in disk $B(\Gamma_{t+10}(v_0), rpw + d(\Gamma))$. Note that $dist_{\Gamma_{t+10}}(p, v_0) \leq d(\Gamma)$, all subtrees $T'(v_j)$ (as they are part of $T'(v_1)$) take no more than rpw space around the pole $(v_{1_x}, v_{1_y}, z), z \in \mathbb{R}$ and therefore in projection to X0Y plane lie within the disk $B(\Gamma_{t+10}(v_0), rpw + d(\Gamma))$.

After this step all morphs move processing vertices in projection to X0Y plane only within disk $B(\Gamma_{t+10}(v_0), rpw + d(\Gamma))$.

Lemma 24. *Step 10 is crossing-free.*

Proof. Step is crossing-free because all vertices of the path P are horizontally separated from other subtrees of other vertices on the plane $X0Y$ (all already lifted subtrees have height at most $n-1$ and all processing vertices have z-coordinates at least n) and the only not-separated edge (v_0, v_1) moves along such a vector, that its projection does not change angle in $X0Y$ plane. □

Steps 11-13 depend on whether we turned $T'(v_0)$ during Step 2 or not.

Case 1: if $pr(v_0, v_1)$ and $T'(v_0)$ did not overlap after Step 1 and $T'(v_0)$ was not rotated during Step 2. In this case in the drawing Γ_{t+11} lifted subtree of v_0 lies in $XZ^+_{v_0}$. See Fig. 22.

Case 2: the overlap happened and $T'(v_0)$ was rotated twice during Step 2. Then $T'(v_0)$ lies in $XZ^-_{v_0}$ in Γ_{t+11}. See Fig. 23.

Below we describe each of the Steps 11-13 for both cases separately.

Step 11: First part of xy-correcting (see Fig. 22 and Fig. 23).

This step consists of one morphing step $\langle \Gamma_{t+11}, \Gamma_{t+12} \rangle$, which depends on the position of $T'(v_0)$.

Case 1: If during Step 2 $T'(v_0)$ was not rotated, which means that the edge (v_0, v_1) was not parallel to 0X, then the movement involves internal vertices of the path. In this case all internal vertices of P with their subtrees $T'(v_j)$ move along the same horizontal vector so that after this movement edge (v_0, v_1) will lie in half-plane parallel to Y0Z and $|v_{1_y} - v_{0_y}| = |\mathcal{C}(v_1)_x - \mathcal{C}(v_0)_x|$. The direction is chosen so that the angle between $pr((v_0, v_1))$ in Γ_{t+11} and $pr((v_0, v_1))$ in Γ_{t+12} is minimal.

Case 2: If during Step 2 $T'(v_0)$ was rotated, which means that the edge (v_0, v_1) was parallel to 0X, then $pr((v_0, v_1))$ is still parallel to 0X, because we have not changed its direction. By definition of Step 2 we know that $T'(v_0)$ in Γ_{t+11} lies in $XZ^-_{v_0}$. We are rotating $T'(v_0)$ around the pole through v_0 to lie in $YZ^+_{v_0}$.

Lemma 25. *Step 11 is crossing-free.*

Proof. **Case 1:** All vectors of movement in $T(v_1)$ are the same, which 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 $B(\Gamma_{t+11}(v_0), rpw + d(\Gamma))$ throughout this morphing step and therefore can not cross with lifted subtrees $T'(v)$ of other non-processing vertices v which projections lie within $B(\Gamma_{t+11}(v), rpw + d(\Gamma))$. Also (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}.

Case 2: Rotation is a crossing-free morph and in projection happens within $B(\Gamma_{t+11}(v_0), rpw + d(\Gamma))$. For the same reasons as in Case 1 no crossings happen. □

Step 12: Go down (see Fig. 21)

$\langle \Gamma_{t+12}, \Gamma_{t+13} \rangle$ morphing step is a vertical morph, common for both cases. In Γ_{t+12} drawing z-coordinates of internal vertices of P are wrong with respect to v_0, they are

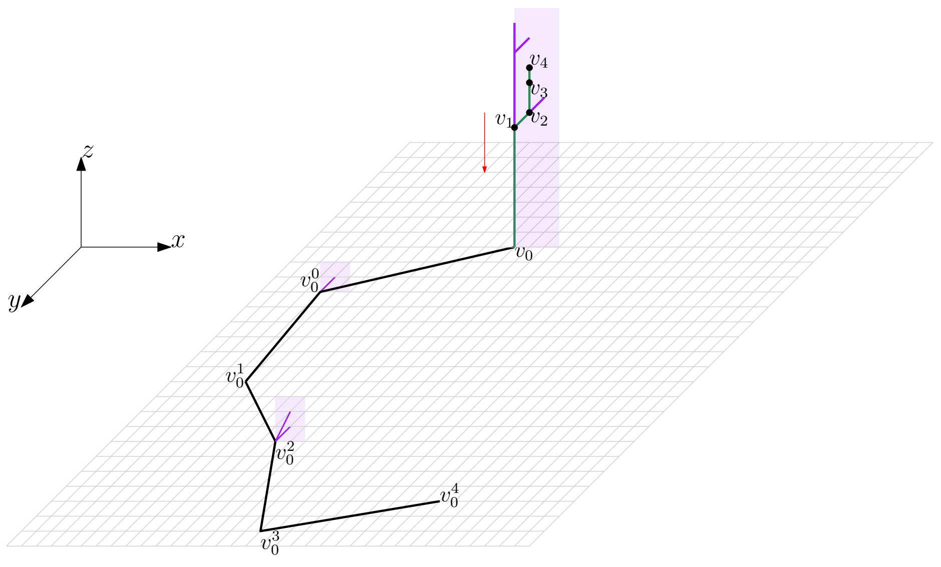

Figure 21: **Step 12.** $T(v_1)$ moves down to the canonical height.

n more than the canonical ones. In both cases in this step we will move all vertices $v_1, \dots, v_m$ with all their subtrees down with vector $(0, 0, -n)$.

Lemma 26. *Step 12 is crossing-free.*

Proof. **Case 1:** Step is crossing-free because $T'(v_0)$ and $T(v_1)$ lie in distinct parallel planes and do not intersect in projection in Γ_{t+12}. Vertical morph does not change the projection of the drawing so separation remains. Edge (v_0, v_1) and $T'(v_0)$ after Step 11 lie in different planes too.

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 $XZ^-_{v_0}$. They can not make any crossings because they do not intersect in projection during all this morph. □

Step 13: Second part of xy-correcting (see Fig. 22 and Fig. 23).

This step consists of morphing step $\langle \Gamma_{t+13}, \Gamma_{t+14} \rangle$ different in Cases 1 and 2.

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

Case 2: $T(v_1)$ and (v_0, v_1) are in canonical positions with respect to v_0 after Step 12: 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 0X 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+14}, they all now form new lifted subtree of vertex v_0 that will be used in later *Lift*() procedures.

Lemma 27. *Step 13 is crossing-free.*

Proof. **Case 1:** During the morph projections of (v_0, v_1), $T(v_1)$, $T'(v_0)$ do not cross. In the end of the step $T(v_1)$ and (v_0, v_1) lie in canonical positions with respect to v_0, $T'(v_0)$ was already in canonical position with respect to v_0 by condition (I), so at the end of the morph they also do not cross.

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

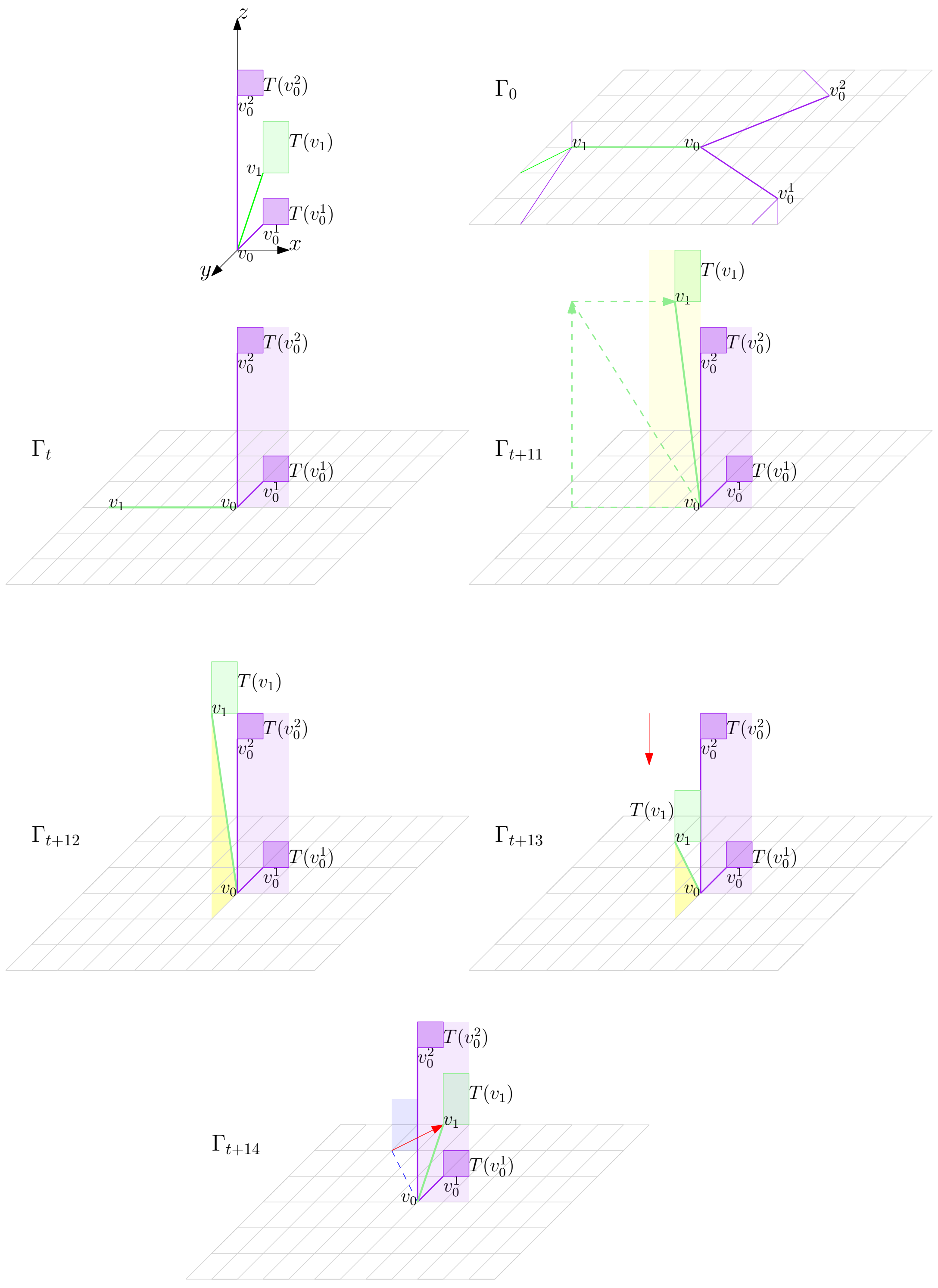

Figure 22: **Step 11-13.** Case 1

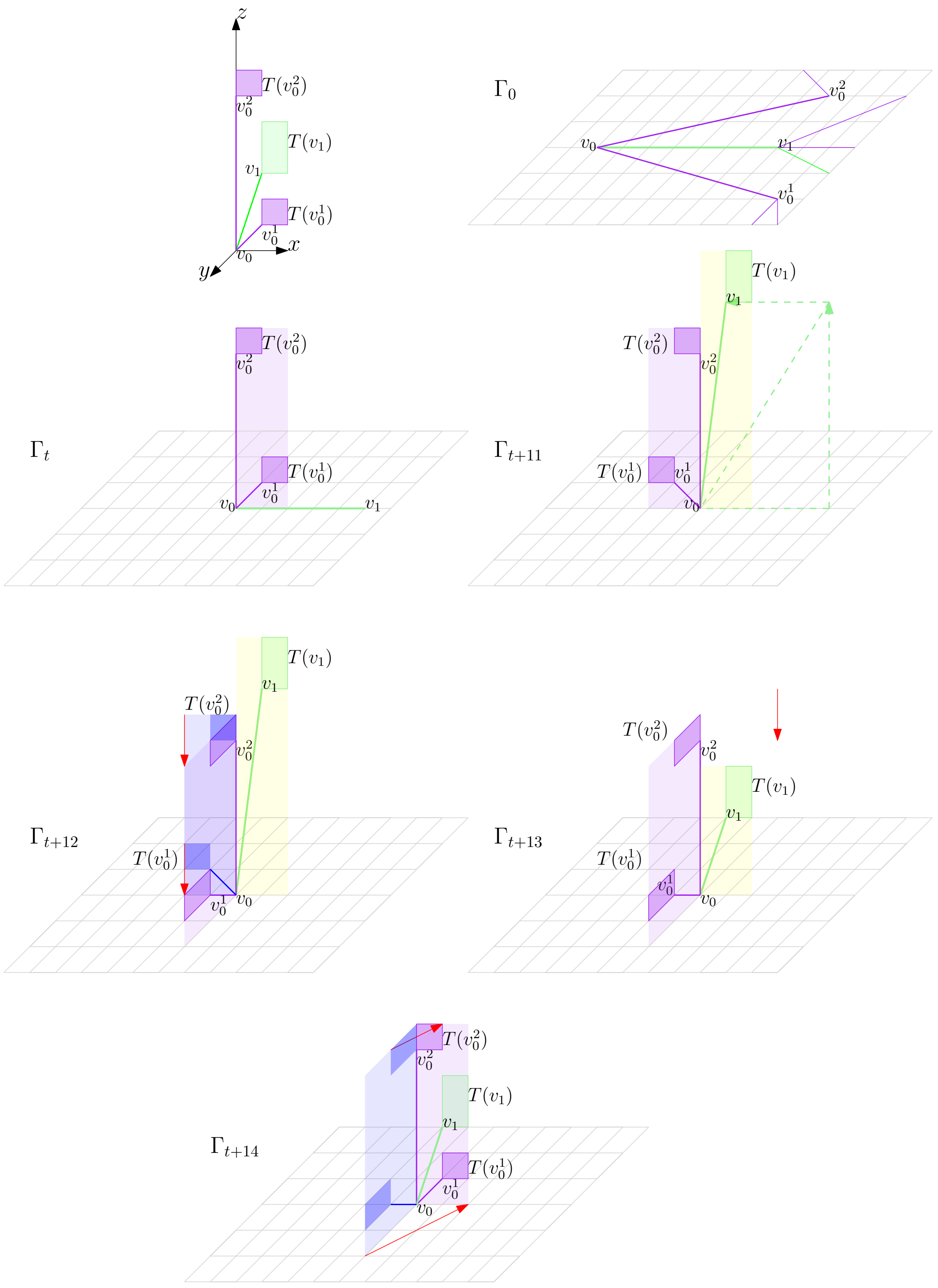

Figure 23: **Step 11-13.** Case 2

3.2 Correctness of the algorithm

Lemma 28. *Conditions (I) and (II) hold after performing Lift(P_i) for each $1 \leq i \leq m$.*

Proof. (I) the drawing of $T'(v)$ in Γ_t is the canonical drawing of $T'(v)$ with respect to v for any $v \in V(T)$

Base of the induction: $i = 1$:

Note that after performing the stretching step the whole T lies on the X0Y plane. Since none of the paths are lifted, condition (I) trivially holds.

The induction step:

By induction hypothesis every $T'(v)$ for internal vertex v lies in position needed before *Lift*(P_i) procedure. After Step 5 all internal vertices have canonical x, y-coordinates with respect to vertex v_1 and after Step 6 — canonical z-coordinates with respect to v_1 also. Steps 7 and 8 guarantee that $T'(v_j), j = 1, \ldots, m$ are in canonical position with respect to to their roots rotated to horizontal plane in positive x-direction. Steps 1 and 8 are mutually inverse planar morphs for every $T'(v_j), 1 \leq j \leq m$ which means that canonical coordinates with respect to the roots will remain the same after Step 8 but in horizontal direction.

Step 9 lifts subtrees of internal vertices into vertical canonical position and $T(v_1)$ is in canonical position with respect to vertex v_1. Steps 10-13 move $T(v_1)$ along with v_1, so after *Lift*(P) procedure $T(v_1)$ is in canonical position with respect to v_1.

Steps 10, 11, 13 place v_1 in canonical xy-position with respect to v_0 and Step 12 makes it y-canonical. So after Step 13 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 X0Y, their lifted subtrees had not moved during *Lift*(P) procedure and are in canonical positions with respect to their roots by induction hypothesis.

(II) vertices of paths $P_k, k > i$ are lying within X0Y plane

Base of the induction: $i = 1$:

The condition (II) holds since the entire tree T lies in the X0Y plane.

The induction step:

By induction hypothesis before *Lift*(P) procedure all vertices of non-processed paths lie in X0Y plane. Internal vertices of the path P can not lie in other non-processed paths than P by definition of path decomposition. That means that after *Lift*(P) in which we move only processed paths, i.e. lifted subtrees, or internal vertices of the P, all vertices that lie on non-processed paths will still lie in X0Y plane. □

Lemma 29. *In morph $\langle \Gamma = \Gamma_0, \ldots, \Gamma_l = \Gamma' \rangle$ from morphing through lifting paths algorithm every intermediate drawing $\Gamma_i, 1 \leq i \leq l$ is a grid drawing.*

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

The base case is trivial: 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.

By induction hypothesis in the beginning of *Lift*(P) procedure all vertices lie on lattice points of the grid. During *Lift*(P) 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. Turning of the subtrees in horizontal planes in Step 7 is integer by definition (see [6]). In all other steps the lifted subtrees of internal vertices are moved along with their roots by the integer vector.

As all coordinates at the beginning of *Lift*(P) were integer and all vectors of movement of all vertices in every step were integer, in every intermediate drawing during *Lift*(P) and after *Lift*(P) procedure all vertices have integer coordinates. □

3.3 Complexity of the algorithm

Let us estimate the grid size (we also call it space) required by the algorithm.

Our graph $T = (V, E)$, $|V| = n$ initially has drawing $\Gamma = \Gamma_0$ that takes space $l(\Gamma) \times w(\Gamma) \times 1$.

First step of the algorithm multiplies needed space 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 drawing takes no more then n, during *Lift*() procedure we may get vertices at height at most $n + n$.

So the required space is:

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

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

The estimation $rpw(T) = \mathcal{O}(\log n)$ is given in [5] and is less than $d(\Gamma) = \Omega(\sqrt{n})$ asymptotically.

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

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

Proof. Bound $\mathcal{O}(n)$ to the number of paths in $\mathcal{P}$ is obvious. □

4 Morphing through lifting edges

In this section we describe another algorithm that morphs a planar drawing Γ of tree T to the canonical drawing $\mathcal{C}(T)$ of T. This time one iteration of our algorithm lifts simultaneously a set of edges of the same depth.

Once again we consider an input drawing $\Gamma = \Gamma_0$ that is lying in $X0Y$ plane.

Step 0: Preprocessing

This step consists of one stretching morph $\langle \Gamma, \Gamma_1 \rangle$. We multiply each coordinate of each of the vertices in Γ by $\mathcal{S}_1 = 2 \cdot rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1)$.

Step 0 is a crossing-free morph because stretching morph is crossing-free by Lemma 3.

4.1 Procedure $\overline{Lift}$(*edges*)

Let $\mathcal{K}$ be the partition of edges of T into sets by depth of their vertices in the tree, sorted by depth in decreasing order. Let m be the depth of T, then $\mathcal{K} = \{K_1, \ldots, K_m\} : \ \forall i \ \forall e = (u, v) \in K_i \ min(dpt(u), dpt(v)) = m - i$.

We call *start of an edge e — $st(e)$ —* the end vertex of this edge with the minimum depth in T. The *end of an edge e — $end(e)$ —* is on the other hand the end vertex of e with maximum depth in T.

Note that for every path in a fixed path decomposition of T every K_i contains at most one edge from this path.

We lift up sets K_i from $\mathcal{K}$ from $i = 1$ to $i = m$ with procedure $\overline{Lift}(K_i)$ consisting of Steps 1-5. Note that this algorithm does not depend on any path decomposition.

Let K_i be the input set, Γ_t be the drawing of T before lifting set K_i.

Similar to the previous section let *lifted subtree* $T'(v_j)$ be the portion of subtree $T(v_j)$ that has been lifted by the of execution of $\overline{Lift}(K_j)$ where $j < i$. Conditions that we maintain throughout the algorithm are the following:

(I) 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) vertices that lie on edges in a set that is not yet processed are lying in X0Y plane.

Analogously we prove by induction that for every $i : 1 \leq i \leq m$ after lifting the set K_i conditions hold.

Lemma 30. *For every edge $e = (v, u)$ and its start vertex v in Γ_1 there is a lattice point $z_e \in e$ such that:*

1. $B(\Gamma_1(z_e), rpw \cdot d(\Gamma)) \subset B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$
2. *for distinct edges $e_1, e_2 \in K_i \forall i = 1, \ldots, m$ disks $B(\Gamma_1(z_{e_1}), rpw)$ and $B(\Gamma_1(z_{e_2}), rpw)$ are disjoint.*
3. *for distinct edges $e_1, e_2 \in K_i \forall i = 1, \ldots, m$ regions*

$$\mathcal{F}_{e_1} = \{x \in X0Y : dist_{\Gamma_1}(x, (z_{e_1}, u)) \leq rpw\}$$

and

$$\mathcal{F}_{e_2} = \{x \in X0Y : dist_{\Gamma_1}(x, (z_{e_2}, u)) \leq rpw\}$$

are disjoint.

Proof. Let us fix an edge $e = (v, u)$. By Lemma 5, there is a lattice point z lying on e in $B(\Gamma_1(v), d(\Gamma))$. Then let point z_e be $(v_x + (z_x - v_x) \cdot 4 \cdot rpw \cdot d(\Gamma), v_y + (z_y - v_y) \cdot 4 \cdot rpw \cdot d(\Gamma), 0)$. It satisfies the conditions of the lemma:

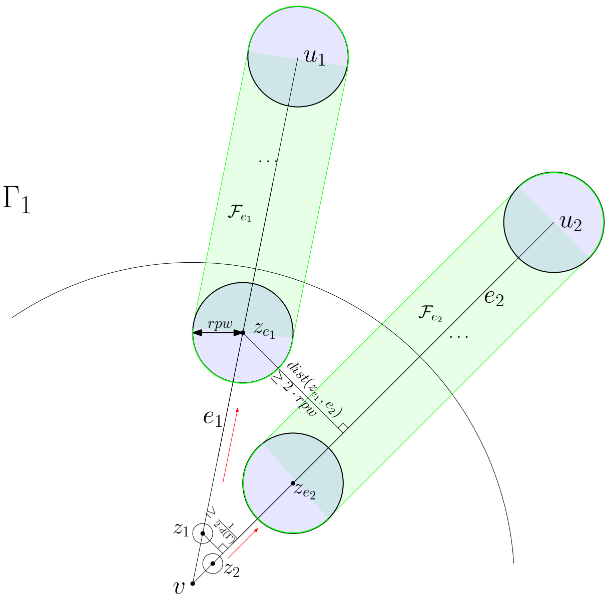

Figure 24: Finding points z_{e_1}, z_{e_2} for edges e_1, e_2 with common start endpoint v. The border of the regions $\mathcal{F}_{e_1}, \mathcal{F}_{e_2}$ is colored green.

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

2. For edges $e_1, e_2 \in K_i$ with different start vertices v_1, v_2 we get disjointedness of the corresponding disks from the fact that $B(\Gamma_1(z_{e_j}), rpw) \subset B(\Gamma_1(v_j), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ and disks $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ do not cross for different v.

   For edges $e_1, e_2 \in K_i$ with a common start vertex v we get points z_1, z_2 from Lemma 5 in Γ_1. We have $dist(z_1, z_2) \geq 1$ because z_1, z_2 are lattice points of the grid. Then by definition of z_{e_1}, z_{e_2} we have $dist_{\Gamma_1}(z_{e_1}, z_{e_2}) \geq 2 \cdot rpw$ and for different edges e in K_i disks $B(\Gamma_1(z_e), rpw)$ do not intersect.

3. For $e_1 = (v, u_1), e_2 = (v, u_2)$ by Lemma 1 $dist(z_1, e_2) \geq \frac{1}{2 \cdot d(\Gamma)}$, because $z_1, (v, z_2)$ can be interpreted as drawing within $B(\Gamma_1(v), d(\Gamma))$ with the diameter at most $2 \cdot d(\Gamma)$. Then $dist(z_{e_1}, e_2) \geq (4 \cdot rpw \cdot d(\Gamma)) \cdot \frac{1}{2 \cdot d(\Gamma)} = 2 \cdot rpw$.

   Minimum distance between $(z_{e_1}, u_1), (z_{e_2}, u_2)$ is realized at one of the endpoints of the segments, for $z_{e_j}, j = 1, 2$ it is at least $2 \cdot rpw$ as we know from above. For vertices $u_j, j = 1, 2$ by Lemma 5 disk $B(\Gamma_t(u_j), 2 \cdot rpw)$ does not intersect with 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.

   For edges e_1, e_2 with different start vertices we have that e_1, e_2 do not have common endpoints by the definition of K_i. Then $e_1 = (v_1, v_2), e_2 = (v_3, v_4)$ and for the same reasons as above the corresponding regions are disjoint.

$\square$

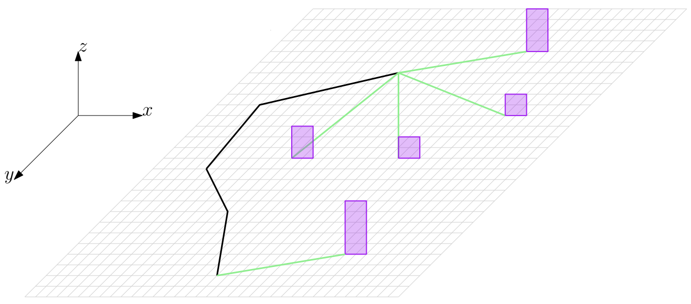

Figure 25: Drawing Γ_t in the beginning of the procedure $\overline{Lift}(K_i)$, lifted subtrees are violet. K_i consists of green edges.

Step 1: Shrink (see Fig. 26).

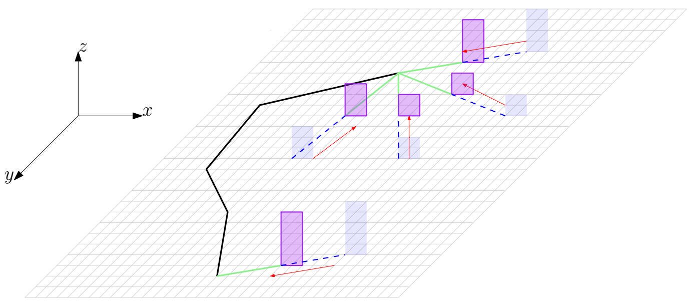

Figure 26: **Step 1.**

In the morphing step $\langle \Gamma_t, \Gamma_{t+1} \rangle$ for every edge $e \in K_i$ we move vertex $end(e)$ along with its lifted subtree towards $st(e)$ until end vertex reaches point z_e of the corresponding edge e, all vectors of movement are horizontal. Point z_e for vertex $st(e)$ and the edge e are defined as in Lemma 30.

Lemma 31. *Step 1 is crossing-free.*

Proof. Vertices that lie in the X0Y plane do not move or move along their incident edges, so no intersections can happen in X0Y.

For the moving subtrees we know from Lemma 30 that they 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 $e \in K_i$.

From Conditions (I) and (II) it follows that no intersections happen during this morphing step. □

Step 2: Go up (see Fig. 27).

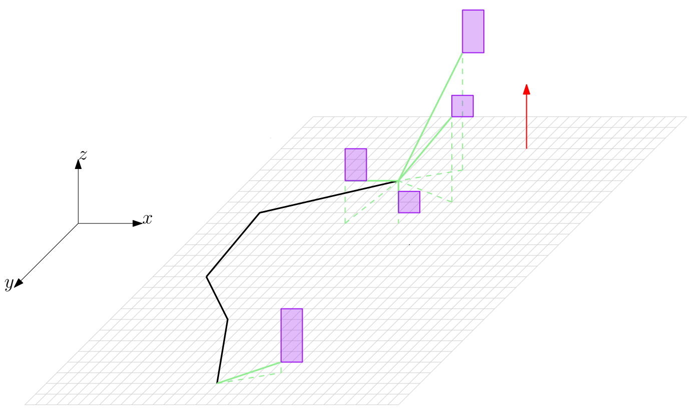

Figure 27: **Step 2.**

In morphing step $\langle \Gamma_{t+1}, \Gamma_{t+2} \rangle$ we move all end vertices $end(e)$ in the set K_i along the vector $(0, 0, \mathcal{C}(end(e))_z - \mathcal{C}(st(e))_z)$. Each $T'(end(e))$ moves along the same vector as $end(e)$.

Lemma 32. *Step 2 is crossing-free.*

Proof. For every vertex v let $End(v)$ be a set of edges, for which v is a start vertex. All edges of $End(v)$ are contained in one set K_i because their depth is defined by $dpt(v)$. That means that if we are lifting some edges with start vertex v, then v does not have any lifted subtree in the beginning of $\overline{Lift}(K_i)$ procedure. In projection to X0Y plane in Γ_{t+1} lifted subtrees of the same or different vertices do not intersect.

During this morphing step we do not change projection and move every lifted subtree by the same vertical vector, so no intersections can happen between the subtrees and between the edges of the same subtree. Vertices of unprocessed sets are lying still in X0Y plane and can not make any intersections too. □

Step 3: Mapping (see Fig. 28).

Morphing step $\langle \Gamma_{t+2}, \Gamma_{t+3} \rangle$ is a mapping morph, see Section 2.2. For every lifted subtree $T'(v_j)$, where $v_j = end(e), e \in K_i$, we define the half-planes of the mapping morph as follows: half-plane α is $XZ^+_{v_j}$ by Condition (I), half-plane β is part of the vertical plane containing the edge e in such direction that $e \notin \beta$, the common vertical pole of α and β is a pole through v_j.

All mapping steps are done simultaneously for all subtrees of end vertices of the edges of K_i.

Lemma 33. *Step 3 is crossing-free.*

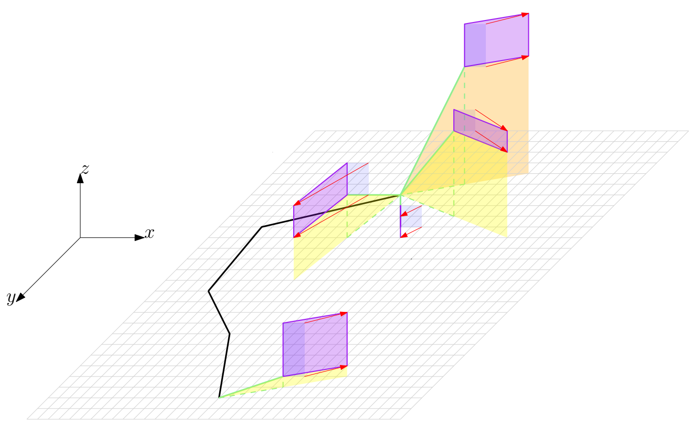

Figure 28: **Step 3.**

Proof. By Lemma. 7 mapping is a crossing-free morph and no intersections happen in every $T'(v_j)$.

Movement of every $T'(v_j)$ for $v_j = end(e)$ happens in projection to X0Y plane in the region $\mathcal{F}_e$ defined for e and $st(e)$: distance from vertices of $T'(v_j)$ to e decreases and e has at least $rpw + 1$ integer points on it so in Γ_{t+1} and Γ_{t+2} projections of vertices of $T'(v_j)$ lie in $\mathcal{F}_e$.

Different lifted subtrees do not intersect as they do not intersect in projection to X0Y during this step. Other vertices do not move and also make no intersections. $\square$

Step 4: Shrink more (see Fig. 29).

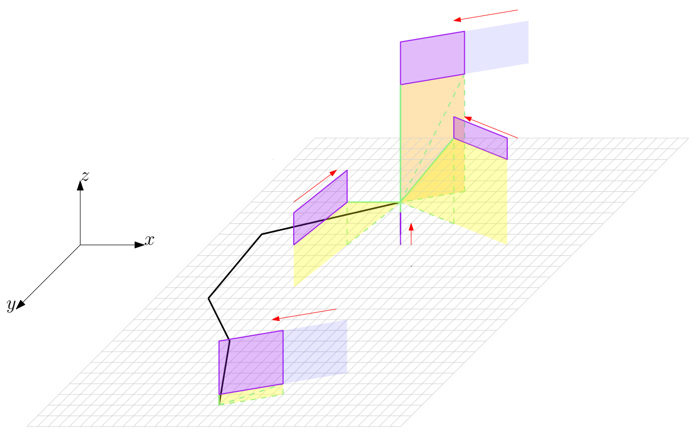

Figure 29: **Step 4.**

The morphing step $\langle \Gamma_{t+3}, \Gamma_{t+4} \rangle$ is a horizontal morph. For each $v_j = end(e), e \in K_i$ we define a horizontal vector of movement as following. If e is a vertical edges in canonical drawing then this vector is $(\Gamma_{t+3}(st(e))_x - \Gamma_{t+3}(end(e))_x, \Gamma_{t+3}(st(e))_y - \Gamma_{t+3}(end(e))_y, 0)$, in this case subtree $T'(end(e))$ is moving towards vertical pole through $st(e)$ until the image of the edge e becomes vertical. As in canonical drawing only one edge is vertical at every vertex, we get at most one edge vertical at every vertex $st(e), e \in K_i$.

If e is not a vertical edge in canonical drawing, then $\mathcal{C}(end(e))_x - \mathcal{C}(st(e))_x = 1$ and we move the whole subtree $T'(end(e))$ towards the pole through $\Gamma_{t+3}(st(e))$ 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 34. *Step 4 is crossing-free.*

Proof. Every lifted subtree moves inside its half-plane. That means that different subtrees $T'(end(e_j))$ with the same $st(e_j)$ do not intersect in projection to X0Y plane and though do not intersect with each other.

Subtrees with different $st(e_j)$ in projection lie in non-crossing disks $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ by Lemma 30 and also can not intersect during this morph. □

Step 5: Collide planes (see Fig. 30).

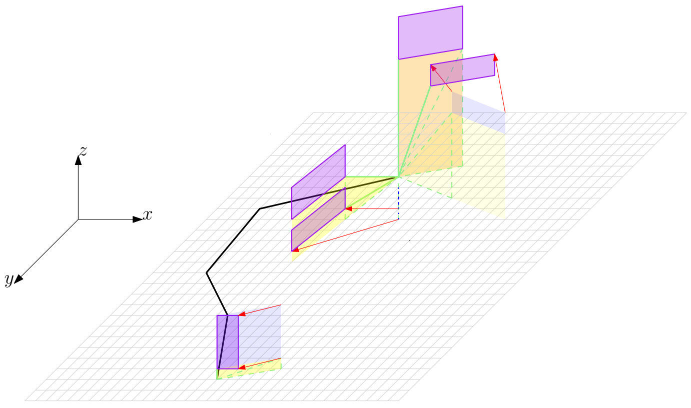

Figure 30: **Step 5.**

During the following steps $\langle \Gamma_{t+4}, \Gamma_{t+5} \rangle, \ldots, \langle \Gamma_{t+5+\log k}, \Gamma_{t+5+\log k+1} \rangle$ we iteratively divide half-planes that contain $T'(end(e)), e \in K_i$ around each vertex $st(e), e \in K_i$ in pairs, pairs are formed of neighboring half-planes in clockwise order around the pole through the corresponding $st(e)$. If in some iteration there are odd number of planes around some pole, the plane without pair does not move in this iteration.

In every iteration we map the drawing of one plane in the pair to another simultaneously in all pairs. As around each vertex we can have at most $k = \Delta(T)$ number of half-planes, we need at most $\log k + 1 + 1$ number of mapping steps to collide all planes in one and to rotate the resulting image to $XZ^+_{st(e)}$

Lemma 35. *Step 5 is crossing-free.*

Proof. Mapping is a crossing-free morph, no intersection happen in every lifted subtree in every iteration.

After every step every $T'(end(e)), e \in K_i$ is in canonical position with respect to $st(e)$ but mapped from $XZ^+_{st(e)}$ to some other half-plane. This is true in Γ_{t+4} and remains true through all these morphing steps as we do not change set of half-planes. In every mapping of one plane to another by the definition of mapping 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 plane α containing one subtree to plane β containing another we will have in plane β two subtrees drawing which can be obtained by mapping from $X0Z^+_{st(e)}$ their canonical drawing with respect to $st(e)$. So during each morphing step and at the end of every morphing step no intersections can happen.

Every mapping step for every $st(e)$ is happening in projection to X0Y in its disk $B(\Gamma_1(v), rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1))$ so mappings for different $st(e)$ can not intersect too.

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

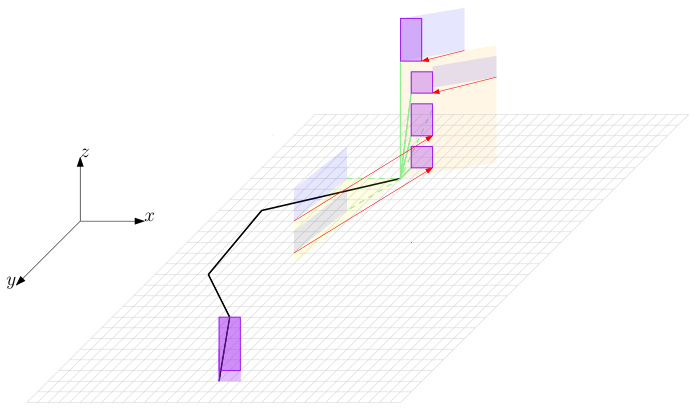

Figure 31: Drawing Γ_{t+6} after colliding all the planes.

4.2 Correctness of the algorithm

Lemma 36. *Conditions (I) and (II) hold after performing* $\overline{Lift}(K_i)$ *for each* $1 \leq i \leq m$.

Proof. (I) the drawing of $T'(v)$ in Γ_t is the canonical drawing of $T'(v)$ with respect to v for any $v \in V(T)$

Base of the induction: $i = 1$:

Note that after performing the stretching step the whole T lies on the X0Y plane. Since none of the paths are lifted, condition (I) trivially holds.

The induction step:

By induction hypothesis every $T'(v)$ for end vertex of any edge in K_i lies in position needed before $\overline{Lift}(K_i)$ procedure. After Step 2 all end vertices of the edges of K_i have canonical z-coordinates with respect to the start vertices of their edges. Then Step 4 for any edge e in K_i makes distances from the vertex $end(e)$ and vertices of its subtree to the pole through $st(e)$ proportional to their canonical y-coordinate. After Step 5 all $end(e), e \in K_i$ vertices and their lifted subtrees lie in the corresponding $XZ^+_{st(e)}$ 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 subtree in the beginning of the procedure $\overline{Lift}(K_i)$ and after this procedure their lifted subtree consists of the end vertices of the edges in K_i and their lifted subtrees.

As for the other vertices in X0Y, their lifted subtrees had not moved during $\overline{Lift}(K_i)$ procedure and are in canonical positions with respect to their roots by induction hypothesis.

(II) vertices that lie on edges in a set that is not yet processed are lying within X0Y plane

Base of the induction: $i = 1$:

The condition (II) holds since the entire tree T lies in the X0Y plane.

The induction step:

By induction hypothesis before $\overline{Lift}(K_i)$ procedure all vertices that lie on edges in a set that is not yet processed lie in X0Y plane. End vertices of the edges in K_i can not lie in $K_j, j > i$ by definition of the partition $\mathcal{K}$. That means that after $\overline{Lift}(K_i)$ 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 X0Y plane.

□

Lemma 37. *In morph $\langle \Gamma = \Gamma_0, \ldots, \Gamma_l = \Gamma' \rangle$ from morphing through lifting edges algorithm every intermediate drawing $\Gamma_i, 1 \le i \le l$ is a grid drawing.*

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

The base case is trivial. 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.

By induction hypothesis in the beginning of $\overline{Lift}(K_i)$ procedure all vertices lie on lattice points of the grid. During $\overline{Lift}(K_i)$ 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. In all other steps the lifted subtrees of end vertices of the edges are moved along with their roots by the integer vector.

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

4.3 Complexity of the algorithm

The estimation on the grid size required by the algorithm is similar to the one in previous section.

In the beginning of the algorithm the given drawing $\Gamma = \Gamma_0$ of graph $T = (V, E)$, $|V| = n$ takes space $l(\Gamma) \times w(\Gamma) \times 1$.

First step of the algorithm multiplies needed space 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.

So the space algorithm requires during all steps is at most:

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

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

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

There are l number of sets K_i, where l is the depth of T. For every procedure $\overline{Lift}(K_i)$ we need at most $6 + \log k$ number of morphing steps, where $k = \Delta(T)$ — maximum degree of a vertex in T.

So the total number of steps in the algorithm is:

$$\mathcal{O}(dpt(T) \cdot \log \Delta(T))$$

Theorem 2. *For every two planar straight-line grid drawings Γ, Γ' of tree T with n vertices there exists a crossing-free 3D-morph $\mathcal{M} = \langle \Gamma = \Gamma_0, \ldots, \Gamma_k = \Gamma' \rangle$ that takes $\mathcal{O}(dpt(T) \cdot \log \Delta(T))$ steps and grid of size $\mathcal{O}(d^3 \cdot \log n \times d^3 \cdot \log n \times n)$ to perform, where $dpt(T)$ is the depth of tree T, $d = max(d(\Gamma), d(\Gamma'))$ is maximum of the diameters of the given drawings Γ, Γ', $\Delta(T)$ is maximum degree of a vertex in T. In this morph every intermediate drawing $\Gamma_i, 0 \le i \le k$ is a straight-line 3D grid drawing.*

5 Trade-off

Recall that $\mathcal{L}(T)$ is the ordered set of paths induced by the long-path decomposition, see Section 1. Let $Long(T)$ be the set of paths from $\mathcal{L}(T)$, consisting of the paths which length is at least $\sqrt{n}$, ordered as 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. 32 for an example of partition of a tree into $Long(T), Short(T)$.

Lemma 38. *1. $|Long(T)| \leq \sqrt{n}$*

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

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

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

$$\Rightarrow |Long(T)| \leq \sqrt{n}$$

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 does not exist in T_i. We arrive to a contradiction.

□

Let us divide edges in $Short(T)$ into sets $Sh_1, \ldots Sh_{\lfloor\sqrt{n}\rfloor}$ by the following rule: 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, see Fig. 32. Note that as depth of all trees T_k are at most $\sqrt{n}$, $Sh_1, \ldots Sh_{\lfloor\sqrt{n}\rfloor}$ will contain all the edges of these subtrees.

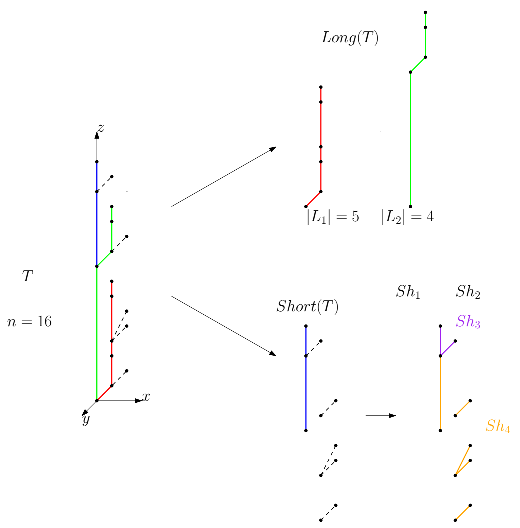

Figure 32: Example of partition of the edges into set of paths $Long(T)$ and sets of edges $Sh_i, i \in \{1, \ldots, \sqrt{n} = 4\}$. Sets Sh_1, Sh_2 are empty, as trees in $Short(T)$ have depth at most 2.

The **trade-off algorithm** works as follows:

In the beginning we perform a stretching step with $\mathcal{S}_1 = 2 \cdot rpw \cdot d(\Gamma) \cdot (4 \cdot d(\Gamma) + 1)$, the same as in Section 4. This stretching constant is big enough to perform *Lift*() and $\overline{Lift}()$ procedures crossing-free.

After that, we lift edges from sets Sh_1 to $Sh_{\lfloor \sqrt{n} \rfloor}$ by $\overline{Lift}(Sh_i)$ procedure (see Section 4). Due to our choice of constant $\mathcal{S}_1$ different subtrees are separated from each other and this is a valid operation. This part of our algorithm takes $\mathcal{O}(\sqrt{n} \cdot \log \Delta(T))$ steps in total by Theorem 2.

Then, we lift paths in $Long(T)$ in reverse order by *Lift*() procedure (see Section 3). As $|Long(T)| \leq \sqrt{n}$ and each *Lift*() procedure consists of a constant number of morphing steps, this part of the algorithm takes $\mathcal{O}(\sqrt{n})$ steps in total.

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

As $\Delta(T) \leq n$ we immediately get the following corollary:

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

6 Conclusion

In this thesis, we have studied algorithms for crossing-free $3D$-morphs between two planar straight-line grid drawings of an n-vertex tree T where each intermediate morphing step produces a $3D$ straight-line grid drawing of T. We have got the algorithm such that it takes $\mathcal{O}(\sqrt{n} \log n)$ morphing steps and grid of a size $\mathcal{O}(d^3 \cdot \log n \times d^3 \cdot \log n \times n)$ to perform, where $d = max(d(\Gamma), d(\Gamma'))$ is maximum of the diameters of the given drawings Γ, Γ'. Our algorithm is better in number of steps and in the (horizontal) size of the grid than the best known algorithm operating in the plane [6].

We suppose that our result may not be optimal in number of steps and grid size since we do not have a matching lower bound. However, our algorithm shows that the third dimension can help to reduce the number of steps in morphing grid drawings and gives rise to an interesting open problem: to what extent one can further reduce the required number of steps and grid size for morphing 3D grid drawings of trees?

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] Soroush Alamdari, Patrizio Angelini, Timothy M Chan, Giuseppe Di Battista, Fabrizio Frati, Anna Lubiw, Maurizio Patrignani, Vincenzo Roselli, Sahil Singla, and Bryan T Wilkinson. Morphing planar graph drawings with a polynomial number of steps. In S. Khanna, editor, *24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '13)*, pages 1656–1667. SIAM, 2013. `doi:10.1137/1.9781611973105.119`.

[3] P. Angelini, G. Da Lozzo, G. Di Battista, F. Frati, M. Patrignani, and V. Roselli. Morphing planar graph drawings optimally. In J. Esparza, P. Fraigniaud, T. Husfeldt, and E. Koutsoupias, editors, *41st International Colloquium on Automata, Languages and Programming (ICALP '14)*, volume 8572 of *LNCS*, pages 126–137. Springer, 2014. `doi:10.1007/978-3-662-43948-7_11`.

[4] Patrizio Angelini, Fabrizio Frati, Maurizio Patrignani, and Vincenzo Roselli. Morphing planar graph drawings efficiently. In S. Wismath and A. Wolff, editors, *21st International Symposium on Graph Drawing (GD '13)*, volume 8242 of *LNCS*, pages 49–60. Springer, 2013. `doi:10.1007/978-3-319-03841-4_5`.

[5] 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.

[6] 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. In *Workshop on Algorithms and Data Structures*, pages 57–70. Springer, 2019.

[7] Fidel Barrera-Cruz, Penny Haxell, and Anna Lubiw. Morphing planar graph drawings with unidirectional moves. In *Mexican Conference on Discrete Mathematics and Computational Geometry*, pages 57–65, 2013. also available at `http://arxiv.org/abs/1411.6185`.

[8] Fidel Barrera-Cruz, Penny Haxell, and Anna Lubiw. Morphing schnyder drawings of planar triangulations. *Discrete & Computational Geometry*, 61(1):161–184, 2019.

[9] Michael Bender and Martin Farach-Colton. The level ancestor problem simplified. volume 321, pages 190–201, 03 2002. `doi:10.1007/3-540-45995-2_44`.

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

[11] S.S. Cairns. Deformations of plane rectilinear complexes. *The American Mathematical Monthly*, 51(5):247–252, 1944. `doi:10.1080/00029890.1944.11999082`.

[12] Michael S. Floater and Craig Gotsman. How to morph tilings injectively. *Journal of Computational and Applied Mathematics*, 101(1-2):117–129, 1999. `doi:10.1016/S0377-0427(98)00202-7`.

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

[14] 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`.

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

[16] 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`.

[17] Robert E. Horton. Erosional development of streams and their drainage basins: hydro-physical approach to quantitative morphology. *Geological Society of America Bulletin*, 56(3):275–370, 1945. `doi:http://dx.doi.org/10.1130/0016-7606(1945)56[275:EDOSAT]2.0.CO;2`.

[18] Marc Lackenby. The efficient certification of knottedness and Thurston norm. *CoRR*, abs/1604.00290, 2016. URL: `http://arxiv.org/abs/1604.00290`.

[19] Arthur N. Strahler. Hypsometric (area-altitude) analysis of erosional topology. *Geological Society of America Bulletin*, 63(11):1117–1142, 1952. `doi:http://dx.doi.org/10.1130/0016-7606(1952)63[1117:HAAOET]2.0.CO;2`.

[20] Arthur N. Strahler. Quantitative analysis of watershed geomorphology. *Transactions of the American Geophysical Union*, 38(6):913–920, 1957. `doi:http://dx.doi.org/10.1029/tr038i006p00913`.

[21] Carsten Thomassen. Deformations of plane graphs. *Journal of Combinatorial Theory, Series B*, 34(3):244–257, 1983. `doi:10.1016/0095-8956(83)90038-2`.

7 Appendix

In the following, we mention the proofs of the lemmas that are omitted from the main paper.

Proof of Lemma 1:

Proof. Let fix vertex $v \in V$ and $e = (u, w) \in E, v \neq u, w$. Denote by h the height in the triangle (v, u, w) from the vertex v, h is the distance from v to the line that contains the edge (u, w) by its definition. Let S be the area of triangle (u, v, w).

The $S = \frac{1}{2} \cdot h \cdot dist_\Gamma(u, w) \leq \frac{d(\Gamma)}{2} \cdot h$.

On the other hand from the Pick's theorem we know, that the area of the triangle with vertices lying in the lattice points of the grid is at least $\frac{1}{2}$.

Then $\frac{1}{2} \leq S \leq \frac{d(\Gamma)}{2} \cdot h \Rightarrow h \geq \frac{1}{d(\Gamma)}$. □