The Complexity of the Two Dimensional Curvature-Constrained Shortest-Path Problem

Sylvain Lazard*, John Reif† and Hongyan Wang

June 20, 2002

Abstract

The motion planning problems for non-holonomic car-like robots have been extensively studied in the literature. The curvature-constrained shortest-path problem is to plan a path from an initial configuration to a final configuration (where a configuration is defined by a location and an orientation), in the presence of obstacles, such that the path is a shortest among all paths with a prescribed curvature bound. The curvature-constrained shortest-path problem can also be seen as finding a shortest path for a point car-like robot moving forward at constant speed with a radius of curvature bounded from above by some constant. Previously, there is no known hardness result for the 2D curvature constrained shortest-path problem. This paper shows that the above problem in two dimensions is NP-hard, when the obstacles are polygons with a total of N vertices and the vertex positions are given within $O(N^2)$ bits of precision. Our reduction is computed by a family of polynomial-size circuits. This NP-hardness result provides evidence that there are no efficient exact algorithms for curvature-constrained shortest-path planning in arbitrary environments, and it justifies the approaches based on approximation and discretization used in most of the previous papers on curvature-constrained path planning.

*Surface address: INRIA Lorraine-Loria, 615, rue du Jardin Botanique, B.P. 101, 54602 Villers-lès-Nancy cedex, FRANCE E-mail: lazard@loria.fr

†Surface address: Department of Computer Science, Duke University, Durham, NC 27708-0129. E-mail: reif@cs.duke.edu. Supported in part by Grants NSF/DARPA CCR-9725021, CCR-96-33567, NSF IRI- 9619647, ARO contract DAAH-04-96-1-0448, and ONR contract N00014-99-1-0406. This paper was presented in the Workshop on the Algorithmic Foundations of Robotics (WAFR98), Houston, Texas,June, 1998.

1 Introduction

Roughly, a *curvature-constrained path* is a path whose curvature at any point along the path is no larger than a prescribed upper bound. Curvature-constrained path planning arises in a variety of robotics problems, e.g. the motions of robot vehicles controlled by steering mechanisms (see also Latombe [10]). This paper studies the complexity of planning curvature-constrained paths.

To be more precise, let $P : I \to R^d$ be a d-dimensional continuous differentiable path parameterized by arc length $s \in I$, for some real interval I (for simplicity, we regard both the function P and its image $P(I)$ in R^d as a path). The *average curvature* of P in the interval $[s_1, s_2] \subseteq I$ is defined by $\|P'(s_1) - P'(s_2)\|/|s_1 - s_2|$, where P' is the derivative of P. A continuous differentiable path parameterized by the arc length has an *upper-bounded curvature* C if its average curvature is upper bounded by C in every interval of I. A *curvature-constrained path* is a continuous differentiable path parameterized by the arc length that has an upper-bounded curvature C. (Note that the curvature of a curvature-constrained path is upper bounded by C everywhere the curvature is defined.)

A *configuration* is a tuple (p, θ), where p is a d-dimensional vector specifying a location, and θ specifies an orientation. (In 2 dimensions, it suffices that θ be a real number satisfying $0 \le \theta \le 2\pi$.) A 2-dimensional path P is *from* a configuration (p_1, θ_1) *to* a configuration (p_2, θ_2) if for interval $I = [s_1, s_2]$: (i) P has value p_1, p_2 respectively at s_1, s_2, and also (ii) the derivative of P has polar angle θ_1, θ_2, respectively at s_1, s_2.

Given a curvature bound, an initial configuration (p_1, θ_1), a final configuration (p_2, θ_2) and a set of obstacles, the *curvature-constrained shortest-path problem* is to find a path from the initial configuration to the final configuration that is a shortest path among all curvature-constrained paths avoiding the interior of the obstacles and connecting the initial configuration to the final configuration, or reporting that there is no such path.

In the rest of the paper, without further mentioning, paths will be implicitly assumed to satisfy the curvature constraint.

For the case of a region without obstacles, Dubins [7] characterized shortest paths in 2D, and Sussmann [16] recently extended the characterization for shortest paths in 3D.

In the presence of polygonal obstacles, Jacobs and Canny [9] proved the existence of a curvature-constrained shortest path between two configurations, if there exists a curvature-constrained path between these configurations. Fortune and Wilfong [8] gave the first algorithm which decides the existance or not of a shortest path avoiding obstacles. However, their algorithm required time super-exponential in the number n obstacles vertices, and did not actualy compute the exact shortest paths. A polynomial-time approximation algorithm was given by Jacobs and Canny [9], whose running time is $O(n^2(\frac{n+L}{\epsilon})\log n + (\frac{n+L}{\epsilon})^2)$ where L is the total length of the obstacle edges and ϵ represents the precision of the approximation. Their approximation algorithm was later improved by Wang and Agarwal [14] to reduce the running time to $O((\frac{n}{\epsilon})^2 \log n)$ and not to depend on the obstacle size. Agarwal, Raghavan, and Tamaki [1] also developed efficient approximation algorithms for the 2D shortest paths for a restricted class of obstacles (moderate obstacles). In three dimensions, Reif and Wang [13] developed non-uniform discretization approximations for kinodynamic motion planning and curvature-constrained shortest paths, whose computational complexity is considerably lower than those of the algorithms presented in [6, 12] for kinodynamic motion planning. **Note: slightly modified the sentence - check. S.L.**

Note: Shall we be more complete on the references? (see section 1.1 of my paper with pankaj et al. which I put on the website). S.L. Response of JR to S.L.: please add in your more complete and upto date references.

Canny and Reif [5] proved that the problem of finding a shortest 3-dimensional path, with no curvature constraints, and avoiding polygonal obstacles, is NP-hard. (Subsequently, Asano, Kirkpatrick and Yap [4] proved using similar techniques the NP-hardness of optimal motion of a rod on a plane with polygonal obstacles.) As a consequence of the above result, the curvature-constrained shortest-path problem in 3D is at least NP-hard. Although the problem in 2D is long conjectured to be a hard problem, its complexity is never stated or proved. This paper provides the first known complexity result for the curvature-constrained shortest-path problem in 2D. We show that if the number of obstacle vertices N is an input parameter and the vertex positions are given within $O(N^2)$ bits of precision, this problem is NP-hard even in 2D.

Our approach to the complexity result is similar to the *path-encoding* approach developed in [5] (while the detailed techniques are quite different). This approach is to encode the Boolean assignments by paths and then to reduce the 3-SAT problem to finding a shortest path. Recall that a 3-SAT formula in CNF form with n Boolean variables $X_1, \ldots, X_n$ has the form

$$\bigwedge_{i=1,\ldots,m} C_i,$$

where each C_i is a clause of the form $(l_{i1} \vee l_{i2} \vee l_{i3})$. Each literal l_{ij} in turn is either a variable X_k or a negation of it. The 3-SAT problem is to decide whether there is a Boolean assignment to $X_1, \ldots, X_n$ such that the formula is satisfied.

There are basically three steps in the path-encoding approach. For technical reasons, we add in Section 4.5 an additional $2m$ Boolean variables. We arrange the obstacles such that there are 2^N shortest paths, where each path encodes a different Boolean assignment over the resulting $N = 2m + n$ Boolean variables.

For a clause, we can construct the obstacles (to form a "clause filter") such that the path whose encoding does not satisfy this clause is stretched a little longer than those paths with satisfying encodings. We can arrange the obstacles such that all the 2^N paths pass through all m "clause filters". The 3-SAT formula is satisfiable if and only if there exists a path whose length is not stretched more than a certain amount (the details will be in later sections).

The third and final step is to have a reverse of the obstacle arrangement so far. This will make all the 2^N paths end at the same location with the same orientation, which is the final configuration. From the construction, it can be shown that the total number of obstacle vertices used in the above three steps is N^2.

Our techniques for generating and encoding the paths are quite different from those of [5], where 3D obstacles are used and a 3D path is represented by the obstacle edges it passes through. In our case, a path is 2 dimensional and is represented by its orientation at certain common locations. In [5], filtering the 3 literals in a clause can be done in parallel because it is in 3D, while we have to use a more complicated encoding to emulate this in our 2D construction.

This paper is organized as follows. In Section 2 we describe the *basic gadget*, an arrangement of some obstacles, which is the building block of our obstacle construction. In Section 3 we show how to use the basic gadgets to construct what we call *splitters* to generate 2^N shortest paths. In Section 4 we show in detail the reduction from the 3-SAT problem to finding a shortest path. In Section 5 the error analysis completes the proof. In Section 6 we state an open problem and give acknowledgements.

In the rest of the paper, we use the following notations. A 3-SAT problem involves m clauses in n Boolean variables, $X_1, \ldots, X_n$. N is $2m + n$. The number of bits of precision is $\beta = cN^2$, where c is a constant. Without loss of generality, we set the curvature bound to be 1.

2 Basic Gadgets and Their Properties

2.1 Basic gadgets

A *basic gadget* $\mathcal{B}$ (see Figure 1) consists of line-segment obstacles. These line segments form the boundary of two squares joined together at point v. The consecutive line segments of each square touch each other at their endpoints; except there are "holes" at the *start point* s, the *middle point* v and the *target point* t such that the point robot can pass through. The segments are also connected so that a robot cannot escape at the middle point (i.e., the right endpoint of the upper segment of the lower square touches the lower endpoint of the lower left segment of the upper square, and the upper endpoint of the right segment of the lower square touches the lower endpoint of the lower right segment of the upper square).

The purpose of a basic gadget is to force a robot to pass through point v to get to point t once it enters the first square at point s. For a basic gadget, we also require that the distance $|sv|$ from s to v be the same as the distance $|vt|$ from v to t, and that the non-oriented angle $\angle s\,v\,t$ is $\frac{3\pi}{4}$. There are two types of basic gadgets: if t is on the left side of the oriented line from s to v then it is a type L basic gadget, otherwise it is a type R basic gadget. Let b denote the edge length of the two squares of the gadget (i.e., $b = \frac{|sv|}{\sqrt{2}} = \frac{|vt|}{\sqrt{2}}$). All the basic gadgets in discussion are of the same fixed b. Furthermore, we assume in the rest of the paper that $b \in (1, 2)$.

For any two points p, q, define $\overrightarrow{pq}$ to be the ray that starts at point p, passes through point q and has direction going from p to q. If the robot enters the first square at configuration (s, ϑ), then the *input ray* is defined to be the ray that starts at s and has an orientation of ϑ. The *input angle*, denoted θ, is defined to be (a measure in $[-\pi, \pi]$ of) the clockwise angle from ray $\overrightarrow{sv}$ to the input ray for a type L basic gadget, and the counter-clockwise one for a type R basic gadget (see Figure 1). Due to the obstacles comprising the

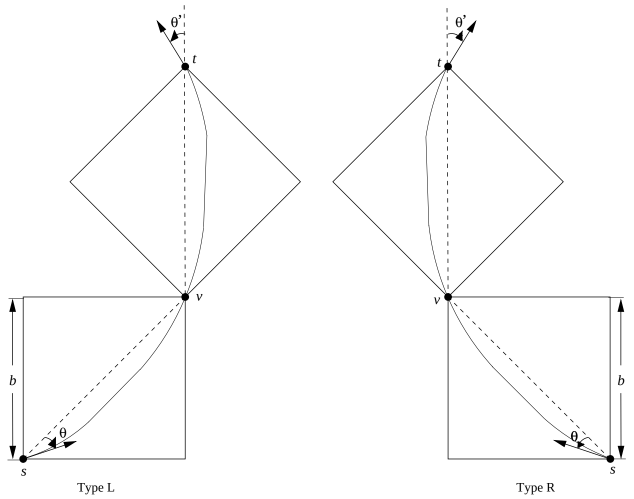

Figure 1: A basic gadget of type L or R with path $P^{(\theta)}$ (which consists of a line segment, a circular arc, and a line segment).

basic gadget, it is implied that the range of the input angle is $[-\pi/4, \pi/4]$. The angle θ is an *extrema* if it is either $-\pi/4$ or $\pi/4$. If the robot exits the second square at configuration (t, ϑ'), the *output ray* is defined to be the ray that starts at t and has on orientation angle ϑ'. The *output angle*, denoted θ', is defined to be (a measure in $[-\pi, \pi]$ of) the counter-clockwise angle from $\overrightarrow{vt}$ to the output ray for a type L basic gadget, and the clockwise one for a type R basic gadget.

Let $P^{(\theta)}$ be a shortest path from s to t with an input angle of θ (see Figure 1), and let $\rho(\theta)$ and $\psi(\theta)$ be the path length and the output angle of $P^{(\theta)}$, respectively. Let $\bar{\theta}$ be such that

$$\rho(\bar{\theta}) = \min_{\theta \in [-\pi/4, \pi/4]} \rho(\theta). \tag{1}$$

Therefore $P^{(\bar{\theta})}$ is a shortest path from s to t, among all paths with different input angles. We call $P^{(\bar{\theta})}$ the *canonical path* of the basic gadget and call $\bar{\theta}$ the *canonical input angle*. Let $\psi(\bar{\theta}) = \bar{\theta}'$ be the *canonical output angle* of canonical path $P^{(\bar{\theta})}$.

Lemma 2.1 *A basic gadget is such that (i) $P^{(\bar{\theta})}$ is unique, (ii) $\psi(\bar{\theta}) = \bar{\theta}$, (iii) $\bar{\theta} \in (0, \frac{\pi}{4})$, and (iv) once a curvature-constrained path enters the gadget at point s, it can only exit at the target point t.*

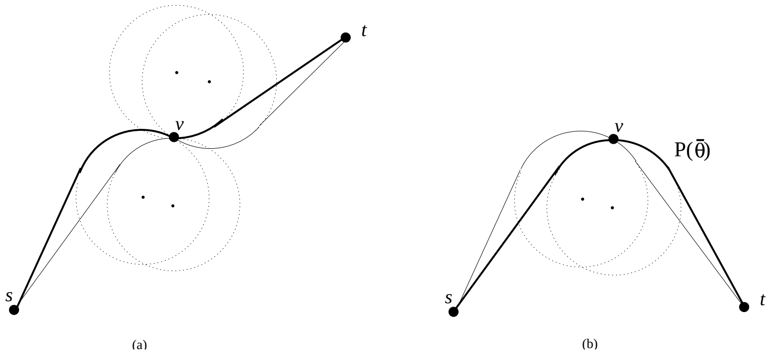

Figure 2: Curvature-constrained paths from s to t through v.

Proof: Since the edge length of the gadget is less than 2 ($b < 2$), no unit disk is totally contained in the gadget. It follows that (iv) once a point robot enters the gadget at point s, then it can only exit at point t, because the region bounded by a curvature-constrained path whose initial and final locations coincide (at s) contains at least one disk of unit radius; see [11] (in Russian and proved in a slightly less general form) or [2].

Since the edge length of the gadget is bigger than 1 ($b > 1$), point t does not belong to the interior of any circle of unit radius passing through v and tangent to a path (in the gadget) at v. By [3], the curvature-constrained shortest path from a configuration, say (v, ϑ), to a final point, say t, that does not belong to the interior of the two circles of unit radius tangent at v to the line through v with orientation ϑ, consists of a circular arc of unit radius followed by a line segment. Therefore, by a symmetric argument on the reversed path from v to s, a curvature-constrained shortest path from s to t, through v, consists of a line segment leaving s, a circular arc (of unit radius) reaching v, another circular arc leaving v, and a line segment reaching t (see Figure 2). Furthermore the two circular arcs lie on the same circle because otherwise the path is not a curvature-constrained shortest path; indeed, by convexity the thin path shown in Figure 2a shortens the thick path. Hence a curvature-constrained shortest path from s to t, through v, consists of a line segment starting at s, followed by a circular arc (of unit radius) that goes through v, followed by a line segment reaching t (see Figure 2b).

Such a path can be represented by a string of thickness zero that goes from s to t and passes through a pulley of radius 1 attached to v on its

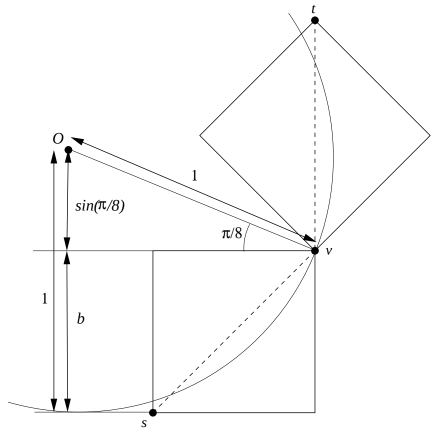

Figure 3: The unit circle centered at O goes through v and is tangent to the horizontal line through s.

boundary. Then, a shortest such path is obtained as an equilibrium of such a mechanical system, where the pulley can freely rotate around v, and two equal forces are applied at the endpoints s and t of the string. An equilibrium is obtained when the sum of the torques of these two forces is zero. Let $\vec{U}_s$ and $\vec{U}_t$ be unit vectors tangent to the path at s and t, respectively, F be the norm of the force applied on the string at s and t, and c be the center of the pulley/disk that rotates around v. Then the sum of the torques about v is $(-F.\vec{U}_1 + F.\vec{U}_2) \wedge \overrightarrow{cv}$ which is equal to zero only if the circular arc of the path is a whole circle (i.e., $\vec{U}_1 = \vec{U}_2$), or cv is the inner bisector of the two line segments of the path. Since the first case does not applies here because no unit circle entirely lies inside the gadget, we get that the shortest path $P^{(\bar{\theta})}$ is (see Figure 2b) symmetric with respect of the interior bisector of the segments sv and vt, and consists of a line segment starting at s, followed by a circular arc (of unit radius) whose midpoint is v, followed by a line segment reaching t. It then follows from that characterization that (i) $P^{(\bar{\theta})}$ is unique, (ii) $\psi(\bar{\theta}) = \bar{\theta}$, and $\bar{\theta} \in (0, \frac{\pi}{4}]$.

Finally, we prove that $\bar{\theta}$ is not an extrema $\pi/4$, implying (iii) $\bar{\theta} \in (0, \frac{\pi}{4})$. $\bar{\theta}$ is equal to $\pi/4$ only if the circular arc of $P^{(\bar{\theta})}$ is tangent to the horizontal line through s (assuming that the position of the gadget is as on Figure 1). Let O denote the center of the circle supporting the circular arc of $P^{(\bar{\theta})}$ and refer to Figure 3. Since O lies on the interior bisector of the segments sv and vt,

O is at distance 1 from the horizontal line through s when $1 = \sin(\pi/8) + b$, which never happens since $b \in (1, 2)$ and $\sin(\pi/8) > 0$. ∎

The major part of the lower-bound proof is to show that by using $O(N)$ basic gadgets, we can construct 2^N different paths from the initial state to the final state such that these 2^N paths are all shortest between the initial state and the final state up to $\beta = cN^2$ bits of precision.

Note: Probably to early to define difference angle (not used before the Analysis) so probably good to remove next paragraph. Also: Is the previous paragraph at the right place, or needed? S.L.

As we will see later in our construction, at some stages, all paths generated so far pass through the target point of the same basic gadget. Thus the paths cannot be distinguished by their locations. They have to be distinguished by their orientations. Since all the paths exit through the same gadget, it implies that they can be distinguished by the output angles. Furthermore, since the output ray of canonical path $P^{(\bar{\theta})}$ is fixed, an output angle of a path $P^{(\theta)}$ can be represented by the angle from the output ray of $P^{(\bar{\theta})}$ to the output ray of $P^{(\theta)}$. We call the angle from an output ray of canonical path $P^{(\bar{\theta})}$ to the output ray of a given path $P^{(\theta)}$ the *difference angle*; it is counted clockwise for type L basic gadgets, and is counted counter-clockwise for type R basic gadgets.

2.2 Gadget sequences

Let $\mathcal{B}_1$ and $\mathcal{B}_2$ be two basic gadgets. $\mathcal{B}_2$ is a *sequel* of $\mathcal{B}_1$ if the start point of $\mathcal{B}_2$ coincides with the target point of $\mathcal{B}_1$ and $\mathcal{B}_1, \mathcal{B}_2$ are rotationally oriented so that the output ray of the canonical path of $\mathcal{B}_1$ is the same as the input ray of the canonical path of $\mathcal{B}_2$. Note that if $\mathcal{B}_1$ and $\mathcal{B}_2$ are gadgets of different types, and $\mathcal{B}_2$ is a sequel of $\mathcal{B}_1$, then by Lemma 2.1(ii), segment vt of $\mathcal{B}_1$ is aligned with segment sv of $\mathcal{B}_2$.

A $\sigma-$*gadget sequence* S is a sequence of gadgets $\mathcal{B}_i$ for $i = 1, \ldots, \sigma$, such that $\mathcal{B}_{i+1}$ is a sequel of $\mathcal{B}_i$, for $i = 1, \ldots, \sigma - 1$, and no non-consecutive gadgets intersect.

The start (resp. target) point of a gadget sequence is the start (resp. target) point of its first (resp. last) gadget. Similarly, the input (resp. output) angle of a gadget sequence is the input (resp. output) angle of its first (resp. last) gadget. Let $\sigma P^{(\theta)}$ denote a shortest curvature-constrained path from the start point to the target point of a $\sigma-$gadget sequence with an input angle of θ. By definition of a gadget sequence, the concatenation of the canonical paths of the gadgets in the sequence forms a curvature-constrained path, called the *canonical path* of a gadget sequence. By Lemma 2.1 we have:

Lemma 2.2 *The canonical path of a $\sigma-$gadget sequence S is a shortest curvature-constrained path from the start point s of S to its target point t, among all curvature-constrained paths from s with an input angle in $[-\pi/4, \pi/4]$ to t. Furthermore, the length of the canonical path is $\sigma\rho(\bar{\theta})$.*

The error analysis of both a basic gadget and a $\sigma-$gadget sequence are given in Section 5.2, where we prove: **Note: Check the section (5.2). Also the following lemma does not quite seem to be at the right place. S.L.**

Lemma 2.3 *In a basic gadget, if the input angle θ is sufficiently close to $\bar{\theta}$ (i.e., $\theta - \bar{\theta} = O(2^{-\beta/2})$)[1], then the length of $P^{(\theta)}$ is the same as the shortest path length $\bar{\rho} = \rho(\bar{\theta})$, up to the allowed β bits of precision. Furthermore, the accumulation of path length error (as given in (4))[2] over a gadget sequence of $\sigma = O(N^2)$ basic gadgets is no more than $2^{-\beta}$.*

2.3 Variant gadgets

The purpose of a variant gadget $\mathcal{B}(\delta)$ is to force the output orientation to differ from the output orientation of a basic gadget $\mathcal{B}$ by some amount distinguishable with β bits of precision. A variant gadget is constructed in the following way; see Figure 4. Let $\mathcal{B}$ be a basic gadget and s, v and t be the start point, the middle point and the target point, respectively. Let ℓ be the line though t such that the angle, oriented counter-clockwise if the variant gadget is of type L and clockwise otherwise, from the ray $\overrightarrow{vt}$ to the line ℓ is $\bar{\theta}$. Let s', v' and t' be the start point, the middle point and the target point of $\mathcal{B}(\delta)$ respectively. Let s' coincide with s, v' lie on the segment sv, let t' lie on the line ℓ, and let $|s'v'| = |v't'|$. The angle, denoted δ, between segment vt and segment $v't'$ is oriented clockwise if the variant gadget of type L, and counter-clockwise otherwise. [3] Similarly as for gadgets, a variant

[1] **Note: Is that right? (its a try) S.L.**

[2] **Note: the relationship between the accumulation of path length error and the equation is not obvious. S.L.**

[3] **Note: There were some inconsistency between the text and the figure for the definition of ℓ and δ. Also the angle between to line is not oriented (explains why $\bar{\theta}$ is the angle from a ray to a line). I hope I got everything ok. S.L.**

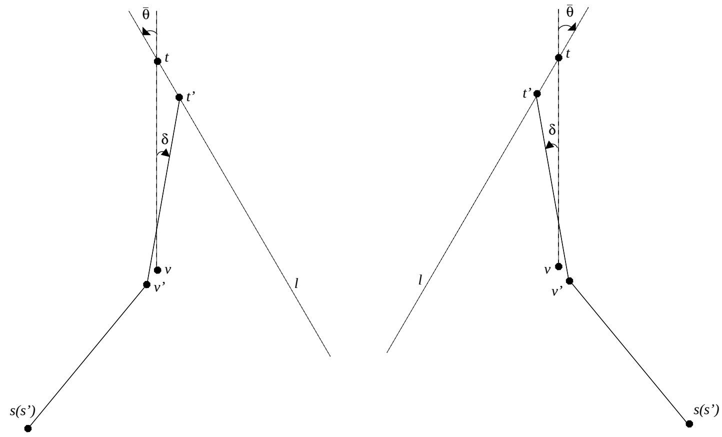

Figure 4: The modification of basic gadgets of type L and R into variant gadgets $\mathcal{B}(\delta)$ of type L and R, respectively.

gadget $\mathcal{B}(\delta)$ consists of line-segment obstacles that form the boundary of two squares whose diameters are segments $s'v'$ and $v't'$.

Let $P_\delta^{(\theta)}$ be a shortest path in variant gadget $\mathcal{B}(\delta)$ from s to t with input angle θ. Let $\bar{\theta}_\delta$ be the input angle that gives a shortest path from s' to t' in the variant gadget $\mathcal{B}(\delta)$.

Lemma 2.4 *For any sufficiently small δ, a variant gadget $\mathcal{B}(\delta)$ is so that (i) $P_\delta^{(\bar{\theta}_\delta)}$ is unique, (ii) $\psi(\bar{\theta}_\delta) = \bar{\theta}_\delta$, (iii) $\bar{\theta}_\delta \in (0, \frac{\pi}{4})$, and (iv) once a curvature-constrained path enters the variant gadget at point s', then it can only exit at the target point t'. Furthermore, the canonical path of a sequence S of variant gadgets is a shortest path from the start point s' of S to the endpoint t' of S, among all curvature-constrained paths from s' with input angle in $[-\pi/4, \pi/4]$ to t'.*

Proof: The edge length of a variant gadget is less than the edge length of a gadget since the length $|s'v'| = |v't'|$ is less than the length $|sv| = |vt|$. Since the edge length of a gadget is less than 2, the edge length of a variant gadget is less than 2. Thus a variant gadget contains no unit disk. (iv) then follows as in the proof of Lemma 2.1. The proof of (i), (ii) and (iii) follows exactly the proof of Lemma 2.1 (except in the last sentence that $\frac{\pi}{8}$ is now $\frac{\pi}{8} + \varepsilon$ where ε is a small constant depending on δ; this does not modify the argument since $\sin(\frac{\pi}{8}+\varepsilon)$ is still positive). Finally, (ii) directly implies that the last claim of the lemma. ∎

Error analysis of variant gadget is given in Section 5.3, where we show:

Lemma 2.5 *For sufficiently small δ, if the input angle θ is sufficiently close to $\bar{\theta}_\delta$, then the length of $P_\delta^{(\theta)}$ is the same as the shortest path length, up to the allowed β bits of precision. Furthermore, the accumulation of path length error over a variant gadget sequence of $\sigma = O(N^2)$ variant gadgets is no more than $2^{-\beta}$.*

3 Generating 2^N Paths

3.1 Mergers

A *merger* consists of an upper half and a lower half. (See Figure 5.) The upper half is constructed from a concatenation of 4 gadgets $\mathcal{B}_i$ (less portions below line ℓ defined below) with start point s_i, middle point v_i, and target point t_i, for $i = 1, \ldots, 4$ where $\mathcal{B}_1, \mathcal{B}_4$ are basic gadgets of type L, and $\mathcal{B}_2, \mathcal{B}_3$ are basic gadgets of type R. The lower half is the symmetric of the upper half with respect to the line ℓ (defined below); it thus consists of 4 gadgets $\mathcal{B}'_i$ (less portions above line ℓ) with start point s'_i, middle point v'_i, and target point t'_i, for $i = 1, \ldots, 4$ where $\mathcal{B}'_1, \mathcal{B}'_4$ are basic gadgets of type R, and $\mathcal{B}'_2, \mathcal{B}'_3$ are basic gadgets of type L.

Let ℓ be the oriented line supporting the input ray of the canonical path of $\mathcal{B}_1$, and assume for simplicity that ℓ is a horizontal line. We only consider the part of $\mathcal{B}_1$ above line ℓ; for simplicity we denote here the part of $\mathcal{B}_1$ above line ℓ by $\mathcal{B}_1$. Let $\mathcal{B}_2$ be a type R basic gadget such that $\mathcal{B}_1$ and $\mathcal{B}_2$ form a $2-$gadget sequence; since $\mathcal{B}_1$ and $\mathcal{B}_2$ are of different type, $t_1 = s_2$ is the midpoint of segment $v_1 v_2$ (by Lemma 2.1(ii)). Note that neither $\mathcal{B}_2$ not the second square of $\mathcal{B}_1$ are intersected by ℓ.

Let $\mathcal{B}'_1$ and $\mathcal{B}'_2$ be the symmetric of $\mathcal{B}_1$ and $\mathcal{B}_2$ with respect to line ℓ, and let $\mathcal{B}_3, \mathcal{B}_4, \mathcal{B}'_3$ and $\mathcal{B}'_4$ be the symmetric of $\mathcal{B}_2, \mathcal{B}_1, \mathcal{B}'_2$ and $\mathcal{B}'_1$, respectively, with respect to the line through t_2 and t'_2.

Let the *input angle* (resp. *output angle*) of a path in a merger be the angle, oriented counter-clockwise, from line ℓ to the input (resp. output) ray of the path. Let $M^{(\theta)}$ (resp. $M'^{(\theta)}$) be a curvature-constrained shortest path from s_1 to t_4 with an input angle θ, that goes through the upper half (resp. lower half) of the merger.

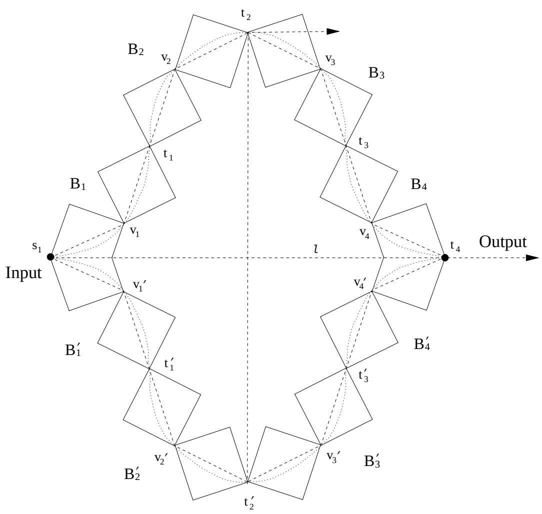

Figure 5: A merger.

The two paths form s_1 to t_4 that consist of concatenation of canonical paths of the gadgets in the merger, are called *canonical path* of the merger. We prove in Lemma 3.2 that the two canonical paths of a merger are curvature-constrained shortest paths. Note that the input and output angles of the two canonical paths are equal to 0 by construction.

We assume in the rest of the paper that the edge length b of the basic gadgets is now in $(1, 1+\frac{\sqrt{2}}{2})$.

Note: These lemmas probably still need some work, in particular the last one whose proof is I think not ok. Also need to think about whether all these following lemmas are actually used? S.L.

Lemma 3.1 *Once a curvature-constrained path enters a merger at point s_1 (i.e., the input angle belongs to $[-\frac{\pi}{4}-\bar{\theta}, \frac{\pi}{4}+\bar{\theta}]$), it can only exit at the point t_4.*

Proof: By construction of the merger, a path enters the merger at s_1 if and only its input angle is in $[-\frac{\pi}{4}-\bar{\theta}, \frac{\pi}{4}+\bar{\theta}]$. As in the proof of Lemma 2.1(iv), it is sufficient to prove that the merger does not contain any disk of unit radius.

Since the edge length b of the basic gadgets is in $(1, 1+\frac{\sqrt{2}}{2})$, no unit disk lies entirely in the basic gadgets of the merger. We now prove using simple trigonometric arguments that no unit disk lies in the first (or the last) element of the merger, the "union" of the basic gadgets $\mathcal{B}_1$ and $\mathcal{B}_1'$.

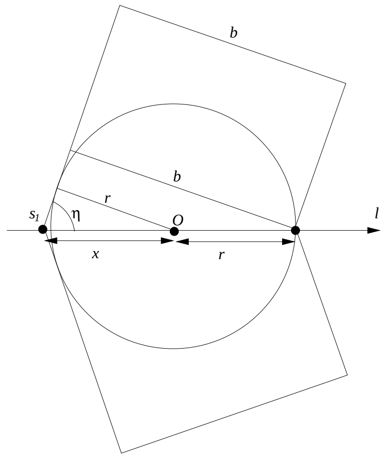

Figure 6: A merger does not contain any disk of unit radius.

Refer to Figure 6. Let O and r be the center and the radius of the biggest disk that lies entirely in the first element of the merger; O lies on ℓ. Let x be the distance between O and s_1, and η be the (non-oriented) angle between ℓ and one of the edges of the merger connected at s_1; η is equal to $\frac{\pi}{4} + \bar{\theta}$. We have $\sin\eta = \frac{r}{x} = \frac{b}{x+r}$. Thus $x = \frac{r}{\sin\eta}$ and $\sin\eta(\frac{r}{\sin\eta} + r) = b$. It follows that $r = \frac{b}{1+\sin\eta}$. Since $\bar{\theta} \in (0, \frac{\pi}{4})$ by Lemma 2.1, $\eta \in (\frac{\pi}{4}, \frac{\pi}{2})$, thus $\sin\eta > \frac{\sqrt{2}}{2}$ and $\frac{1}{1+\sin\eta} \in (0, \frac{1}{1+\sqrt{2}/2}))$. It thus follows from $b \in (1, 1 + \frac{\sqrt{2}}{2})$ that $r < 1$. ▮

Lemma 3.2 *The two canonical paths of a merger are curvature-constrained shortest paths from the start point s_1 to the endpoint t_4, among all curvature-constrained paths from s_1 with input angle in $[-\frac{\pi}{4} - \bar{\theta}, \frac{\pi}{4} + \bar{\theta}]$ to t_4.*

Proof: We first prove that the two canonical paths of a merger are curvature-constrained paths. We assume as before that the line ℓ (supporting the input ray of the canonical paths of the merger) is horizontal and refer to Figure 5. The two gadgets $\mathcal{B}_1$ and $\mathcal{B}_2$ form a $2-$gadget sequence. As mentioned before, since $\mathcal{B}_1$ and $\mathcal{B}_2$ are of different type, $t_1 = s_2$ is the midpoint of segment v_1v_2 (by Lemma 2.1(ii)). Thus segments s_1v_1 and v_2t_2 are parallel, which implies that the input ray of the canonical path of $\mathcal{B}_1$ (which is supported by ℓ) is parallel to the output ray of the canonical path of $\mathcal{B}_2$. Thus, this output ray is horizontal. Now, since t_2' is the symmetric of t_2 with respect to line ℓ

(by construction), the line through t_2 and t_2' is vertical, and since $\mathcal{B}_3$ is the symmetric of $\mathcal{B}_2$ with respect to this line, the output ray of the canonical path of $\mathcal{B}_2$ coincides with the input ray of the canonical path of $\mathcal{B}_3$. Thus $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ and $\mathcal{B}_4$ form a 4−gadget sequence and the concatenation of their canonical paths forms a curvature-constrained path. By symmetry, the concatenation of the canonical paths of $\mathcal{B}_1', \mathcal{B}_2', \mathcal{B}_3'$ and $\mathcal{B}_4'$ is also a curvature-constrained path. Since the canonical paths of the merger are concatenation of canonical of basic gadgets, and any path that enters the merger at s_1 can only escape at t_4 (by Lemma 3.1), it follows that the two canonical paths of the merger are curvature-constrained shortest paths among all the curvature-constrained paths from s_1 with input angle in $[-\frac{\pi}{4} - \bar{\theta}, \frac{\pi}{4} + \bar{\theta}]$ to t_4. ∎

Lemma 3.3 *For any input angle θ sufficiently close to 0, the two curvature-constrained shortest paths $M^{(\theta)}$ and $M'^{(\theta)}$ have same length and output angle within error $O(2^{-\beta})$.*

Proof: If $\theta = 0$, then by definition of the merger, the two canonical paths in the low and upper halves merge as required, and have the same path distance at this point at the exit t_4 of the merger, within error $O(2^{-\beta})$.

When the input angle is $\theta \neq 0$, we appeal to Lemma 5.1 to track the changes in orientations. Lemma 5.1 states that for a gadget and any input angle θ sufficiently close to $\bar{\theta}$, the path $P^{(\theta)}$ reaches the target in distance $\rho(\theta) = \rho(\bar{\theta}) + O((\theta - \bar{\theta})^2) + O(2^{-\beta})$ and with output angle $\psi(\theta) = c_0 + c_1(\theta - \bar{\theta}) + O((\theta - \bar{\theta})^2) + O(2^{-\beta})$, for constants c_0, c_1 (where the sign of c_1 depends on the type of the gadget). By definition, the input angle $\bar{\theta}$ for a gadget corresponds to the input angle 0 for a merger. Thus for any input angle θ sufficiently close to 0, the two curvature-constrained shortest paths $M^{(\theta)}$ and $M'^{(\theta)}$ exist and have same length $4\rho(\bar{\theta})$ within error $O(\theta^2) + O(2^{-\beta})$.

We track the changes in orientations by applying compositions of these linear output angle functions, which are identical in the two halves (except the lower paths have changes that are the negative of the upper paths). Hence we have shown the two paths with input angle θ have the same orientation and path distance at point t_4 within error $O(\theta^2) + O(2^{-\beta})$. Therefore, up to this allowed error, the two paths merge at t_4, and this merger by itself does not double the number of paths. **Note: I have a problem here: If c_0 is not equal to $\bar{\theta}$, then it cannot be ensured that the input angle of B_2 (in the merger) is sufficiently close to $\bar{\theta}$ and Lemma 5.1 does not apply. Another question: Does "θ sufficiently close to 0" mean**

that $\theta = O(2^{-\beta/2})$ so that $\theta^2 = O(2^{-\beta})$? (If so perhaps it should be stated in the various lemmas.) S.L. ▮

3.2 Splitters

By replacing gadgets $\mathcal{B}_4$ and $\mathcal{B}'_4$ in the merger with variant gadgets $\mathcal{B}(\delta)$ as defined in Section 2.3, we obtain a *δ-splitter* $\mathcal{S}(\delta)$. Let $\Delta = \Delta(\delta)$ be the angular orientation change induced by a variant gadget $\mathcal{B}(\delta)$ as controlled by the parameter δ. It can be set to ensure the changes are distinguishable within β bits. When the sub-path that goes through the upper half of the δ-splitter and the sub-path that goes through the lower half of the δ-splitter meet at point t_4, their orientations differ by $2\Delta(\delta)$ due to the variant gadgets. We now to prove that a path is doubled when it goes through a δ-splitter:

Proposition 3.1 *For any θ on the interval $[-\pi/4, \pi/4]$, when a path with input angle θ splits into two paths at the entrance of a δ-splitter, the two paths in the low and upper halves created that way, have orientation $\theta + \Delta(\delta)$ and $\theta - \Delta(\delta)$ at the exit t_4 to the $\delta-$splitter, and have the same path length, within error $O(2^{-\beta})$.*

Proof: If $\theta = \bar{\theta}$, then by definition of the δ-splitter, the two paths in the low and upper halves created that way, clearly split into two paths that have orientation $\bar{\theta} + \Delta(\delta)$ and $\bar{\theta} - \Delta(\delta)$ and have the same path distance at the exit t_4. When the input angle is $\theta \neq \bar{\theta}$, then we need to appeal again (as the case of a merger) to Lemma 5.1 to track the changes in orientations. Again we track the changes in orientations by applying compositions of the linear output angle functions given by Lemma 5.1; these linear output angle functions are again identical in the two halves (except the lower paths have angular changes that are the negative of the upper path angular changes). Hence we have shown the two paths with input angle θ exit t_4 with orientation $\theta + \Delta(\delta)$ and $\theta - \Delta(\delta)$, and they have the same path distance at this point within error $O((\theta - \bar{\theta})^2) + O(2^{-\beta})$. ▮

Hence, given a set of 2^k paths with consecutive input orientation difference 2Δ, the appropriate choice of δ (so that the resulting orientation change caused by $\mathcal{B}(\delta)$ is $\Delta = \Delta(\delta)$) is 1/2 the orientation difference between consecutive paths), a δ-splitter doubles the number of paths.

Therefore, there exists a set $\{\delta_1, \ldots, \delta_N\}$, defining N splitters $\mathcal{S}(\delta_1), \ldots, \mathcal{S}(\delta_N)$ where each splitter $\mathcal{S}(\delta_i)$ provides a change in angular orientation $\Delta_i(\delta_i)$, for

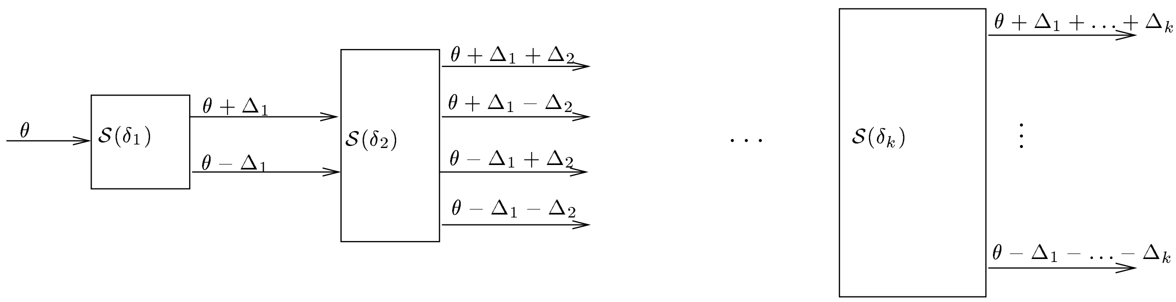

Figure 7: The scheme of using k splitters $\mathcal{S}(\delta_i)$ to generate 2^k distinct orientations and paths, with changes in angular orientation Δ_i, for $i = 1, \ldots, k$.

$i = 1, \ldots, N$ distinguishable within β bits. (Even though we don't know how to compute these δ_i's, we can define and store each of them using a fixed size β circuit.)

The scheme of generating 2^k orientations is illustrated in Figure 7. These splitters $\mathcal{S}(\delta_1), \ldots, \mathcal{S}(\delta_N)$ are arranged in sequence horizontally, with the left entrance of $\mathcal{S}(\delta_i)$ at the same point as the right exit of $\mathcal{S}(\delta_{i-1})$, for each $i > 1$. Note that Δ_i is the change in angular orientation where as δ_i is the displacement parameter associated with the variant gadgets within splitter $\mathcal{S}(\delta_i)$. Note that the figure is illustrative of only the branching of paths, not their spatial configuration, since each of the resulting paths each enter and exit each splitter $\mathcal{S}(\delta_i)$ at the same points, and it is only the orientation angles of entrance and exit that differ for each path.

Summarizing the above, we have from Proposition 3.1:

Proposition 3.2 *For any θ on the interval $[-\pi/4, \pi/4]$, when a path with input angle θ goes through these N splitters $\mathcal{S}(\delta_1), \ldots, \mathcal{S}(\delta_N)$ are arranged in sequence horizontally, it splits into 2^N paths at the exit of the last splitter, with the resulting exiting paths having angle orientation distinguishable within β bits, and have the same path length, within error $O(2^{-\beta})$.*

4 Reduction from 3-SAT

Recall that a 3-SAT formula in n variables $X_1, \ldots, X_n$ has the form

$$\bigwedge_{i=1,\ldots,m} C_i,$$

where each clause C_i is of the form $(l_{i1} \vee l_{i2} \vee l_{i3})$. Each literal l_{ij} in turn is either a variable X_k or a negation of it. The 3-SAT problem is to decide whether there is a Boolean assignment to $X_1, \ldots, X_n$ such that the formula is satisfied.

This section describes the construction of a curvature-constrained shortest-path problem derived from the input 3-SAT formula. It will have polygonal obstacles defined by a total of $O(N)$ vertices (where $N = 2m + n$), each of which will be specified within cN^2 bits of precision, for a constant $c \geq 1$. Thus the resulting curvature-constrained shortest-path problem will be specified by a total of $O(N^{O(1)})$ bits. This reduction from the input $3-$SAT formula to our resulting curvature-constrained shortest-path problem will be computed in $N^{O(1)}$ time, with the exception of certain coefficients (e.g., the displacement parameter δ_i associated with the variant gadgets within splitter $\mathcal{S}(\delta_i)$) which depend on N. Therefore, our reduction can be computed by a family of polynomial-size circuits, where for each $N = 2m + n$, the Nth circuit of the family computes the reduction from any input 3-SAT formula to our resulting curvature-constrained shortest-path problem.

In the previous section we have described how we can construct gadgets to generate 2^N shortest paths, distinguishable within β bits. All the paths end at the same locations but with different orientations. We number the paths $0, \ldots, 2^N - 1$, ordered by their orientations. We introduce extra $2m$ Boolean variables $Y_1, Z_1, \ldots, Y_m, Z_m$. Each path $i \in \{0, \ldots, 2^N - 1\}$ is associated with a Boolean assignment to $N = n + 2m$ variables $Y_1 Z_1 \ldots Y_m Z_m X_1 \ldots X_n$, such that its binary encoded value is i. The basic idea here is that for each clause we can arrange the obstacles such that if a path's encoding does not satisfy the clause, the path will be "stretched" so it is longer than the canonical path by a distinguishable amount, say γ. On the other hand, if the encoding satisfies the clause, it will not be "stretched", i.e. its path length is the same as the length of the canonical path. Therefore, the 3-SAT formula is satisfiable if and only if there exists a path whose length is the same as the canonical path length, which we know beforehand.

4.1 Blockers

The 2^N paths numbered $0, \ldots, 2^N - 1$ have a *lower half* $0, \ldots, 2^{N-1} - 1$ (that is, those with negative θ_i) and *upper half* $2^{N-1}, \ldots, 2^N - 1$ (that is, those with positive θ_i). In the constructions below, we will need to block the lower half 2^{N-1} paths, or the upper half 2^{N-1} paths.

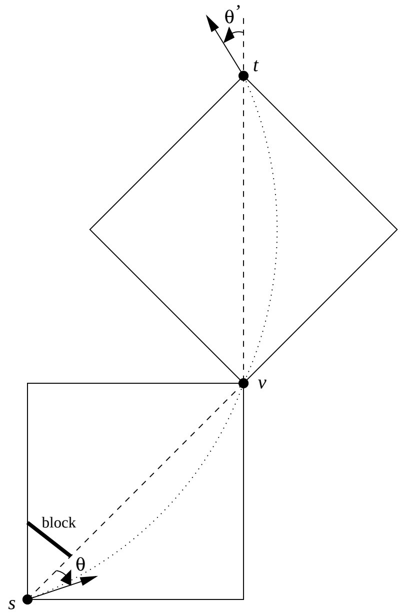

Figure 8: A basic gadget of type L modified to be a lower half blocking gadget.

To block the lower (upper, respectively) half 2^{N-1} paths within a type L basic gadget, we modify the type L basic gadget by placing a *blocking* line segment, perpendicular to line segment sv, from a point $2^{-2\beta}$ to the right of s on line segment sv to the line segment on the lower left (bottom, respectively) boundary of the gadget. The result will be called a *lower (upper, respectively) half blocking type L basic gadget.* (See Figure 8.)

To block the lower (upper, respectively) half 2^{N-1} paths within a type R basic gadget, we modify the type R basic gadget by placing a *blocking* line segment, perpendicular to line segment sv, from a point $2^{-2\beta}$ to the left of s on line segment sv to the line segment on the lower right (bottom, respectively) boundary of the gadget. The result will be called a *lower (upper, respectively) half blocking type 2 basic gadget.* (See Figure 9.)

In the lower half blocking construction in both cases L and R, we can observe by inspection of Figure 8 and Figure 9 that (i) clearly by inspection of Figure 9, those paths with positive orientation (i.e., the upper half 2^{N-1}) are not affected by the blocking segment, but (ii) this blocking segment is sufficiently close to s so that those paths with negative orientation (i.e., the

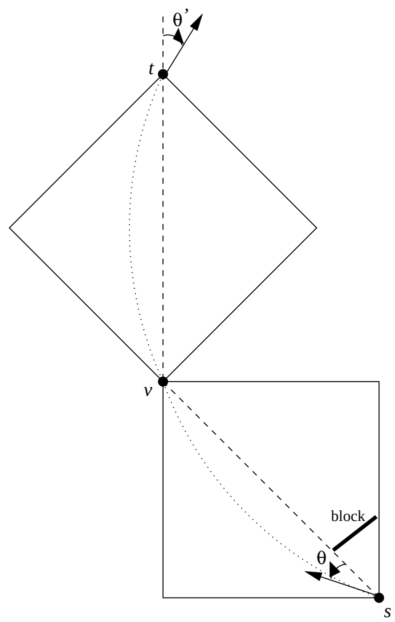

Figure 9: A basic gadget of type R modified to be an lower half blocking gadget.

lower half 2^{N-1}) the lower half 2^{N-1} paths are blocked and cannot proceed further (in particular, due to the unit curvature restriction, which allows less than $2^{-\beta}$ curvature change in the distance $2^{-2\beta}$ to the blocking segment, the lower half paths run into the blocking segment).

The analysis of the upper half blocking construction is similar: we can observe in both cases L and R, that (i) the lower half 2^{N-1} paths are not affected by the blocking segment, but (ii) the blocking segment is sufficiently close to s so that the upper half 2^{N-1} paths are blocked and cannot proceed further (in particular, due to the unit curvature restriction, the upper half paths again run into the blocking segment).

Summarizing the above, we have

Proposition 4.1 *The lower half blocking type L and R basic gadgets block the lower half 2^{N-1} paths and the upper half blocking type L and R basic gadgets block the upper half 2^{N-1} paths.*

4.2 Shufflers

For simplicity in this subsection, let the N Boolean variables $Y_1Z_1 \ldots Y_mZ_mX_1 \ldots X_n$ be temporarily renamed as $V_0V_1 \ldots V_{N-1}$, where $V_{2i-2} = Y_i, V_{2i-1} = Z_i$ for $i = 1, \ldots, m$ and $V_{i+2m-1} = X_i$ for $i = 1, \ldots, n$.

We now consider a rotational permutation of this list of Boolean variables, say $V_iV_{i+1} \ldots V_{N-1}, V_0V_1 \ldots V_{i-1}$. In this Boolean variable V_i is the *most significant bit* (placed in front of the encoding). In our use of 2^N paths to encode true values of these variables, this Boolean variable V_i is considered to be *exposed* iff the lower half of the (ordered) 2^N paths verify the Boolean variable is 0 and the upper half of the paths verify the Boolean variable is 1.

Suppose path $i \in \{0, \ldots, 2^N - 1\}$ has binary encoded value defining a Boolean assignment to the N variables $V_0V_1 \ldots V_{N-1}$, so V_0 is exposed. Now suppose the lower half 2^{N-1} paths are perfectly shuffled with the upper half 2^{N-1} paths, so that they are ordered $0, 2^{N-1}, 1, 2^{N-1}+1, \ldots, 2^{N-1}-1, 2^N-1$. Then each path $i \in \{0, \ldots, 2^N - 1\}$ now has binary encoded value defining a Boolean assignment to the N variables in order defined by a single variable rotation: $V_1V_2 \ldots V_NV_0$, and now the new first variable V_1 is exposed.

The way to achieve this is by a construction called a *shuffler*, which is similar to a splitter with some alterations. Recall the merger defined in Section 3 and the splitter as given in Section 3.2 is derived by replacing gadgets $\mathcal{B}_4$ and $\mathcal{B}'_4$ in the merger with variant gadgets $\mathcal{B}(\delta)$.

Let Δ_0 be the difference in angular orientation between the 2^N consecutive paths $0, \ldots, 2^N-1$. We choose a δ such that the resulting $\delta-$splitter $\mathcal{S}(\delta)$ provides a change in angular orientation $\Delta(\delta) = (2^{N-1} - 1/2)\Delta_0$. the $\delta-$splitter (without the below alterations) simply outputs (a) a *prefix sequence* of 2^{N-1} paths $0, 1, \ldots, 2^{N-1} - 1$ with angular orientation separation of Δ_0, followed by (b) an *shuffle sequence* of 2^N paths $0, 2^{N-1}, 1, 2^{N-1}+1, \ldots, 2^{N-1}-1, 2^N-1$ with angular orientation separation of $\Delta_0/2$, followed by (c) a *suffix sequence* of 2^{N-1} paths $2^{N-1}, 2^{N-1}+1, \ldots, 2^N-1$ with angular orientation separation of Δ_0. That is, it is a merge of the sequence $0, \ldots, 2^N - 1$ with the same $0, \ldots, 2^N - 1$ sequence with a shift of $2^{N-1} - 1/2$.

The alteration of a $\delta-$splitter used to construct a shuffler we simply add blocking segments to the third most (lower and upper) gadgets $\mathcal{B}_3$ and $\mathcal{B}'_3$ so that: (i) type L basic gadget $\mathcal{B}_3$ becomes a lower half blocking type 1 basic gadget that lets the 2^{N-1} paths $2^{N-1}, \ldots, 2^N - 1$ go through the upper half of the shuffler and blocks the 2^{N-1} paths $0, \ldots, 2^{N-1} - 1$ (i.e., the prefix sequence) from going through gadget $\mathcal{B}_3$ in the upper half of the shuffler, and (ii) type R basic gadget $\mathcal{B}'_3$ becomes a upper half blocking gadget that lets the 2^{N-1} paths $0, \ldots, 2^{N-1} - 1$ go through the lower half of the shuffler and blocks the 2^{N-1} paths $2^{N-1}, \ldots, 2^N - 1$ (i.e., the suffix sequence) from going through gadget $\mathcal{B}'_3$ in the lower half of the shuffler. (The blocking in gadgets $\mathcal{B}_3$ and $\mathcal{B}'_3$ is done as given in Proposition 4.1.)

With these alterations, the resulting shuffler simply blocks the above defined prefix sequence and suffix sequence, and so outputs the above defined shuffle sequence of 2^N paths $0, 2^{N-1}, 1, 2^{N-1} + 1, \ldots, 2^{N-1} - 1, 2^N - 1$ with angular orientation separation of $\Delta_0/2$.

In this way, the shuffler has made the second bit to be the most significant bit (in the binary encoding of the path numbers). In general, at most N shuffles can be used to permute the order of the paths so that each path i now has binary encoded value defining a Boolean assignment to the N variables in rotated order, to expose any given Boolean variable. So we can let the paths go through a sequence of at most N shufflers to expose any bit.

As described in [5], in order to filter all the paths whose Boolean variable V_j is set to 0 (or 1, respectively) we need to repeatedly shuffle the paths such that the V_j becomes the most significant bit, and then block the lower half (or upper half, respectively) 2^{N-1} paths by application of Proposition 4.1.

Summarizing the above, we have

Proposition 4.2 *For each of the encoded Boolean variables, N shufflers and* *one blockers suffice to filter all the paths where a given Boolean variable is set to a particular Boolean value.*

(Note: furthermore, if we wish to filter those paths encoding particular Boolean values for a sequence of two or more Boolean variables, we simply repeat this process for each of the variables.)

4.3 Stretchers and Filters

We next describe constructions called *stretchers*. The effect of a stretcher of type 1 is to stretch the paths so that they are longer than the canonical path by a distinguishable amount, where as the effect of a stretcher of type 0 is to not to stretch the paths.

In the above, each path $i \in \{0, \ldots, 2^N - 1\}$ was associated with a binary assignment to $N = n + 2m$ Boolean variables $Y_1 Z_1 \ldots Y_m Z_m X_1 \ldots X_n$, such that its binary encoded value is i.

We now introduce N^2 addition Boolean variables $W_1, \ldots, W_{N^2}$. Here each path $i' \in \{0, \ldots, 2^{N(N+1)} - 1\}$ was associated with a binary assignment to $N(N+1)$ Boolean variables $Y_1 Z_1 \ldots Y_m Z_m\ X_1 \ldots X_n, W_1, \ldots, W_{N^2}$, such that its binary encoded value is i'. Note that the original set of 2^N paths lie in the first 2^N paths of the $2^{N(N+1)}$ paths, where all the additional Boolean variables $W_j = 0$. By N^2 repeated uses of the shuffler of Section 4.2, extended to $N(N+1)$ Boolean variables, we can permute these $2^{N(N+1)}$ paths, corresponding to a rotation of the $N(N+1)$ Boolean variables as $W_1, \ldots, W_{N^2}, Y_1 Z_1 \ldots Y_m Z_m\ X_1 \ldots X_n$.

A stretcher of type 1 consists of these N^2 repetitions of the shuffler, as extended to these to $N(N+1)$ Boolean variables, followed by N^2 repetitions of the flip of the shuffler (which reverses these rotations).

Then Lemmas 5.1 and 5.5 imply that if $\bar{\theta}$ is the canonical start angle entering the first shuffler, an path entering $\mathcal{B}_1$ at its start point reaches the target point of $\mathcal{B}_2$ in distance which is the same as the canonical path distance through the transporter, with $O((\theta - \bar{\theta})^2) + O(2^{-\beta})$ error, and with output angle which is a linear function of θ, with $O((\theta - \bar{\theta})^2) + O(2^{-\beta})$ error. But due to the introduction of the N^2 addition Boolean variables, this error in path length is significant; in particular, it is detectable within cN bits. The error in path length is additive, and so is not reversed by the final N^2 repetitions of the flip of the shuffler (which reverses the rotations). On the other hand, the error of angle orientation is reversed by the final N^2 repetitions of the flip of the shuffler.

A stretcher of type 0 consists of these N^2 repetitions of the merger, extended to these to $N(N+1)$ Boolean variables, followed by N^2 repetitions of the flip of the merger.

After exit from either type of stretcher, the N^2 addition Boolean variables $W_1, \ldots, W_{N^2}$ are not used, and the original N Boolean variables $Y_1 Z_1 \ldots Y_m Z_m X_1 \ldots X_n$ are in their original order.

If we put stretchers of distinct types in the upper and the lower halves of a merger, we get a construction called the *filter*. In the case a stretcher of type 1 is in the upper half of the merger, and a stretcher of type 0 is in the lower half of the merger, then the filter filters out paths whose most significant bit is 1, i.e. paths whose most significant bit is 1 (going through the lower half of the merger) will have lengths the same as the canonical path length, while the paths whose most significant bit is 0 (going through the upper half of the merger) will be stretched by a distinguishable amount. On the other hand, if a stretcher of type 0 is in the upper half of the merger, and a stretcher of type 1 is in the lower half of the merger, then the filter filters out paths whose most significant bit is 0.

4.4 Transporters

We now breifly describe constructions which are called *transporters*. The effect of transporters are to translate the group of paths either upward or downward.

An *upward transporter* consists of a $2-$gadget sequence $\mathcal{B}_1, \mathcal{B}_2$ where $\mathcal{B}_1$ is a type 1 basic gadget and $\mathcal{B}_2$ is a type R basic gadget, positioned exactly the same as the gadgets $\mathcal{B}_1, \mathcal{B}_2$ of a merger, so that $\mathcal{B}_2$ is a sequel of $\mathcal{B}_1$.

A *downward transporter* consists of the flip of an upward transporter.

Since a transporter is a R-gadget sequence, Lemmas 5.1 and 5.5 imply that if $\bar{\theta}$ is the canonical angle entering $\mathcal{B}_1$, then for any input angle θ on the interval $[-\pi/4, \pi/4]$, an path entering $\mathcal{B}_1$ at its start point reaches the target point of $\mathcal{B}_2$ in distance which is the same as the canonical path distance through the transporter, with $O((\theta-\bar{\theta})^2) + O(2^{-\beta})$ error, and with output angle which is a linear function of θ, with $O((\theta-\bar{\theta})^2) + O(2^{-\beta})$ error.

4.5 The 3-SAT Reduction

A naive way to "emulate" a clause, say $X_i \vee \overline{X_j} \vee X_k$, is as follows. First we let the paths to go through a sequence of shufflers so that the bit X_i becomes the most significant bit. We let the paths then go through a filter that stretches all the paths whose X_i is set to 0 by an amount of γ. Second we shuffle the paths again so that X_j becomes the most significant bit. We then let the paths go through a filter that stretches all the paths whose X_j is set to 1 by an amount of γ. Finally we do the same for bit X_k as we did for bit X_i. Therefore, the binary encoding of a path satisfies clause $X_i \vee \overline{X_j} \vee X_k$ if and only if its length is stretched by no more than 2γ.

The above method does not work for the following reason. Take an example of path P_1 and path P_2 and a 3-SAT formula with two clauses. Path P_1 is stretched by 2γ for the first clause and then by 2γ for the second clause, whereas path P_2 is stretched by 3γ for the first clause and then is stretched by γ for the second clause. This means path P_1 satisfies the formula while path P_2 does not. However we cannot tell the difference in path length of P_1 and P_2. This is where the extra variables Y_i, Z_i, for $i = 1, \ldots, m$, are used. For the ith clause, the bits $Y_i Z_i$ specify which of the 3 clause literals the binary encoding will satisfy. Let $y_1 z_1 \ldots y_m z_m x_1 \ldots x_n$ be the binary encoding of a path (i.e. y_i, z_i and x_i are the Boolean assignments for variables Y_i, Z_i and X_i respectively). The binary encoding indicates that the $(2y_i + z_i + 1)$th literal of the ith clause should be satisfied by $x_1 \ldots x_n$.

Proposition 4.3 *For a 3-SAT formula, there exists a satisfying binary encoding, if and only if there exists a binary encoding $y_1 z_1 \ldots y_m z_m x_1 \ldots x_n$, such that $1 \leq 2y_i + z_i + 1 \leq 3$ and $x_1 \ldots x_n$ satisfies the $(2y_i + z_i + 1)$th literal of the ith clause, for $i = 1, \ldots, m$.*

Recall that Y_i is the most significant bit iff the first half of the (ordered) 2^N paths verify $Y_i = 0$ and the second half of the paths verify $Y_i = 1$. We shuffle the 2^N paths so that Y_i becomes the most significant bit. Note that this shuffling corresponds to a rotation of the Boolean variables, where Z_i follows Y_i. As a consequence, Z_i becomes the next most significant bit. Using these two bits, we divide the paths into 4 groups. A group is labeled j, if $2Y_i + Z_i + 1 = j$. Let X_{ij} be the variable for the jth literal of the ith clause. For the paths in group j, for $j = 1, 2, 3$, we send them into a subconstruction that first exposes the bit X_{ij} (i.e. makes X_{ij} the most significant bit) and then filters the most significant bit to be either 0 or 1 as appropriate.

In the meantime, we block all the paths in path group 4. Finally we reverse this construction so that all the paths come to a common locations with different orientations.

Let $y_1 z_1 \dots y_m z_m x_1 \dots x_n$ be the binary encoding of a path. At this moment, this path is not stretched by an amount of γ, if and only if $1 \leq 2y_i + z_i + 1 \leq 3$ and $x_1 \dots x_n$ satisfies the $(2y_i + z_i + 1)$th literal of the ith clause.

The spatial layout of the clause filter constructions. To insure that the shufflers and filter of the three groups not blocked do not overlap or intersect in the plane, we use a sequence of transporters, as defined in Section 4.4, to route the first three path groups into distinct and consecutive vertical sections of the plane, and then use transporters and mergers to route them together. In particular, the first group is routed by a sequence of two upward transporters, the second group is routed by an upward transporter followed by a downward transporter, and the third group is routed by a sequence of two downward transporters. The reverse process is used to route the groups back together.

Furthermore, to layout the sequence of all the clause filter constructions we also use a horizontal sequence of transporters, as defined in Section 4.4, which clearly provides a planar layout. We define the final point p_f to be at the exit to the final clause filter.

This completes our description of the construction for filtering all paths satisfying each given clause of the Boolean formula.

Initialization of the Construction. We have shown that we can arrange the obstacles so that there exist 2^N shortest paths from an initial configuration. The initial location is chosen to be at the start point of the first gadget defined in the construction. To complete the construction, we apply Lemma **??** to show that the initial orientation exists (note that since our NP-reduction is via circuits, we do not need formally compute this initial orientation; just need to show it exists). For a given $\bar{\theta}$, we can determine b by the solution of a quadratic equation given in (**??**). We need to specify b only up to β bits of precision.

Summarizing the above, and applying Proposition 4.3, we have

Lemma 4.1 *The initialization and construction for filtering all paths satisfying all the clauses requires a total of $O(N^2)$ mergers, shufflers, filters, and other basic and variant gadgets, which can be placed in the plane without intersection.*

The error analysis of the 3-SAT Construction, and in particular the proof of Lemma 2.3, is given in Section 5.4. This implies our main result:

Theorem 4.1 *Using circuits of polynomial size we can give a reduction from any given a 3-SAT formula in CNF form with n Boolean variables and m clauses, with $N = 2m + n$, to a unit curvature-constrained path problem with $N^{O(1)}$ segments and specified within $\beta = O(N^2)$ bit precision, such that the formula is satisfied by a Boolean assignment iff there is a unit curvature-constrained path of specified length L, and otherwise all unit curvature-constrained paths are of length $\leq L - \Omega(2^{-\beta})$.*

5 Error Analysis

We will prove here

Lemma 5.1 *For any input angle θ on the interval $[-\pi/4, \pi/4]$, such that θ is sufficiently close to $\bar{\theta}$, the path $P^{(\theta)}$ exists and reaches the target in distance $\rho(\theta) = \rho(\bar{\theta}) + O((\theta - \bar{\theta})^2) + O(2^{-\beta})$ and with output angle $\psi(\theta)$ which is $c_0 + c_1(\theta - \bar{\theta}) + O((\theta - \bar{\theta})^2) + O(2^{-\beta})$ for constants c_0, c_1 (where the sign of c_1 depends on whether the basic gadget is type L or type R).*

We will also show this implies Lemma 2.3.

5.1 Error analysis of the difference angle of a gadget sequence

For a basic gadget (see Figure 10), let $x = \theta - \bar{\theta}$ be the angle from the input ray of canonical path $P^{(\bar{\theta})}$ to the input ray of path $P^{(\theta)}$, oriented (as for θ) clockwise if the gadget is of type L and counter-clockwise otherwise. We define $f(x) = \psi(\bar{\theta} + x) - \bar{\theta}$ $(= \psi(\theta) - \psi(\bar{\theta})$ by Lemma 2.1(ii)) as the angle from the output ray of $P^{(\bar{\theta})}$ to the output ray of $P^{(\theta)}$, oriented (as for $\psi(\theta)$) counter-clockwise if the gadget is of type L and clockwise otherwise. We call x the *input difference angle* and $f(x)$ the *output difference angle.* We also define $g(x) = f(-x)$.

Lemma 5.2 *In a basic gadget, for any input angle θ sufficiently close to $\bar{\theta}$, the path $P^{(\theta)}$ exists and reaches the target with output angle $\psi(\theta) = \bar{\theta} + c_0 +$*

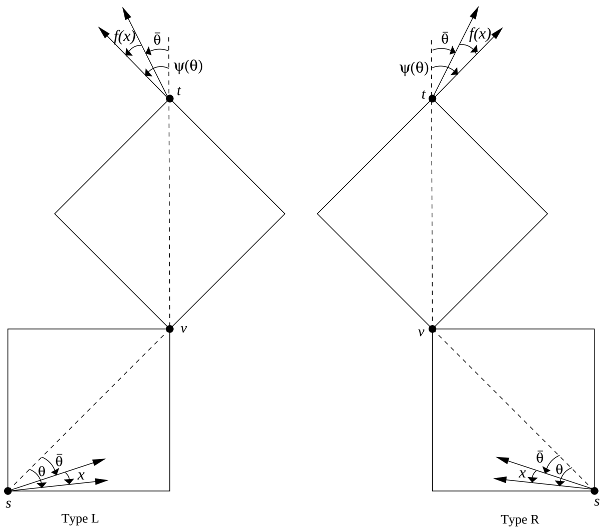

Figure 10: Angles x and $f(x)$ depending on the type of the gadget.

$c_1(\theta - \bar{\theta}) + O((\theta - \bar{\theta})^2) + O(2^{-\beta})$[4] for constants c_0, c_1, where the sign of c_1 depends on whether the basic gadget is type L or type R.

Proof: When $|x|$ is close enough to 0, the output difference angle $f(x)$ can be represented by a converging Taylor series as,

$$f(x) = a_0 + \sum_{j=1}^{\lambda} a_j x^j + O(2^{-\beta}),$$

where $\lambda > 0$ is an integer and that $\sum_{j>\lambda} a_j x^j = O(2^{-\beta})$, for any x in a predetermined range. Since $f(0) = 0$, $a_0 = 0$. Let k be the integer such that $a_j = 0$, for $j = 1, \ldots, k-1$, and $a_k \neq 0$.

Let θ_1 and θ_2 be two different input angles and are represented with $\beta = O(N^2)$ bits of precision. Let δ_i be the output difference angle for a path with input angle θ_i after a gadget sequence of σ basic gadgets, for $i = 1, 2$. It is easy to see that to distinguish δ_1 and δ_2, it requires $\Omega(\beta k^\sigma)$ bits of precision. After a sequence of $O(N)$ gadget, the number of bits required to represent difference angles is $\Omega(\beta k^N)$. If $k > 1$ we are in trouble since we cannot afford such a large number of bits of precision.

[4]Note: Careful, $c_0 + \ldots$ is the output difference angle and not the output angle. SL

Instead of showing that $k = 1$, we do the following trick to deal with the case when $k > 1$. We rotate a basic gadget around the entering point by an angle of ϵ. By doing this, we change an entering angle of x to $x + \epsilon$. Let h be a map that maps an entering angle before the rotation to an exiting angle (after the rotation). Thus,

$$\begin{aligned}
h(x) &= f(x+\epsilon) \\
&= \sum_{j=1}^{\lambda} a_j (x+\epsilon)^j + O(2^{-\beta}) \\
&= \sum_{j=1}^{\lambda} a_j \epsilon^j + \left(\sum_{j=1}^{\lambda} j a_j \epsilon^{j-1} \right) x \\
&\quad + \left(\sum_{j=1}^{\lambda} j(j-1) a_j \epsilon^{j-2}/2 \right) x^2 + \ldots + O(2^{-\beta}) \\
&= c_0(\epsilon) + c_1(\epsilon)x + c_2(\epsilon)x^2 + \ldots + O(2^{-\beta}) \\
&= c_0(\epsilon) + c_1(\epsilon)x + O(x^2) + O(2^{-\beta}),
\end{aligned}$$

where $c_0(\epsilon) = \sum_{j=1}^{\lambda} a_j \epsilon^j$ and $c_1(\epsilon) = \sum_{j=1}^{\lambda} j a_j \epsilon^{j-1}$. We require an $\epsilon > 0$ such that $c_1(\epsilon) \neq 0$ so that the angular changes have nonzero linear terms. Furthermore, this ϵ needs to have at most β bits. To do this, we choose an ϵ randomly, with a uniform random choice, within the range of small values $[0, 2^{-\beta}]$. Since $c_1(\epsilon)$ is a non-zero polynomial in ϵ, the Schwartz-Zippel Lemma [17, 15] insures that with probability 1, the coefficient $c_1(\epsilon)$ is nonzero. Once such an ϵ is chosen, $c_i = c_i(\epsilon)$ are constants, for $i = 0, 1$. This concludes the lemma. ▮

Consider now a $\sigma-$gadget sequence S of basic gadgets $\mathcal{B}_i$ for $i = 1, \ldots, \sigma$, such that $\mathcal{B}_{i+1}$ is a sequel of $\mathcal{B}_i$, for $i = 1, \ldots, \sigma - 1$. We define

$$F(S, x) = F_\sigma(F_{\sigma-1}(\ldots F_1(x)) \ldots), \tag{2}$$

where F_i is equal to f if gadgets $\mathcal{B}_i$ and $\mathcal{B}_{i-1}$ have different types, and is equal to g otherwise, for $i = 2, \ldots, \sigma$. Therefore, if x is the input difference angle of gadget $\mathcal{B}_1$, $F(S, x)$ is the output difference angle of gadget $\mathcal{B}_\sigma$, at the end of a gadget sequence S. **Note: I modified the definition of F (I don't think it was correct). S.L. Note 2: F does not seem to be used. SL.**

Lemma 5.3 *In a $\sigma-$gadget sequence, for any input angle θ sufficiently close to $\bar{\theta}$, the path $\sigma P^{(\theta)}$ exists and reaches the target with output difference angle*

$$h^{(\sigma)}(x) = c_0^{(\sigma)}(\epsilon) + c_1^{(\sigma)}(\epsilon)x + O(x^2) + O(\sigma 2^{-\beta}), \tag{3}$$

where $c_1^{(\sigma)} = c_1(\epsilon)^{\sigma}$, and the coefficient $c_1^{(\sigma)}(\epsilon) \neq 0$. **Note: Are the indices right? SL.**

Proof: The difference angle after a $2-$gadget sequence is

$$\begin{aligned} h^{(2)}(x) &= c_0(\epsilon) + c_1(\epsilon)(c_0(\epsilon) + c_1(\epsilon)x + \epsilon) + O(x^2) + O(2^{-\beta}) \\ &= c_0^{(2)}(\epsilon) + c_1^{(2)}(\epsilon)x + O(x^2) + O(2^{-\beta}), \end{aligned}$$

where $h^{(2)}$ means $h(h(x))$,

$$c_0^{(2)}(\epsilon) = c_0(\epsilon) + c_0(\epsilon)c_1(\epsilon) + \epsilon c_1(\epsilon),$$

and

$$c_1^{(2)}(\epsilon) = (c_1(\epsilon))^2.$$

Thus the coefficient $c_1^{(2)}(\epsilon)$ is nonzero (for the same choice of ϵ as in the proof of the previous lemma). By induction on σ we get the result.

Note that the function $c_0^{(\sigma)}$ is bounded by $O(c_0(\epsilon)^{\sigma})$. Since ϵ is fixed, we only need to provide up to β bits of precision for $c_0^{(\sigma)}$ and $c_1^{(\sigma)}$. ∎

5.2 Path length of gadget sequences

We prove here the following lemma.

Lemma 5.4 *???????????????????? In a gadget sequence of $\sigma = O(N^2)$ basic gadgets, for any input angle θ sufficiently close to $\bar{\theta}$ (i.e., $\theta - \bar{\theta} = O(2^{-\beta/2})$), the length of path $P^{(\theta)}$ is the same as the length of the shortest path $P^{(\bar{\theta})}$, up to the allowed β bits of precision (i.e., $\rho(\theta) - \rho(\bar{\theta}) = O(2^{-\beta})$). Furthermore, the accumulation of path length error over a gadget sequence of $\sigma = O(N^2)$ basic gadgets is no more than $2^{-\beta}$.*

Consider, as in Section 2.2, a canonical path of a gadget sequence. By Lemma 2.2, the canonical path of a gadget sequence S is a shortest path from the start point s of S to the endpoint t of S, among all paths from (s, θ_1) to (t, θ_2), where $\theta_i \in [0, \pi/2]$, and the length of the canonical path is $\sigma\bar{\rho}$, where σ is the number of basic gadgets of S and $\bar{\rho} = \rho(\bar{\theta})$, where $\bar{\theta}$ is as defined in (1).

There exists a small real number $\epsilon_1 > 0$, such that if $-\epsilon_1 < \theta - \bar{\theta} < \epsilon_1$, then $\rho(\theta)$ can be represented by a converging Taylor series,

$$\rho(\theta) = \rho(\bar{\theta}) + \sum_{j=1}^{\mu} d_j(\theta - \bar{\theta})^j + O(2^{-\beta}),$$

where $\mu > 0$ is some integer and that $\sum_{j>\mu} d_j(\theta - \bar{\theta})^j = O(2^{-\beta})$ for all $\theta \in [\bar{\theta} - \epsilon_1, \bar{\theta} + \epsilon_1]$, and d_j are real constants. Since $\bar{\rho} = \rho(\bar{\theta})$ is a minimum, and we have avoided extrema where $\bar{\theta}$ is $-\pi/4$ or $\pi/4$,

$$\frac{d\rho}{d\theta}|_{\theta=\bar{\theta}} = 0.$$

This implies that

Proposition 5.1 $d_1 = 0$.

This proves the part of Lemma 5.1 concerning $\rho(\theta)$.

Then there exists $\epsilon_2 > 0$, such that $\epsilon_2 < \epsilon_1$ and that

$$\sum_{j=2}^{j=\mu} (\epsilon_2)^j < 2^{-\beta}. \tag{4}$$

Hence there exists ϵ_σ, $0 < \epsilon_\sigma < \epsilon_2$, such that the accumulation of path length error (as given in (4)) over a gadget sequence of $\sigma = O(N^2)$ basic gadgets is no more than $2^{-\beta}$. This proves Lemma 2.3. ∎

5.3 Error analysis of a variant gadget

Consider, as in Section 2.3 and Figure 4 a variant gadget $\mathcal{B}(\delta)$ derived from basic gadget $\mathcal{B}$. Recall the angle, oriented clockwise, between segment vt and segment $v't'$ is δ if the variant gadget of type L, and the angle is $-\delta$ if the variant gadget of type R. We assume the angle between segment vt and ℓ is oriented clockwise. We also assume that $\delta = o(2^{-\beta})$.

Let $\rho_\delta(\theta)$ and $\psi_\delta(\theta)$ be the path length and the output angle of $P_\delta^{(\theta)}$, respectively. We can represent by converging Taylor series (extending the above proof for functions $\rho(\theta)$ and $\psi(\theta)$ of the basic gadget) the variant gadget functions $\rho_\delta(\theta)$ and $\psi_\delta(\theta)$, as follows:

There exists a small constant real number $\epsilon_1 > 0$, where for all θ such that $-\epsilon_1 < \theta - \bar{\theta} < \epsilon_1$, we have:

$$\rho_\delta(\bar{\theta}_\delta + \gamma) = \bar{\rho}_\delta + d' \cdot \gamma^2 + O(\gamma^3), \tag{5}$$

and

$$\psi_\delta(\bar{\theta}_\delta + \gamma) = \bar{\theta}_\delta + c' \cdot \gamma + O(\gamma^2), \tag{6}$$

where d', and c' are some constants. Hence we have:

Lemma 5.5 *For any input angle θ on the interval $[-\pi/4, \pi/4]$, such that θ is sufficiently close to $\bar{\theta}_\delta$, and for sufficiently small δ, the path $P_\delta^{(\theta)}$ exists and reaches the target in distance $\rho_\delta(\theta) = \rho_\delta(\bar{\theta}_\delta) + O((\theta - \bar{\theta}_\delta)^2) + O(2^{-\beta})$ and with output angle $\psi_\delta(\theta)$ which is $c'_0 + c'_1(\theta - \bar{\theta}_\delta) + O((\theta - \bar{\theta}_\delta)^2) + O(2^{-\beta})$ for constants c'_0, c'_1 (where the sign of c_1 depends on whether the basic gadget is type L or type R).*

This also implies (as in the proof of Lemma 2.3) Lemma 2.5. ∎

5.4 Error Analysis of the 3-SAT Construction

Finally, to show that the 3-SAT construction works, we have to show the following two things. First, we have to show that the 2^N paths have the same length as the canonical path up to $\beta = cN^2$ number of bits of precision, i.e., the accumulated error in path length over all the gadgets is no more than $2^{-\beta}$. Second, we have to show that the 2^N paths are distinguishable with β bits of precision, i.e., we can construct 2^N paths such that the output angles of these paths are all different with β bits of precision.

Since each path has to pass through $O(N^2)$ gadgets, to show the first claim, it is sufficient to show that for each gadget, the difference in path length is no more than $2^{-\beta}/O(N^2)$. Let $\beta' = -(\beta/2 + \log\beta)$. Let θ be an input angle; we require that $|\theta - \bar{\theta}| < 2^{-\beta'}$. By Lemma 5 and Lemma 12, we have that $\rho(\theta)$ can be approximated as a polynomial in $(\theta - \bar{\theta})$ and its second coefficient (for the linear term) is 0. This implies that $|\rho(\theta) - \rho(\bar{\theta})| = O((\theta - \bar{\theta})^2) = O(2^{-2\beta'}) = O(2^{-\beta - 2\log\beta}) = O(2^{-\beta}/\beta^2) < 2^{-\beta}/O(N^2)$.

On the other hand, we also require that $|\theta - \bar{\theta}| \geq 2^{-3\beta/4}$. The intuition behind this is that we don't want input angles which are too close to $\bar{\theta}$, because the output angles may also become too close to the canonical output angle to be distinguished within β bits of precision. Recall that the difference angle has a linear term $c_1^{(\sigma)}(\theta - \bar{\theta})$ as proved in Lemma 10, where $c_1^{(\sigma)} = c_1^{(\sigma)}(\epsilon)$ depends on a small rotation angle ϵ. We can choose an ϵ such that $c_1^{(\sigma)} \geq 2^{-\beta/4}$. This implies that the difference angles cover a range of $[2^{-\beta}, 2^{-3\beta/4}]$ and $[-2^{-3\beta/4}, -2^{-\beta}]$. This range contains at least $2^{\beta/4}$ distinct values within β bits of precision. Therefore we have enough different output angles to make 2^N paths. This concludes the NP-hardness proof (Theorem 4.1) of the 2D curvature-constrained shortest-path problem.

6 Open Problems and Acknowledgments

It remains an open problem to prove the NP-hardness of the 2D curvature-constrained shortest-path problem in the case where the obstacles are all connected.

We wish to thank Pankaj Agarwal and Zheng Sun for their very helpful comments on this paper and thank Zheng Sun Guangwei Yuan and for their very useful aid in preparation of the figures. We also wish to thank the referees for their useful suggestions in improving the proofs and in particular for Propositions **??** and **??**.

References

[1] P.K. Agarwal, P. Raghavan, and H. Tamaki. Motion planning for a steering-constrained robot through moderate obstacles. In *Proc. 27th Symp. on Theory of Computing*, pages 343–352, 1995.

[2] H.-K. Ahn, O. Cheong, J. Matoušek, and A. Vigneron. Reachability by paths of bounded curvature in convex polygons. In *Proc. of the 32th Annu. ACM Sympos. Theory Comput.*, 2000, pp. 251–259.

[3] J.-D. Boissonnat and X.-N. Bui. Accessibility region for a car that only moves forwards along optimal paths. Research Report INRIA 2181, 1994.

[4] T. Asano, D. Kirkpatrick and C. Yap. $D1$ optimal motion of a rod, In *Proc. Symp. on Computational Geometry*, 1997.

[5] J. Canny and J. Reif. New lower bound techniques for robot motion planning problems. In *Proc. 28th Symp. on Foundations of Computer Science*, pages 49–60, 1987.

[6] J. Canny, A. Rege, and J.H. Reif, An Exact Algorithm for Kinodynamic Planning in the Plane. 6th Annual ACM Symposium on Computational Geometry, Berkeley, CA, June 1990, pp. 271-280. Published in Discrete and Computational Geometry, Vol. 6, 1991, pp. 461-484.

[7] L.E. Dubins. On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents. *Amer. J. Math.*, 79:497–516, 1957.

[8] S. Fortune and G. Wilfong. Planning constrained motion. *Annals of Mathematics and Artificial Intelligence*, 3:21–82, 1991.

[9] P. Jacobs and J. Canny. Planning smooth paths for mobile robots. In Z. Li and J. Canny, editors, *Nonholonomic Motion Planning*, pages 271–342. Kluwer Academic Publishers, 1992.

[10] J.-C. Latombe. Robot Motion Planning. Kluwer Academic Publishers, Boston, 1991.

[11] G. Pestov and V. Ionin. On the largest possible circle imbedded in a given closed curve. *Dok. Akad. Nauk SSSR*, 127 (1959), 1170–1172.

[12] J.H. Reif and S. Tate, Approximate Kinodynamic Planning Using L2-norm Dynamic Bounds. Computers and Mathematics with Applications, Vol. 27, No.5, 1994, pp.29-44.

[13] J. Reif and H. Wang. Non-uniform discretization approximations for Kinodynamic motion planning and its applications. SIAM Journal of Computing (SICOMP), 30(1):161-190, 2000.

[14] H. Wang and P. K. Agarwal. Approximation algorithms for curvature-constrained shortest paths. In *Proceedings of the 7th ACM-SIAM Symposium on Discrete Algorithms*, pages 409–418, 1996.

[15] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities, J. ACM **27**, 701–717, (1980).

[16] H.J. Sussmann. Shortest 3-dimensional paths with a prescribed curvature bound. In *Proc. 34th Conf. on Decision and Control*, pages 3306–3312, 1995.

[17] R. E. Zippel. Probabilistic algorithms for sparse polynomials, in "Proc. EUROSAM '79", Lecture Notes in Computer Science **72** (1991), 216–226.