Path-cost Analysis and Real-Time Path Computation in Weighted Regions

Norman Jaklin Utrecht University*   Mark Tibboel Nspyre†   Roland Geraerts Utrecht University*

Abstract

The *Weighted Region Problem* is defined as the problem of finding a cost-optimal path in a weighted planar polygonal subdivision. Searching for paths on a grid representation of the scene is fast and easy to implement. However, grid representations do not capture the exact geometry of the scene. Hence, grid paths can be inaccurate or might not even exist at all. Methods that work on an exact representation of the scene can approximate an optimal path up to an arbitrarily small ϵ-error. However, these methods are computationally inefficient and thus not well-suited for real-time applications. In this paper, we analyze the quality of optimal paths on a 8-neighbor-grid. We prove that the costs of such a path in a scene with weighted regions can be arbitrarily high in the general case. If all regions are aligned with the grid, we prove that the costs are at most $\left(4+\sqrt{4-2\sqrt{2}}\right)$ times the costs of an optimal path. In addition, we present a new hybrid method called *Vertex-based Pruning* (VBP). VBP computes paths that are ϵ-optimal inside a pruned subset of the scene. Experiments show that VBP paths can be computed at interactive rates, and are thus well-suited as an input for advanced path-following strategies in robotics, crowd simulation or gaming applications.

CR Categories: F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems—Geometrical problems and computations I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems I.2.1 [Artificial Intelligence]: Applications and Expert Systems—Games

Keywords: path planning, weighted region problem, grid representation

*{n.s.jaklin, r.j.geraerts}@uu.nl
†mark.tibboel@nspyre.nl

1 Introduction

The *Weighted Region Problem* (WRP) has been introduced by Mitchell and Papadimitriou [1991] as a generalization of the classical path planning problem. Instead of a polygonal scene with static obstacles and traversable space, the input for the WRP is a polygonal planar subdivision with positive weights for each polygonal region. Each polygonal region is traversable, but yields different traversal costs depending on its weight. The goal is to compute a path between two query points which is optimal with respect to the traversal costs. Optimal paths in weighted regions are proven to consist of straight-line segments, and each straight-line segment runs through only one type of region. The bending points of an optimal path lie on the region boundaries, where the path obeys *Snell's law of refraction*. The costs for a path are defined as the sum of the weighted Euclidean lengths of each straight-line segment.

The WRP has applications in many fields such as video games [van der Sterren 2002; Kamphuis et al. 2005], simulations [Reece et al. 2000], or robotics [Chestnutt et al. 2007]. Moving entities such as virtual humans or robots need to steer through a scene with various region types. However, the WRP is proven to be unsolvable in the *Algebraic Computation Model over the Rational Numbers* (ACM$\mathbb{Q}$) [Carufel et al. 2012]. In other words, a solution to an instance of the WRP cannot be expressed as a closed formula in ACM$\mathbb{Q}$. This justifies the search for approximation algorithms to compute paths that are near-optimal within a certain costs-bound. A widely-used approach is to approximate the exact geometry of the scene with a rectilinear grid. An efficient graph-search algorithm such as the popular A^*-algorithm [Hart et al. 1968] can then be used on this grid. This approach is fast and easy to implement, but a grid does not capture the exact geometry of the scene. On the other hand, existing approximation algorithms [Aleksandrov et al. 1998; Aleksandrov et al. 2005; Sun and Reif 2001] to solve the WRP are computationally expensive and not suited for real-time applications with many virtual agents.

Contributions. We analyze the quality of 8-neighbor-grid paths compared to optimal paths in weighted regions. We show that optimal grid paths can be arbitrarily expensive for fixed grid resolutions. We then show that in scenes where all regions are aligned with the grid, the costs of an optimal grid path are at most $\left(4+\sqrt{4-2\sqrt{2}}\right)$ times the costs of an optimal path. We generalize the topological property of two paths being *homotopic* in the context of the WRP by defining *region-homotopic* paths. In addition, we present a new hybrid method called *Vertex-Based Pruning* (VBP), which combines an efficient A^* search on a grid with existing approximation algorithms on the exact geometry of the scene. The paths computed with VBP are ϵ-optimal inside a pruned subset of the scene. This means that for any given $\epsilon > 0$, the paths are at most $(1+\epsilon)$ times as expensive as an optimal path inside the pruned scene. Furthermore, the paths follow the exact geometry of the scene and thus overcome problems inherent to approximate grid paths. In a comparative study, we show that VBP improves upon the running times of the Steiner point approximation algorithm by Aleksandrov et al. [1998] and thus allows the computation of high-quality paths at interactive rates. We also show empirically that a path computed with VBP and an overall optimal path are region-homotopic in 97.3% of the tested cases. This means that we can guarantee the same ϵ-error bound as the original Steiner point method in most cases.

The rest of this paper is organized as follows: In Section 2, we give an overview of related work on the WRP, existing approximation algorithms, and its application areas. In Section 3, we give preliminary definitions used in our subsequent analysis. In Section 4, we analyze the quality of A^* grid paths and prove the claimed upper bound on the path costs when all regions are aligned with the grid. In Section 5, we present our new VBP method. A comparative study between VBP and the Steiner point method by Aleksandrov et al. [1998] is conducted in Section 6. We conclude and summarize our contributions and discuss future extensions in Section 7.

2 Related Work

Mitchell and Papadimitriou [1991] were the first to study the WRP as a generalization of the classical path planning problem. They presented the first approximation algorithm, which has a running time of $O(n^8 \log \frac{nNW}{w\epsilon})$, where n is the number of vertices, N is the maximum integer coordinate of any vertex of the triangulation, and w and W are the lowest and highest weight of the regions, respectively. Due to the high computational complexity, the algorithm is mainly of theoretical interest. The authors discuss several fundamental properties of the WRP that do not hold in the classical Path Planning Problem. One interesting property is that an optimal path in weighted regions obeys *Snell's law of refraction* at its bending points. In other words, such a path crosses the borders of two adjacent regions in the same way as a ray of light crosses the border of two different materials.

Several approximation algorithms for the WRP were presented in the following years. Mata and Mitchell [1997] created a graph called *Pathnet* to approximate optimal solutions to the WRP. The method makes use of cones around all vertices, which limit the paths that can extend from a vertex. The Pathnet can be constructed in $O(kn^3)$, where k is the number of such cones. Once the Pathnet is constructed, in can find ϵ-approximate paths in $O(n \log n)$ time, where $\epsilon = O(\frac{W/w}{k\theta_{min}})$. W/w is the ratio of the maximum and minimum weight, respectively, and θ_{min} is the minimum internal face angle of the subdivision.

Aleksandrov et al. [1998] presented an ϵ-approximation scheme to solve the WRP up to an arbitrary $\epsilon > 0$. The method uses Steiner points that are added to the edges of all triangles in the scene with a logarithmic distribution. Because this distribution leads to infinitesimally small distances between Steiner points near the vertices, the authors define a *vertex vicinity*, which is a circle around each vertex that is void of Steiner points. Within each triangle, all Steiner points and vertices are connected by additional edges. Path planning queries can then be answered using Dijkstra's algorithm [1959] (improved by using Fibonacci heaps). The running time is $O(mn \log(mn) + nm^2)$, where m is the total number of Steiner points. Aleksandrov et al. [2005] also presented a variant of the Steiner point method, in which the Steiner points are placed on the bisectors of the triangles. This variant has a better running time, but the paper lacks the description of some critical cases required for a thorough practical implementation. In our new hybrid method, we therefore use this original Steiner point method [Aleksandrov et al. 1998] on a pruned graph; see Section 5.

Sun and Reif [2001] presented an ϵ-approximation algorithm called *BUSHWHACK*. Similar to Aleksandrov et al. [1998], they use Steiner points on the edges of the triangulation. Instead of searching the graph with Dijkstra's algorithm [1959], they introduced a new graph-search method that exploits the underlying geometrical properties of the scene. Using an *interval* data structure, the BUSHWHACK algorithm can find ϵ-optimal paths in $O(\frac{n}{\epsilon}(\log \frac{1}{\epsilon} + \log n) \log \frac{1}{\epsilon})$ time.

Carufel et al. [2012] presented a fundamental theoretical result on the WRP. The authors proved that the WRP is unsolvable in the *Algebraic Computation Model over the Rational Numbers* (ACM$\mathbb{Q}$). This means that a solution to an instance of the WRP cannot be expressed as a closed formula in ACM$\mathbb{Q}$. This result further justifies the search for ϵ-approximate solutions to the problem.

Ferguson and Stentz [2006] presented *Field D^**, an interpolation-based planning and replanning method well-suited for scenarios with weighted regions. Field D^* computes paths that may follow any direction and are thus not restricted to angles of $\frac{\pi}{2}$ at its bending points. The input for Field D^* is a grid representation of the weighted scene, in which all regions are aligned with the grid. By contrast, our new VBP method can handle an arbitrary scene given as a planar polygonal subdivision with weighted regions.

Research has also been conducted on grid path analysis for the classical path planning problem without weights on both $2D$ and $3D$ grids. Nash [2012] presented in his PhD thesis a unified proof structure to derive upper bounds on the length of grid paths. In addition to square grids, the author considers triangular and hexagonal grids for non-weighted scenarios in $2D$ and cubic grids in $3D$. Nash et al. [2010] also presented an upper bound on the length of paths in a 26-neighbor $3D$ grid without weights.

Approximation algorithms for variants of the WRP have also been discussed. Aleksandrov et al. [2010] introduced a data structure called *All Points Query* (APQ), which can be used to efficiently find ϵ-optimal paths for all-pairs queries on an instance of the WRP. APQ has a high construction time, and it is therefore mainly useful for answering many queries on the same scene. Cheng et al. [2010] presented a method to compute *homotopic* paths that are ϵ-optimal in an instance of the WRP. Gheibi et al. [2013] recently considered a variant of the WRP with weighted arrangements of lines instead of bounded triangulated subdivisions. Jaklin et al. [2013] presented a path following method called *MIRAN*. The input for the MIRAN method is a path through weighted regions, which is then smoothed and traversed by a virtual agent based on its region preferences. This justifies the search for solutions to the WRP that can be performed at interactive rates, which we address in this paper.

3 Preliminaries

In this section, we give definitions and general assumptions we will be using throughout this work. We start by introducing *region-homotopic* paths. Afterwards, we discuss grids and *grid-optimal* paths.

One important topological property in classical path planning without weighted regions are *homotopic* paths. Two paths with the same fixed start and goal positions are homotopic, if there exists a homotopic function that continuously maps one path into the other without having to cut open the path or intersect obstacle polygons in the scene. The question whether two paths are homotopic can be important when analyzing error bounds for approximations on optimal paths. To use this property in our analysis in Section 4, we therefore generalize it to weighted regions in the following way:

Given the scene as a polygonal subdivision of the plane, let $\mathcal{R} = \{(r_1, w_1), (r_2, w_2), ..., (r_n, w_n)\}$ be the set of its n region polygons r_i together with a weight $w_i > 0$. Without loss of generality, we assume that any two adjacent region polygons have different weights. In other words, if two adjacent regions had the same weight in the given representation of the scene, we assume them to be merged into one region polygon. Let π_1, π_2 be two paths connecting the same start and goal positions s and g in $\mathcal{R}$, respectively. In order to generalize the definition of homotopic paths, we map the given weighted scenario to an instance of the classical Path Planning Problem by declaring all regions that either of the two paths intersect as free space. All remaining regions are declared as hard obstacles with an infinite weight. More formally, we let $\mathcal{R}(\pi_1, \pi_2) = \{(r, w) \in \mathcal{R} \mid \pi_1 \text{ or } \pi_2 \text{ intersects the interior of } r\}$. We then define $\mathcal{R}'(\pi_1, \pi_2) = \{(r_1, w'_1), (r_2, w'_2), ..., (r_n, w'_n)\}$ as the same set of region polygons r_i as in $\mathcal{R}$, but with the following weights:

$$w'_i = \begin{cases} 1 & \text{if } (r_i, w_i) \in \mathcal{R}(\pi_1, \pi_2) \\ \infty & \text{else .} \end{cases}$$

Definition 1. *We say that π_1 and π_2 are* region-homotopic *in $\mathcal{R}$, if and only if they are homotopic in $\mathcal{R}'(\pi_1, \pi_2)$.*

We consider a grid with square grid cells to approximate the scene. Each cell in the grid is weighted with the highest weight of all regions that intersect the cell. From a graph-search point of view, we use the center points of each grid cell to represent the cell as a vertex in the graph.

We focus on 8-*neighbor grids* that allow movement on the grid in up to 8 directions. Thus, any grid path consists of horizontal, vertical and diagonal straight-line segments. If such a segment runs along the edge shared by two grid cells, the smaller weight of the two cells counts for this segment. This choice reflects paths that are tangent to a high-cost region without intersecting the interior of that region. Furthermore, we only consider grid cell center points as input for path planning queries.

By $\mathcal{C}(\cdot)$, we denote the function to measure the costs of a path π. If π consists of k straight-line segments π_i, and each π_i runs through one region with weight w_i $(1 \leq i \leq k)$, we let

$$\mathcal{C}(\pi) = \sum_{i=1}^{k} w_i ||\pi_i||,$$

with $||\cdot||$ being the Euclidean norm. Since an optimal (non-grid) path in weighted regions is proven to consist of straight-line segments [Mitchell and Papadimitriou 1991], we can use this cost function for both grid paths and optimal paths.

Let $\gamma \in \mathbb{R}^+$ be the size of each grid cell. Let C_1 and C_2 be two adjacent grid cells with weights w_1 and w_2, respectively. Given the above cost function, we can conclude the following. If C_1 and C_2 are horizontally or vertically connected by a straight-line segment l, the costs for moving from one cell to the other are $\mathcal{C}(l) = \frac{1}{2}\gamma w_1 + \frac{1}{2}\gamma w_2 = \frac{1}{2}\gamma(w_1 + w_2)$. If C_1 and C_2 are diagonally connected, the costs are $\mathcal{C}(l) = \frac{1}{2}\gamma\sqrt{2}(w_1 + w_2)$.

In addition to the above, we use the following definition to distinguish between optimal paths on the grid and optimal paths on the exact geometry of the scene.

Definition 2. *We call a grid path* grid-optimal, *if it is optimal among all other possible grid-paths with respect to the cost function $\mathcal{C}(\cdot)$.*

4 Quality of Grid-Optimal Paths

In this section, we analyze the quality of grid-optimal paths with respect to optimal paths in weighted regions. First, we show that a grid-optimal path can be arbitrarily expensive, if a fixed cell size is given. Afterwards, we focus on scenarios in which all regions are aligned with a fixed grid. We show that even in this simplified scenario, a grid-optimal path and an optimal path are not necessarily region-homotopic. We can, however, prove an upper bound on the costs of grid-optimal paths when all regions are aligned with the grid, which is the main result in this section.

Let us assume a grid with a fixed resolution. If an optimal path and a grid path are not region-homotopic, the grid path can be arbitrarily expensive. This already holds in the classical path planning situation in which no weighted regions are given, whenever the paths are not homotopic. Figure 1 shows an example in which a grid-optimal path Γ and an optimal path π^* that connect points s and g are not (region-)homotopic. This is due to using an overly coarse grid resolution. By increasing the height of the obstacle polygon P, we can make Γ become arbitrarily expensive. This also shows that a grid path does not necessarily need to exist at all in the general case.

The situation is different when all regions are aligned with the grid. Given a particular scene with weighted regions, we can always adjust the grid resolution to achieve this. We might require a very small grid resolution if the scene features complex shapes with many vertices, and this yields high running times for search algorithms on the grid. By contrast, in applications with simple rectangular shapes or when the exact geometric shape of a region is less important (e.g. in grid-based games, or when a region resembles an abstract feature such as *dangerous* or *attractive*), aligning all regions with the grid is a feasible way to reduce the complexity of the scene. We now discuss important properties and analyze the

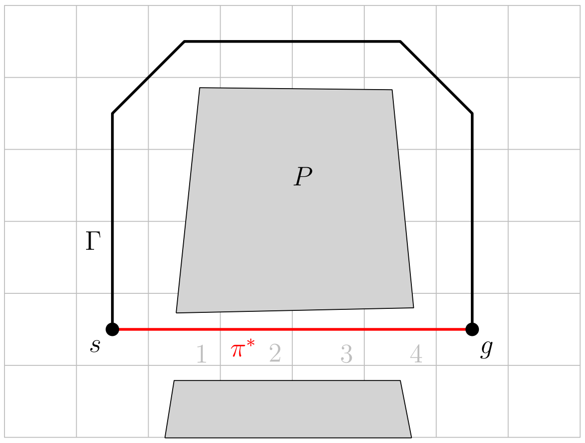

Figure 1: *Example of a two paths connecting points s and g. The grid-optimal path Γ can become arbitrarily expensive compared to the costs of an optimal path π^*, if the grid resolution is too coarse. Grid cells $1, 2, 3$ and 4 are not traversable for Γ because obstacle polygon P intersects them.*

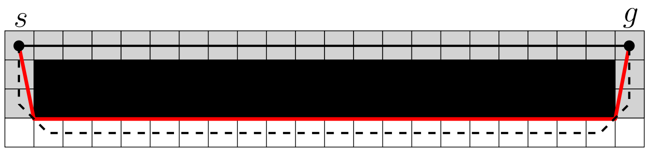

Figure 2: *An example in which the grid-optimal path Γ (solid black) is not region-homotopic to the optimal path π^* (solid red). The grid path Γ' that is region-homotopic to π^* (dashed black) has slightly higher costs than Γ. The grid cell size is $\gamma = 1$. The white region has a low weight of 1, the gray region has a weight of 1.3, and the black region has a very high (infinite) weight. The path costs are: $\mathcal{C}(\pi^*) \approx 26.63, \mathcal{C}(\Gamma) \approx 27.3, \mathcal{C}(\Gamma') \approx 27.45$.*

quality of grid-optimal paths in this special case of the Weighted Region Problem.

The example in Figure 1 might give rise to the assumption that a grid-optimal path and an optimal path are always region-homotopic if we ensure that all regions are aligned with the grid. However, this is not the case. Aligning all regions with the grid does ensure that there exists a grid path in the same homotopy class in which an optimal path is contained, but it does not need to be grid-optimal. Figure 2 shows an example in which the grid-optimal path Γ (solid black) and the optimal path π^* (solid red) are not region-homotopic, even when all regions are aligned with the grid. The grid path following the same regions as π^* is denoted as Γ' (dashed black), and its costs are approximately 27.45. The costs for Γ, in comparison, are approximately 27.3.

In the remainder of this section, we prove an upper bound on the costs of grid-optimal paths with respect to the costs of an optimal path when all regions are aligned with the grid. To this end, we first discuss a simpler scenario in the following Lemma 1. We will then use the result of this lemma in the subsequent proof of Theorem 1. Note that the result of Lemma 1 was independently proven for obstacle-free scenarios [Ferguson and Stentz 2006], and also for different types of grids and the general case in which grid cells can be blocked [Nash 2012]. We give an alternative proof in which we consider arbitrary weights and grid cell sizes, and we use the notation and definitions that better fit our discussion in Theorem 1.

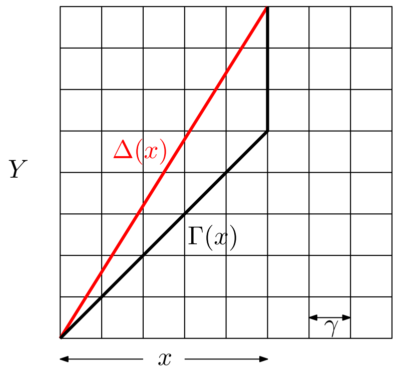

Figure 3: *The situation discussed in Lemma 1.*

Lemma 1. *Let R be an axis-aligned rectangle, rasterized with a grid of cell size $\gamma \in \mathbb{R}^+$. Let each grid cell be weighted with the same weight $w > 0$, and let Γ be a grid-optimal path within R that runs from one corner of R to the diagonally opposite corner of R. Let Δ be the diagonal of R. Then it holds that $\mathcal{C}(\Gamma) \leq (1+\epsilon)\,\mathcal{C}(\Delta)$, with $\epsilon = \sqrt{4 - 2\sqrt{2}} - 1 \approx 0.08$, independent of γ and w.*

Proof. Let X, Y be the number of grid cells of R in x- and y-dimension, respectively. To prove the claimed ϵ-bound for R, we do the following: Instead of R we consider the square with $\max(X, Y)$ cells in both dimensions. We then keep one dimension, say y, fixed and consider x as a variable to find the x-value between 0 and $\max(X, Y)$ that maximizes the ϵ-error between the path costs. The grid path from $(0, 0)$ to (x, Y) is denoted as $\Gamma(x)$, and the line segment between $(0, 0)$ and (x, Y) is denoted as $\Delta(x)$. The situation is shown in Figure 3 for $\max(X, Y) = Y$.

We can now consider ϵ as a function in x:

$$\epsilon(x) = \frac{\mathcal{C}(\Gamma(x)) - \mathcal{C}(\Delta(x))}{\mathcal{C}(\Delta(x))}.$$

The costs for the line segment $\Delta(x)$ are $w(\sqrt{(\gamma x)^2 + (\gamma Y)^2})$, and the costs for $\Gamma(x)$ are the costs for the diagonal steps $w(\sqrt{2}\gamma x)$ plus the costs for the vertical steps $w(\gamma(Y - x))$. Thus,

$$\epsilon(x) = \frac{w\gamma\left(\sqrt{2}x + (Y - x) - \sqrt{x^2 + Y^2}\right)}{w\gamma\sqrt{x^2 + Y^2}}.$$

It is easy to see that γ and w can be canceled from the fraction. We can now find the maximum ϵ-error by computing the root x_0 of the first derivative $\epsilon'(x)$ and evaluating $\epsilon(x_0)$. The first derivative is $\epsilon'(x) = Y(\sqrt{2}Y - Y - x)(Y^2 + x^2)^{-\frac{3}{2}}$. It holds that $\epsilon'(x) = 0 \Leftrightarrow x = \sqrt{2}Y - Y$, and $\epsilon(\sqrt{2}Y - Y) = \sqrt{4 - 2\sqrt{2}} - 1 \approx 0.08$. $\square$

Theorem 1. *Let all regions be aligned with the grid. Let Γ be a grid-optimal path, and let π^* be an optimal path. It holds that $\mathcal{C}(\Gamma) \leq \left(4 + \sqrt{4 - 2\sqrt{2}}\right)\mathcal{C}(\pi^*)$.*

Proof. We first give a brief outline of the proof. The main idea is to exploit the fact that an optimal path π^* consists of straight-line segments, with each segment running through only one type of region. We split up π^* at its bending points, inducing a sequence of rectangles of grid cells. Each rectangle in this sequence contains

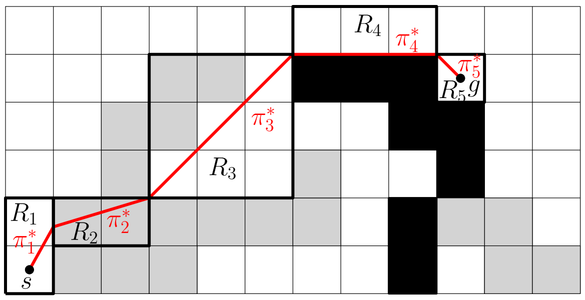

Figure 4: *An example scene with three different types of regions: white (low weight), gray (medium weight) and black (high weight). The shown optimal path between s and g consists of 5 straight-line segments π_i^*. Each segment induces a rectangular set of grid cells R_i.*

one of the segments of π^*. We then construct grid-optimal paths inside each rectangle. For each grid-optimal path, we show that its costs are at most $\left(\sqrt{4-2\sqrt{2}}\right)+2\gamma$ times the costs of the segment of the optimal path in the same rectangle. By exploiting the fact that the costs for any two consecutive line segments of π^* are at least γ, we can deduce the claimed upper bound on the overall path costs.

One property we are going to use in the proof requires the costs of a path being at least as high as its Euclidean length. This property only holds if all weights are not smaller than 1. The Weighted Region Problem is defined for arbitrary positive weights and thus allows weights smaller than 1. However, we can assume all weights being greater or equal to 1 without loss of generality in the following way: Let w_{min} be the smallest weight. If $w_{min} \geq 1$, we do not have to do anything. If $0 < w_{min} < 1$, we multiply all weights by $\frac{1}{w_{min}}$. This ensures that all weights are not smaller than 1. In addition, the relation between any two weights stays the same under this scaling operation. This ensures that an optimal path in the scene with scaled weights consists of the the same straight-line segments as the corresponding optimal path in the non-scaled scene. We can therefore assume $w \geq 1$ for all weights w throughout the remainder of this proof.

We start the proof by splitting up π^* into k segments π_i^*, $1 \leq i \leq k$. The locations at which we split up π^* are its bending points. Thus, each segment π_i^* is a straight-line segment running through only one type of region [Mitchell and Papadimitriou 1991]. For each segment π_i^*, we consider the rectangular part R_i of the grid in which the segment is contained. To be precise, we let A_i and B_i be the grid cells in which the endpoints of π_i^* lie. Except for possibly π_1^* and π_k^*, these endpoints lie on cell edges or cell corners of the grid. Thus, they lie on more than one grid cell, which makes the choice ambiguous. Among the candidate grid cells that contain the endpoints, we pick the two cells A_i and B_i that span the smallest rectangle of grid cells. We then define R_i as the rectangular set of grid cells that is spanned by A_i and B_i. Figure 4 shows an example scene with an optimal path between s and g, and the corresponding segments π_i^* and rectangles R_i. The grid cells in R_i do not need to be all of the same region type (see e.g. R_3 in Figure 4). However, each π_i^* passes through a (horizontally, vertically or diagonally connected) sequence of grid cells in R_i that are all of the same region type. The corner grid cells A_i and B_i are the first and last grid cells of this sequence in R_i.

We would now like to assess the path costs by comparing the parts of Γ that run through each rectangle R_i with each segment π_i^*. However, we cannot guarantee that Γ intersects each rectangle R_i in which the segment π_i^* is contained. This only holds trivially for

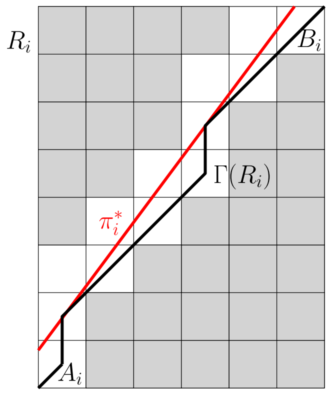

Figure 5: *Rectangle R_i with corner grid cells A_i and B_i. The path $\Gamma(R_i)$ is grid-optimal, and it connects the opposite corners of R_i that belong to A_i and B_i. Since π_i^* induces a connected sequence of cells of the same weight (shown in white), we can always define $\Gamma(R_i)$ to follow the same region type.*

R_1 and R_k because both Γ and π^* connect the same points s and g. All other rectangles might not contain parts of Γ because Γ and π^* need not to be region-homotopic, as shown before in Figure 2.

Since we have no guarantee that a grid-optimal path and an optimal path are region-homotopic, we have to compare each segment π_i^* with a different grid path. To this end, we define k grid paths $\Gamma(R_i)$, each of which is fully contained in R_i. We define $\Gamma(R_i)$ so that it connects the two opposite corners of R_i that belong to the grid cells A_i and B_i. Since π_i^* induces a connected sequence of grid cells with weight w_i, we can always define $\Gamma(R_i)$ such that it runs through the same type of region as π_i^* does. This step is similar to the approximation of a straight-line segment in a pixel raster using Bresenham's line algorithm [Bresenham 1965]; see Figure 5.

Each $\Gamma(R_i)$ connects the corner points of rectangle R_i, and the sequence of all rectangles R_i connects the cells that contain s and g. We can conclude that the sum of the costs of all $\Gamma(R_i)$ covers the costs of the grid-optimal path Γ, i.e. $\mathcal{C}(\Gamma) \leq \sum_{i=1}^{k} \mathcal{C}(\Gamma(R_i))$.

We now prove that the upper bound we are aiming for holds for the union of all grid paths $\Gamma(R_i)$. In other words, we show that $\sum_{i=1}^{k} \mathcal{C}(\Gamma(R_i)) \leq \left(4+\sqrt{4-2\sqrt{2}}\right) \mathcal{C}(\pi_i^*)$ to conclude the proof. To this end, we discuss two different cases, depending on whether k is an odd or an even number.

Case 1: k is even. We define for each $1 \leq j \leq \frac{k}{2}$: $c_j := \mathcal{C}(\pi_{2j-1}^*) + \mathcal{C}(\pi_{2j}^*)$. The term c_j describes the combined costs of every two consecutive segments of π^*. It holds that these combined costs are each at least as large as the grid cell size, i.e. $c_j \geq \gamma$. To see this, note that every two consecutive straight-line segments of π^* are at least as long as the side of one grid cell; see Figure 6. Since we assume that all weights are not smaller than 1, we can conclude that $c_j = \mathcal{C}(\pi_{2j-1}^*) + \mathcal{C}(\pi_{2j}^*) \geq ||\pi_{2j-1}^*|| + ||\pi_{2j}^*|| \geq \gamma$. Given that, we can deduce the following useful property:

$$\sum_{j=1}^{\frac{k}{2}} c_j \geq \sum_{j=1}^{\frac{k}{2}} \gamma = \frac{k}{2}\gamma \Leftrightarrow 2\sum_{j=1}^{\frac{k}{2}} c_j \geq k\gamma \qquad (1)$$

We can now use Lemma 1 to prove an upper bound on the path costs for each grid-optimal path $\Gamma(R_i)$ in the following way: We have to compare the potentially cheapest segment π_i^* with the grid-path $\Gamma(R_i)$. Figure 7 (top) shows an example of this situation. Since π_i^* starts on one side of a corner cell of the rectangle R_i and ends on one side of the opposite corner cell, the cheapest possible connection inside R_i is between the points p and q as shown in the figure. Note that the situation is symmetrically flipped if the height of R_i is larger than its width, or if π_i^* starts in a different corner of R_i. In Figure 7 (bottom), we show how π_i^* can be shifted to start in the bottom left corner of R_i, and that we can construct $\Gamma(R_i)$ in a way that it connects p and q, and then runs from q to the top right corner. This allows us to use Lemma 1 for the part of $\Gamma(R_i)$ that connects p and q, and we can conclude the following:

$$\sum_{i=1}^{k} \mathcal{C}(\Gamma(R_i)) \leq \sum_{i=1}^{k} \left(\left(\sqrt{4 - 2\sqrt{2}} \right) \pi_i^* + 2\gamma \right) \tag{2}$$

The summand 2γ in Equation 2 refers to the fact that $\Gamma(R_i)$ is by 2γ longer than the connection between p and q; see Figure 7 (bottom). Note that the shifted paths might intersect region cells with different weights. This is not a problem because we only use the shifting as an illustration to compare the costs for the initial paths with the initially underlying weight.

We can now use Equations 1 and 2 to conclude the proof for case 1:

$$\begin{aligned}
\sum_{i=1}^{k} \mathcal{C}(\Gamma(R_i)) &\overset{(2)}{\leq} \sum_{i=1}^{k} \left(\left(\sqrt{4 - 2\sqrt{2}} \right) \pi_i^* + 2\gamma \right) \\
&= k2\gamma + \left(\sqrt{4 - 2\sqrt{2}} \right) \sum_{i=1}^{k} \pi_i^* \\
&\overset{(1)}{\leq} 2 \cdot 2 \sum_{i=1}^{\frac{k}{2}} c_j + \left(\sqrt{4 - 2\sqrt{2}} \right) \sum_{i=1}^{\frac{k}{2}} c_j \\
&= \left(4 + \sqrt{4 - 2\sqrt{2}} \right) \sum_{i=1}^{\frac{k}{2}} c_j \\
&= \left(4 + \sqrt{4 - 2\sqrt{2}} \right) \mathcal{C}(\pi^*).
\end{aligned}$$

Case 2: k is odd. In this case, we can deduce the same upper bound as in case 1 for the first $k-1$ segments of π^*. For the last remaining segment π_k^*, we use the fact that this segment ends in the cell center point g. It therefore holds that $\mathcal{C}(\pi_k^*) \geq 0.5\gamma$, whereas the grid path $\Gamma(R_k) \leq 0.5\sqrt{2}\gamma$. We can further conclude that $\frac{\mathcal{C}(\Gamma(R_k))}{\mathcal{C}(\pi_k^*)} \leq \frac{0.5\sqrt{2}\gamma}{0.5\gamma} = \sqrt{2}$. Since $\sqrt{2} < \left(4 + \sqrt{4 - 2\sqrt{2}}\right)$, we can deduce the same upper bound as in case 1 for the last segment, and thus for the whole path. This concludes the proof of the theorem. □

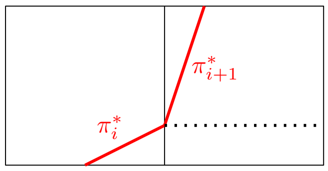

Figure 6: *The sum of any two consecutive segments of the optimal path are at most as long as the side of one grid cell. If the opposite was true, the segment π_{i+1}^* would have to run below the dotted line, which is impossible due to* Snell's law of refraction.

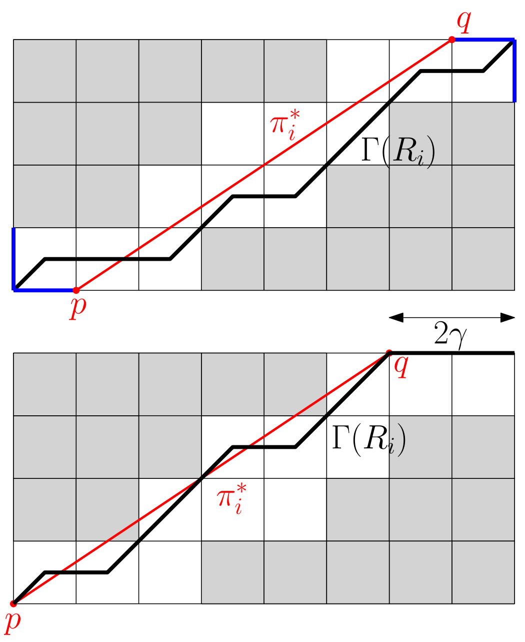

Figure 7: *An example of the cheapest possible path segment π_i^* in rectangle R_i.* Top: *The possible start and endpoints of π_i^* are shown in blue in the bottom left and upper right corner cells. The cheapest connection in this case is the straight-line between p and q.* Bottom: *An illustration of how to compare the path costs. π_i^* can be shifted to the bottom left corner, and a corresponding grid-optimal path $\Gamma(R_i)$ can be used to compare the costs using Lemma 1.*

Algorithm 1 **VERTEX-BASED PRUNING (VBP)**

Input. weighted polygonal environment with edges E, vertices V, weights W; start position s and goal position g; error bound $\epsilon > 0$.
Output. a path π between s and g that is ϵ-optimal in the same homotopy class as a grid-optimal path between s and g.

```
1: $\Gamma \leftarrow$ run $A^*$ on a grid with weights $W$ and an admissible
   heuristic.
2: $E' \leftarrow$ PRUNEGRAPH($\Gamma, E, V$)
3: $\pi \leftarrow$ run the Steiner Point method on $E'$ with error bound $\epsilon$.
4: return $\pi$
```

5 A Hybrid Method: Vertex-based Pruning

In this section, we present a novel hybrid method called *Vertex-based Pruning* (VBP) to efficiently compute a path that respects the exact geometry of a scene with weighted regions. The idea is to combine an efficient A^*-search [Hart et al. 1968] on a coarse grid representation of the scene with the ϵ-optimal *Steiner point method* by Aleksandrov et al. [1998].

The VBP method is described as pseudocode in Algorithm 1. To guarantee grid-optimality for the grid path Γ in the first step of the algorithm, we need to use an admissible heuristic. A heuristic is admissible, if it never overestimates the actual costs from a node to the goal. If all weights are greater than 1, we can use the Euclidean distance to the goal as an admissible heuristic. If the given instance

Algorithm 2 PRUNEGRAPH

Input. grid path Γ; environment with edges E and vertices V.
Output. pruned environment as a set of edges E'.

```
1: initialize $E'$ as an empty set
2: for all bending points $b$ of $\pi$ do
3:     Find the set $closestV$ of closest vertices from $b$ in $V$
4:     for all vertices $v$ in $closestV$ do
5:         for all edges $e$ incident to $v$ do
6:             Add $e$ to $E'$
7: for all segments $s$ of $\pi$ do
8:     for all edges $e$ in $E$ do
9:         if $s$ intersects $e$ then
10:            Add $e$ to $E'$
11: return $E'$
```

of the WRP features weights smaller than 1, we can multiply the Euclidean distance to the goal with the minimum weight w_{min}.

The actual pruning step is described in Algorithm 2. Given a grid path π as the result of an initial A^* search, we iterate over all bending points of π. For each bending point b, we compute the triangle vertices of the environment that are closest to b. We then take all triangle edges incident to these vertices. Note that this set of edges does not yet need to form a connected graph. To address this, we also add all edges that π intersects, if they have not already been added during the first step of the algorithm. This ensures that the resulting subgraph is connected.

After the pruning step, we run the Steiner point method on the pruned graph. The Steiner point method is proven to compute ϵ-optimal paths for any given $\epsilon > 0$ [Aleksandrov et al. 1998]. Thus, the VBP method will always compute a path π that is guaranteed to be ϵ-optimal in the homotopy class of the grid-path Γ. As discussed in Section 4, an overall optimal path and a grid-optimal path do not always need to be region-homotopic. However, in the experiments conducted in Section 6, we empirically determined that the number of cases is small in which the two paths are not region-homotopic.

6 Experiments

In this section, we discuss the experiments we have conducted to measure the performance of the new VBP method as described in the previous section. First, we introduce the scenarios we have tested in Section 6.1. Afterwards, we compare the performance of the VBP method against the original Steiner point method [Aleksandrov et al. 1998] in Section 6.2. In addition, we measure empirically how often the VBP method succeeds to compute the same ϵ-optimal path as the original Steiner point method.

6.1 The Tested Scenarios

We have tested the VBP method on five different scenes. All scenes are displayed in Figure 8. The size of the first four scenes is 100×50 units. The size of the fifth scene is 410×290 units.

The first scene is called *Puddle*. It resembles a puddle in the center with weight 20, surrounded by a forest with weight 5. Below the puddle runs a road with weight 1, and below the road is a grass lawn with weight 3. This scene is small and simple, which makes it easy to visually check the computed paths. In addition, it resembles a typical scenario that could occur in a gaming or simulation application.

The second scene is called *Bars*. It features several vertical bars with alternating weights of 1 and 5. The triangles span the whole height of the scene, which makes it an interesting test case for the VBP method: We expect the VBP method to result in a graph that is close to the Steiner graph of the whole scene.

The third scene is called *High-low*. It features a high-cost region with weight 20 at the bottom, a medium-weight region in the center with weight 3, and a low-cost region at the top with weight 1. The scene is well-suited to display a property of the Weighted Region Problem that does not occur in the classical Path Planning Problem: An optimal path can cross the same triangle multiple times. If s and g both lie in the high-cost region, an optimal path might use parts of the low-cost region at the top.

The fourth scene is called *Zigzag*. It resembles a sandy road with weight 6, a grass field above it with weight 3, and alternating parts of road with weight 1 and water with weight 20. This is an interesting test case: An optimal path from left to right should follow the sandy road, but alternate between the top and bottom border of this region, resulting in a zig-zag path.

The fifth scene is called *Forest*. It resembles a path with weight 3 that runs through a deep forest with weight 99. There are several puddles on the way with weight 20. At the left side of the scene, there is an attractive spot such as a panoramic view over a valley with weight 1. The two parallel rectangular regions on the top resemble fallen tree logs that block the path, but can be traversed (by climbing or ducking) with weight 2. The scene is larger than the other four, so that we can test the effects of our pruning method compared to the original Steiner point method when the number of triangles is high. Furthermore, it resembles a real-life scenario that could occur in a gaming or simulation application.

6.2 Performance of the VBP method

We have tested the Steiner point method by Aleksandrov et al. [1998] against the VPB method in all five scenes. All experiments have been conducted on an *AMD Phenom II*$^{\text{TM}}x4$ 3.4Ghz processor with 4 GB RAM. All measurements are averaged over 50 deterministic runs each.

Table 1 shows the results of the comparison of the two methods. For each scene, we have used 5 different ϵ-error bounds ranging between 0.1 and 0.5, and we have measured the construction times of the graphs, the times needed to answer path planning queries on the graph, the number of nodes explored during the search, and the overall costs of the resulting paths. The particular query points and the resulting paths for each scene are displayed in Figure 8.

The following conclusions can be drawn from the experiments. Compared to the original Steiner point method, the VBP method needs additional time to perform the initial A^* search on the whole scene. Depending on how much of the scene can be pruned, the VBP method may then save time in constructing the Steiner points and additional edges. Once the pruned graph is constructed, VBP improves on the query times in all tested scenes for all ϵ-error bounds. This property is key to achieve real-time performance, and it is reflected in the results of our experiments.

As expected, VBP does not outperform the Steiner point method in the *Bar* scene. The scene contains triangles that span the whole height of the scene, and paths are planned from the left side to the right side. The pruned graph is therefore almost as big as the graph of the whole scene. In this case, the additional time to perform the A^* search dominates the time that can be saved due to the pruning step of the algorithm. The larger the ϵ-error bound the fewer overall Steiner points are required. With fewer required Steiner points, the difference in the number of Steiner points on the whole graph and the pruned graph is small. This yields higher construction times

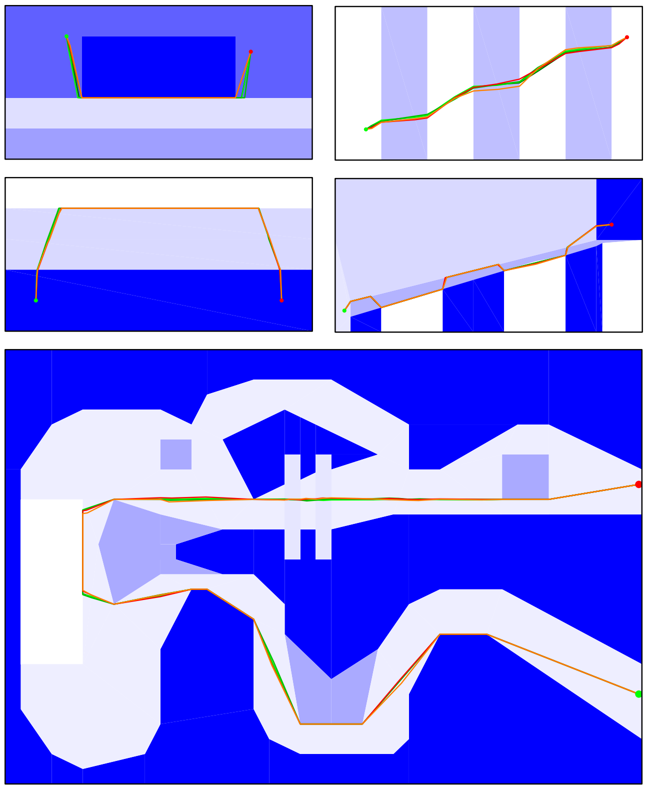

Figure 8: *The five scenes tested in our experiments. From top to bottom and left to right:* Puddle, Bars, High-low, Zigzag, *and* Forest. *The higher the weight for a region the darker its shade of blue. We also show the paths computed with the Steiner point method by Aleksandrov et al. [1998] and the VBP method for different ϵ-error bounds: $\epsilon = 0.1$ (green), $\epsilon = 0.2$ (lightgreen), $\epsilon = 0.3$ (darkgreen), $\epsilon = 0.4$ (red), and $\epsilon = 0.5$ (orange). Note that both methods computed the same paths in all shown cases.*

for VBP and ϵ ranging from 0.3 to 0.5. Thus, the corresponding improvement in query times is also comparably small.

VBP performs slightly better in the *High-low* scene. It improves the query times and number of explored nodes more than in the *Bar* scene. However, due to the geometry of the scene, the construction time is still slightly higher compared to the original Steiner point method for $\epsilon = 0.5$.

In the *Zigzag*, *Puddle* and particularly the *Forest* scene, the strength of VBP becomes apparent. These scenes are more complex and feature scenes that are more likely to occur in gaming or simulation applications. Here, the difference between the pruned graph and the initial graph is big. The time that can be saved to explore parts of the graph that are not relevant strongly dominates the additional time required for the initial A^* search. This yields an overall improvement in both construction times, query times, and the number of nodes explored during the search.

Furthermore, all paths computed with VBP are equal to the paths computed with the Steiner point method. This is, however, not necessarily the case in general because the initial grid-path and an optimal path might not be region-homotopic as discussed in Section 4. We therefore conducted a second type of experiment, in which we ran both methods again for a large number of path queries on each scene. We measured the overall number of paths that yielded different costs for the VBP and Steiner point methods. The goal was

Table 2: *Empirical results of how often the VBP method computes the same path as the original Steiner point method by Aleksandrov et al. [1998].*

Scene	Number of paths	Number of path differences	Success rate
Puddle	6561	23	99.6%
Bars	6561	0	100%
High-low	6561	132	97.9%
Zigzag	6561	201	96.9%
Forest	9000	587	93.4%
All averaged	35244	943	**97.3%**

to empirically determine in how many cases the paths computed by both methods are not region-homotopic.

We uniformly sampled each scene to generate different start and goal positions for each query. For the first four scenes, we picked 9 different x- and y-coordinates, yielding 81 start and 81 goal positions. For each combination of start and goal positions, we computed a path with both methods, yielding a total of 6561 different paths for each of the four scenes. For the larger *Forest* scene, we generated a total of 9000 different paths with both methods by uniformly sampling start and goal positions.

Table 2 shows the results of this experiment. On average, both methods computed the same paths in 97.3% of all cases. This matches our observations in Section 4 and gives further justification to believe that grid-optimal paths and optimal paths are not region-homotopic in only a limited number of scenarios, when the weights and path costs are very close to each other. We can conclude that the same ϵ-error bound as for the Steiner point method applies to VBP paths in most cases. In the few remaining cases, VBP paths are still ϵ-optimal with respect to an optimal path in the pruned scene.

7 Conclusion

In this paper, we have studied the quality of 8-neighbor grid paths in the context of the Weighted Region Problem (WRP). We have proven a first upper bound on the path costs of grid-optimal paths when all regions are aligned with the grid. Furthermore, we have presented a new hybrid path planning method called Vertex-based Pruning (VBP). We have discussed the quality of VBP paths with both theoretical and empirical analyses. VBP needs to construct a new graph for every path planning query. However, the experiments we have conducted show that comparably large ϵ-error bounds of 0.5 are already sufficient to compute high-quality paths that follow the exact geometry of both simple and complex weighted scenes. The significantly improved query times of VBP compared to the Steiner point method [Aleksandrov et al. 1998] enables the computation of such paths at interactive rates.

For future work, it would be interesting to improve on the upper bound of grid-optimal path costs in grid-aligned regions. The upper bound of $4+\sqrt{4-2\sqrt{2}}$ is a first approach, but it is still coarse. By further analyzing the underlying geometric properties of optimal paths in such scenarios, we believe that the least upper bound is closer to the upper bound of the unweighted variant of the problem, which is $\sqrt{4-2\sqrt{2}} \approx 1.08$; see Lemma 1 in Section 4. As for the new VBP method, it would be interesting to combine it with an region-based path following method such as MIRAN [Jaklin et al. 2013], and to conduct a comparative study with other existing ϵ-approximation and grid-based algorithms. Combining the pruning step of VBP with other advanced graph-search strategies might also lead to interesting variants of VBP.

In conclusion, we believe that our work can help to further understand the underlying mathematical principles of the WRP. Furthermore, we believe that the VBP method can be used to improve path planning in weighted regions in a variety of fields such as gaming, simulation and robotics applications.

8 Acknowledgements

This research has been supported by the COMMIT project (http://www.commit-nl.nl/). Norman Jaklin and Roland Geraerts are part of the Institute of Information and Computing Sciences, Utrecht University, 3584 CC Utrecht, the Netherlands.

References

ALEKSANDROV, L., LANTHIER, M., MAHESHWARI, A., AND SACK, J.-R. 1998. An epsilon-approximation for weighted shortest paths on polyhedral surfaces. In *Proceedings of the 6th Scandinavian Workshop on Algorithm Theory*, Springer-Verlag, SWAT '98, 11–22.

ALEKSANDROV, L., MAHESHWARI, A., AND SACK, J.-R. 2005. Determining approximate shortest paths on weighted polyhedral surfaces. *Journal of the ACM 52*, 1 (Jan.), 25–53.

ALEKSANDROV, L., DJIDJEV, H., GUO, H., MAHESHWARI, A., NUSSBAUM, D., AND SACK, J.-R. 2010. Algorithms for approximate shortest path queries onweighted polyhedral surfaces. *Discrete & Computational Geometry 44*, 4, 762–801.

BRESENHAM, J. E. 1965. Algorithm for computer control of a digital plotter. *IBM Systems Journal 4*, 1, 25–30.

CARUFEL, J.-L. D., GRIMM, C., MAHESHWARI, A., OWEN, M., AND SMID, M. 2012. Unsolvability of the weighted region shortest path problem. In *European Workshop on Computational Geometry (EuroCG)*, 65–68.

CHENG, S.-W., JIN, J., VIGNERON, A., AND WANG, Y. 2010. Approximate shortest homotopic paths in weighted regions. In *Algorithms and Computation*, O. Cheong, K.-Y. Chwa, and K. Park, Eds., vol. 6507 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 109–120.

CHESTNUTT, J., NISHIWAKI, K., KUFFNER, J., AND KAGAMI, S. 2007. An adaptive action model for legged navigation planning. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, 196–202.

DIJKSTRA, E. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik 1*, 1, 269–271.

FERGUSON, D., AND STENTZ, A. 2006. Using interpolation to improve path planning: The field d* algorithm. *Journal of Field Robotics 23*, 79–101.

GHEIBI, A., MAHESHWARI, A., AND SACK, J.-R. 2013. Weighted region problem in arrangement of lines. In *CCCG*, Carleton University, Ottawa, Canada.

HART, P., NILSSON, N., AND RAPHAEL, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics 4*, 2, 100 –107.

JAKLIN, N., COOK IV, A., AND GERAERTS, R. 2013. Real-time path planning in heterogeneous environments. *Computer Animation and Virtual Worlds 24*, 285–295.

KAMPHUIS, A., ROOK, M., AND OVERMARS, M. 2005. Tactical path finding in urban environments. *Proceedings of the First International Workshop on Crowd Simulation*, 51–60.

MATA, C., AND MITCHELL, J. 1997. A new algorithm for computing shortest paths in weighted planar subdivisions (extended abstract). *Proceedings of the thirteenth annual symposium on Computational Geometry*, 264–273.

MITCHELL, J. S. B., AND PAPADIMITRIOU, C. H. 1991. The weighted region problem: finding shortest paths through a weighted planar subdivision. *Journal of the ACM 38*, 1, 18–73.

NASH, A., KOENIG, S., AND TOVEY, C. A. 2010. Lazy theta*: Any-angle path planning and path length analysis in 3d. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010.*

NASH, A. 2012. *Any-Angle Path Planning*. PhD thesis, University of Southern California.

REECE, D., KRAUS, M., AND DUMANIOR, P. 2000. Tactical movement planning for individual combatants. *Proceedings of the 9th Conference on Computer Generated Forces and Behavioral Representation.*

SUN, Z., AND REIF, J. 2001. Bushwhack: An approximation algorithm for minimal paths through pseudo-euclidean spaces. In *In Proceedings of the 12th Annual International Symposium on Algorithms and Computation*, Springer, 160–171.

VAN DER STERREN, W. 2002. Tactical path-finding with A*. *Game Programming Gems 3*, 294–306.

Scene	ϵ	Method	Construction time (ms)	Query time (ms)	Nodes explored	Path costs
Puddle	0.1	Steiner point method	49911.9	222.3	7441	231.5
		VBP	**18396.8**	**77.6**	**3561**	231.5
	0.2	Steiner point method	2943.4	29.0	2856	231.5
		VBP	**1084.5**	**10.7**	**1367**	231.5
	0.3	Steiner point method	617.7	8.7	1560	231.8
		VBP	**283.1**	**3.1**	**751**	231.8
	0.4	Steiner point method	255.3	3.6	986	232.2
		VBP	**164.2**	**1.4**	**479**	232.2
	0.5	Steiner point method	167.0	1.8	675	232.3
		VBP	**125.0**	**0.7**	**331**	232.3
Bars	0.1	Steiner point method	6993.7	62.8	4908	273.6
		VBP	**5281.6**	**48.5**	**3957**	273.6
	0.2	Steiner point method	450.3	8.4	1774	273.8
		VBP	**394.6**	**6.7**	**1468**	273.8
	0.3	Steiner point method	**166.1**	2.4	909	273.8
		VBP	171.9	**1.9**	**776**	273.8
	0.4	Steiner point method	**117.6**	0.9	542	274.0
		VBP	136.5	**0.8**	**475**	274.0
	0.5	Steiner point method	**120.7**	0.5	370	274.2
		VBP	131.5	**0.4**	**329**	274.2
High-low	0.1	Steiner point method	127198.0	471.1	7887	592.7
		VBP	**117328.3**	**448.8**	**7409**	592.7
	0.2	Steiner point method	7895.9	60.9	3016	592.8
		VBP	**7245.3**	**59.9**	**2721**	592.8
	0.3	Steiner point method	1409.0	17.7	1648	592.8
		VBP	**1335.7**	**16.6**	**1494**	592.8
	0.4	Steiner point method	487.5	7.3	1039	592.8
		VBP	**475.6**	**6.9**	**946**	592.8
	0.5	Steiner point method	**251.8**	3.5	723	592.9
		VBP	252.2	**3.4**	**662**	592.9
Zigzag	0.1	Steiner point method	438767.4	2238.3	26245	343.6
		VBP	**303882.8**	**1789.0**	**21004**	343.6
	0.2	Steiner point method	28167.3	256.5	10334	343.6
		VBP	**19252.0**	**203.3**	**8333**	343.6
	0.3	Steiner point method	5089.8	72.8	5796	343.7
		VBP	**3547.5**	**57.8**	**4717**	343.7
	0.4	Steiner point method	1591.8	30.0	3765	343.8
		VBP	**1192.1**	**24.2**	**3096**	343.8
	0.5	Steiner point method	683.3	15.3	2645	343.8
		VBP	**565.3**	**12.4**	**2199**	343.8
Forest	0.1	Steiner point method	163325.0	1141.1	47969	2461.2
		VBP	**16857.2**	**224.6**	**15413**	2461.2
	0.2	Steiner point method	9638.0	127.8	17710	2461.4
		VBP	**1049.6**	**26.8**	**5700**	2461.4
	0.3	Steiner point method	1794.7	34.3	9370	2461.9
		VBP	**351.9**	**7.8**	**3031**	2461.9
	0.4	Steiner point method	600.6	13.4	5744	2462.5
		VBP	**245.0**	**3.1**	**1863**	2462.5
	0.5	Steiner point method	300.4	6.4	3826	2464.2
		VBP	**225.4**	**1.4**	**1243**	2464.2

Table 1: *Comparison of the Steiner point method by Aleksandrov et al. [1998] and the VBP method on all five scenes with ϵ-error bounds ranging from* 0.1 *to* 0.5.