

Optimization of Triangular Networks with Spatial Constraints

Valentin R. Koch* and Hung M. Phan †

April 03, 2019

Abstract

A common representation of a three dimensional object in computer applications, such as graphics and design, is in the form of a triangular mesh. In many instances, individual or groups of triangles in such representation need to satisfy spatial constraints that are imposed either by observation from the real world, or by concrete design specifications of the object. As these problems tend to be of large scale, choosing a mathematical optimization approach can be particularly challenging. In this paper, we model various geometric constraints as convex sets in Euclidean spaces, and find the corresponding projections in closed forms. We also present an interesting idea to successfully maneuver around some important nonconvex constraints while still preserving the intrinsic nature of the original design problem. We then use these constructions in modern first-order splitting methods to find optimal solutions.

AMS Subject Classification: Primary 26B25, 65D17; Secondary 49M27, 52A41, 90C25.

Keywords: alignment constraint, convex optimization, Douglas–Rachford splitting, maximum slope, minimum slope, oriented slope, projection methods.

1 Introduction and Motivation

The notation in the paper is fairly standard and follows largely [3]. \mathbb{R} denotes the set of real numbers. By “ $x := y$ ”, or equivalently “ $y =: x$ ”, we mean that “ x is defined by y ”. The *assignment operators* are denoted by “ \leftarrow ” and “ \rightarrow ”. The angle between two vectors \vec{n} and \vec{q} is denoted by $\angle(\vec{n}, \vec{q})$. The cross product of \vec{n}_1 and \vec{n}_2 in \mathbb{R}^3 is denoted by $\vec{n}_1 \times \vec{n}_2$.

1.1 Abstract Problem Formulation

Throughout the paper, we assume that $n \in \{3, 4, \dots\}$ and $X = \mathbb{R}^n$ with standard inner product $\langle \cdot, \cdot \rangle$ and induced norm $\| \cdot \|$. Assume that $G = (V_0, E_0)$ is a given *triangular mesh* on \mathbb{R}^2 where V_0 is the set of vertices and E_0 is the set of (undirected) edges, i.e.,

$$V_0 := \{p_i := (p_{i1}, p_{i2}) \in \mathbb{R}^2 \mid i \in \{1, 2, \dots, n\}\},$$

$$E_0 \subseteq \{p_i p_j \equiv p_j p_i \mid p_i, p_j \in V_0\}.$$

*Design and Creation Products (DCP), Autodesk, Inc. Email: valentin.koch@autodesk.com

†Department of Mathematical Sciences, Kennedy College of Sciences, University of Massachusetts Lowell, MA 01854, USA. E-mail: hung.phan@uml.edu

From G , we can derive the set of triangles of the mesh as follows

$$T_0 := \{\Delta = (p_i p_j p_k) \mid \{p_i p_j, p_j p_k, p_k p_i\} \subseteq E_0\}.$$

We first aim to

$$(1a) \quad \text{find a vector } z = (z_1, \dots, z_n) \in X$$

such that the points

$$(1b) \quad \{P_i := (p_{i1}, p_{i2}, z_i)\}_{i \in \{1, \dots, n\}} \text{ satisfy a given set of constraints.}$$

Clearly, the points $\{P_i\}_{i \in \{1, \dots, n\}}$ also form a corresponding triangular mesh S in three dimensions. Therefore, if there is no confusion, we will also use E_0, V_0, T_0 to denote the sets of vertices, edges, and triangles of the three dimensional mesh.

Several types of constraints for triangular meshes are listed below:

- **interval constraints:** For a given subset I of $\{1, 2, \dots, n\}$, for all $i \in I$, the entries z_i must lie in a given interval of \mathbb{R} .
- **edge slope constraints:** For a given subset E of the edges E_0 , and for every edge $P_i P_j \in E$, the slope

$$s_{ij} := \frac{z_i - z_j}{\sqrt{(p_{i1} - p_{j1})^2 + (p_{i2} - p_{j2})^2}}$$

must lie in a given subset of \mathbb{R} .

- **edge alignment constraints:** For a given pair of edges $P_i P_j, P_m P_n \in E_0$, the edges must have the same slope $s_{ij} = s_{mn}$.
- **surface alignment constraints:** For a given pair of triangles $\Delta_i, \Delta_j \in T_0$ the *normal vectors* \vec{n}_{Δ_i} and \vec{n}_{Δ_j} must be parallel.
- **surface orientation constraints:** For a given subset T of the triangles T_0 and for each triangle $\Delta \in T$, there is a given set of vectors $Q_\Delta \subset \mathbb{R}^3$ such that for each $\vec{q} \in Q_\Delta$, the *angle* between the *normal vector* \vec{n}_Δ and \vec{q} must lie in a given subset $\theta_{\vec{q}}$ of $[0, \pi]$.

Suppose there are J constraints imposed on a model. For $j \in \{1, 2, \dots, J\}$, we denote by C_j the set of all points that satisfies the j -th constraint. Thus, (1) can be written in the mathematical form

$$(2) \quad \text{find a point } x \in C := \bigcap_{j \in \{1, 2, \dots, J\}} C_j,$$

which we refer to as *feasibility problem*. Moreover, of the infinitude of possible solutions for (2), one may be particularly interested in those that are *optimal* in some sense. For instance, it could be desirable to find a solution that minimizes the slope change between adjacent triangles, a solution that minimizes the volume between the initial triangles and the triangles in the final solution, or variants and combinations thereof. If more than one objective function is of interest, it is common to additively combine these functions, perhaps by scaling the functions based on their different levels of importance. In summary, our goal is to solve the problem

$$(3) \quad \min F(z) \text{ subject to } z \in C := \bigcap_{j \in \{1, 2, \dots, J\}} C_j,$$

where F may be a sum of (scaled) objective functions. The function F is typically nonsmooth which prevents the use of standard optimization methods.

1.2 Computer-Aided Design for Architecture and Civil Engineering Structures

The abstract problem formulation in Section 1.1 has some concrete applications in computational surface generation of triangular meshes. In particular, in Computer-Aided Design (CAD), triangular meshes are widely used in various engineering disciplines. For example, in architecture and civil engineering, existing and finished ground surfaces are represented by triangulated irregular networks. Slopes are relevant in the context of drainage, in both, civil engineering (transportation structures), and architecture (roof designs), as well as in the context of surface alignments such as solar farms, embankment dams, and airport infrastructure layouts.

A concrete problem that arises in civil engineering design is the grading of a parking lot. Within a given area, the engineer has to define grading slopes such that the parking lot fits within existing structures, the drainage requirements on the lot are met, and safety and comfort is taken into account. Besides these requirements, the engineer would like to change the existing surface as little as possible, in order to save on earthwork costs.

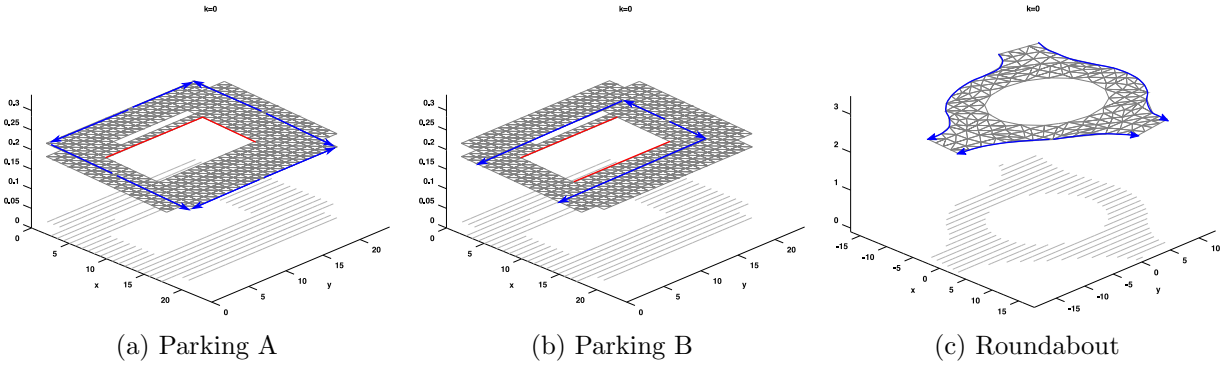


Figure 1: Drainage schemes for various engineering structures.

Consider Figure 1. The triangular mesh in Figure 1a represents the existing ground of a planned parking lot. The engineer would like the water to drain away from the building, and along the blue drain lines into the four corners. Red lines indicate where the triangle edges need to be aligned. In Figure 1b, the engineer would like to study a different scheme, where the drainage happens in parallel, on either side of the building. Lastly, Figure 1c represents the triangular mesh of existing ground for a roundabout, where a minimum slope is required from the inner circle to the outer one, and water needs to drain from the road at the top along the outer circle to the roads on either side at the bottom.

1.3 Objective and Outline of This Paper

This paper aims to provide a framework for modeling practical design problems using geometric constraints in three dimensions. These problems can then be solved by iterative optimization methods. This involves the introduction and computation of projection/proximity operators of new constraints and objective functions. Once all required formulas are accomplished, they will be used in iterative optimization methods to obtain the solutions.

The paper is organized as follows. Section 2 contains an overview of iterative methods that will be employed. From Section 3 to Section 6, we derive the projection operators of the above spatial constraints in closed forms. Particularly in Section 6, we present an interesting idea to *modify* certain nonconvex constraints into convex ones that retain the intrinsic nature of the original design problem. To the best of our knowledge, this is the *first* methodology to successfully maneuver around such

nonconvexity. Utilizing these constructions, we include in Section 7 some optimization problems of interest. Finally, we present the numerical experiment in Section 8 and some remarks in Section 9.

2 Methods Overview

2.1 Projections onto Constraint Sets

A constraint set $C \subseteq X$ is the collection of all feasible data points, i.e., points that satisfy some requirements. Suppose the given data point $z \in X$ is not feasible (i.e., $z \notin C$), we aim to *modify* z so that the newly obtained point x is feasible (i.e., $x \in C$); and we would like to do it with *minimal* adjustment on z . This task can be achieved by using the projection onto C . Recall that the projection of z onto C , denoted by $P_C z$, is the solution of the optimization problem

$$P_C z = \operatorname{argmin}_{x \in C} \|z - x\| = \{x \in C \mid \|z - x\| = \min_{y \in C} \|z - y\|\}.$$

It is well known that if C is nonempty, closed, and *convex*¹, then $P_C z$ is singleton, see, for example, [21, Theorem 2.10].

2.2 Proximity Operators

Suppose $f : X \rightarrow (-\infty, +\infty]$ is a proper convex lower semicontinuous function² and x is a given point in X . Then it is well known (see [3, Section 12.4]) that the function

$$X \rightarrow (-\infty, +\infty] : y \mapsto f(y) + \frac{1}{2}\|x - y\|^2$$

has a unique minimizer, which we will denote by $\operatorname{Prox}_f(x)$. The induced operator $\operatorname{Prox}_f : X \rightarrow X$ is called the *proximal mapping* or *proximity operator* of f (see [19]). Note that if f is the *indicator function* of a set C (the indicator function ι_C is defined by $\iota_C(x) = 0$ if $x \in C$ and $\iota_C(x) = +\infty$ otherwise), then $\operatorname{Prox}_f = P_C$. Thus, proximity operators are generalizations of projections.

2.3 Iterative Methods for Optimization Problems

Iterative optimization methods are often used for solving (3), which may require the computations of proximity and projection operators for the functions and constraint sets involved.

It turns out that all spatial constraints encountered in our settings are convex and closed. Hence, their projections always exist and are unique. Moreover, we also successfully obtain explicit formulas for these projections. In the coming sections, we will make the formulas as convenient as possible for software implementation. As we will make proximity operators available for several types of objective functions, any iterative optimization methods that utilize proximity operators, for example, [7, 8, 13, 18], can be used to solve the corresponding problems. Let us describe one such method, the Douglas–Rachford (DR) splitting algorithm. The DR algorithm emerged from the field of differential equations [14], and was later made widely applicable in other areas thanks to the seminal work [18].

To formulate DR algorithm, we first use indicator functions to convert (3) into

$$(4) \quad \min \quad F(x) + \iota_{C_1} + \iota_{C_2} + \cdots + \iota_{C_J} \quad \text{subject to} \quad x \in X.$$

¹ C is convex if for all $x, y \in C$ and $\lambda \in [0, 1]$, we have $(1 - \lambda)x + \lambda y \in C$

²See, e.g., [20] and [3] for relevant materials in convex analysis

So, it suffices to present DR for the following general optimization problem

$$(5) \quad \min \sum_{i=1}^m f_i(x) \quad \text{subject to} \quad x \in X,$$

where each f_i is a proper convex lower semicontinuous function on X . The DR operates in the product space $\mathbf{X} := X^m$ with inner product $\langle \mathbf{x}, \mathbf{y} \rangle := \sum_{i=1}^m \langle x_i, y_i \rangle$ for $\mathbf{x} = (x_i)_{i=1}^m$ and $\mathbf{y} = (y_i)_{i=1}^m$. Set the starting point $\mathbf{x}_0 = (z, \dots, z) \in \mathbf{X}$, where $z \in X$. Given $\mathbf{x}_k = (x_{k,1}, \dots, x_{k,m}) \in \mathbf{X}$, we compute

$$(6a) \quad \bar{x}_k := \frac{1}{m} \sum_{i=1}^m x_{k,i},$$

$$(6b) \quad \forall i \in \{1, \dots, m\} : \quad x_{k+1,i} := x_{k,i} - \bar{x}_k + \text{Prox}_{\gamma f_i}(2\bar{x}_k - x_{k,i}),$$

$$(6c) \quad \text{then update} \quad \mathbf{x}_{k+1} := (x_{k+1,1}, \dots, x_{k+1,m}).$$

Then the sequence $(\bar{x}_k)_{k \in \mathbb{N}}$ converges to a solution of (5). We note that $(\mathbf{x}_k)_{k \in \mathbb{N}}$ and $(\bar{x}_k)_{k \in \mathbb{N}}$ are referred to as *governing* and *monitored* sequences, respectively.

It is worth mentioning that when all f_i 's are indicator functions of the constraints (thus, all proximity operators become projections), then (5) becomes a pure feasibility problem

$$(7) \quad \text{find a point} \quad x \in C := \bigcap_{j \in \{1, 2, \dots, J\}} C_j.$$

Therefore, all optimization methods that work for (5) can also be used for (7). Nevertheless, it is worth mentioning that there are many iterative projection methods that are specifically designed for feasibility problems. At the first glance, it might be tempting to solve such problem by finding the projection onto the intersection C directly. However, this is often not possible due to the complexity of C . A workaround is to utilize the projection P_{C_j} onto each constraint, if the explicit formula is available. Then with an initial point, one can *iteratively execute* the projections P_{C_j} 's to derive a solution for (7). Let $z_0 \in X$ be the initial data point. The following are two simplest instances among iterative projection methods (see [3, 10, 12] and the references therein)

- **cyclic projections:** Given z_k , we update $z_{k+1} := Tz_k$ where $T := P_{C_J} P_{C_{J-1}} \cdots P_{C_2} P_{C_1}$.
- **parallel projections:** Given z_k , we update $z_{k+1} := Tz_k$ where $T := \frac{1}{J} (P_{C_1} + P_{C_2} + \cdots + P_{C_J})$.

In addition, when projection methods succeed (see [11] for some interesting examples), they have various attractive features: they are easy to understand, simple for implementation and maintenance, and sometime can be very fast. We refer the readers to [2, 4, 10, 12] for more comprehensive discussions on projection methods.

3 Projections onto Linear Constraints

By linear constraints, we refer to any constraint on the triangular mesh that can be represented by a system of linear equalities and inequalities. Indeed, this class includes several important constraints in design problems. In this section, we will analyze those constraints and their projectors.

3.1 Interval Constraint

Similar to [4, Section 2.2], we assume that I is a subset of $\{1, 2, \dots, n\}$ and $(l_i)_{i \in I}, (u_i)_{i \in I} \in \mathbb{R}^I$ are given. Define

$$Y := \{z = (z_1, z_2, \dots, z_n) \in X \mid \forall i \in I: l_i \leq z_i \leq u_i\}.$$

Then one can readily check that Y is closed and convex. The following explicit formula is for the projection onto Y , whose proof is straightforward, thus, omitted.

$$\begin{aligned} P_Y : X &\rightarrow X : (z_1, z_2, \dots, z_n) \mapsto (x_1, x_2, \dots, x_n), \\ \text{where } x_i &= \begin{cases} \max\{l_i, \min\{u_i, z_i\}\}, & i \in I, \\ z_i, & i \in \{1, 2, \dots, n\} \setminus I. \end{cases} \end{aligned}$$

3.2 Edge Minimum Slope Constraint

Let $P_i = (p_{i1}, p_{i2}, z_i)$ and $P_j = (p_{j1}, p_{j2}, z_j)$. The designer may require that the (directional) slope from P_j to P_i must be no less than a threshold level s_{ij} . More specifically, since the value $p_{i1}, p_{i2}, p_{j1}, p_{j2}$ are fixed, we can write this constraint as

$$z_i - z_j \geq \alpha_{ij} := s_{ij} \sqrt{(p_{i1} - p_{j1})^2 + (p_{i2} - p_{j2})^2},$$

which we will call the *edge minimum slope constraint*. This constraint is a linear inequality, thus, convex. The projection formula onto this type of constraint can be derived analogously to [4, Section 2.3].

3.3 Low Point Constraint

Definition 3.1 (low point). A point $P_j = (p_{j1}, p_{j2}, z_j)$ on the mesh is called a low point if each point $P_i = (p_{i1}, p_{i2}, z_i)$ connected to P_j satisfies the edge minimum slope constraint

$$z_i - z_j \geq \alpha_i := s_i \sqrt{(p_{i1} - p_{j1})^2 + (p_{i2} - p_{j2})^2},$$

where all $s_i \in \mathbb{R}$ are given.

We can treat low point constraint as a *single* constraint even though it is the intersection of finitely many edge slope constraints. The following result shows how to project onto this constraint. First, we need a simple lemma

Lemma 3.2. *Let $a, b \in \mathbb{R}$ and $k \geq 1$. Assume that $ka \leq b + (k-1)\max\{a, b\}$. Then $a \leq b$.*

Proof. By assumption, we must have either $ka \leq b + (k-1)a$ or $ka \leq b + (k-1)b$, and any one of them readily implies $a \leq b$. \square

Theorem 3.3 (projection onto a low point constraint). *Let $\alpha_2, \dots, \alpha_m \in \mathbb{R}$. Define the set*

$$E := \{(x_1, \dots, x_m) \in \mathbb{R}^m \mid \forall i \in \{2, \dots, m\}: x_i - x_1 \geq \alpha_i\}.$$

Let $z := (z_1, \dots, z_m) \in \mathbb{R}^m$. Let $\delta_1 := z_1$ and let $\delta_2 \leq \delta_3 \leq \dots \leq \delta_m$ be the values $\{z_i - \alpha_i\}_{i \in \{2, \dots, m\}}$ in nondecreasing order. Let k be the largest number in $\{1, \dots, m\}$ such that $\delta_k \leq (\delta_1 + \dots + \delta_k)/k$. Then the projection $x := (x_1, x_2, \dots, x_m) := P_E z$ is given by

$$x_1 = (\delta_1 + \dots + \delta_k)/k \quad \text{and} \quad \forall i \in \{2, \dots, m\}: x_i = \max\{x_1, z_i - \alpha_i\} + \alpha_i.$$

Proof. First, x is the solution of

$$\begin{aligned} (8a) \quad & \min \quad (x_1 - z_1)^2 + \dots + (x_m - z_m)^2 \\ (8b) \quad & \text{s.t.} \quad x_1 - x_i + \alpha_i \leq 0, \quad i \in \{2, \dots, m\}. \end{aligned}$$

Let $y := (y_1, \dots, y_m)$ where $y_1 = x_1$ and $y_i := x_i - \alpha_i$ for $i \in \{2, \dots, m\}$. Without relabeling, we may assume $\delta_i = z_i - \alpha_i$ for all $i \in \{2, \dots, m\}$. Then (8) becomes

$$\begin{aligned} (9a) \quad & \min \quad \varphi(y) := (y_1 - \delta_1)^2 + \dots + (y_m - \delta_m)^2 \\ (9b) \quad & \text{s.t.} \quad g_i(y) := y_1 - y_i \leq 0, \quad i \in \{2, \dots, m\}. \end{aligned}$$

To find the (unique) solution, we use Lagrange multipliers: there exist $\lambda_2, \dots, \lambda_m$, such that

$$\begin{aligned} (10a) \quad & (1/2)\nabla\varphi(y) + \lambda_2\nabla g_2(y) + \dots + \lambda_m\nabla g_m(y) = 0, \\ (10b) \quad & \forall i \in \{2, \dots, m\} : \quad \lambda_i g_i(y) = 0, \quad \lambda_i \geq 0, \quad g_i(y) \leq 0. \end{aligned}$$

Then (10a) implies

$$\begin{aligned} & (y_1 - \delta_1) + \lambda_2 + \dots + \lambda_m = 0 \\ & \forall i \in \{2, \dots, m\} : \quad (y_i - \delta_i) - \lambda_i = 0. \end{aligned}$$

So $y_1 \leq \delta_1$ because $\lambda_2, \dots, \lambda_m \geq 0$. By substitution, we get

$$(11) \quad y_1 + y_2 + \dots + y_m = \delta_1 + \delta_2 + \dots + \delta_m.$$

Also, (10b) reads as, for each $i \in \{2, \dots, m\}$,

$$0 = \lambda_i g_i(y) = (y_i - \delta_i)(y_1 - y_i), \quad y_i - \delta_i \geq 0, \quad \text{and} \quad y_1 - y_i \leq 0.$$

It follows that either $y_i = \delta_i \geq y_1$ or $y_1 = y_i \geq \delta_i$, i.e.,

$$(12) \quad \forall i \in \{2, \dots, m\} : \quad y_i = \max\{y_1, \delta_i\}.$$

Substituting (12) into (11) yields

$$(13) \quad \delta_1 + \dots + \delta_m = y_1 + \max\{y_1, \delta_2\} + \dots + \max\{y_1, \delta_m\}.$$

So for all $j \in \{1, \dots, m\}$, (13) implies

$$(14) \quad \delta_1 + \dots + \delta_m \geq \underbrace{y_1 + \dots + y_1}_{j \text{ terms}} + \delta_{j+1} + \dots + \delta_m \implies y_1 \leq \frac{\delta_1 + \dots + \delta_j}{j}.$$

Next, let k be the largest number in $\{1, \dots, m\}$ that satisfies

$$(15) \quad \delta_k \leq \frac{\delta_1 + \dots + \delta_k}{k}.$$

The number k is well defined since at least (15) is true for $k = 1$. Now we claim that

$$(16) \quad y_1 = \frac{\delta_1 + \dots + \delta_k}{k}$$

by considering two cases.

Case 1: $k = m$. Using (15), (13), and $\delta_2 \leq \delta_3 \leq \dots \leq \delta_m$, we have

$$\begin{aligned} m\delta_m &\leq \delta_1 + \dots + \delta_m = y_1 + \max\{y_1, \delta_2\} + \dots + \max\{y_1, \delta_m\} \\ &\leq y_1 + (m-1) \max\{y_1, \delta_m\}. \end{aligned}$$

Then Lemma 3.2 implies $\delta_m \leq y_1$. Using this in (13), we derive $\delta_1 + \dots + \delta_m = my_1$ which is (16).

Case 2: $k < m$. By the choice of k , we have $\frac{\delta_1 + \dots + \delta_{k+1}}{k+1} < \delta_{k+1}$, which implies

$$\frac{\delta_1 + \dots + \delta_k}{k} < \delta_{k+1}.$$

Combining with (14), we conclude that $y_1 < \delta_{k+1}$. Using $y_1 < \delta_{k+1}$ and (15) in (13), we obtain

$$\begin{aligned} k\delta_k + \delta_{k+1} + \dots + \delta_m &\leq (\delta_1 + \dots + \delta_k) + \delta_{k+1} + \dots + \delta_m \\ &= y_1 + \max\{y_1, \delta_2\} + \dots + \max\{y_1, \delta_m\} \\ &= y_1 + \underbrace{\max\{y_1, \delta_2\} + \dots + \max\{y_1, \delta_k\}}_{k-1 \text{ terms}} + \delta_{k+1} + \dots + \delta_m \\ &\leq y_1 + (k-1) \max\{y_1, \delta_k\} + \delta_{k+1} + \dots + \delta_m. \end{aligned}$$

This implies

$$k\delta_k \leq y_1 + (k-1) \max\{y_1, \delta_k\}.$$

Again, Lemma 3.2 implies $\delta_k \leq y_1$. Now using $\delta_k \leq y_1 < \delta_{k+1}$ in (13), we have

$$\delta_1 + \dots + \delta_m = ky_1 + \delta_{k+1} + \dots + \delta_m, \quad \text{which implies (16).}$$

So, (16) is true. Finally, we compute y_i 's from (12) and (16), then use them to derive x_i 's. \square

3.4 Edge Alignment Constraint

On the triangular mesh, the designer may want a *constant slope* on a particular path, in which case we say the path is “aligned”. Such a path is sometimes called a *feature line*. To formulate this constraint, suppose the feature line is given by adjacent points A_1, A_2, \dots, A_m on the triangular mesh where $A_i = (a_{i1}, a_{i2}, x_i) \in \mathbb{R}^3$. For $A_i A_{i+1}$, the length of its “shadow” on the xy -plane is the euclidean distance

$$(17) \quad \delta_i := \|(a_{i+1,1}, a_{i+1,2}) - (a_{i,1}, a_{i,2})\| = \sqrt{(a_{i+1,1} - a_{i,1})^2 + (a_{i+1,2} - a_{i,2})^2}.$$

Define also

$$(18) \quad t_1 := 0, \quad t_2 := \delta_1, \quad t_3 := \delta_1 + \delta_2, \quad \dots, \quad t_m := \delta_1 + \dots + \delta_{m-1}.$$

Then the alignment constraint is written as

$$(19) \quad \forall i \in \{1, \dots, m-2\}: (x_{i+1} - x_i)/(t_{i+1} - t_i) = (x_{i+2} - x_{i+1})/(t_{i+2} - t_{i+1}).$$

Theorem 3.4 (projection onto an edge alignment constraint). *Suppose the points $(a_{i1}, a_{i2}) \in \mathbb{R}^2$ with $i \in \{1, \dots, m\}$, forms a path in \mathbb{R}^2 . Let δ_i and t_i be given respectively by (17) and (18). Let F be the set of points $(x_1, \dots, x_m) \in \mathbb{R}^m$ such that (19) is satisfied.*

Let $z = (z_1, \dots, z_m) \in \mathbb{R}^m$. Then the projection $P_F z \in \mathbb{R}^m$ is given by

$$\forall i \in \{1, \dots, m\}: (P_F z)_i = f(t_i) = \alpha + \beta t_i.$$

where $f(t) = \alpha + \beta t$ is the least squares line for the points $(t_i, z_i) \in \mathbb{R}^2$, $i \in \{1, \dots, m\}$.

Proof. First, F is convex since all constraints in (19) are linear. Next, we consider the points (t_i, z_i) in \mathbb{R}^2 . The problem is to find (x_1, \dots, x_m) such that the points (t_i, x_i) are aligned and

$$\|x - z\|^2 = \sum_{i=1}^m (x_i - z_i)^2 \quad \text{is minimized.}$$

This is the *least squares* problem for the points (t_i, z_i) . The proof is complete. \square

3.5 Surface Alignment Constraint

The designer may want to patch several adjacent triangles on the mesh into a single polygon, in which case we say that these triangles are “aligned”. This is equivalent to requiring all vertices of the triangles to lie on the same plane. So we have the following result.

Theorem 3.5 (projection onto a surface alignment constraint). *Let (a_{i1}, a_{i2}) , $i \in \{1, \dots, m\}$ be a collection of points in \mathbb{R}^2 that are not on the same line. Let F be the set of all points $(x_1, \dots, x_m) \in \mathbb{R}^m$ such that the points $\{(a_{i1}, a_{i2}, x_i)\}_{i \in \{1, \dots, m\}}$ lie on the same plane in \mathbb{R}^3 . Let $z = (z_1, \dots, z_m) \in \mathbb{R}^m$. Then the projection $P_F z \in \mathbb{R}^m$ is given by*

$$\forall i \in \{1, \dots, m\} : \quad (P_F z)_i = f(a_{i1}, a_{i2}) = \alpha + \beta a_{i1} + \gamma a_{i2},$$

where $f(t_1, t_2) = \alpha + \beta t_1 + \gamma t_2$ is the least squares plane for the points $(a_{i1}, a_{i2}, z_i) \in \mathbb{R}^3$, $i \in \{1, \dots, m\}$.

Proof. Clearly, F is a convex set. Let $x := (x_1, \dots, x_m) = P_F z$, then x minimizes

$$\|x - z\|^2 = \sum_{i=1}^m |x_i - z_i|^2$$

subject to the constraint that (a_{i1}, a_{i2}, x_i) , $i \in \{1, \dots, m\}$, lie on the same plane in \mathbb{R}^3 . Thus, it is equivalent to finding the least squares plane $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ for the points (a_{i1}, a_{i2}, z_i) , $i \in \{1, \dots, m\}$, which has unique solution since these points are not on the same line. The conclusion follows. \square

4 Projections onto General Surface Slope Constraints

Surface slope constraints are any requirement imposed on the *slope* of a triangle. In this section, we provide a general setup for projections onto such constraints.

4.1 Normal Vector and Surface Slope Constraint

Let $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$ be the standard basis of \mathbb{R}^3 . Given three points $P_1 = (p_{11}, p_{12}, h_1)$, $P_2 = (p_{21}, p_{22}, h_2)$, and $P_3 = (p_{31}, p_{32}, h_3)$ in \mathbb{R}^3 , the normal vector to the plane $P_1 P_2 P_3$ is the cross product

$$(20) \quad \vec{n} = \begin{pmatrix} p_{11} - p_{31} \\ p_{12} - p_{32} \\ h_1 - h_3 \end{pmatrix} \times \begin{pmatrix} p_{21} - p_{31} \\ p_{22} - p_{32} \\ h_2 - h_3 \end{pmatrix} =: \begin{pmatrix} a_1 \\ b_1 \\ t_1 \end{pmatrix} \times \begin{pmatrix} a_2 \\ b_2 \\ t_2 \end{pmatrix} = \begin{pmatrix} b_1 t_2 - b_2 t_1 \\ a_2 t_1 - a_1 t_2 \\ a_1 b_2 - a_2 b_1 \end{pmatrix},$$

where the new variables $a_1, a_2, b_1, b_2, t_1, t_2$ are defined correspondingly, e.g., $a_1 := p_{11} - p_{31}$, etc. Also, we always assume that the “shadows” of P_1, P_2, P_3 on xy -plane (p_{11}, p_{12}) , (p_{21}, p_{22}) , (p_{31}, p_{32}) are not on the same line. Thus, $a_1 b_2 - a_2 b_1 \neq 0$.

So we can rescale and use

$$\vec{n} = \left(\frac{b_1 t_2 - b_2 t_1}{a_1 b_2 - a_2 b_1}, -\frac{a_1 t_2 - a_2 t_1}{a_1 b_2 - a_2 b_1}, 1 \right).$$

If we define

$$(21) \quad \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} =: \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} \iff \begin{pmatrix} u \\ v \end{pmatrix} := \frac{1}{a_1 b_2 - a_2 b_1} \begin{pmatrix} b_2 & -b_1 \\ -a_2 & a_1 \end{pmatrix} \begin{pmatrix} t_1 \\ t_2 \end{pmatrix},$$

then $\vec{n} = (-u, -v, 1)$. Obviously, surface slope constraints depend solely on the normal vector \vec{n} . Thus, we can represent a surface slope constraint as

$$g(u, v) \leq 0.$$

An important case of such constraints is the *surface orientation constraint* below.

Definition 4.1 (surface orientation constraint). Let Δ be a triangle with normal vector $\vec{n} = (-u, -v, 1) \in \mathbb{R}^3$ as above. Let $\vec{q} = (q_1, q_2, q_3) \in \mathbb{R}^3 \setminus \{0\}$ be a unit vector and θ be an angle in $[0, \pi]$, the constraint

$$\angle(\vec{n}, \vec{q}) \leq \theta, \quad \text{or equivalently,} \quad \cos \angle(\vec{n}, \vec{q}) \geq \cos \theta,$$

is called the *surface orientation constraint* specified by the pair (\vec{q}, θ) .

In Section 4.2, we develop the general framework for projection onto surface orientation constraint. Then in Sections 5 and 6 respectively, we will consider two special surface orientation constraints: the *surface maximum slope constraint* and the *surface oriented minimum slope constraint*.

4.2 Projection onto a Surface Slope Constraint

Consider a single triangle determined by three points $Q_1 = (p_{11}, p_{12}, w_1)$, $Q_2 = (p_{21}, p_{22}, w_2)$, and $Q_3 = (p_{31}, p_{32}, w_3)$ in \mathbb{R}^3 . Without loss of generality, we can assume

$$w_1 + w_2 + w_3 = 0.$$

Projecting onto a slope constraint is to find the new heights h_1, h_2, h_3 that is a solution to the problem

$$\begin{aligned} \min_{(h_1, h_2, h_3) \in \mathbb{R}^3} \quad & \| (h_1, h_2, h_3) - (w_1, w_2, w_3) \|^2 \\ \text{subject to} \quad & \text{the triangle } P_1 P_2 P_3 \text{ satisfies a given slope constraint,} \\ & \text{where } P_1 = (p_{11}, p_{12}, h_1), P_2 = (p_{11}, p_{12}, h_2), \text{ and } P_3 = (p_{11}, p_{12}, h_3). \end{aligned}$$

Defining a_i, b_i, t_i, u, v as in (20) and (21), we have $h_1 = a_1 u + b_1 v + h_3$ and $h_2 = a_2 u + b_2 v + h_3$, i.e.,

$$(22) \quad \begin{pmatrix} h_1 \\ h_2 \\ h_3 \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & 1 \\ a_2 & b_2 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ h_3 \end{pmatrix}.$$

Suppose also the slope constraint on the triangle $P_1 P_2 P_3$ is written as $g(u, v) \leq 0$. Then the projection problem is converted to

$$(23a) \quad \min_{(u, v, h_3) \in \mathbb{R}^3} \quad \phi(u, v, h_3) := \left\| \begin{pmatrix} a_1 & b_1 & 1 \\ a_2 & b_2 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ h_3 \end{pmatrix} - \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} \right\|^2$$

$$(23b) \quad \text{subject to} \quad g(u, v) \leq 0.$$

Next, suppose further changing variables is necessary, for instance, (u, v) is replaced by the new variables (\hat{u}, \hat{v}) under a linear transformation

$$\begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} := Q \begin{pmatrix} u \\ v \end{pmatrix}, \quad \text{where } Q \text{ is an invertible matrix.}$$

Then (23) is equivalent to

$$(24a) \quad \min_{(\hat{u}, \hat{v}, h_3) \in \mathbb{R}^3} \left\| \begin{pmatrix} \hat{a}_1 & \hat{b}_1 & 1 \\ \hat{a}_2 & \hat{b}_2 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \hat{u} \\ \hat{v} \\ h_3 \end{pmatrix} - \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} \right\|^2$$

$$(24b) \quad \text{subject to } \hat{g}(\hat{u}, \hat{v}) := g(u, v) \leq 0,$$

where $\begin{pmatrix} \hat{a}_1 & \hat{b}_1 \\ \hat{a}_2 & \hat{b}_2 \end{pmatrix} := \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} Q^{-1}$. That means we can treat (24) as (23) with *new* coefficients \hat{a}_i, \hat{b}_i and constraint function \hat{g} .

Next, we will simplify the model (23) further. Since (23b) does not involve h_3 , we can convert problem (23) into two variables (u, v) as follows: first, set the derivative of ϕ with respect to h_3 to zero

$$\begin{aligned} \nabla_{h_3} \phi &= 2 \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \left[\begin{pmatrix} a_1 & b_1 & 1 \\ a_2 & b_2 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ h_3 \end{pmatrix} - \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} \right] \\ &= 2[(a_1 + a_2)u + (b_1 + b_2)v - 3h_3 - (w_1 + w_2 + w_3)] = 0. \end{aligned}$$

Since $w_1 + w_2 + w_3 = 0$, we obtain

$$(26) \quad h_3 = -(1/3)[(a_1 + a_2)u + (b_1 + b_2)v].$$

Substituting (26) into (23a), we have

$$(27a) \quad \phi(u, v, h_3) = \left\| \begin{pmatrix} a_1 & a_2 & 1 \\ a_2 & b_2 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -\frac{a_1+a_2}{3} & -\frac{b_1+b_2}{3} \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} - \begin{pmatrix} w_1 \\ w_2 \\ -(w_1 + w_2) \end{pmatrix} \right\|^2$$

$$(27b) \quad =: \frac{1}{9} \left[\frac{1}{2} \begin{pmatrix} u & v \end{pmatrix} \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} - \begin{pmatrix} w_a & w_b \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + Z \right]$$

where Z is some constant independent of (u, v) and

$$(28a) \quad A := 2(a_1^2 + a_2^2 - a_1 a_2) > 0, \quad B := 2(b_1^2 + b_2^2 - b_1 b_2) > 0,$$

$$(28b) \quad C := 2a_1 b_1 + 2a_2 b_2 - a_1 b_2 - a_2 b_1, \quad \text{and}$$

$$(28c) \quad \begin{pmatrix} w_a & w_b \end{pmatrix} := 3 \begin{pmatrix} w_1 & w_2 \end{pmatrix} \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}.$$

Thus, (23) is equivalent to the problem

$$(29a) \quad \min_{(u, v) \in \mathbb{R}^2} \varphi(u, v) := \frac{1}{2} \begin{pmatrix} u & v \end{pmatrix} \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} - \begin{pmatrix} w_a & w_b \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

$$(29b) \quad \text{subject to } g(u, v) \leq 0.$$

As long as we can find the solution (u, v) , we can obtain (h_1, h_2, h_3) by using (26) and (22). In the case new variables \hat{u}, \hat{v} are used, we will use the corresponding coefficients $(\hat{a}_i, \hat{b}_i, \hat{u}, \hat{v})$ in place of (a_i, b_i, u, v) . Finally, we show that $\varphi(u, v)$ is strictly convex.

Lemma 4.2. Suppose $a_1, a_2, b_1, b_2 \in \mathbb{R}$ such that $a_1 b_2 - a_2 b_1 \neq 0$. Define A, B, C by (28a)–(28b). Then $\begin{pmatrix} A & C \\ C & B \end{pmatrix} \succ 0$. Consequently, the function $\varphi(u, v)$ in problem (29) is strictly convex.

Proof. Note from (27) that $\begin{pmatrix} A & C \\ C & B \end{pmatrix} = M^T M$ where $M := \begin{pmatrix} a_1 & a_2 & 1 \\ a_2 & b_2 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -\frac{a_1+a_2}{3} & -\frac{b_1+b_2}{3} \end{pmatrix}$. Since $a_1 b_2 - a_2 b_1 \neq 0$, M has full column rank, which implies $M^T M$ is positive definite. It follows that φ is strictly convex. \square

5 Projections onto Surface Maximum Slope Constraints

Adopting the notation in Section 4.1, we let $P_1 P_2 P_3$ be a triangle in \mathbb{R}^3 with normal vector $\vec{n} = (-u, -v, 1)$. In certain cases, it is required that the angle between \vec{n} and a given direction \vec{q} must not exceed a given threshold. For example, suppose $P_1 P_2 P_3$ represents the desired ground, that cannot be too steep with respect to gravity, i.e., the slope of the plane $P_1 P_2 P_3$ must not exceed a threshold $s := s_{\max} \in \mathbb{R}_+$. Then the angle between \vec{n} and $\mathbf{e}_3 := (0, 0, 1)$ must satisfy $\angle(\vec{n}, \mathbf{e}_3) \leq \tan^{-1}(s)$, which is equivalent to

$$(30a) \quad \cos \angle(\vec{n}, \mathbf{e}_3) = \frac{\langle \vec{n}, \mathbf{e}_3 \rangle}{\|\vec{n}\|} \geq \cos(\tan^{-1}(s)) = \frac{1}{\sqrt{1+s^2}}$$

$$(30b) \quad \iff \frac{1}{\sqrt{u^2 + v^2 + 1}} \geq \frac{1}{\sqrt{1+s^2}} \iff u^2 + v^2 - s^2 \leq 0.$$

Definition 5.1 (surface maximum slope constraint). We call (30) the *surface maximum slope constraint* with maximum slope s . This is a special case of surface orientation constraint where $(\vec{q}, \theta) = (\mathbf{e}_3, \tan^{-1}(s))$ (see Section 1.1).

Using the general model (29), we convert the projection onto surface maximum slope constraint to

$$(31a) \quad \min_{(u,v) \in \mathbb{R}^2} \varphi(u, v) = \frac{1}{2} (u \ v) \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} - (w_a \ w_b) \begin{pmatrix} u \\ v \end{pmatrix}$$

$$(31b) \quad \text{subject to } u^2 + v^2 - s^2 \leq 0,$$

where A, B, C, w_a, w_b are given by (28). Rescaling by $(u, v, w_a, w_b) \leftarrow \left(\frac{u}{s}, \frac{v}{s}, \frac{w_a}{s}, \frac{w_b}{s}\right)$, we obtain an equivalent problem

$$(32a) \quad \min_{(u,v) \in \mathbb{R}^2} \varphi(u, v) = \frac{1}{2} (u \ v) \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} - (w_a \ w_b) \begin{pmatrix} u \\ v \end{pmatrix}$$

$$(32b) \quad \text{subject to } g(u, v) := u^2 + v^2 - 1 \leq 0.$$

This problem can be solved by several ways including numerical methods. For example, (32) is a special case of trust region subproblem which can be solved by means of generalized eigenvalue problems [1].

As this is a projection problem that is needed in iterative optimization methods, it is desirable to have a closed form solution. Thus, in the rest of this section, we will aim to find such solution via Lagrange multipliers, also known as Karush-Kuhn-Tucker (KKT) conditions, see [16, 17] or [6, Theorem 11.5], and Ferrari's method for quartic equations [15].

First, due to Lemma 4.2, (32) is the problem of minimizing a strictly convex quadratic function $\varphi(u, v)$ over a closed, bounded, convex set in \mathbb{R}^2 . Thus, there exists a unique solution. To solve (32), we start by finding the vertex (u_0, v_0) of $\varphi(u, v)$, which is the unique solution of

$$\nabla\varphi(u, v) = \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} - \begin{pmatrix} w_a \\ w_b \end{pmatrix} = 0.$$

Now we check if the vertex (u_0, v_0) is inside or outside the feasible region:

Case 1: $g(u_0, v_0) \leq 0$ (inside). Then (u_0, v_0) is the solution of (32).

Case 2: $g(u_0, v_0) > 0$ (outside). Then $\nabla\varphi(u, v) \neq 0$ for all $g(u, v) \leq 0$. Observe that for each value $\eta \geq 0$, the level set $\varphi(u, v) = \eta$ is an ellipse in \mathbb{R}^2 whose center is the vertex (u_0, v_0) (see Figure 2).

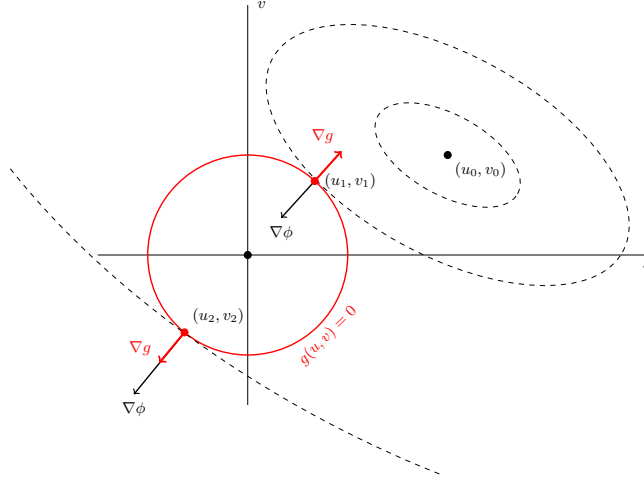


Figure 2: Level sets of $\varphi(u, v)$ vs. feasible region.

Hence, the tangent point of the unit circle $g(u, v) = 0$ and the *smallest* elliptical level set of φ that intersects the circle (see Figure 2), denoted by (u_1, v_1) , is the unique solution of the minimization problem (32); and any tangent point of the unit circle $g(u, v) = 0$ and the *largest* elliptical level set of φ that intersects the circle (see Figure 2), denoted by (u_2, v_2) , is a solution of the maximization problem

$$\max_{(u,v) \in \mathbb{R}^2} \varphi(u, v) \quad \text{subject to} \quad g(u, v) \leq 0.$$

Based on this observation, there exist Lagrange multipliers $\lambda_1 \geq 0$ and $\lambda_2 \leq 0$ such that (u_1, v_1, λ_1) and (u_2, v_2, λ_2) satisfy the Lagrange multipliers system

$$(33a) \quad \nabla\varphi(u, v) + \frac{\lambda}{2}\nabla g(u, v) = 0,$$

$$(33b) \quad u^2 + v^2 - 1 = 0.$$

Note that $\nabla\varphi(u_1, v_1) \neq 0$ and $\nabla\varphi(u_2, v_2) \neq 0$, so we must have $\lambda_1 > 0$ and $\lambda_2 < 0$. Moreover, since the minimization problem has a unique solution, we therefore conclude that *the system (33) must possess a unique solution (u_1, v_1, λ_1) with $\lambda_1 > 0$, and at least a solution (u_2, v_2, λ_2) with $\lambda_2 < 0$.*

In summary, *Case 2* reduces to finding the unique solution (u, v, λ) of (33) with $\lambda > 0$. First, (33a) yields

$$(34) \quad \begin{pmatrix} A + \lambda & C \\ C & B + \lambda \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} w_a \\ w_b \end{pmatrix}.$$

By Lemma 4.2 and the assumption that $\lambda > 0$, the matrix in (34) is positive definite. It follows that (34) has a unique solution

$$(35a) \quad u = \frac{w_a(B + \lambda) - w_b C}{(A + \lambda)(B + \lambda) - C^2} = \frac{\lambda w_a + w_a B - w_b C}{\lambda^2 + (A + B)\lambda + AB - C^2},$$

$$(35b) \quad v = \frac{w_b(A + \lambda) - w_a C}{(A + \lambda)(B + \lambda) - C^2} = \frac{\lambda w_b + w_b A - w_a C}{\lambda^2 + (A + B)\lambda + AB - C^2},$$

Next, substituting u and v into (33b) yields

$$\begin{aligned} [\lambda^2 + (A + B)\lambda + AB - C^2]^2 &= [w_a \lambda + (w_a B - w_b C)]^2 + [w_b \lambda + (w_b A - w_a C)]^2 \\ &= (w_a^2 + w_b^2)\lambda^2 + 2(w_a^2 B + w_b^2 A - 2w_a w_b C)\lambda \\ &\quad + (w_a B - w_b C)^2 + (w_b A - w_a C)^2. \end{aligned}$$

Defining the constants accordingly, we rewrite as

$$(\lambda^2 + R_1 \lambda + R_2)^2 = R_3 \lambda^2 + 2R_4 \lambda + R_5.$$

One can simply solve this equation by the classic Ferrari's method for quartic equations [9]. Nevertheless, since there is a unique positive λ , we will find its explicit formula following Ferrari's technique. We introduce a real variable y

$$(36a) \quad (\lambda^2 + R_1 \lambda + R_2 + y)^2 = R_3 \lambda^2 + 2R_4 \lambda + R_5 + 2(\lambda^2 + R_1 \lambda + R_2)y + y^2$$

$$(36b) \quad = (R_3 + 2y)\lambda^2 + 2(R_4 + R_1 y)\lambda + R_5 + 2R_2 y + y^2.$$

We choose y such that the right hand side is a perfect square in λ . Thus, the right hand side must have zero discriminant

$$\begin{aligned} [R_4 + R_1 y]^2 - (R_3 + 2y)(R_5 + 2R_2 y + y^2) &= 0 \\ \iff -2y^3 + [R_1^2 - R_3 - 4R_2]y^2 + [2R_1 R_4 - 2R_2 R_3 - 2R_5]y + R_4^2 - R_3 R_5 &= 0. \end{aligned}$$

This is a cubic equation in y , so we use Cardano's method [9] to find one real solution y_0 . Then (36) becomes

$$(37) \quad (\lambda^2 + R_1 \lambda + R_2 + y_0)^2 = (R_3 + 2y_0) \left(\lambda + \frac{R_4 + R_1 y_0}{R_3 + 2y_0} \right)^2.$$

As discussed above, this equation must have at least one positive and one negative solutions. Next, we solve for the (unique) positive λ :

Case 2a: $R_3 + 2y_0 < 0$. Then $\lambda = -(R_4 + R_1 y_0)/(R_3 + 2y_0)$ must be the unique solution, which is a contradiction. Thus, this case cannot happen.

Case 2b: $R_3 + 2y_0 = 0$. Then $y_0 = -R_3/2$ and $\lambda^2 + R_1 \lambda + R_2 + y_0 = 0$. So

$$\lambda = \frac{1}{2} \left(-R_1 \pm \sqrt{R_1^2 - 4R_2 + 2R_3} \right).$$

Since there is only one positive λ , we take $\lambda = (-R_1 + \sqrt{R_1^2 - 4R_2 + 2R_3})/2$.

Case 2c: $R_3 + 2y_0 > 0$. Define $r := \sqrt{R_3 + 2y_0}$. Then (37) becomes

$$(\lambda^2 + R_1 \lambda + R_2 + y_0)^2 = \left(\lambda r + \frac{R_4 + R_1 y_0}{r} \right)^2.$$

This leads to two quadratic equations in λ

$$(38) \quad \lambda^2 + (R_1 \pm r) \lambda + (R_2 + y_0 \pm \frac{R_4 + R_1 y_0}{r}) = 0.$$

Now if $R_4 + R_1 y_0 = 0$, then (38) consists of two quadratic equations that have constant term $R_2 + y_0$. Thus, it yield an even number (possibly none) of positive solutions. Therefore, we must have $R_4 + R_1 y_0 \neq 0$. Moreover, we must take the equation with negative constant term. So we set $r \leftarrow r \cdot \text{sgn}(R_4 + R_1 y_0)$ and take only the equation

$$\lambda^2 + (R_1 - r) \lambda + (R_2 + y_0 - \frac{R_4 + R_1 y_0}{r}) = 0.$$

It follows that the positive λ is

$$\lambda = \frac{1}{2} \left(r - R_1 + \sqrt{(r - R_1)^2 - 4(R_2 + y_0 - \frac{R_4 + R_1 y_0}{r})} \right).$$

Next, we obtain u and v from (35) and then rescale variables $(u, v) \leftarrow (su, sv)$. Finally, we obtain (h_1, h_2, h_3) by using (26) and (22).

6 Projections onto Surface Oriented Minimum Slope Constraints

6.1 Motivation from a Nonconvex Constraint

Let $P_1 P_2 P_3$ be a triangle with the normal vector $\vec{n} = (-u, -v, 1)$ as in Section 4.1. In some cases, it is required that the angle $\angle(\vec{n}, \vec{q}) \geq \alpha$ for some given vector \vec{q} and number α . This happens, for example, if $P_1 P_2 P_3$ must have a slope at least $s := s_{\min} \in \mathbb{R}_+$. In this case, the angle between \vec{n} and $\mathbf{e}_3 = (0, 0, 1)$ satisfies

$$(39a) \quad \angle(\vec{n}, \mathbf{e}_3) \geq \tan^{-1}(s)$$

$$(39b) \quad \Leftrightarrow \cos \angle(\vec{n}, \mathbf{e}_3) = \langle \vec{n}, \mathbf{e}_3 \rangle / \|\vec{n}\| \leq \cos(\tan^{-1}(s)) = 1/\sqrt{1+s^2}$$

$$(39c) \quad \Leftrightarrow \frac{1}{\sqrt{u^2 + v^2 + 1}} \leq \frac{1}{\sqrt{1+s^2}} \quad \Leftrightarrow \quad u^2 + v^2 - s^2 \geq 0.$$

We refer to (39) as the *surface minimum slope constraint* with minimum slope s . This is clearly a *non-convex* constraint in (u, v) . Despite the projection onto this constraint is still available, nonconvexity may prevent iterative methods from convergence. It is worth to mention that similar minimum slope constraints are also critical in road design problem [4], which is again nonconvex and thus, somewhat hinders the theoretical analysis. Therefore, we will next present a novel idea to modify this constraint in a way such that minimum slope is preserved.

6.2 The Surface Oriented Minimum Slope Constraint

Condition (39a) implies the angle between \vec{n} and the xy -plane satisfies

$$(40) \quad \angle(\vec{n}, xy\text{-plane}) \leq (\pi/2) - \tan^{-1}(s).$$

In some cases, it is reasonable to align the plane $P_1 P_2 P_3$ toward a given location/direction. For instance, in civil engineering drainage, the designer may want to direct the water to certain drains.

Suppose we want $P_1P_2P_3$ inclined towards a unit direction $\vec{q} = (q_1, q_2, 0) \in \mathbb{R}^3$. Then we can fulfill (40) by requiring $\angle(\vec{n}, \vec{q}) \leq \frac{\pi}{2} - \tan^{-1}(s)$, i.e.,

$$\cos \angle(\vec{n}, \vec{q}) = \langle \vec{n}, \vec{q} \rangle / \|\vec{n}\| \geq \cos \left(\frac{\pi}{2} - \tan^{-1}(s) \right) = s / \sqrt{1 + s^2},$$

Substituting $\vec{n} = (-u, -v, 1)$ and $\vec{q} = (q_1, q_2, 0)$, we obtain $\frac{-q_1u - q_2v}{\sqrt{u^2 + v^2 + 1}} \geq \frac{s}{\sqrt{1 + s^2}}$, which is equivalent to

$$(41) \quad q_1u + q_2v < 0, \quad u^2 + v^2 + 1 - \frac{1+s^2}{s^2}(q_1u + q_2v)^2 \leq 0.$$

Hence, we arrive at the following definition.

Definition 6.1 (surface oriented minimum slope constraint). We call (41) the *surface oriented minimum slope constraint* specified by (\vec{q}, s) , where $\vec{q} = (q_1, q_2, 0) \in \mathbb{R}^3$ is a unit vector and $s \in \mathbb{R}_{++}$ is the minimum slope.

Definition 6.1 is a special case of the *surface orientation constraint* in Section 1.1 where $(\vec{q}, \theta) = (\vec{q}, \frac{\pi}{2} - \tan^{-1}(s))$. It is worth mentioning that the constraint (41) not only guarantees surface minimum slope but also generates a *convex* feasible set.

6.3 The Projection onto (41)

By employing Section 4.2, the projection onto the surface oriented minimum slope constraint is given by the solution to the problem

$$\begin{aligned} \min_{(u,v) \in \mathbb{R}^2} \quad & (a_1u + b_1v + h_3 - w_1)^2 + (a_2u + b_2v + h_3 - w_2)^2 + (h_3 - w_3)^2 \\ \text{subject to} \quad & (41). \end{aligned}$$

Again, we first simplify this problem. Define $Q := \begin{pmatrix} q_1 & q_2 \\ q_2 & -q_1 \end{pmatrix}$ where $(q_1, q_2, 0)$ is the unit direction vector that defines the constraint (41). Then $Q^2 = \text{Id}$, which implies $Q = Q^T = Q^{-1}$. Next, we change variables

$$\begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} := Q \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} q_1u + q_2v \\ q_2u - q_1v \end{pmatrix} \Leftrightarrow \begin{pmatrix} u \\ v \end{pmatrix} = Q \begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} = \begin{pmatrix} q_1\hat{u} + q_2\hat{v} \\ q_2\hat{u} - q_1\hat{v} \end{pmatrix}.$$

Then the second inequality in (41) becomes

$$(q_1\hat{u} + q_2\hat{v})^2 + (q_2\hat{u} - q_1\hat{v})^2 + 1 - \left(1 + \frac{1}{s^2}\right)\hat{u}^2 = \left(-\frac{1}{s^2}\right)\hat{u}^2 + \hat{v}^2 + 1 \leq 0.$$

Thus, (41) becomes

$$(42) \quad \hat{u} < 0, \quad \hat{v}^2 - \frac{\hat{u}^2}{s^2} + 1 \leq 0.$$

Following Section 4.2, we change coefficients (without relabeling)

$$(43) \quad \begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} \leftarrow Q \begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix},$$

and obtain an equivalent problem

$$(44) \quad \min_{(\hat{u}, \hat{v}) \in \mathbb{R}^2} \quad \frac{1}{2} \begin{pmatrix} \hat{u} & \hat{v} \end{pmatrix} \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} - (w_a \quad w_b) \begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} \quad \text{subject to} \quad (42).$$

where A, B, C, w_a, w_b are defined by (28) with the new coefficients a_1, a_2, b_1, b_2 from (43). Next, we change variables by $(u, v, A, B, w_b) \leftarrow \left(\frac{\hat{u}}{s}, \hat{v}, sA, \frac{B}{s}, \frac{w_b}{s} \right)$, then (44) is equivalent to

$$(45a) \quad \min_{(u,v) \in \mathbb{R}^2} \varphi(u, v) = \frac{1}{2} \begin{pmatrix} u & v \end{pmatrix} \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} - \begin{pmatrix} w_a & w_b \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

$$(45b) \quad \text{subject to } g_1(u, v) := u < 0, \quad g_2(u, v) := v^2 - u^2 + 1 \leq 0.$$

Since all matrices in (43) are nonsingular, the new coefficients a_1, a_2, b_1, b_2 satisfy $a_1 b_2 - a_2 b_1 \neq 0$, which implies that $\varphi(u, v)$ is strictly convex by Lemma 4.2. The feasible set (45b) is the left half of a hyperbola, thus, also a convex region (see Figure 3). Therefore, (45) is the problem of minimizing a strictly convex quadratic function over a closed convex region, which must yield a unique solution.

Similar to Section 5, we now will present a way to solve (45) by Lagrange multipliers. First, we find the vertex (u_0, v_0) of $\varphi(u, v)$, which is the unique solution of

$$\nabla \varphi(u, v) = \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} - \begin{pmatrix} w_a \\ w_b \end{pmatrix} = 0.$$

Case 1: (u_0, v_0) is feasible, i.e., $u_0 < 0$ and $g_2(u_0, v_0) \leq 0$. Then clearly (u_0, v_0) is the unique solution of (45).

Case 2: (u_0, v_0) is not feasible. Then the unique solution (u_1, v_1) of (45) is the tangent point of the left branch hyperbola curve $C := \{(u, v) \mid u < 0, v^2 - u^2 + 1 = 0\}$ and the smallest level set of $\varphi(u, v)$ that intersects C , which is an ellipse centered at (u_0, v_0) . Figure 3 illustrates two possible scenarios.

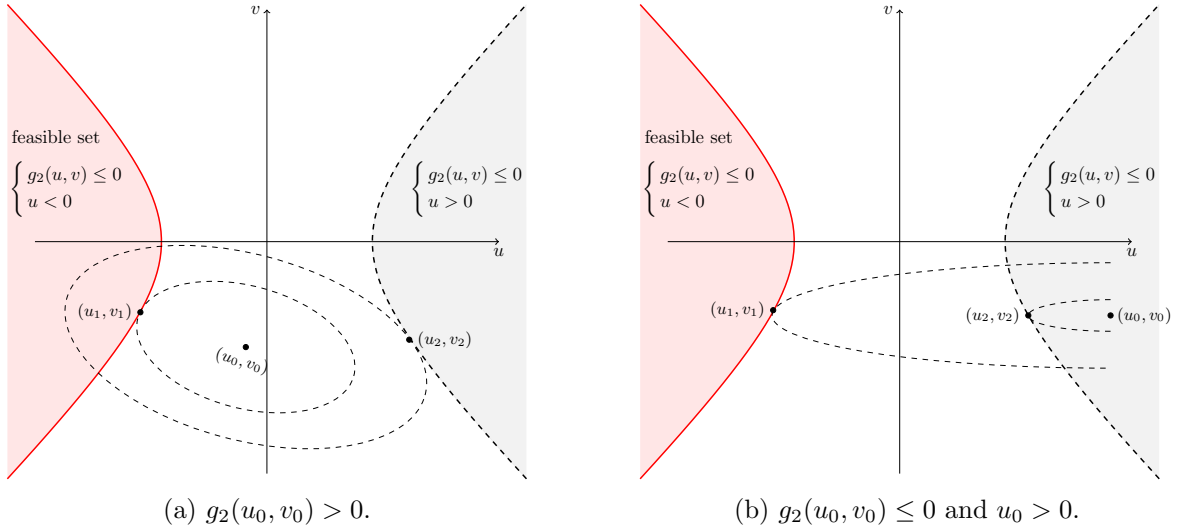


Figure 3: Level sets of $\varphi(u, v)$ vs. feasible region.

In both cases, we see that $g_1(u, v) = u < 0$ is always an inactive constraint. Therefore, the Lagrange multipliers system reduces to

$$(46a) \quad \nabla \varphi(u, v) + \frac{\lambda}{2} \nabla g_2(u, v) = 0,$$

$$(46b) \quad g_2(u, v) = v^2 - u^2 + 1 = 0.$$

Utilizing Figure 3, we conclude the following:

If $g_2(u_0, v_0) > 0$ (see Figure 3a), then (46) must have a unique solution (u_1, v_1, λ_1) with $\lambda_1 > 0$ and $u_1 < 0$, and a unique solution (u_2, v_2, λ_2) with $\lambda_2 > 0$ and $u_2 > 0$.

If $g_2(u_0, v_0) \leq 0$ and $g_1(u_0, v_0) = u_0 > 0$ (see Figure 3b), then (46) must have a unique solution (u_1, v_1, λ_1) with $\lambda_1 > 0$ and $u_1 < 0$. Also, (46) must have at least one solution (u_2, v_2, λ_2) with $\lambda_2 < 0$.

In summary, *Case 2* reduces to finding the solution (u_1, v_1, λ_1) of (46) where $\lambda_1 > 0$ and $u_1 < 0$. It then follows that (u_1, v_1) is the unique solution of (45). First, (46a) is

$$(47) \quad \begin{pmatrix} A - \lambda & C \\ C & B + \lambda \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} w_a \\ w_b \end{pmatrix}.$$

Define

$$\begin{aligned} D &:= (A - \lambda)(B + \lambda) - C^2 = -\lambda^2 + (A - B)\lambda + (AB - C^2), \\ D_u &:= w_a(B + \lambda) - w_b C = \lambda w_a + w_a B - w_b C, \\ D_v &:= w_b(A - \lambda) - w_a C = -\lambda w_b + (w_b A - w_a C). \end{aligned}$$

Suppose $D \neq 0$, then $u = D_u/D$ and $v = D_v/D$. Substituting into (46b) yields

$$\begin{aligned} \left[\lambda^2 + (B - A)\lambda + (C^2 - AB) \right]^2 &= (\lambda w_a + w_a B - w_b C)^2 - [\lambda w_b + (w_a C - w_b A)]^2 \\ &= (w_a^2 - w_b^2) \lambda^2 + 2(w_a^2 B + w_b^2 A - 2w_a w_b C) \lambda \\ &\quad + (w_a B - w_b C)^2 - (w_a C - w_b A)^2. \end{aligned}$$

By defining the constants accordingly, we rewrite the above identity as

$$(\lambda^2 + R_1 \lambda + R_2)^2 = R_3 \lambda^2 + 2R_4 \lambda + R_5.$$

Again, one can analyze this equation analogously to Section 5. However, complication arises since there are possibly more than one positive λ 's. Instead, we expand

$$\lambda^4 + 2R_1 \lambda^3 + (R_1^2 + 2R_2 - R_3) \lambda^2 + 2(R_1 R_2 - R_4) \lambda + R_2^2 - R_5 = 0,$$

and use the classic Ferrari's method for quartic equation. Next, for each $\lambda > 0$, we solve (47) as follows.

Case 2a: $D \neq 0$. Then we simply compute $u = D_u/D$ and $v = D_v/D$.

Case 2b: $D = 0$. If either $D_u \neq 0$ or $D_v \neq 0$, then (47) has no solution (u, v) . So we now consider the remaining case that $D_u = D_v = 0$. Then from (47) and (46b), we have

$$\begin{aligned} Cu + (B + \lambda)v &= w_b, \\ v^2 - u^2 + 1 &= 0. \end{aligned}$$

Since $B > 0$ and $\lambda > 0$, the first equation implies $v = (w_b - Cu)/(B + \lambda)$. So the second one becomes $(w_b - Cu)^2 - (B + \lambda)^2 u^2 + (B + \lambda)^2 = 0$, i.e.,

$$(51) \quad [C^2 - (B + \lambda)^2] u^2 - 2w_b C u + [w_b^2 + (B + \lambda)^2] = 0.$$

If the discriminant $(w_b C)^2 - (C^2 - (B + \lambda)^2)(w_b^2 + (B + \lambda)^2) \geq 0$, then we obtain two solutions u . Since there is a unique pair (u, v) with $u < 0$, the quadratic equation (51) must yield one positive and one negative solutions

$$u = \frac{w_b C \pm \sqrt{(w_b C)^2 - (C^2 - (B + \lambda)^2)(w_b^2 + (B + \lambda)^2)}}{C^2 - (B + \lambda)^2},$$

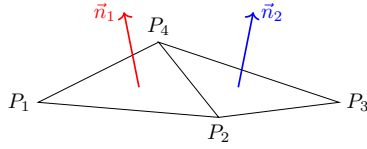
and we must also have $C^2 - (B + \lambda)^2 < 0$. Therefore, the negative solution u is

$$u = \frac{w_b C + \sqrt{(w_b C)^2 - (C^2 - (B + \lambda)^2)(w_b^2 + (B + \lambda)^2)}}{C^2 - (B + \lambda)^2}.$$

Next, among all (u, v) 's, choose the unique pair with $u < 0$. Then rescale variables $u \leftarrow su$. Finally, we obtain (h_1, h_2, h_3) by (26) and (22).

7 Curvature Minimization

In some design problems, the designer may wish to construct a surface that is as “smooth” as possible. This problem is referred to as minimizing the *curvature* between adjacent triangles in the mesh. In this section, we will address this problem.



Given the points $P_i = (p_{i1}, p_{i2}, h_i)$, $i = 1, 2, 3, 4$, so that they form two adjacent triangles $\Delta_1 = P_1 P_2 P_4$ and $\Delta_2 = P_2 P_3 P_4$ (see Figure 4).

Figure 4: Curvature between two triangles.

Define $a_i := p_{i1} - p_{41}$ and $b_i := p_{i2} - p_{42}$, for $i = 1, 2, 3$. Then the respective normal vectors are

$$\begin{aligned} \vec{n}_1 &= \begin{pmatrix} a_1 \\ b_1 \\ h_1 - h_4 \end{pmatrix} \times \begin{pmatrix} a_2 \\ b_2 \\ h_2 - h_4 \end{pmatrix} = \begin{pmatrix} -b_2 h_1 + b_1 h_2 + (b_2 - b_1) h_4 \\ a_2 h_1 - a_1 h_2 + (a_1 - a_2) h_4 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}, \\ \vec{n}_2 &= \begin{pmatrix} a_2 \\ b_2 \\ h_2 - h_4 \end{pmatrix} \times \begin{pmatrix} a_3 \\ b_3 \\ h_3 - h_4 \end{pmatrix} = \begin{pmatrix} -b_3 h_2 + b_2 h_3 + (b_3 - b_2) h_4 \\ a_3 h_2 - a_2 h_3 + (a_2 - a_3) h_4 \\ a_2 b_3 - a_3 b_2 \end{pmatrix}. \end{aligned}$$

We rescale and obtain

$$\begin{aligned} \vec{n}_1 &= \left(\frac{-b_2 h_1 + b_1 h_2 + (b_2 - b_1) h_4}{a_1 b_2 - a_2 b_1}, \frac{a_2 h_1 - a_1 h_2 + (a_1 - a_2) h_4}{a_1 b_2 - a_2 b_1}, 1 \right), \\ \vec{n}_2 &= \left(\frac{-b_3 h_2 + b_2 h_3 + (b_3 - b_2) h_4}{a_2 b_3 - a_3 b_2}, \frac{a_3 h_2 - a_2 h_3 + (a_2 - a_3) h_4}{a_2 b_3 - a_3 b_2}, 1 \right). \end{aligned}$$

The curvature can be represented by the difference between \vec{n}_1 and \vec{n}_2 , i.e.,

$$\vec{\delta}_{12} := \vec{\delta}_{\Delta_1 \Delta_2} = \vec{n}_1 - \vec{n}_2 =: \begin{pmatrix} \langle u, h \rangle \\ \langle v, h \rangle \end{pmatrix},$$

where $h := (h_1, h_2, h_3, h_4)$, $u := (u_1, u_2, u_3, u_4)$, $v := (v_1, v_2, v_3, v_4)$,

$$(52a) \quad u_1 = -b_2 / (a_1 b_2 - a_2 b_1) \quad , \quad u_2 = b_1 / (a_1 b_2 - a_2 b_1) + b_3 / (a_2 b_3 - a_3 b_2) \quad ,$$

$$(52b) \quad u_3 = -b_2 / (a_2 b_3 - a_3 b_2) \quad , \quad u_4 = -(u_1 + u_2 + u_3) \quad ,$$

$$(52c) \quad v_1 = a_2 / (a_1 b_2 - a_2 b_1) \quad , \quad v_2 = -a_1 / (a_1 b_2 - a_2 b_1) - a_3 / (a_2 b_3 - a_3 b_2) \quad ,$$

$$(52d) \quad v_3 = a_2 / (a_2 b_3 - a_3 b_2) \quad , \quad v_4 = -(v_1 + v_2 + v_3).$$

So for each pair (Δ_i, Δ_j) of adjacent triangles, we find the corresponding vectors $h_{ij} = (h_1^{ij}, h_2^{ij}, h_3^{ij}, h_4^{ij}) \in \mathbb{R}^4$, $u_{ij}, v_{ij} \in \mathbb{R}^4$, and compute the corresponding curvature $\vec{\delta}_{ij} = (\langle u_{ij}, h_{ij} \rangle, \langle v_{ij}, h_{ij} \rangle)$. We then aim to minimize all the curvatures between adjacent triangles. Thus, we arrive at the objective

$$G_{1,*}(x) := \sum_{\text{all triangle pairs } (\Delta_i, \Delta_j)} \|\vec{\delta}_{ij}\|_*$$

where $\|\cdot\|_*$ can be either 1-norm or max-norm in \mathbb{R}^2 . For 1-norm, the objective is

$$(53) \quad G_{1,1}(x) = \sum_{\text{all triangle pairs } (\Delta_i, \Delta_j)} |\langle u_{ij}, h_{ij} \rangle| + |\langle v_{ij}, h_{ij} \rangle|.$$

For max-norm, the objective is

$$(54) \quad G_{1,\infty}(x) = \sum_{\text{all triangle pairs } (\Delta_i, \Delta_j)} \max\{|\langle u_{ij}, h_{ij} \rangle|, |\langle v_{ij}, h_{ij} \rangle|\}.$$

Remark 7.1 (simplified computations for symmetric cases). Suppose the two dimensional mesh satisfies the following symmetry: for every adjacent triangles $P_1P_2P_4$ and $P_2P_3P_4$, there exists $t \in \mathbb{R}$ such that

$$\overrightarrow{P_4P_1} + \overrightarrow{P_4P_3} = t\overrightarrow{P_4P_2}.$$

Then it follows that $(a_1, b_1) + (a_3, b_3) = t(a_2, b_2)$. So we can deduce $a_1b_2 - a_2b_1 = a_2b_3 - a_3b_2$. From (52), we have

$$u = \frac{-b_2}{a_1b_2 - a_2b_1}(1, -t, 1, t-2) \quad \text{and} \quad v = \frac{a_2}{a_1b_2 - a_2b_1}(1, -t, 1, t-2),$$

i.e., u and v are parallel. This simplifies the computations for (53) and (54).

Since our optimization methods require proximity operators, we will derive the necessary formulas. It is sometimes convenient to compute the proximity operator Prox_f via the proximity operator of its *Fenchel conjugate* f^* , which is defined by $f^* : X \rightarrow \mathbb{R} : x^* \mapsto \sup_{x \in X} (\langle x^*, x \rangle - f(x))$. Indeed, if $\gamma > 0$, then (see, e.g., [3, Theorem 14.3(ii)])

$$\forall x \in X : \quad x = \text{Prox}_{\gamma f}(x) + \gamma \text{Prox}_{\gamma^{-1}f^*}(\gamma^{-1}x).$$

We also recall a useful formula from [5, Lemma 2.3]: if $f : X \rightarrow \mathbb{R}$ is convex and positively homogeneous, $\alpha > 0$, $w \in X$, and $h : X \rightarrow \mathbb{R} : x \mapsto \alpha f(x - w)$, then

$$\text{Prox}_h(x) = w + \alpha \text{Prox}_f\left(\frac{x - w}{\alpha}\right) = x - \alpha \text{Prox}_{f^*}\left(\frac{x - w}{\alpha}\right).$$

Theorem 7.2. Let $\{u_i\}_{i \in I}$ be a system of finitely many vectors in \mathbb{R}^n , and

$$f : \mathbb{R}^n \rightarrow \mathbb{R} : x \mapsto \max_{i \in I} |\langle u_i, x \rangle|.$$

Then $f^* = \iota_D$ where $D := \text{conv} \bigcup_{i \in I} \{u_i, -u_i\}$. Consequently, $\text{Prox}_f = \text{Id} - P_D$.

Proof. Suppose $x^* \in D$, then we can express

$$x^* = \sum_{i \in I} \lambda_i u_i \quad \text{where} \quad \sum_{i \in I} |\lambda_i| \leq 1.$$

It follows that for all $x \in X$,

$$\langle x^*, x \rangle - f(x) = \sum_{i \in I} \lambda_i \langle u_i, x \rangle - \max_{i \in I} |\langle u_i, x \rangle| \leq \left(\sum_{i \in I} \lambda_i - 1 \right) \max_{i \in I} |\langle u_i, x \rangle| \leq 0.$$

So, $f^*(x^*) = \sup_{x \in X} [\langle x^*, x \rangle - f(x)] \leq 0$. Notice that equality happens if we set $x = 0$. Thus, $f^*(x^*) = 0$.

Now suppose $x^* \notin D$. Since D is nonempty, closed, and convex, the classic separation theorem implies that there exists $x \in X$ such that

$$\langle x^*, x \rangle > \langle u, x \rangle \quad \text{for all } u \in D.$$

This leads to $\langle x^*, x \rangle - f(x) > 0$. Since f is homogeneous, we have

$$f^*(x^*) \geq \langle x^*, \lambda x \rangle - f(\lambda x) \rightarrow +\infty \quad \text{as } \lambda \rightarrow +\infty.$$

So $f^*(x^*) = +\infty$. Therefore, we can conclude that $f^* = \iota_D$. \square

Finally, Examples 7.3 and 7.4 provide the necessary formulas to compute the proximity operators of the objectives in (53) and (54), respectively.

Example 7.3. Given $\alpha > 0$, a vector $u \in \mathbb{R}^n \setminus \{0\}$ and the function

$$f : \mathbb{R}^n \rightarrow \mathbb{R} : x \mapsto \alpha |\langle u, x \rangle|.$$

By Theorem 7.2, the proximity operator of f is

$$\text{Prox}_f = \text{Id} - P_D \quad \text{where } D := [-\alpha u, \alpha u].$$

The explicit projection onto D is given by (see [5, Theorem 2.7])

$$P_D x = \min \left\{ 1, \max \left\{ -1, \frac{\langle \alpha u, x \rangle}{\|\alpha u\|^2} \right\} \right\} \alpha u = \min \left\{ \alpha, \max \left\{ -\alpha, \frac{\langle u, x \rangle}{\|u\|^2} \right\} \right\} u.$$

Example 7.4. Given two vectors $u, v \in \mathbb{R}^n$ and

$$f(x) : \mathbb{R}^n \rightarrow \mathbb{R} : x \mapsto \max\{|\langle u, x \rangle|, |\langle v, x \rangle|\}.$$

By Theorem 7.2, the proximity operator of f is

$$\text{Prox}_f = \text{Id} - P_D \quad \text{where } D := \text{conv}\{u, v, -u, -v\}.$$

In general, Prox_D is the projection onto a parallelogram in \mathbb{R}^n .

8 Experiments

With the formulas for proximity and projection operators, we are ready to apply iterative methods to solve feasibility and optimization problems. In particular, we present an application to the civil engineer problem in Section 1.2 which is part of our motivation.

Experiment setup: In each of the three problems outlined in Figure 1, we aim to minimize the surface curvature using the objective function (53), and subject to the requirements that: the maximum slope of all triangles does not exceed 4%; each triangle must incline toward its closest drain line (marked in

blue) with minimum slope of 0.5%; and the triangle edges (marked in red) must be aligned. The DR algorithm (6) will be used and it will stop when distance between two consecutive *governing* iterations are less than the tolerance $\varepsilon = 0.001$ and the *monitored* iteration meets all constrained with (same) tolerance ε .

Results: In Figures 5, 6, and 7, we show the solutions after various iterations of the DR algorithm. Triangles colored in red violate the design constraints (maximum and/or minimum slopes), whereas triangles in green satisfy the constraints. Below the surfaces are the contours, which become more regular with increasing iterations due to curvature minimization objective.

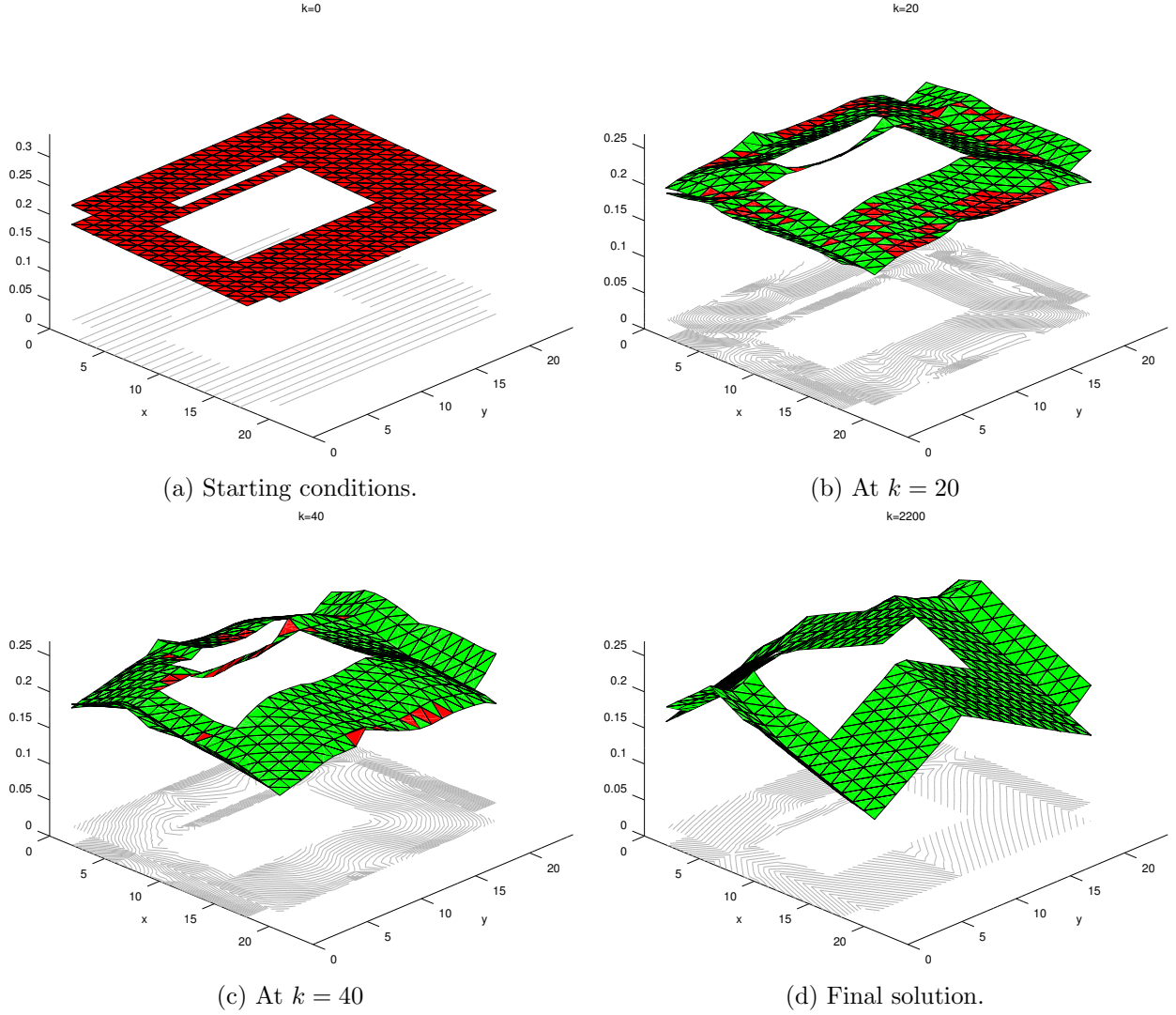


Figure 5: Grading design for a parking lot with corner drainage.

9 Conclusion

The manipulation of triangle meshes has many applications in computer graphics and computer-aided design. The paper presents a general framework for triangular design problems with spatial constraints. In particular, we model several important constraints and costs in suitable forms so that projection

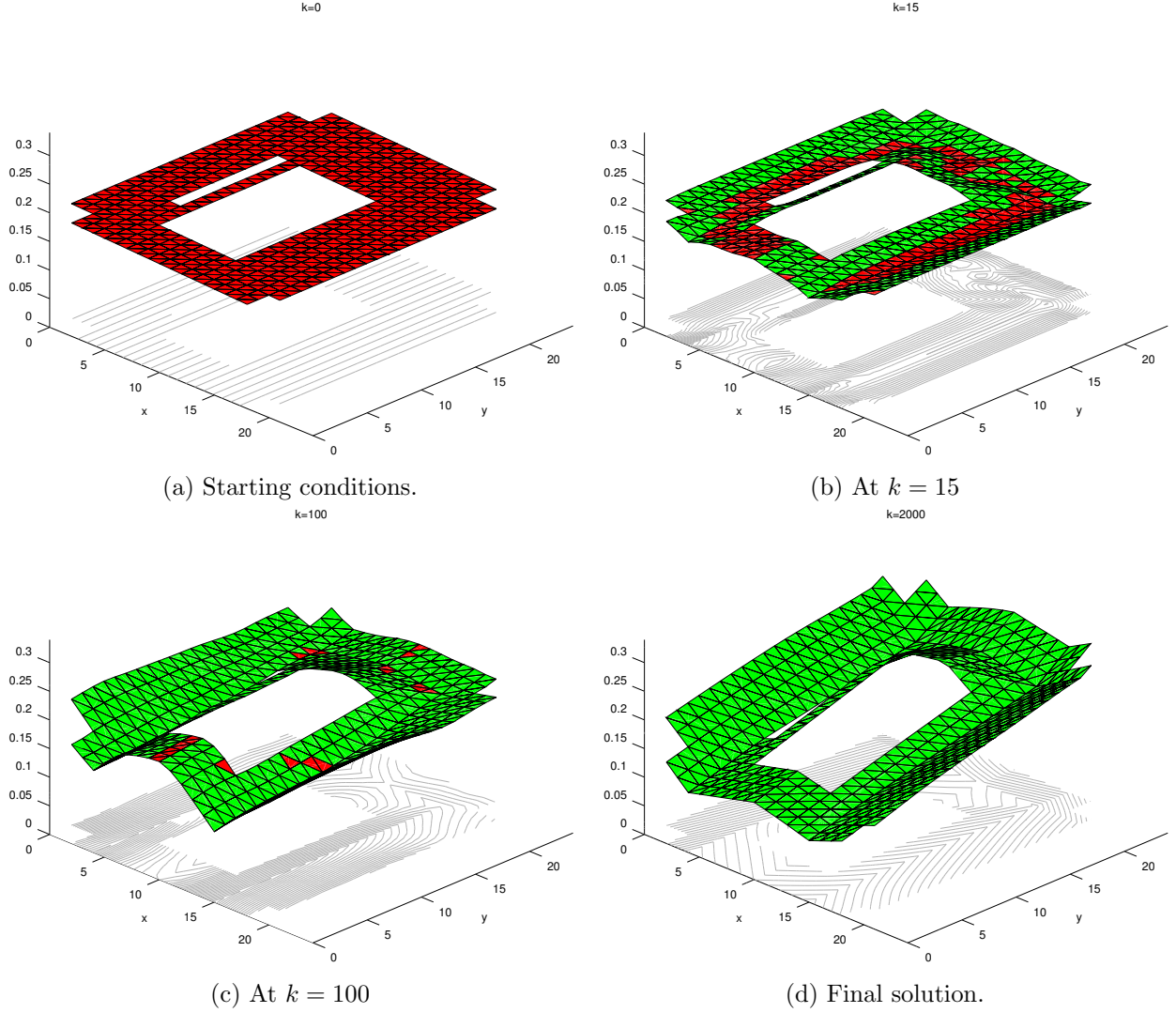


Figure 6: Grading design for a parking lot with side drainage.

and proximity operators can be computed explicitly. With the help of iterative splitting methods, we are able to solve some complex design problems on these triangular meshes. Therefore, modeling constraints and their proximity operators, and using them in modern first-order optimization methods can be a successful approach to solve large-scale problems in industry and science.

Acknowledgement

This research is partially supported by Autodesk, Inc. The authors are grateful to the Editors and two anonymous referees for their constructive suggestions that allow us to improve the original presentation.

References

- [1] S. Adachi, S. Iwata, Y. Nakatsukasa, and A. Takeda, Solving the trust-region subproblem by a generalized eigenvalue problem, *SIAM Journal on Optimization* 27 (2017), 269–291.

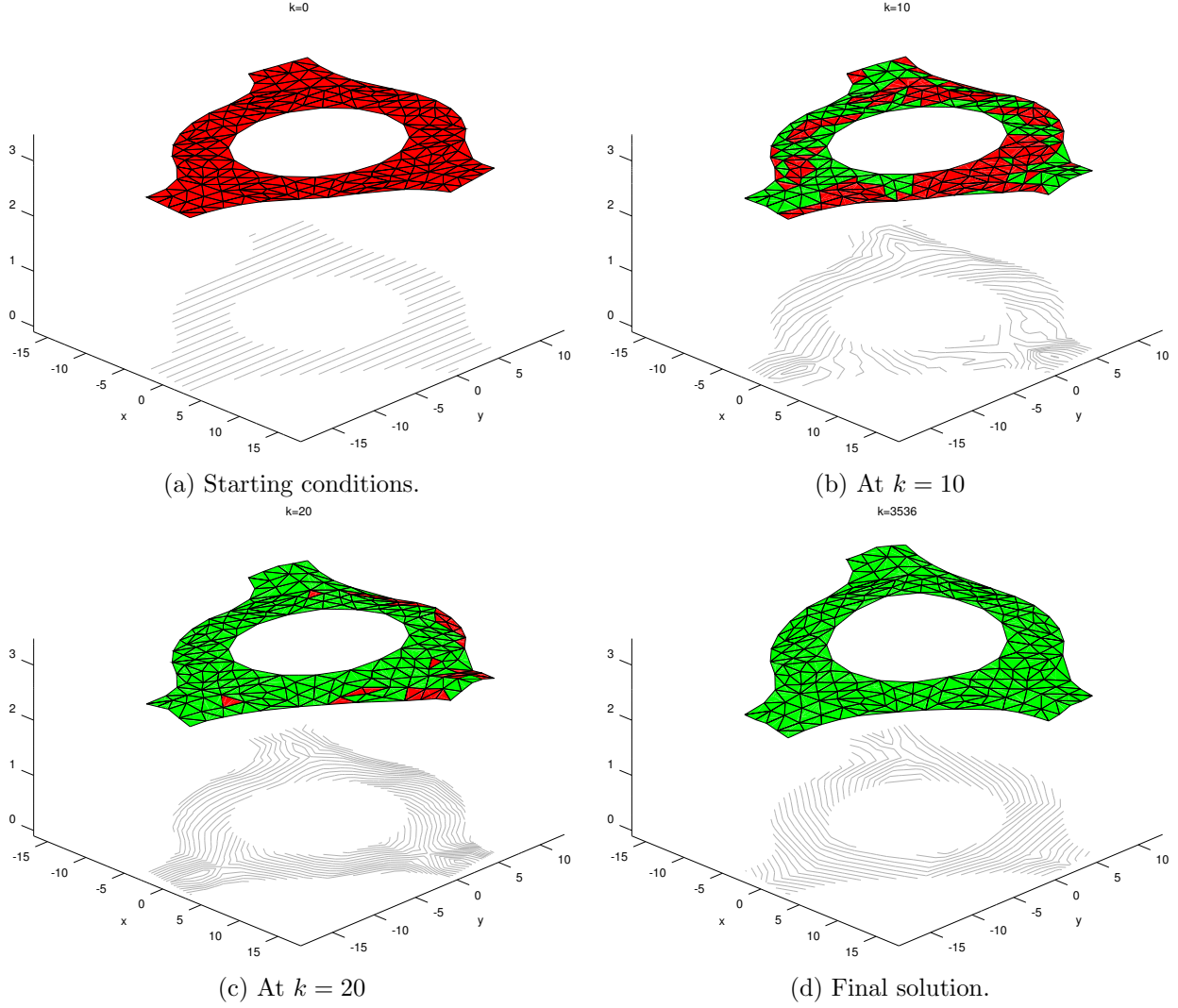


Figure 7: Grading design for a roundabout.

- [2] H.H. Bauschke and J.M. Borwein, On projection algorithms for solving convex feasibility problems, *SIAM Review* 38 (1996), 367–426.
- [3] H.H. Bauschke and P.L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, second edition, CMS Books in Mathematics, Springer, New York (2017).
- [4] H.H. Bauschke and V.R. Koch, Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces, *Infinite products of operators and their applications, Contemporary Mathematics* (2015) 636, 1–40.
- [5] H.H. Bauschke, V.R. Koch, and H.M. Phan, Stadium norm and Douglas-Rachford splitting: a new approach to road design optimization, *Operations Research* 64 (2016), 201–218.
- [6] A. Beck, *Introduction to Nonlinear Optimization: Theory, Algorithms and Applications with Matlab*, MOS-SIAM Series on Optimization 19, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; Mathematical Optimization Society (MOS), Philadelphia, PA (2014).

- [7] R.I. Boţ, E.R. Csetnek, and A. Heinrich, A primal-dual splitting algorithm for finding zeros of sums of maximal monotone operators, *SIAM Journal on Optimization* 23 (2013), 2011–2036.
- [8] L.M. Briceño Arias and P.L. Combettes, A monotone + skew splitting model for composite monotone inclusions in duality, *SIAM Journal on Optimization* 21 (2011), 1230–1250.
- [9] G. Cardano, *The great art or the rules of algebra*, The M.I.T. Press, Cambridge, Massa.-London (1968), translated from the Latin and edited by T. Richard Witmer.
- [10] A. Cegielski, *Iterative methods for fixed point problems in Hilbert spaces*, Lecture Notes in Mathematics 2057, Springer, Heidelberg (2012).
- [11] Y. Censor, W. Chen, P.L. Combettes, R. Davidi, and G.T. Herman, On the effectiveness of projection methods for convex feasibility problems with linear inequality constraints, *Computational Optimization and Applications* 51 (2012), 1065–1088.
- [12] Y. Censor and S.A. Zenios, *Parallel optimization: Theory, algorithms, and applications*, Numerical Mathematics and Scientific Computation, Oxford University Press, New York (1997).
- [13] P.L. Combettes and J.-C. Pesquet, A proximal decomposition method for solving convex variational inverse problems, *Inverse Problems* 24 (2008), 065014 (27pp).
- [14] J. Douglas and H.H. Rachford, On the numerical solution of heat conduction problems in two and three space variables, *Transactions of the American Mathematical Society* 82 (1956), 421–439.
- [15] R. Irving, *Beyond the quadratic formula*, The Mathematical Association of America, Washington DC (2010).
- [16] W. Karush, *Minima of functions of several variables with inequalities as side conditions*, ProQuest LLC, Ann Arbor, MI (1939), Thesis (SM)–The University of Chicago.
- [17] H.W. Kuhn and A.W. Tucker, Nonlinear programming, *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability* (1950), 481–492, University of California Press, Berkeley and Los Angeles (1951).
- [18] P.-L. Lions and B. Mercier, Splitting algorithms for the sum of two nonlinear operators, *SIAM Journal on Numerical Analysis* 16 (1979), 964–979.
- [19] J.-J. Moreau, Proximité et dualité dans un espace hilbertien, *Bulletin de la Société Mathématique de France*, 93 (1965), 273–299.
- [20] R.T. Rockafellar, *Convex analysis*, Princeton University Press, Princeton, NJ, (1970).
- [21] A. Ruszczyński, *Nonlinear optimization*, Princeton University Press, Princeton, NJ (2006).