Surface Fitting and Registration of Point Clouds using Approximations of the Unsigned Distance Function

Simon Flöry and Michael Hofer

Geometric Modeling and Industrial Geometry Research Group, Vienna University of Technology,
Wiedner Hauptstraße 8-10, 1040 Wien, Austria

Abstract

Many problems in computer aided geometric design and geometry processing are stated as least–squares optimizations. Least–squares problems are well studied and widely used but exhibit immanent drawbacks such as high sensitivity to outliers. For this reason, we consider techniques for the registration of point clouds and surface fitting to point sets based on the l_1-norm. We develop algorithms to solve l_1–registration and l_1–fitting problems and explore the emerging non–smooth minimization problems. We describe efficient ways to solve the optimization programs and present results for various applications.

Key words: Curve fitting, Surface fitting, Registration, l_1–Approximation, Optimization, Proximal Bundle Method, Linear Program, Second Order Cone Programming

1. Introduction

Numerous ways of solving problems in geometry processing and computer aided geometric design involve a formulation of the target requirements as solution of an optimization process. Many of these optimizations are phrased in a least–squares sense, that is the l_2–norm $\|\mathbf{r}\|_2 := \sqrt{\sum_i r_i^2}$ of some residue $\mathbf{r} \in \mathbb{R}^p$ to be minimized. Theoretical backing for using least–squares methods is given by the Gauss–Markov theorem which basically states, that if the input data to the problem fulfills some statistical properties, e.g. zero mean, unique variance and zero covariance, the least–squares way is the best (linear) way to go.

However, as praxis has shown, the input data does not always meet these requirements. Real world data poses severe challenges to the least–squares approach. One particular

Email address: floery@geometrie.tuwien.ac.at (Simon Flöry).

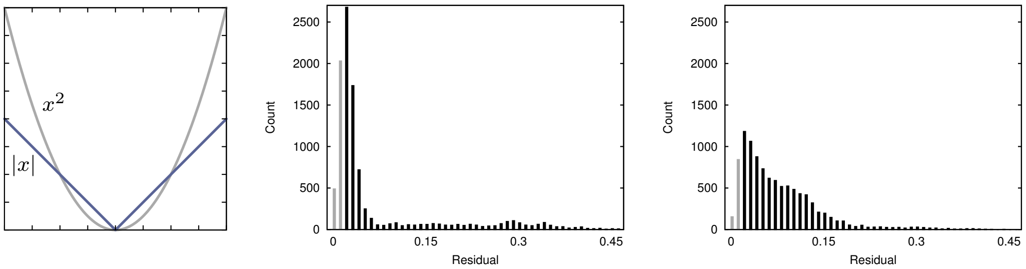

Fig. 1. The l_1–norm weights outliers much less than the l_2–norm. Moreover, it puts more emphasis on small residues (cf. function graph of $f(x) = |x|$ and $f(x) = x^2$, left). Both properties can be seen in typical histograms of residues for l_1– (center) and l_2–approximations (right). These plots were computed on discrete input data. The histograms' two left most bins (marked in grey) are below the average sampling density and bias the analysis. We will discuss the residual distribution in more detail in Sec. 5.

problem is outliers, which naturally occur in various ways in physical measurement processes. Several techniques have been proposed to improve the robustness of l_2–norm optimizations and applications of these methods have found their way into geometry processing.

We choose a different approach and go beyond least–squares. Instead of turning to robust variants of l_2–approximations we choose a norm known to be more robust by itself, the l_1–norm $\|\mathbf{r}\|_1 = \sum_i |r_i|$ (cf. Fig. 1). l_1–techniques are far less popular in CAGD which may be due to the challenges of non–smooth optimization resulting from the absolute values in the definition of $\|.\|_1$. Geometric insights allow us to explore the l_1–approach in an elegant way. In many cases, the least–squares techniques are equivalent to a minimization of the squared distances in the setup. In the l_1–norm, this results in working with (the absolute value of) the signed distance function. We show ways to derive new geometric approximation algorithms from this premise. Please note that distances will still be taken in the Euclidean norm, but instead of squaring the distance values (l_2–norm) we employ the l_1–norm.

1.1. *Related Work*

In this work we will show, how well-known problems such as the registration of two point clouds or the reconstruction of B-spline curves or surfaces from point sets can be stated in an l_1–sense. Accordingly, our survey on related literature concentrates on previous literature on registration and surface reconstruction techniques, improvements to least–squares methods, and rare uses of the l_1–norm.

Curve and Surface Fitting

The approximation of point clouds by B-spline curves (in the plane) or by B-spline surfaces (in space) is a well-known problem in computer-aided design (Dierckx, 1993). Instead of attempting a comprehensive review of related literature, we concentrate on previous work motivating our way of tackling the problem. Inspired by active contour methods in image processing (Blake and Isard, 1998), we iteratively minimize the absolute value of the signed distance function between the point cloud and the approximating shape (see Fig. 2). Previous related work on curve and surface fitting assigns each element of the point cloud the point of the closest distance to the fitting shape (Hoschek and Lasser, 1993). In these so–called foot points, the *squared distance* between curve/surface and data points is approximated. The degree of approximation distinguishes the different methods. Following the naming conventions of (Wang et al., 2006), point-to-point methods simply minimize the distance between data points and foot points (Plass and Stone, 1983; Hoschek, 1988; Hoppe et al., 1994; Goshtasby, 2000; Weiss et al., 2002). Both, point-to-tangent plane methods (Blake and Isard, 1998) and curvature based methods (Wang et al., 2006) significantly improve upon convergence of the point-to-point methods.

Registration

The alignment of two or more point clouds has attracted some attention over the last decades. The vast body of contributions may be organized with respect to the input configuration of the point clouds (none vs. rough initial alignment). Additionally, constraints on the solution space (rigid vs. non–rigid registration) may be used as classification criteria. In this framework, our contribution classifies as a local, rigid method. The de–facto standard in local, rigid least–squares registration is the *Iterative Closest Point* (short ICP) method nearly simultaneously proposed by (Besl and McKay, 1992) and (Chen and Medioni, 1992). For a pair of roughly aligned input point sets, closest point correspondences are established and used to minimize the squared distance between the two shapes. Variants of these methods are widely used nowadays, see (Rusinkiewicz and Levoy, 2001) for a survey. Multiple point sets may be matched pair by pair (Levoy et al., 2000; Bernardini and Rushmeier, 2002), the major drawback being accumulative errors. *Simultaneous* or *multi–view* registration leaves this shortcoming behind by considering all systems to be aligned at the same time and distributing alignment errors equally (Bergevin et al., 1996; Pulli, 1999).

l_1– and Robust l_2–Techniques

Efforts to increase the robustness of least–squares methods are well summarized in (Pighin and Lewis, 2007). Weighted least–squares basically break with the equal variance assumption of the Gauss-Markov theorem. Intuitively, measurements with large errors are considered outliers and weighted weaker. Iterative Reweighted Least Squares (Holland and Welsch, 1977) for example suggest to weight the summands of the current error term with the reciprocal powers of the previous iteration's residuals. (Sharf et al., 2008) rely partially on this method in a surface reconstruction framework based on a space-time model. Apart from weighting the residuals, least–squares have been combined with RANSAC (Fischler and Bolles, 1981). This so called *least median squares* method (Rousseeuw and Leroy, 1987) has been used among others for multi–view registration of range scan images (Masuda and Yokoya, 1995). Related applications of robust statistics to curve and surface reconstructions comprise (Faber and Fisher, 2001) who employ true geometric distances to fit implicite polynomials to point clouds.

The l_1–norm appears early in geometric optimization problems. (Weber, 1909) states the problem of finding the optimal location of a new industrial site with minimal sum of distances to a set of existing sites. The problem was soon traced back to Fermat in the 17^{th} century and is known as the *Fermat-Weber* problem since then. (Weiszfeld, 1937) was first to propose an iterative approach to solve the problem that was thoroughly investigated by (Kuhn, 1973). The solution of the Fermat-Weber problem is usually

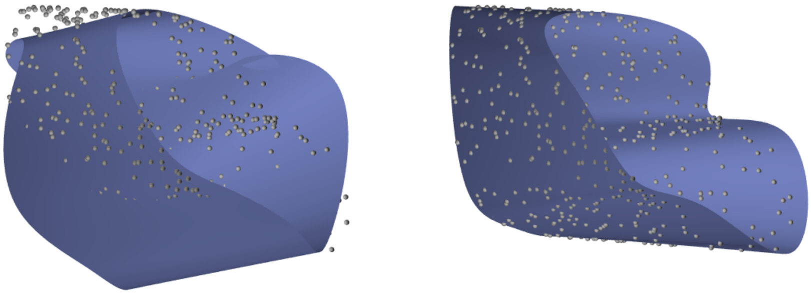

Fig. 2. A surface fitting example. (Left) The initial setup. (Right) The unsigned distance between point cloud (grey) and surface (blue) is iteratively minimized. The approximating surface is deformed until it fits the point cloud.

called the l_1*-median* or *geometric median*. Recently, (Lipman et al., 2007) presented a projection operator for surface reconstruction that resembles locally the l_1-median. In this context, the terminology of *signed* or *unsigned distance function* is frequently used. If a point set's normal field is required to have unique orientation, the distance function is called signed. Otherwise, it is denoted as unsigned. See the discussion in (Alliez et al., 2007) for pointers to recent literature.

The l_1–norm is used infrequently to solve matching problems. One prominent exception is the method of *sum of absolute differences* in translational image registration. (Barnea and Silverman, 1972) discretize the solution space and use the l_1–norm of the difference images as similarity measure. This method is still widely used nowadays in modern video compression software (Richardson, 2003). References to optimization literature covering non–smooth minimization (among others, sums of absolute values) are given in Sec. 4.

The paper is organized as follows. First, we are introducing the curve and surface fitting problem and describe ways to solve it in the l_1–norm. In the following, we consider the registration of two point clouds by minimizing the unsigned distance function between the shapes. Both sections will lead to non–smooth optimization problems that are solved subsequently. We conclude with several examples.

2. Curve and Surface Fitting

In this section, we give a general formulation of the curve and surface fitting problem which we solve subsequently in various ways with l_1–approximation algorithms. Let $P = \{\mathbf{p}_k \in \mathbb{R}^d : k = 0, \dots, n\}$ be an unordered set of points, called the *point cloud* henceforth. Depending on the dimension of the elements of P we distinguish between the curve fitting problem in the plane ($d = 2$) and the surface fitting problem in Euclidean three-space ($d = 3$). It's eligible to ask for a curve fitting in $\mathbb{R}^3$, however, we are not going to address this issue in the following.

For now, let us consider the surface fitting problem, all statements below are easily taken to the two–dimensional curve case. Given a point cloud, we say an approximating surface $\mathbf{x}$ solves the surface fitting problem with respect to an arbitrary distance measure d^* if

$$d^*(\mathbf{x}, P) := \sum_{k=0}^{n} d^*(\mathbf{x}, \mathbf{p}_k) \tag{1}$$

is minimal.

Examples for d^* include the squared distance function $d^* = d^2$ mentioned in the introduction, typically resulting in least–squares problems. In this work, we are going to consider the *unsigned distance function* $|d(\mathbf{x}, \mathbf{p}_k)|$. The unsigned distance function is the absolute value of the *signed distance function* d to $\mathbf{x}$, which is defined as solution of the Eikonal equation $\|\nabla d\| = 1$ with $d(\mathbf{x}) = 0$. For distance minimizing purposes, the use of the absolute value is necessary to ensure a minimum of the distance function to $\mathbf{x}$.

Either distance measure we choose, minimizing Equ. (1) remains a highly non–linear minimization problem. Non–linearity motivates an iterative approach. At each step, we specify locations on $\mathbf{x}$, in which the distance to the point cloud is described and subsequently minimized. The first part, usually termed *parametrization*, computes for each data point $\mathbf{p}_k$ a sample $\mathbf{x}_k$ on $\mathbf{x}$. Most commonly, these samples are chosen to be the points on $\mathbf{x}$ of shortest distance to the elements $\mathbf{p}_k$ of P (Hoschek, 1988). We will call these points on $\mathbf{x}$ the *foot points* henceforth. Computation of a foot point can be easily achieved in a Newton iteration minimizing $\|\mathbf{x}(u_k, v_k) - \mathbf{p}_k\|^2$, given that $\mathbf{x}$ is a parametric surface. The second part, the approximation of the unsigned distance function in $\mathbf{x}_k$, will be the topic of the remainder of this section. At this point, we may give the following rough algorithmic plot of our solution of the surface fitting problem.

Algorithm 1 *For a given point cloud P and initial approximating surface $\mathbf{x}$, a general surface fitting algorithm involves the following steps:*

(i) For each element $\mathbf{p}_k$ of P, obtain the foot point $\mathbf{x}_k$ of the closest distance from $\mathbf{p}_k$ to $\mathbf{x}$.

(ii) In these foot points, approximate the unsigned distance function (w.r.t. the current surface).

(iii) Solve this non–smooth optimization problem and update the surface. If the magnitude of this update is above a certain threshold T_f, continue at step (i).

Fig. 2 shows an application of this algorithm in a surface fitting example.

2.1. *Point Distance Minimization*

Given the foot point $\mathbf{x}_k$ to $\mathbf{p}_k \in P$, a simple way to describe the distance function d would be the point-to-point distance between $\mathbf{p}_k$ and $\mathbf{x}_k$,

$$|d(\mathbf{x}, \mathbf{p}_k)| \approx \|\mathbf{p}_k - \mathbf{x}_k\|. \tag{2}$$

For a fixed surface $\mathbf{x}$, this equation is exact. But in the course of an optimization, the approximating surface will change to better adapt to the geometry of the point cloud. Please note, that with changing $\mathbf{x}$, the foot points of $\mathbf{p}_k$ are likely to change as well. The literature on surface reconstruction tends towards ignoring this dependency. Accordingly, we keep the foot points $\mathbf{x}_k$ fixed over the course of a single iteration step.

Getting more specific about the approximating surface, we define it to be a tensor product B-spline surface $\mathbf{x}(u, v) = \sum_{i=0}^{m} N_i(u, v)\mathbf{d}_i$ with B-spline basis functions $N_i(u, v)$ and control points $\mathbf{d}_i$. The foot point of a data point $\mathbf{p}_k$ is denoted by $\mathbf{x}_k = \mathbf{x}(u_k, v_k)$. The unknowns of the optimization will be the displacement vectors $\delta = (\delta_i)_{i=0,\ldots,m}$ of the B-spline surface control points $\mathbf{d}_i$. We write $\mathbf{x}^\delta(u, v) = \sum_i N_i(u, v)(\mathbf{d}_i + \delta_i)$ for the unknown new surface shape. With this notation in mind, we can rewrite above's approximation of the unsigned distance function and obtain the error term,

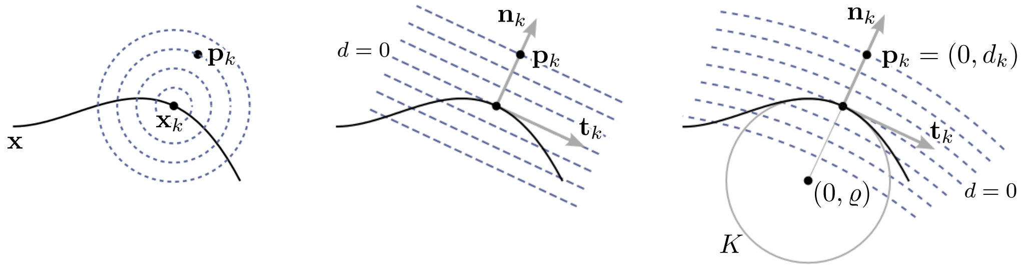

Fig. 3. Level sets of the unsigned distance function approximations to a curve. From left to right: point—to–point distance, point–to–tangent distance and curvature based second order distance minimization.

$$f_{P1}(\mathbf{x}^\delta, P) = \sum_{k=0}^{n} \|\mathbf{p}_k - \mathbf{x}_k^\delta\|. \tag{3}$$

The argument of the norm depends linearly on the updates δ to the control points of the fitting surface. Thus, f_{P1} is convex and continuous but not differentiable everywhere. At the zeros of each of its summands, it is only C^0. A minimization based on this error term poses a non–smooth optimization problem. The level sets of this approximation are concentric spheres centered at $\mathbf{x}_k$, an illustration for a planar curve is shown in Fig. 3, left.

2.2. *Tangent Distance Minimization*

Going one step further, linearization of d in $\mathbf{x}_k$ gives,

$$d(\mathbf{x}, \mathbf{p}_k) \approx d(\mathbf{x}_k) + \nabla d(\mathbf{x}_k)^T \cdot (\mathbf{p}_k - \mathbf{x}_k). \tag{4}$$

As d is the distance to the surface, $d(\mathbf{x}_k) = 0$ holds. Furthermore, due to $\|\nabla d\| = 1$ by definition, we set $\nabla d(\mathbf{x}_k)$ to be the unit normal $\mathbf{n}_k$ of $\mathbf{x}$ in the foot point $\mathbf{x}_k$. All together, this yields a new error term,

$$f_{T1}(\mathbf{x}^\delta, P) = \sum_{k=0}^{n} \left|\mathbf{n}_k^T \cdot (\mathbf{p}_k - \mathbf{x}_k^\delta)\right|, \tag{5}$$

which describes the distance of $\mathbf{p}_k$ to the tangent plane of $\mathbf{x}$ in $\mathbf{x}_k$.

As the sign of the tangent plane distance depends on which side of the tangent plane the point $\mathbf{p}_k$ is lying (more precisely speaking, on the orientation of $\mathbf{n}_k$), we sum up over the absolute values of this approximation of the distance function. The argument of the (convex) absolute value function is linear in δ and thus f_{T1} is convex as well. The absolute value function is not differentiable for points on the tangent plane. Thus, the summands of Equ. (5) are only C^0 there and we obtain another non–smooth optimization problem. The level sets of this approximation of d are planes parallel to the tangent plane (cf. Fig 3, middle).

Considering a deforming fitting surface, an approximation based on the tangent plane involves more information about the surface geometry and promises better results than a simple point-to-point measure. However, in areas of significant curvature, the tangent plane is still a poor approximation of the surface. The following section tries to overcome this drawback.

2.3. *Second Order Distance Minimization*

In the following discussion of a second order approximant of the signed distance function we are going to address the planar curve case first. Following that, we will generalize the two–dimensional results to surfaces in three dimensions. A quadratic approximation of the squared distance function has been presented in (Pottmann and Hofer, 2003) and we follow a similar approach here.

Consider the local Frenet frame F spanned by the tangent $\mathbf{t}_k = \dot{\mathbf{x}}(u_k)$ and the normal $\mathbf{n}_k$ in a point $\mathbf{x}_k$ of a C^2-curve $\mathbf{x}$ (cf. Fig. 3, right). As above, we assume that $\mathbf{x}_k$ is the foot point of the shortest distance from the curve to a data point $\mathbf{p}_k$ and we write points $\mathbf{y}$ in this local coordinate system in the form (y_0, y_1) where y_0 denotes the coordinate value with respect to the tangent and y_1 that in direction of the normal. In this context, the center of the osculating circle K with radius $\varrho = 1/\kappa$ has coordinates $(0, \varrho)$. Furthermore, $\mathbf{x}_k = (0, 0)$ and $\mathbf{p}_k = (0, d)$ hold, where d is the signed distance from $\mathbf{p}_k$ to $\mathbf{x}_k$ (see Fig. 3, right). For now, we choose a local orientation of the curve such that $\varrho < 0$.

Lemma 1 *Let* $\mathbf{x}$ *be a* C^2*-curve in* $\mathbb{R}^2$ *and* d, ϱ, $\mathbf{y} = (y_0, y_1)$ *as defined above. Then,*

$$d(\mathbf{x}, \mathbf{y}) \approx d_c(\mathbf{y}) = \frac{1}{2(d - \varrho)} y_0^2 + y_1$$

is a second order approximation of the signed distance to $\mathbf{x}$.

Proof: A proof is achieved in two steps. First, we show that it is sufficient to consider the distance to the osculating circle. Second, we derive above expression for d_c. The first step is done with a result about the geometry of offset curves: For any offset curve $\mathbf{x}^+$ to $\mathbf{x}$ the center of curvature for a point $\mathbf{x}_k^+ \in \mathbf{x}^+$ on $\mathbf{n}_k$ coincides with the center of the osculating circle K. Therefore, the distance functions to $\mathbf{x}$ and to K agree up to second order. For the second part of the proof, we observe that the distance for any point $\mathbf{y}$ to K is given by

$$d(K, \mathbf{y}) = \sqrt{y_0^2 + (y_1 - \varrho)^2} - |\varrho|.$$

For $\varrho < 0$ locally, $d < \varrho < 0$ is impossible (otherwise, the center of K would not be the center of curvature). Thus, second order Taylor approximation of $d(K, \mathbf{y})$ gives above formula.□

The coordinate in direction of the tangent enters above approximant squared. However, the coordinate in direction of the normal contributes its sign. This is what we expect for a signed distance function approximant where the sign of the distance depends on the orientation of the normal. As the factor $\frac{1}{2(d-\varrho)}$ is always greater than zero, the level sets of d_c are concave parabolas (see Fig. 3, right). For the special cases $d = \varrho$, $d = \infty$, $\varrho = -\infty$, d_c reduces to the point-to-tangent distance given by y_1.

Let $\mathbf{x}$ now be a C^2-surface with unit normal vector $\mathbf{n}_k$ in the foot point $\mathbf{x}_k$. We choose $\mathbf{n}_k$ as third coordinate axes in a right-handed coordinate systems F. The remaining two directions of F are given by $\mathbf{k}^0$ and $\mathbf{k}^1$, the two principal directions to the principal curvatures κ_0 and κ_1. In umbilical points, we choose any two orthogonal tangential directions such that F is right-handed.

Lemma 2 *For a C^2-surface $\mathbf{x}$, $\varrho_i = 1/\kappa_i$ the principal curvature radii and d the signed distance from $\mathbf{p}_k$ to its foot point $\mathbf{x}_k$ on $\mathbf{x}$,*

$$d(\mathbf{x}, \mathbf{y}) \approx d_s(\mathbf{y}) = \frac{1}{2(d - \varrho_0)} y_0^2 + \frac{1}{2(d - \varrho_1)} y_1^2 + y_2,$$

yields a second order approximant of the signed distance function to $\mathbf{x}$.

Proof: As for the planar curve case, we begin by showing that it is enough to compute the distance to a second order approximation of $\mathbf{x}$ in $\mathbf{x}_k$. As the theory on the curvature of offset surfaces provides, the signed distance to $\mathbf{x}$ is, up to second order, identical to the distance to tori T_i, obtained by either rotating the first principal curvature circle around the axis of the second one or vice versa. Which torus we finally choose doesn't matter and two different ways may complete this proof. First, we could compute a second order Taylor approximation of a point's distance to T_i, which, however, involves some effort. Or, we obtain the derivatives for the Taylor approximation according to the argumentation of (Pottmann and Hofer, 2003), who reduce the computation of the partial derivatives to the planar curve case.□

Similar to the planar curve case, the tangential coordinates of a point $\mathbf{y}$ appear squared in d_s. The coordinate in normal direction is signed. As the factors of the squares in d_s are greater than zero, the level sets of d_s are elliptic paraboloids.

So far, all computations for a second order approximant happened in a local Frenet frame F. For using above results, we need to convert to a global coordinate system, which can be done according to

$$y_{0,k} = (\mathbf{p}_k - \mathbf{x}_k^\delta)^T \cdot \mathbf{k}_k^0, \qquad y_{1,k} = (\mathbf{p}_k - \mathbf{x}_k^\delta)^T \cdot \mathbf{k}_k^1, \qquad y_{2,k} = (\mathbf{p}_k - \mathbf{x}_k^\delta)^T \cdot \mathbf{n}_k.$$

With this conversion in mind, we can finally write down the error term for surface fitting based on a second order approximation of the signed distance function,

$$f_{S1}(\mathbf{x}^\delta, P) = \sum_{k=0}^{n} |\frac{1}{2(d_k - \varrho_k^0)} [(\mathbf{p}_k - \mathbf{x}_k^\delta)^T \cdot \mathbf{k}_k^0]^2 + \\ \frac{1}{2(d_k - \varrho_k^1)} [(\mathbf{p}_k - \mathbf{x}_k^\delta)^T \cdot \mathbf{k}_k^1]^2 + (\mathbf{p}_k - \mathbf{x}_k^\delta)^T \cdot \mathbf{n}_k|. \quad (6)$$

We already pointed out, in the computation of d_s, that this approximant may become negative. Thus, in order to achieve a minimum of f_{S1}, we need to take the absolute value of the single summands. We end up with a non–smooth and non-convex optimization problem.

2.4. *Relation to Least–Squares Approximations*

The concepts that lead to the above three different approximations of the *unsigned* distance function are naturally related to similar efforts in the computation of approximants of the *squared* distance function. The idea of point-to-point distance minimization was the basis to some of the first solutions of the fitting problem (Hoschek, 1988). Due to its simplicity, this method has been popular ever since. The point-to-tangent distance technique was proposed by (Blake and Isard, 1998) as an improvement to the slow convergence of the point-to-point method. Finally, curvature based curve fitting has been studied by (Wang et al., 2006).

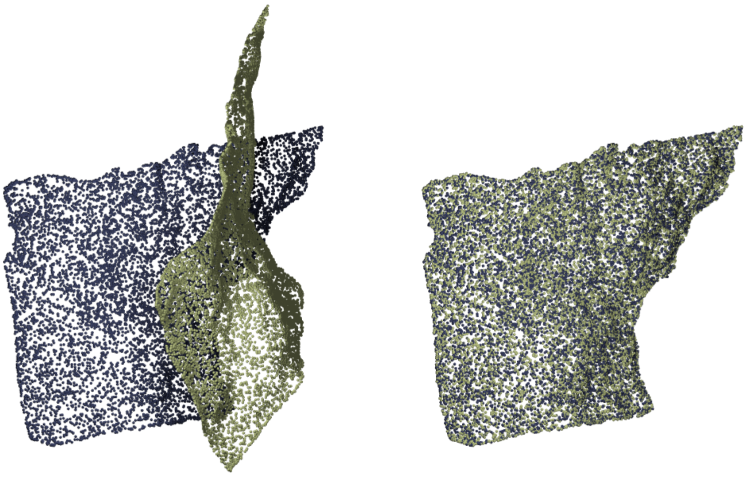

Fig. 4. Local, rigid registration of two point clouds: starting from the initial setup (left), the final alignment (right) is obtained as solution of a non–smooth minimization problem.

3. Registration

Another prominent example of least–squares methods in geometry processing is registration. The acquisition of coordinate samples on an object's surface (e.g. by 3D laser scanning) may produce data from different view points, necessary to capture the whole of the object. The alignment of these data sets, given in different coordinate systems, in a single global coordinate system is probably the best known application of registration.

When it comes to registration, we distinguish between several types: *global* registration aligns the data roughly. For this purpose, most of the existing literature identifies sets of features on the data that are matched subsequently. *Local* registration on the other hand assumes that some global alignment has already happened and improves the positioning of the data to a final spatial position. While we call the registration of two data sets a *pairwise* registration, *simultaneous* or *multi–view* registration considers the alignment of several scans at the same time. Please note that some authors refer to this latter task also as global registration.

In this section we are going to examine, in how far the idea of l_1–norm distance minimization may be applied to the *local* registration problem. In the course of discussion we will first address pairwise alignments. We consider the alignment of three-dimensional point clouds only. All results can easily be adapted to the planar case.

Let $X = \{\mathbf{x}_i \in \mathbb{R}^3 : i = 0, \ldots, m\}$ and $Y = \{\mathbf{y}_i \in \mathbb{R}^3 : i = 0, \ldots, n\}$ be two three dimensional point clouds. We call X the *fixed* or *target* system to which the *moving* system Y shall be registered. In our considerations, registration will ask for that rigid body motion α that minimizes some distance measure d^* between X and Y,

$$d^*(X, \alpha(Y)) := \sum_{i=0}^{n} d^*(X, \alpha(\mathbf{y}_i)). \tag{7}$$

Again we choose d^* to be the *unsigned* distance function $|d|$. Non-rigid registration is not the scope of this work.

Minimization of Equ. (7) poses a non–linear optimization problem. We are going to solve it iteratively. Motivated by the framework of (Besl and McKay, 1992) and (Chen and Medioni, 1992), we assign each element of the moving system a corresponding point in the fixed system. For our purposes, the closest point relation will serve as correspondence. In the closest neighbor in the fixed system, we approximate the signed distance between the two shapes and minimize it.

Algorithm 2 *A general registration algorithm to align two shapes X and Y locally involves the following steps:*

- *(i) For each element $\mathbf{y}_i$ of the moving point cloud Y, compute the foot point $\mathbf{x}_i$ of the shortest distance between $\mathbf{y}_i$ and X.*
- *(ii) In these foot points $\{\mathbf{x}_i\}$, compute an approximation of d.*
- *(iii) Obtain α by minimizing this non–smooth optimization problem and update the position of Y. If the distance between the two point clouds is above a certain threshold T_r, continue at step (i).*

Besides a description of ways to approximate d, which will take most of the remainder of this section, we require a method to describe the unknown rigid body motion α. Basically, any affine motion $A\mathbf{y}+\mathbf{b}$, $A \in \mathbb{R}^{3\times 3}$ and $\mathbf{b} \in \mathbb{R}^3$, can be identified as a point in a 12-dimensional affine space $\mathbb{R}^{12}$. This affine space includes the rigid body motions (A orthogonal) as a 6–dimensional manifold M_6. We could let the entries of A and $\mathbf{b}$ enter the optimization as unknowns, constrained to meet the rigid body motion requirements. However, we prefer to linearize the motion α,

$$\alpha(\mathbf{y}) \approx \mathbf{m}(\mathbf{y}) = \mathbf{y} + \mathbf{v}(\mathbf{y}),$$

and we solve the registration problem for $\mathbf{v}$ (Pottmann et al., 2004). The linear velocity vector field $\mathbf{v}$ of α is of the form

$$\mathbf{v}(\mathbf{y}) = \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{y},$$

with $\bar{\mathbf{c}}, \mathbf{c} \in \mathbb{R}^3$ (Pottmann and Wallner, 2001). Hence, our minimization problem will be of dimension 6,

$$\min_{(\bar{\mathbf{c}},\mathbf{c})} \sum_{i=0}^{n} |d(X, \mathbf{y}_i + \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{y}_i)|$$

with $\bar{\mathbf{c}}, \mathbf{c}$ as unknowns. Note that $\mathbf{m}(\mathbf{y}) = \mathbf{y} + \mathbf{v}(\mathbf{y})$ is an affine but not a Euclidean motion. We can either project the affine motion $\mathbf{m}(\mathbf{y})$ onto M_6 (Pottmann and Wallner, 2001) or solve the rigid registration problem with known correspondences

$$\min_{\alpha} d^2(\mathbf{m}(Y), \alpha(Y))$$

as proposed in (Botsch et al., 2006). As we expect only small displacements in our local registration, we apply the first method.

At this point, we have accomplished the theoretical framework for our solution of the registration problem. What remains to be done is the description and discussion of approximations of the signed distance function. From an abstract point of view, the fitting problem and the registration problem show some similarities. Considering the fitting problem, we set up pairs of corresponding points (each data point gets a foot point assigned) and we describe the distance for such a pair in terms of the unknown displacement of the fitting surface. For registration, we establish correspondences between the elements of the data sets X and Y and approximate distances between these closest points, unknown with respect to the linear velocity vector field of some $\alpha \in M^6$. Therefore, it will come as no surprise that the basic ideas of the following approximations are related to the curve and surface fitting error terms.

3.1. *Point Distance minimization*

Let us start by considering two corresponding points, a moving point $\mathbf{m}(\mathbf{y}_i) = \mathbf{y}_i + \mathbf{v}(\mathbf{y}_i)$ and $\mathbf{x}_i$, the closest point of $\mathbf{y}_i$ in X. Without taking any further geometric information about the neighborhoods of either $\mathbf{x}_i$ or $\mathbf{y}_i$ into account, we approximate the unsigned distance function by

$$|d(X, \alpha(\mathbf{y}_i))| \approx \|\mathbf{x}_i - \mathbf{m}(\mathbf{y}_i)\|.$$

Based upon this term, we define an approximate unsigned distance between X and Y as sum over all these point-to-point distances,

$$r_{P1}(X, \alpha(Y)) = \sum_{i=0}^{n} \|\mathbf{x}_i - \mathbf{m}(\mathbf{y}_i)\|.$$

Comparing this error term to the point distance minimization error term of Sec. 2.1, we see the close resemblance between the two solutions. This similarity also holds for the analytic characteristics of r_{P1}- It is C^0 at the zeros of the terms $\mathbf{x}_i - \mathbf{m}(\mathbf{y}_i)$, that are linear w.r.t. the unknowns $(\bar{\mathbf{c}}, \mathbf{c})$. An optimization relying on this error term will pose a non–smooth optimization problem.

3.2. *Tangent Distance Minimization*

Instead of minimizing the distance between $\mathbf{m}(\mathbf{y}_i)$ and $\mathbf{x}_i$ we could derive a local approximation of X around $\mathbf{x}_i$ and minimize the unsigned distance from $\mathbf{m}(\mathbf{y}_i)$ to this local approximation. A simple way to describe the target point cloud in a neighborhood of $\mathbf{x}_i$ would be the tangent plane of X in $\mathbf{x}_i$. Tangent planes for discrete point sets are a well surveyed topic (see e.g. (Mitra and Nguyen, 2003)) and we obtain them by fitting a plane to the points in a fixed radius neighborhood around $\mathbf{x}_i$. For $\mathbf{y}_i \in Y$, let $\mathbf{x}_i$ be the closest point in X and $\mathbf{n}_i$ an estimated normal in $\mathbf{x}_i$. Then,

$$|d(X, \alpha(\mathbf{y}_i))| \approx |\mathbf{n}_i^T \cdot (\mathbf{x}_i - \mathbf{m}(\mathbf{y}_i))|,$$

approximates the signed distance from moving $\mathbf{y}_i$ to X by a distance–to–tangent plane measure. Similar to the fitting error term f_{T1}, the above approximation can be seen as a linear approximation of the unsigned distance function. As the distance to a plane is signed (the sign depends on the—for our purposes arbitrary—orientation of the normal), we take its absolute value. By summing up, we obtain another approximation of the unsigned distance between X and Y,

$$r_{T1}(X, \alpha(Y)) = \sum_{i=0}^{n} \left|\mathbf{n}_i^T \cdot (\mathbf{x}_i - \mathbf{m}(\mathbf{y}_i))\right|.$$

As already pointed out for the related error term in surface fitting, the absolute value function is not differentiable at the zero of its argument; as a result we will face a non–smooth convex optimization problem.

3.3. *Second Order Distance Minimization*

Our discussion of the fitting problem has shown that we can do better in describing the surface around $\mathbf{x}_i$. This was achieved by including curvature information. For point sets, numerous methods for estimating curvature exist. We choose to fit a surface to a neighborhood of $\mathbf{x}_i$ (Yang and Lee, 1999). For such a local surface patch, we compute a curvature based approximant of the signed distance function according to Lemma 2. Summing up the single second order distance approximations yields, with the same notation as in Sec. 2.3,

$$r_{S1}(X, \alpha(Y)) = \sum_{i=0}^{n} |\frac{1}{2(d_i - \varrho_i^0)}[(\mathbf{x}_i - \mathbf{m}(\mathbf{y}_i))^T \cdot \mathbf{k}_i^0]^2 + \\ \frac{1}{2(d_i - \varrho_i^1)}[(\mathbf{x}_i - \mathbf{m}(\mathbf{y}_i))^T \cdot \mathbf{k}_i^1]^2 + (\mathbf{x}_i - \mathbf{m}(\mathbf{y}_i))^T \cdot \mathbf{n}_i|, \quad (8)$$

as approximation of the distance between target and moving system. This objective states a non–smooth and non–convex optimization problem.

3.4. *Relation to Least–Squares Approximations*

All three distance approximations above do not only relate closely to the fitting error terms of Sec. 2. Their basic ideas are well-known in registration literature, however in least–squares contexts. The first contributions to the registration problem exhibit the first two approaches presented here. (Besl and McKay, 1992) proposed a point–to–point registration while (Chen and Medioni, 1992) discuss a point–to–tangent plane approach in their work. Second order approximation of the squared distance function and the use of spatial kinematics for registration were introduced by (Pottmann et al., 2004).

Based on the systematic similarity between least–squares approaches and our work, an extension from pairwise to multi–view registration is straightforward. In rounding up our discussion of signed distance registration, consider multiple given systems X_i. We choose X_0 as fixed system and describe with

$$\mathbf{v}^{i0}(\mathbf{x}) = \bar{\mathbf{c}}^i + \mathbf{c}^i \times \mathbf{x},$$

the motion of the i-th system towards X_0. As before, $\bar{\mathbf{c}}^i$ and $\mathbf{c}^i$ are the unknowns of our optimization problem. We can express the relative velocity between two moving systems X_j and X_k with,

$$\mathbf{v}^{jk} = \mathbf{v}^{j0} - \mathbf{v}^{k0}.$$

Let $\mathbf{x} \in X_j$ be the closest point of $\mathbf{y} \in X_k$, $k \neq j$. Considering X_j temporarily fixed, we describe the distance between $\mathbf{x}$ and $\mathbf{y} + \mathbf{v}^{jk}(\mathbf{y})$ with any of the three methods above. Summation over all distance approximations yields the non–smooth objective function of the multi–view registration problem which bears $\bar{\mathbf{c}}^i$ and $\mathbf{c}^i$ as solution.

4. Optimization

We have seen how unsigned distance fitting and registration problems lead to non–smooth optimization problems. In this section, we briefly sketch the solution of such minimizations and discuss three non–smooth optimization methods.

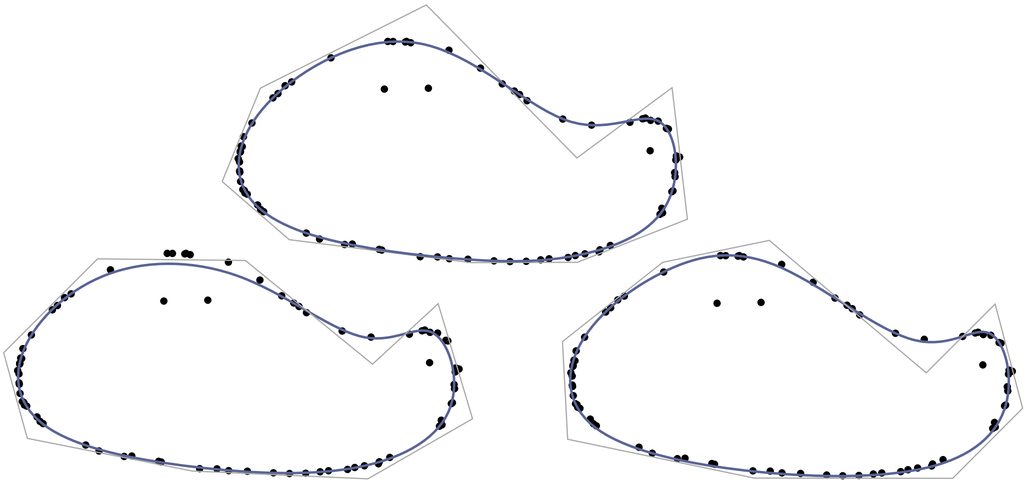

Fig. 5. l_1–fitting is not sensitive to outliers (top). A standard least–squares fitting is biased (bottom, left), whereas a robust variant shows better results (bottom, right). However, the robust version's parameter setup is not straightforward. We also show the control polygons (grey) of the approximating cubic B-spline curves.

To ease the following discussion, we change to a different notation for our problems' objective functions. Please note, that we are going to leave the previous meanings of the variables behind. By reviewing the results of Sec. 2 and Sec. 3 we see that we basically face three types of functions to minimize. For $\mathbf{x} \in \mathbb{R}^n$ as unknown, the point-to-point distance error terms for both fitting and registration give the first type of objective functions, $\sum_i \|A_i\mathbf{x}+\mathbf{b}_i\|$. The tangent distance terms belong to the second class of target functions, $\sum_i |\mathbf{b}_i^T \cdot \mathbf{x}+c_i|$. Finally sums of the form $\sum_i |\mathbf{x}^T A_i\mathbf{x}+\mathbf{b}_i^T \cdot \mathbf{x}+c_i|$ originate from fittings and registrations based on the third class of objectives, namely the second order signed distance approximations. As we have already stated in the previous sections, the first two types are convex while the third is not.

4.1. *Proximal Bundle Method*

In smooth optimization theory, the objective function is differentiable at least once and information about the first or higher order derivatives enter the optimization process. In the case of a non–smooth optimization problem, the objective function is not differentiable everywhere and an adapted idea of gradients is introduced. Non–smooth optimization algorithms such as the *proximal bundle method* (Kiwiel, 1990) build upon this concept.

For now, let f denote the convex non–smooth objective function of the minimization problem $\min_{\mathbf{x}\in\mathbb{R}^n} f$. We call $\mathbf{s} \in \mathbb{R}^n$ a *sub-gradient* of f in $\mathbf{x}$, if

$$f(\mathbf{z}) \geq f(\mathbf{x}) + \mathbf{s}^T \cdot (\mathbf{z} - \mathbf{x}), \quad \forall \mathbf{z} \in \mathbb{R}^n.$$

The set of all sub–gradients in a point $\mathbf{x}$ is called the *sub–differential* $\partial f(\mathbf{x})$ of f in $\mathbf{x}$. In any point x where f is differentiable, $\partial f(\mathbf{x})$ comprises only a single vector, namely the gradient $\nabla f(\mathbf{x})$. It is common to derive a general theory of convex derivatives with help

$\Vert \text{arg} \Vert$	$\Vert A\mathbf{x}+\mathbf{b}\Vert$	$\vert\mathbf{b}^T\cdot\mathbf{x}+c\vert$	$\vert\mathbf{x}^TA\mathbf{x}+2\mathbf{b}^T\cdot\mathbf{x}+c\vert$	$\vert \text{arg} \vert$
$\text{arg}=\mathbf{0}$	$\{\mathbf{s}\in\mathbb{R}^n:\Vert A\mathbf{z}+\mathbf{b}\Vert\geq\mathbf{s}^TA^{-1}(A\mathbf{z}+\mathbf{b}),\ \forall\mathbf{z}\in\mathbb{R}^n\}$	$\text{conv}\{\pm\mathbf{b}\}$	$\text{conv}\{\pm 2(A\mathbf{x}+\mathbf{b})\}$	$\text{arg}=0$
$\text{arg}\neq\mathbf{0}$	$A^T(A\mathbf{x}+\mathbf{b})/\Vert A\mathbf{x}+\mathbf{b}\Vert$	$-\mathbf{b}$	$(-2)(A\mathbf{x}+\mathbf{b})$	$\text{arg}<0$
		$\mathbf{b}$	$2(A\mathbf{x}+\mathbf{b})$	$\text{arg}>0$

Table 1
Sub–differentials $\partial f(\mathbf{x})$ for the objectives of curve/surface fitting and registration. conv$S$ denotes the set of convex combinations of elements of S.

of $\partial f(\mathbf{x})$, see e.g. (Rockafellar, 1972). Table 1 summarizes the sub-gradients of the three types of objective functions introduced above.

In a point $\mathbf{x}$, each sub-gradient $\mathbf{s}$ supports a hyperplane $h=\{\mathbf{z}\in\mathbb{R}^n:\ f(\mathbf{x})+\mathbf{s}^T\cdot(\mathbf{z}-\mathbf{x})\}$, a linear and lower bound approximation to f in $\mathbf{x}$. Proximal bundle methods make iteratively extensive use of these linearization. Assuming that $\mathbf{x}^0$ is an arbitrary starting point, typical iterations for these algorithms are of the form,

$$\mathbf{x}^{k+1}=\arg\min\hat{f}(\mathbf{x})+\frac{1}{2t_k}\Vert\mathbf{x}-\mathbf{x}^k\Vert^2. \tag{9}$$

Here, $\hat{f}$ denotes a linear approximation of f (the *cutting plane model*) that aggregates knowledge from a bundle of previous sub–gradients and corresponding hyperplanes, respectively. The second term controls the maximal allowed step–width (proximity) in an iteration, as the minimum of $\hat{f}$ may be unbounded. The approximate objective function of Equ. (9) yields a quadratic optimization problem with linear constraints that (or its dual) can be solved with methods of smooth optimization theory (Nocedal and Wright, 1999).

The challenges in realizations of proximal bundle methods are found to be rules to maintain and update the bundle of sub–gradients defining $\hat{f}$ and the proximity parameter t_k. It can be shown that these methods give for convex f an ε–optimal global minimizer $\mathbf{x}^*$, this means that $f(\mathbf{x}^*)\leq f(\mathbf{z})+\varepsilon$, $\forall\mathbf{z}\in\mathbb{R}^n$, and user–defined ε (Kiwiel, 1990). For non–convex problems, we can expect to find a local minimum close to $\mathbf{x}^0$.

4.2. *Linear Programs and Second-Order Cone Programming*

For the two convex objectives we can get more specific. Instead of applying a general but approximative solver such as the proximal bundle method, our goal will be to turn the non–smooth problems into smooth optimization problems. This will happen at cost of an increase of dimension and the inclusion of constraints.

Let us first consider the tangent distance minimization term. There exists a well-known reformulation of sums over absolute values of a scalar–valued linear function (see e.g. (Boyd and Vandenberghe, 2004)), that is

$$\min_{\mathbf{x}}\sum_{i=0}^{m}\vert\mathbf{b}_i^T\cdot\mathbf{x}+\mathbf{c}_i\vert \quad\Longleftrightarrow\quad \begin{aligned}&\min_{\mathbf{z}=(\mathbf{x},\mathbf{y})}\sum_{i=0}^{m}y_i\\ &\text{subject to }\ -y_i\leq\mathbf{b}_i^T\cdot\mathbf{x}+\mathbf{c}_i\leq y_i,\quad\forall i=0,\dots,m\end{aligned} \tag{10}$$

where $\mathbf{y}=(y_i)\in\mathbb{R}^{m+1}$ are new auxiliary variables that, added to $\mathbf{x}\in\mathbb{R}^n$, increase the dimension of the optimization problem to $n+m+1$. What we get in return is a linear

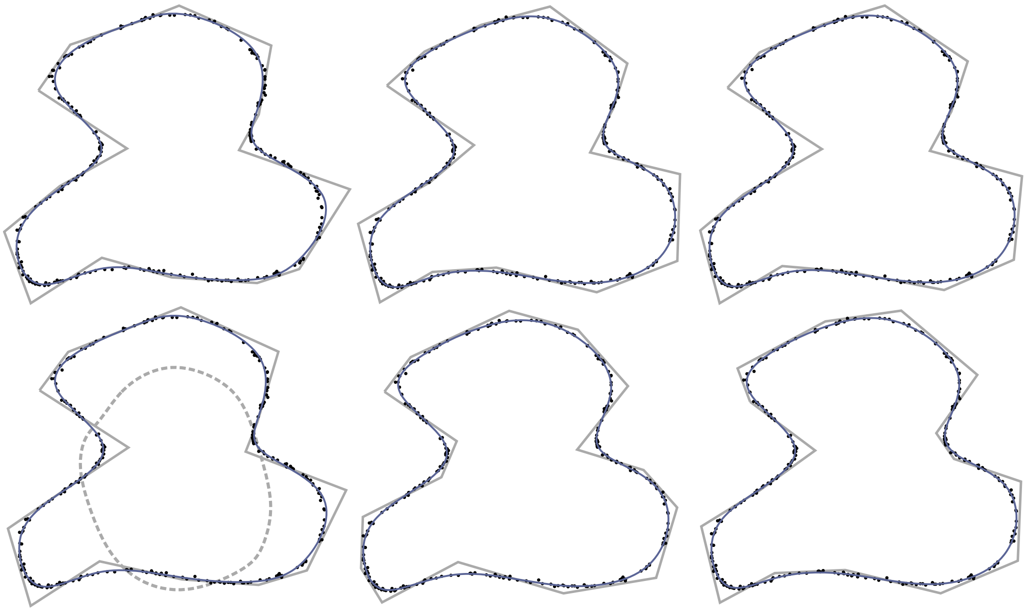

Fig. 6. Top row, from left to right: l_1–fitting with error terms f_{P1}, f_{T1} and f_{S1}. The bottom row shows the least–squares counterparts. The initial approximating curve is depicted bottom left as dashed curve. The results are of comparable quality. The point–to–point methods show less tangential displacements of control points (grey polygon).

program, that can be solved with a variety of algorithms, e.g. the well-known Simplex method or Interior–Point methods (Nocedal and Wright, 1999).

It is obvious to apply above conversion to the remaining two types of objective functions. For the first problem, the norm of a vector–valued linear function, we obtain

$$\min_{\mathbf{x}} \sum_{i=0}^{m} \|A_i\mathbf{x} + \mathbf{b}_i\| \qquad \Longleftrightarrow \qquad \begin{aligned} &\min_{\mathbf{z}=(\mathbf{x},\mathbf{y})} \sum_{i=0}^{m} y_i \\ &\text{subject to } \|A_i\mathbf{x} + \mathbf{b}_i\| \leq y_i, \quad \forall i = 0, \ldots, m. \end{aligned} \tag{11}$$

Please note, that norms are by definition always positive and we can omit the lower bound. We see, that we still get a linear objective function. However, the constraints are not linear but quadratic and the theory of linear programs does not apply.

Instead, this kind of problem is a *second order cone programming* problem. In cone programming, a linear function is minimized over the intersection of an affine set and a Cartesian product of cone constraints. For our needs, the unit second–order cone,

$$C_0 = \{(\mathbf{x}, t) : \mathbf{x} \in \mathbb{R}^p, t \in \mathbb{R}, \|\mathbf{x}\| \leq t\},$$

is well suitable. C_0 plays an important role in wide fields of second-order cone programming.

Intersection of the standard cone with hyperplanes yields conic sections; this indicates why second-order cone programming (short SOCP) is suitable for linear programs with convex quadratic constraints. Note as well that by choosing $\mathbb{R}^p_+$ as cone, we can express any standard linear program as cone programming problem. Cone programming is therefore some kind of generalization of linear programming.

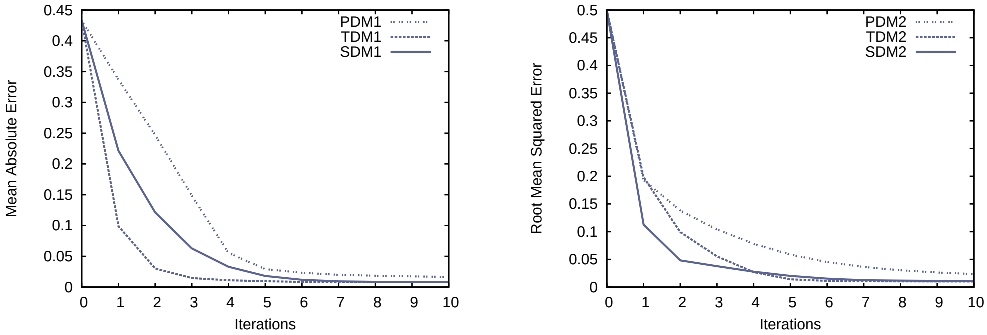

Fig. 7. For l_1–fitting (left), the point–to–tangent error terms performs best. The point–to–point error term is slowest while the non–convex curvature based error term performs moderate. For least–squares based fitting algorithms, the curvature based error term is convex and performs best (right).

By applying an affine mapping to C_0 we obtain as general form of a SOCP problem

$$\min_{\mathbf{z}} \mathbf{g}^T \cdot \mathbf{z}$$
$$\text{subject to } \|A_i\mathbf{z} + \mathbf{b}_i\| \leq \mathbf{c}_i^T \cdot \mathbf{z} + \mathbf{d}_i, \quad \forall i = 0, \ldots, m.$$

Equ. (11) matches this structure without further modification. For the solution of SOCP problems, primal–dual interior–point methods are widely employed. A detailed description of these algorithms is out of scope of this work and we remain with references to the literature (Nesterov and Nemirovskii, 1994; Lobo et al., 1998; Alizadeh and Goldfarb, 2003).

For the third, non-convex problem a similar conversion into a SOCP problem does not work as the lower bound quadratic constraints are non–convex. We have to remain with the proximal bundle method for finding a minimum. As final note in this discussion of solvers for non–smooth minimization problems we want to mention that it is straightforward to add further linear constraints to either the proximal bundle method, linear programs or SOCP problems.

5. Results and Discussion

This section presents several applications of the above optimization problems. According to the layout of the discussion, we are going to show results for curve and surface fitting first. Examples for the registration of point clouds will follow subsequently. In the first part on point cloud approximations, we will focus on curve fitting results. Planar examples let us compare the quality of different approximations much easier.

In Sec. 2, we have derived three different objectives to fit a B-spline curve or surface to a point cloud. Optimization based solely on these terms yields mathematically correct—though visually not pleasing—solutions. Typically, we observe strong oscillations. In order to tackle this issue, a weighted *regularization* or *smoothing* term is added to the fitting objective,

$$\min_{\delta} f(\mathbf{x}^{\delta}, P) + \lambda \cdot s(\mathbf{x}^{\delta}).$$

We call λ the *smoothing weight* and will give details on its choice in Table 2. In l_2–approximation, a discretized version of a measure for the bending energy,

Example	1	2	3	4	5
n	300	300	$75 + 3$	300	$500/1000 + 4$
dim	2×20	2×12	2×13	2×18	$3 \times 2 \times 11/3 \times 19 \times 19$
it	all 20	all 20	all 40	all 20	10/10/20
T (s)	2.08/1.7/34.15	1.43/61.75	0.2/0.2/1.61	1.8/1.8	5.28
	0.01/0.03/1.06	1.15/1.85			209.67 (l_1)/19.86 (l_2)
λ	1/0.1/0.1	1/1	10^{-4}/*/0.1	$10^{-4}/10^{-4}$	10^{-3}
	$10^{-3}/10^{-4}/10^{-4}$	0.04/0.04			10^{-4} (l_1)/10^{-3} (l_2)

Table 2
Configuration parameters and running times for all curve and surface fitting examples (slashes "/" separating different methods, as described in the text). For point clouds with n elements, an optimization problem of dimension *dim* was solved in *it* iterations, taking T seconds. The smoothing weight was initialized to λ and halved at each of the first 10 iterations. Smoothing weight $\lambda = *$ for the robust least–squares method in Example 3 was set manually twice during optimization.

$$s_2(\mathbf{x}^\delta) = \int_{\mathbf{x}} \|\ddot{\mathbf{x}}^\delta\|^2$$

is widely used, cf. (Wang et al., 2006). In our l_1–optimization framework, we refrain from using the squared norm of the second derivatives and base our regularization on the un–squared norm instead,

$$s_1(\mathbf{x}^\delta) = \int_{\mathbf{x}} \|\ddot{\mathbf{x}}^\delta\|.$$

An approximative computation of this term with, for example, the trapezoidal rule integrates straightforward into our non–smooth optimization framework.

Besides the smoothing weights, Table 2 and Table 3 list further characteristics of the presented examples. All code was written in C++ and run on a *Intel Core2 Duo CPU T7700 2.4 GHz* computer. As general non–smooth solver we employed a relaxed proximal bundle method (Huebner and Tichatschke, 2008; Huebner, 2009). For the minimization of linear programs we used the GNU Linear Programming Kit GLPK (GNU Project, 2009) and CVXOPT to solve second order cone programs (Boyd and Vandenberghe, 2004; Dahl and Vandenberghe, 2009). All these solvers are freely available under the terms of the GNU General Public License (GPL).

Example 1 The first example employs all three fitting terms to approximate a point cloud with a cubic B-spline curve. The results are visualized and compared to the l_2–approximations in Fig. 6. The convergence speed of either method is depicted in Fig. 7. For the convergence plots of the l_1–methods, the approximation error at each iteration step is described by the *mean absolute error*,

$$\frac{1}{n+1} \sum_{k=0}^{n} \|\mathbf{p}_k - \mathbf{x}_k\|.$$

For the l_2–fittings, the *root mean squared error*

$$\sqrt{\frac{1}{n+1} \sum_{k=0}^{n} \|\mathbf{p}_k - \mathbf{x}_k\|^2}.$$

is computed after each iteration step.

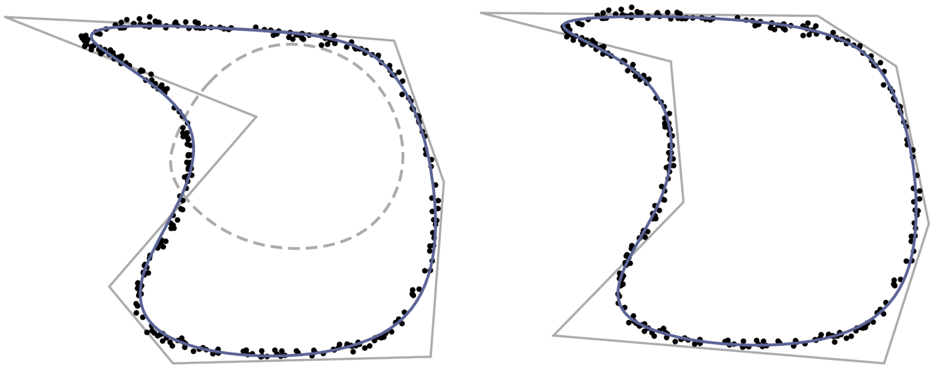

Fig. 8. The non–smooth objectives of point–to–point (left) and point–to–tangent (right) approximation can be solved exactly. The latter method's tangential displacement for control points lets the curve fit the point cloud much faster than the point–to–point objective. The initial curve (dashed, grey) is shown in the left figure.

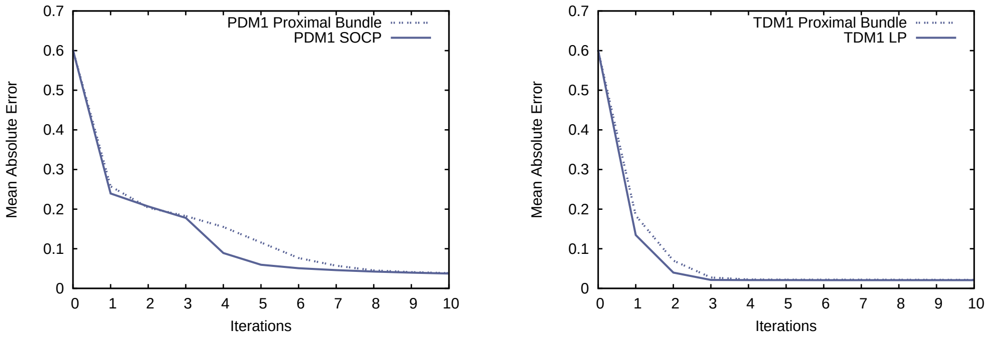

Fig. 9. Both the point–to–point (left) and point–to–tangent plane (right) minimizations may be turned into smooth though constrained optimization problems. These may be solved exactly. An exact solution yields better convergence but requires more computational time.

For the l_1–approximations, the tangent distance minimization method performs best. It outperforms the curvature based method, a non-convex optimization problem with local minima. This first round of l_1–examples was obtained by employing the general proximal bundle method. Comparison to the l_2–methods shows similar convergence behavior. The point–to–point methods are slowest in either scenario. The final approximations show that some control points lack tangential displacement. For this example as well as for all the remaining examples, the fitting problem was initialized manually.

Example 2 In Sec. 4 we have seen that both the point–to–point and the point–to–tangent objectives can be converted from an unconstrained non–smooth optimization problem into a constrained smooth minimization. Opposed to the approximative general solver for non–smooth systems, the constrained smooth programs may be minimized exactly (and thus converge faster at global scope). Fig. 9 confirms our expectations. The improved convergence comes at the price of longer per iteration running times (cf. Table 2). While the increase in computational cost for solving the linear program is moderate (by a factor of 1.6), the point–to–point error term's second order cone program is expensive to minimize. The convergence plot for the constrained smooth point–to–

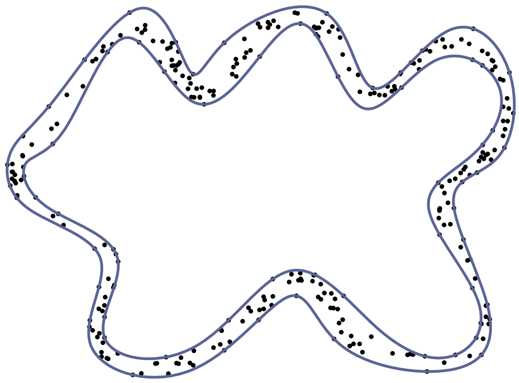

Fig. 10. One-sided point cloud approximations simplify the smooth point–to–tangent optimization.

point optimization shows a kink where the approximating curve started to adapt to the point cloud's upper left bulge. The point–to–tangent approximation fits the point set's shape much faster (cf. Fig. 8).

Example 3 Our main motivation to study curve and surface fitting in the l_1–norm has been the sensitivity of the l_2–norm to outliers. In Fig. 5, three outliers were added to the target point cloud. The l_1–norm approximation is not affected visibly by the additional noise in the data. However, the l_2–fitting is biased significantly. We tested a modified, more robust l_2–error term on the same data set. After some initial unweighted iterations, each point's error term gets weighted by the inverse square of the point's distance to the curve at the beginning of the iteration (Holland and Welsch, 1977),

$$\min_{\delta} \sum_{i=0}^{n} \frac{1}{\|\mathbf{x}_k - \mathbf{p}_k\|^2} f(\mathbf{x}^\delta, \mathbf{p}_k) + \lambda \cdot s_2(\mathbf{x}^\delta).$$

As the final fitting indicates, this modified version is more robust w.r.t. outliers. However, we found that the optimization gets unstable once the weights are added. In particular, the smoothing weight requires adaption to maintain the balance between the new fitting errors and the current regularization impact. Changing the smoothing weight turned out to be a troublesome task.

Example 4 In Sec. 4 we noted that additional linear side conditions may be added to the optimization problem. For curve fitting, one–sided approximations are of certain interest (Flöry, 2009). A closed curve $\mathbf{x}$ is said to approximate the outer boundary of a point cloud P, if it minimizes the distance to P and if P lies in the interior of $\mathbf{x}$. The latter requirement yields linear constraints, one for each data point $\mathbf{p}_k \in P$. Let $\mathbf{x}_k$ be the foot point of $\mathbf{p}_k$ on the approximating curve $\mathbf{x}$ and $\mathbf{n}_k$ the outward oriented normal of $\mathbf{x}$ in $\mathbf{x}_k$. Then,

$$\mathbf{n}_k^T \cdot (\mathbf{p}_k - \mathbf{x}_k^\delta) \leq 0, \qquad \forall k = 0, \ldots, n$$

constrains the fitting to approximate an outer boundary of the target point cloud. Inner boundary reconstructions are obtained by changing the sign of the normal.

Comparing the outer boundary linear constraints to the constraints of the smooth point–to–tangent optimization in Equ. (10), we see that the constraints differ only in the lower bounds. Hence, the one–sided fitting constraints combine well with the point–to–tangent plane error term in its smooth formulation. It replaces half the variable bounds

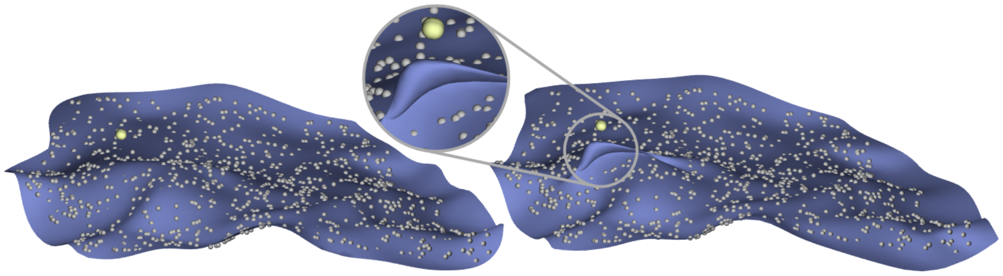

Fig. 11. Surface fitting in the l_1–norm (left) is more robust than least–squares methods (right). The outliers (depicted in yellow) bias the l_2–result.

Example	6	7	8	9
m, n	10k, 10k	100, $100 + 3/29$k, 27k	10k, 10k	1k, 1k
it	all 20	30/10	all 40	all 10
T (s)	53.65/6.87/48.49	0.54 (l_1) / ε (l_2)	13.18 (l_1)	1.45 (l_1 constr.)
	all ε	9.88 (l_1) / ε (l_2)	ε (l_2)	3.97 (l_1 unconstr.)
				0.38 (l_2 constr.)

Table 3
Configuration parameters and running times for the registration examples. A moving point cloud with n elements was aligned to a target system counting m points. The optimization lasted *it* iterations and took T seconds. A running time of ε indicates that the computational cost was below 0.01 seconds.

by constant bounds and thus *simplifies* the linear program. An examination of the computational costs for the result of Fig. 10 revealed slightly shorter running times.

Example 5 So far, we have only presented results for curve fitting. All variants of the fitting algorithm discussed above may be applied to surface approximation without any modification. Fig. 2 shows a reconstruction example with a B-spline surface closed in one parameter direction. Fig. 11 visualizes the robustness of l_1–fittings with respect to outliers.

Example 6 We continue our result review with registration examples (cf. Table 3). At first, we examine registration with all three error terms in either the l_1 or the l_2–norm. We refrain from presenting the whole three times two matrix of results but give a representative final alignment for an l_1 point–to–tangent plane optimization (cf. Fig. 4). The convergence plots of Fig. 12 indicate that the point–to–point minimizations perform worst. For both the l_1– and l_2–registrations, the curvature based term outperforms the tangential term slightly. While this is to be expected for the least–squares approach, the non–convex curvature based l_1–optimization problem may get stuck in local minima. Regarding the computational cost, unconstrained l_2–registration requires a 6×6 system of linear equations to be solved. Hence, we regard the computational effort negligible. This does not hold for the non–smooth objectives which require some effort to be solved with a proximal bundle method. Please note that the point–to–point as well as the point–to–tangent plane error term may be turned into a constrained smooth optimization problem just as outlined above.

Example 7 For curve and surface fitting we have seen, that the robustness of l_1–optimization improves the results in the presence of outliers significantly. For registration,

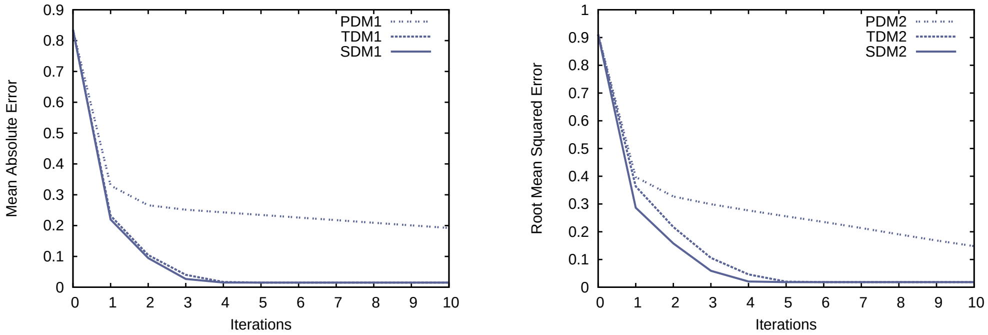

Fig. 12. For registration in the l_1–norm (left) and the l_2–norm (right), the point–to-point error term performs worst. The curvature based technique is slightly faster than the point–to–tangent plane alternative. This must not be expected for the l_1–registration as it solves non–convex optimization problems subsequently.

we want to repeat those experiments. In Fig. 13, we compare the performance of a planar l_1–registration to that of a least–squares alignment. The moving point cloud comprises three significant outliers. Apart from those, the problem would be a zero residual problem. In Fig. 14, we show a robust non–smooth registration opposed to a least–squares alignment. The data of this spatial example was obtained with a stereo and active illumination based 3D scanner, capable of acquiring 17 frames per second (Weise et al., 2007). The high frame rate reduces the quality of the coordinate samples and outliers are very common. We triangulated the data for better visualization. Both tests confirm the robustness of the l_1–methods.

Example 8 This robustness of l_1–optimization is due to a lesser weighting of samples with large residues. This is best seen in Fig. 1 (right) in a comparison of the absolute value function $f(x) = c_0|x|$ and a parabola $f(x) = c_1x^2$. From the graphs we can immediately deduce another property. The l_1–norm considers small residues stronger than the l_2–norm. These two properties are confirmed by histograms of a l_1– and a l_2–registration in Fig. 1 (center and right). The discrete nature of the aligned shapes renders the histogram unreliable for residues smaller than half the average sampling density. For this reason, the corresponding bins are greyed out.

Example 9 General registration yields final alignments with target and moving point cloud penetrating each other. This is to be expected as the distance between the two shapes is minimized. For several applications however, it is of interest to achieve penetration–free alignments (Huang et al., 2006). This can be accomplished in a very similar fashion to one–sided fitting by constraining the solution space. Consider a closed target shape. Then, a penetration–free alignment is achieved by constraining the registration to either the interior or exterior of the target point cloud. Let $\mathbf{x}_i$ be the closest point of a moving sample $\mathbf{y}_i$ and $\mathbf{n}_i$ the normal in $\mathbf{x}_i$. With this notation,

$$\mathbf{n}_i^T \cdot (\mathbf{m}(\mathbf{y}_i) - \mathbf{x}_i) \geq 0, \qquad \forall i = 0, \ldots, n$$

constrains the alignment to that side of the target shape the normals are pointing to. We think of open target shapes as artificially closed to apply above definition.

We see that these constraints are linear in the displacement's velocity vector field and may be added to any discussed non–smooth optimization solver. Again, in the case of

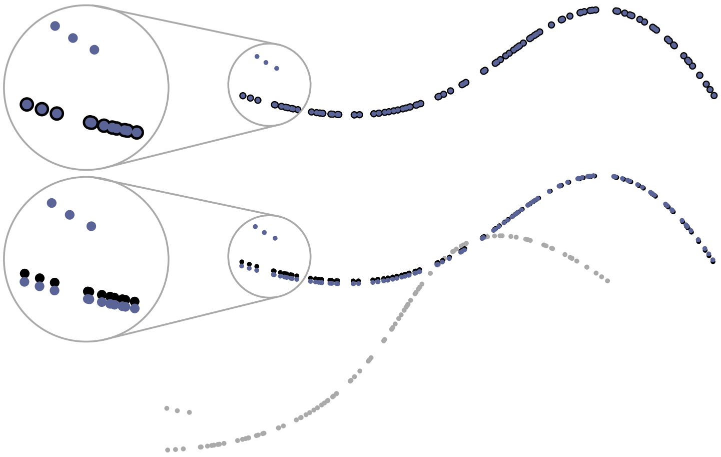

Fig. 13. The moving point cloud (blue) includes three outliers. The l_1–registration (top) is more robust than the l_2–registration (bottom). The initial position of the moving point cloud is shown in grey w.r.t. the target point cloud of the least–squares alignment.

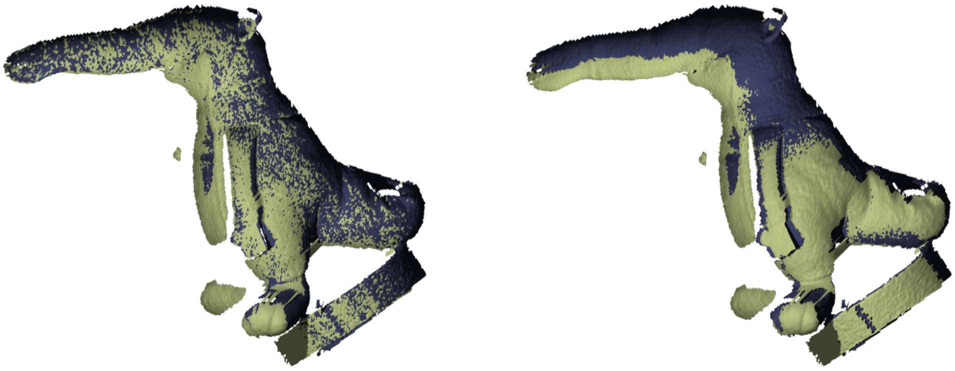

Fig. 14. The l_1–registration aligns the two input point clouds well (left). By contrast, the least–squares registration result is tilted (right). Both point clouds are noisy and include numerous outliers.

the point–to–tangent plane error term, integration of these constraints simplifies the optimization problem. Opposed to the one–sided curve and surface fitting examples, the decrease in computation time is more significant for registration. This is due to the high number of constraints that gets simplified (one constraint per data point). Typically, a penetration–free registration took half the time of an unconstrained registration in the l_1–norm. Please note that the corresponding least–squares registration is an optimization problem with quadratic objective and linear constraints that requires some computational effort as well. In Fig. 15, we show results of an unconstrained and a penetration–free alignment. Again, the point clouds have been converted to triangular meshes after registration for better visualization.

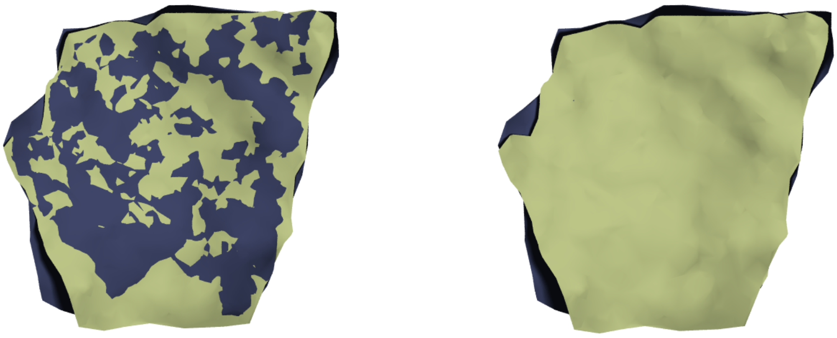

Fig. 15. Unconstrained registration (left) yields a final alignment with mutual penetration of target and moving point cloud. The inclusion of linear constraints—eventually simplifying the smooth variant of the point–to–tangent plane l_1–registration—achieves a penetration–free alignment (right).

6. Conclusions and Future Work

In the present work, we have derived approximations of the unsigned distance function to B-spline surfaces and point clouds. We use these approximations to solve the geometric matching problems of B-spline surface reconstruction from point clouds and local, rigid registration of point sets. Therefore, we employ an optimization framework to solve the emerging non–linear non–smooth problems in an iterative way. Previous work on fitting and registration relies mostly on least–squares methods. In contrast, minimizing the unsigned distance function yields more robust but non–smooth optimization problems. Comparison to l_2–methods reveals that our techniques performs better in the presence of outliers. The moderate increase in computational effort relieves us from administrating additional parameters, that robust l_2–methods typically introduce to the computation. Special applications such as one–sided fitting or penetration-free registration even simplify the optimization problems.

We see several directions for future work. It will be interesting to see how l_1–minimization adapts to other research fields of CAGD and geometry processing. There are numerous algorithms based on least–squares methods that may benefit from the robustness inherent to the l_1–norm. Moreover, a thorough theoretical investigation of convergence characteristics for the minimization of the point–to–point, point–to–tangent plane and curvature based error terms is of particular interest.

Acknowledgements

This work was supported by the Austrian Science Fund (FWF) under grant P18865-N13.

References

Alizadeh, F., Goldfarb, D., 2003. Second-order cone programming. Math. Program. 95 (1), 3–51.

Alliez, P., Cohen-Steiner, D., Tong, Y., Desbrun, M., 2007. Voronoi-based variational reconstruction of unoriented point sets. In: Proc. SGP '07. pp. 39–48.

Barnea, E. I., Silverman, H. F., 1972. A class of algorithms for fast digital image registration. IEEE Transactions on Computers C-21 (2), 179–186.

Bergevin, R., Soucy, M., Gagnon, H., Laurendeau, D., 1996. Towards a general multi-view registration technique. IEEE Trans. Pattern Anal. Mach. Intell. 18 (5), 540–547.

Bernardini, F., Rushmeier, H. E., 2002. The 3D model acquisition pipeline. Comput. Graph. Forum 21 (2), 149–172.

Besl, P. J., McKay, N. D., 1992. A method for registration of 3D shapes. IEEE Trans. Pattern Anal. Mach. Intell. 14 (2), 239–256.

Blake, A., Isard, M., 1998. Active Contours. Springer.

Botsch, M., Pauly, M., Gross, M., Kobbelt, L., 2006. PriMo: coupled prisms for intuitive surface modeling. In: Proc. SGP'06. pp. 11–20.

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press.

Chen, Y., Medioni, G. G., 1992. Object modelling by registration of multiple range images. Image Vision Comput. 10 (3), 145–155.

Dahl, J., Vandenberghe, L., 2009. Cvxopt, version 1.1.1.
URL `http://abel.ee.ucla.edu/cvxopt/`

Dierckx, P., 1993. Curve and surface fitting with splines. Oxford University Press.

Faber, P., Fisher, R. B., 2001. Euclidean fitting revisited. In: Proc. IWVF-4. pp. 165–175.

Fischler, M. A., Bolles, R. C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.

Flöry, S., 2009. Fitting curves and surfaces to point clouds in the presence of obstacles. Computer Aided Geometric Design 26, 192–202.

GNU Project, 2009. Gnu Linear Programming Kit, Version 4.37.
URL `http://www.gnu.org/software/glpk/`

Goshtasby, A. A., 2000. Grouping and parameterizing irregularly spaced points for curve fitting. ACM Trans. Graphics 19 (3), 185–203.

Holland, P., Welsch, R., 1977. Robust regression using iteratively reweighted least squares. Comm. in Statistics A6, 813–827.

Hoppe, H., DeRose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schweitzer, J., Stuetzle, W., 1994. Piecewise smooth surface reconstruction. In: Proc. SIGGRAPH '94. pp. 295–302.

Hoschek, J., 1988. Intrinsic parametrization for approximation. Computer Aided Geometric Design 5 (1), 27–31.

Hoschek, J., Lasser, D., 1993. Fundamentals of Computer Aided Geometric Design. AK Peters.

Huang, Q.-X., Flöry, S., Gelfand, N., Hofer, M., Pottmann, H., 2006. Reassembling fractured objects by geometric matching. ACM Trans. Graphics 25 (3), 569–578, Proc. SIGGRAPH 2006.

Huebner, E., 2009. Bundle method, version 1.2.
URL `http://www.mathematik.uni-trier.de/∼huebner/software.html`

Huebner, E., Tichatschke, R., 2008. Relaxed proximal point algorithms for variational inequalities with multi-valued operators. Optim. Meth. Softw. 23 (6), 847–877.

Kiwiel, K. C., 1990. Proximity control in bundle methods for convex nondifferentiable minimization. Math. Program. 46, 105–122.

Kuhn, H. W., 1973. A note on Fermat's problem. Math. Program. 4, 98–107.

Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., Fulk, D., 2000. The digital michelangelo project: 3D scanning of large statues. In: Proc. SIGGRAPH 2000. pp. 131–144.

Lipman, Y., Cohen-Or, D., Levin, D., Tal-Ezer, H., 2007. Parameterization-free projection for geometry reconstruction. ACM Trans. Graph. 26 (3), 22.

Lobo, M. S., Vandenberghe, L., Boyd, S., Lebret, H., 1998. Applications of second-order cone programming. Linear Algebra and its Applications 284 (1-3), 193–228.

Masuda, T., Yokoya, N., 1995. A robust method for registration and segmentation of multiple range images. Comput. Vis. Image Underst. 61 (3), 295–307.

Mitra, N. J., Nguyen, A., 2003. Estimating surface normals in noisy point cloud data. In: Proc. SCG '03. pp. 322–328.

Nesterov, Y., Nemirovskii, A., 1994. Interior-Point Polynomial Algorithms in Convex Programming. Vol. 13 of SIAM Studies in Applied Mathematics. SIAM.

Nocedal, J., Wright, S. J., 1999. Numerical Optimization. Springer.

Pighin, F., Lewis, J. P., 2007. Practical least-squares for computer graphics. In: SIGGRAPH '07 Courses. ACM, pp. 1–57.

Plass, M., Stone, M., 1983. Curve-fitting with piecewise parametric cubics. In: Proc. SIGGRAPH '83. pp. 229–239.

Pottmann, H., Hofer, M., 2003. Geometry of the squared distance function to curves and surfaces. In: Visualization and Mathematics III. Springer, pp. 223–244.

Pottmann, H., Leopoldseder, S., Hofer, M., 2004. Registration without ICP. Computer Vision and Image Understanding 95 (1), 54–71.

Pottmann, H., Wallner, J., 2001. Computational Line Geometry. Springer.

Pulli, K., 1999. Multiview registration for large data sets. In: 3DIM. pp. 160–168.

Richardson, I. E., 2003. H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia. Wiley.

Rockafellar, R., 1972. Convex Analysis. Princeton University Press.

Rousseeuw, P. J., Leroy, A. M., 1987. Robust regression and outlier detection. Wiley.

Rusinkiewicz, S., Levoy, M., 2001. Efficient variants of the icp algorithm. In: 3DIM. pp. 145–152.

Sharf, A., Alcantara, D. A., Lewiner, T., Greif, C., Sheffer, A., Amenta, N., Cohen-Or, D., 2008. Space-time surface reconstruction using incompressible flow. ACM Trans. Graph. 27 (5), 1–10.

Wang, W., Pottmann, H., Liu, Y., 2006. Fitting B-spline curves to point clouds by squared distance minimization. ACM Trans. Graphics 25 (2), 214–238.

Weber, A., 1909. Über den Standort der Industrien, 1. Teil: Reine Theorie des Standortes. J.C.B. Mohr.

Weise, T., Leibe, B., Gool, L. V., June 2007. Fast 3D scanning with automatic motion compensation. In: Proc. CVPR'07. pp. 1–8.

Weiss, V., Andor, L., Renner, G., Várady, T., 2002. Advanced surface fitting techniques. Comput. Aided Geom. Des. 19 (1), 19–42.

Weiszfeld, E., 1937. Sur le point pour lequel la somme des distances de points donnés est minimum. Tohoku Mathematics Journal 43, 355–386.

Yang, M., Lee, E., 1999. Segmentation of measured point data using a parametric quadric surface approximation. Computer-Aided Design 31 (7), 449–457.